UNIVERSIDAD POLITÉCNICA DE MADRID
ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES
Departamento de Automática, Ingeniería Electrónica
e Informática Industrial

SLAM Geométrico con Modelado Basado en Curvas Spline

Tesis Doctoral

Autor

Luis Pedraza Gómara
Ingeniero Industrial

Directores

Diego Rodríguez-Losada González
Doctor Ingeniero Industrial

Fernando Matía Espada
Doctor Ingeniero Industrial

Madrid, 2009

Tribunal nombrado por el Magfco. Y Excmo. Sr. Rector de la Universidad Politécnica de Madrid, el día ..... de ..... de 200...

Presidente : .....

Vocales : .....

.....

.....

Secretario : .....

Suplentes : .....

.....

Realizado el acto de defensa y lectura de Tesis el día ..... de ..... de 200... en la E.T.S.I. Industriales de la U.P.M., acuerdan otorgar la calificación de .....

EL PRESIDENTE LOS VOCALES

EL SECRETARIO

Este libro está dedicado a mis padres:
Esther y Antonio.
Por su sacrificio, por su ilusión, por su amor.

Agradecimientos

Mientras escribo estas líneas, me parece casi mentira que todo esté a punto de terminar. ¡Se ve la luz al final de túnel! Han sido unos cuantos años bastante intensos, y el camino, aunque apasionante, no ha estado exento de obstáculos y piedras de las que cortan; de esas que se meten dentro del zapato, hacen daño al caminar y cuesta mucho sacar. Pero no había más remedio: había que seguir moviéndose.

Ahora llega el momento de parar, tomar aire profundamente —este frío aire de noviembre— y hacer balance. Los últimos meses han estado marcados por una constante, una idea fija que pesaba como una losa sobre el pecho y la mente: escribir la tesis. El tiempo se agotaba. Pero *mi parto*, como lo llamo cariñosamente, ya está entre tus manos y te dispones a leerlo. ¿Mereció la pena el esfuerzo? Tú, mejor que nadie, y el tiempo, lo decidiréis. Mis objetivos personales, creo, se han cumplido. Yo sólo quería hacer algo útil, algo nuevo y original, algo que se entendiese. Teniendo en cuenta lo complicado que es conseguir todo ello *en esto del SLAM*, creo que puedo estar razonablemente satisfecho con el resultado. También quería hacer algo que se usara. Pero esto último ya no está en mi manos.

"Parir es bonito, pero casi siempre doloroso..."

Al echar la vista atrás, me doy cuenta de que este libro no es obra únicamente mía. Mucha gente ha intervenido, de una manera u otra, en el florecimiento de las ideas, el desarrollo de las ecuaciones, la programación de los algoritmos, la interpretación de los resultados y, casi, casi, hasta en la redacción de los capítulos y la elección del papel para su impresión. Y no sólo desde el punto de vista intelectual, sino también del personal y humano. Aunque sólo sea por haberme enseñado a sacar el coraje y la fuerza, no se sabe muy bien de dónde, para decir:

"Esto va a salir. Por mis narices que sale. Y lo voy a terminar."

Quisiera comenzar recordando a la gente con la que compartí más de dos años en el Instituto de Automática Industrial del CSIC. Allí di mis primeros y tambaleantes pasos en esto de la investigación, y descubrí lo que sospechaba: que me gustaba mucho. El camino no siempre es fácil, a menudo asalta la duda, y siempre acecha la incertidumbre. Pero al final merece la pena tener la oportunidad de hacer cosas nuevas y alucinantes; y por ello me siento un privilegiado. Gracias a Manuel Armada, mi primer jefe, siempre comprensivo y de buen humor. Y gracias a mis compañeros de aquel entonces; algunos ya eran amigos, otros lo son desde entonces, todos ellos lo serán siempre.

"Excelente, como siempre."

Gracias a la Consejería de Educación de la Comunidad de Madrid por la beca FPI que me concedió en octubre de 2003 y que, durante la mayor parte de los cuatro años de su duración, me permitió dedicarme con libertad a la apasionante tarea de la investigación. Si algo me ha quedado claro en este tiempo, es que para que las cosas funcionen, se tienen que dar una serie de condiciones necesarias, aunque no siempre suficientes. Entre ellas están disponer del suficiente tiempo, claridad mental, motivación y ayuda. Y todas ellas pasan casi siempre por disponer de una adecuada financiación. Quiero pensar que en este país mío las cosas empiezan a funcionar como deben, y que cada vez más gente se da cuenta de que es mejor remar todos en la misma dirección.

Gracias a Gamini Dissanayake y a Jaime Valls Miró, que me acogieron en el *ARC Centre of Excellence for Autonomous Systems* de Australia durante 4 meses en 2006. Allí me escucharon, se sorprendieron, y me ayudaron a encontrar ese escurridizo y fino hilo del cual comenzar a tirar. Al principio no me entendían —tal vez ni yo me entendía—: *"Vamos a sentarnos y nos cuentas de qué va eso de los splines"*. Qué importante es a veces tener a alguien dispuesto a escuchar, a involucrarse, y a dedicar diez minutos de su tiempo a intentar comprender tus ideas. Gracias, Gamini y Jaime, por no limitaros a escuchar al *españolito* que llegó con cuatro meses para hacer muchas cosas. Gracias por acercaros a la mesa del rincón y preguntar: "Hey, Luis, ¿cómo va eso? ¿Vemos juntos el código?".

"Yo lo haría así. Pero no creo que te salga..."

Y al final salió.

Gracias a mis directores de tesis, Fernando Matía y Diego Rodríguez-Losada. Gracias Fernando por abrirme las puertas del Grupo de Control Inteligente en ese momento que tanto necesitaba un *cambio de aires*. Gracias por estar siempre dispuesto a facilitar mi tarea, y ayudarme a ver el lado bueno de las cosas. Gracias Diego por involucrarte, por apasionarte conmigo con *esto de los splines*, y por sujetarme por el brazo cada vez que esas malditas piedras en los zapatos comenzaban a hacer insoportable el caminar. Ser un director de tesis es mucho más que sentarse y escuchar. Mucho más que ayudar a identificar y resolver un problema. Mucho más que ser un guía. Ser un director de tesis es ser un incansable compañero de camino. Gracias por haber estado ahí los últimos cuatro años.

"Escribe, escribe..."

Gracias también a los dos directores del Departamento de Automática, Ingeniería Electrónica e Informática Industrial de la UPM que he conocido: Rafael Aracil y Agustín Jiménez. Gracias a su esfuerzo y dedicación se dan esas condiciones que comentaba, y que permiten que, no sólo mi trabajo, sino el de todos los miembros del departamento, se realice bajo las mejores condiciones posibles; algo que interesa a todos. Gracias también a Ramón Galán, el hombre tranquilo siempre dispuesto

a ayudar. Y un agradecimiento muy especial a las pacientes explicaciones de Pablo San Segundo, guía de lujo en la búsqueda del "clique perdido".

"Luis, ¿qué necesitas?"

Gracias también a Tere, Rosa y Carlos por ayudarme a desenvolverme en ese terreno oscuro y farragoso que me aterra más que el de las ecuaciones y los teoremas: el de los formularios, las recogidas de firmas, y la interpretación de normativas y resoluciones no siempre claras. Y gracias por hacerlo siempre con una sonrisa dibujada en la boca.

"Esto te lo tiene que firmar Agustín, ¿eh?"

Gracias a mis compañeros durante estos últimos cuatro años. Iñaki, Alberto, Javier, Lupe, Kike... Con ellos a veces he compartido mucho más que un despacho o la regleta de los enchufes. Gracias por esos ratos de charla, por esas bromas, por compartir las mismas preocupaciones, los mismos sueños.

"¿Un café?"

Gracias a mis amigos, a los cuales (lo siento) no he cuidado como debería en los últimos tiempos. Casi mejor así... que *no ha estado el horno para muchos bollos.* Gracias por vuestra preocupación, por escribirme, por preguntar. Nos vemos pronto.

"Luis, ¿qué tal va todo? ¿Cuándo terminas?"

Gracias a Juanje Gómez por guiarme a través del mundo del *Photoshop*, del *Illustrator*, de las paletas de herramientas que aparecen y desaparecen. Por mostrarme los rudimentos del mundo del diseño que tanto me atrae y al que espero poder dedicar algún día más tiempo —espero que no por necesidad—. Sin su ayuda, esta tesis habría sido mucho menos bonita.

"Yo lo veo bien, pero mejor quedaría centrado."

Gracias a mi familia; a mis padres Antonio y Esther y a mi hermana Elena. Ellos son responsables, para bien o para mal (y creo que más bien lo primero), de gran parte de lo que soy. Su ejemplo de perseverancia, de sacrificio, de ilusión, y de tesón en terminar lo empezado, han sido determinantes para finalizar esta tesis. Sé que no siempre os he contestado. Sé que a veces he estado irascible. Sé que habéis estado preocupados, y que ya no os atrevíais a preguntar. Pero mi genio también os lo debo a vosotros. Gracias, mil gracias siempre.

"Luis, hijo, ¿estás contento?"

Ya sí. Ya está terminada. Espero leer pronto.

También, y aunque muchos ya no están, me gustaría dedicar algunas líneas de agradecimiento a todos esos gigantes a cuyos hombros nos subimos los más pequeños, descubriendo nuevos horizontes y maravillándonos de todo lo que con paciencia y esfuerzo puede conseguir el intelecto humano. A Herón de Alejandría, a Aristóteles, a Arquímedes, al grandísimo Leonardo, a Zhang Heng, a René Descartes, a Pierre Bézier, a Carl de Boor, a Rudolf Kalman... A todos aquellos que decidieron que otro punto de vista era posible. A los que que no temieron enfrentarse a un problema y resolverlo con la razón.

A quienes se arriesgaron a equivocarse. Gracias.

Resumen

Uno de los retos fundamentales de la robótica actual, consiste en obtener mecanismos robustos y eficientes para modelar entornos de creciente complejidad, utilizando máquinas móviles en su exploración. Esto es conocido como el problema de la Localización y Modelado Simultáneos, o SLAM ("Simultaneous Localizacion and Mapping"), en el que la máquina utiliza concurrentemente el mapa que está construyendo para obtener su propia localización.

Dentro de las soluciones al problema SLAM, destacan las probabilísticas; especialmente aquellas que utilizan un Filtro Extendido de Kalman (EKF) para realizar la estimación del mapa. A pesar de existir gran número de soluciones SLAM-EKF en la literatura, aún surgen importantes barreras en la aplicación de los algoritmos. Tal vez una de las más importantes es la que tiene que ver con el modelado geométrico de entornos arbitrariamente complejos, obstáculo que se intenta solventar aquí.

En esta tesis se propone, por primera vez, la utilización de curvas spline como herramienta de modelado. Se aprovecha así la eficacia y versatilidad de estas herramientas —cultivada durante décadas en el dominio de los gráficos por computador, el diseño y la arquitectura—, en el complejo problema de obtener un modelo del entorno cuando no es posible extraer características más sencillas como puntos o segmentos. Se amplía así considerablemente el ámbito de aplicación de populares soluciones existentes, y se abre camino a otras, y a nuevas formas de razonar y obtener información sobre el entorno.

Se desarrollan los mecanismos necesarios para extraer información de los datos suministrados por un sensor ampliamente utilizado en robótica móvil como es el telémetro láser. Se desarrollan las ecuaciones que permiten fusionar eficazmente la teoría matemática de las curvas spline con el marco de trabajo del SLAM-EKF, proponiéndose adecuados esquemas de asociación de datos, modelo de observación, y facilitando algoritmos para extender progresivamente objetos del mapa a medida que se exploran nuevas áreas.

También se propone un método para paliar el coste computacional del algoritmo, creciente de manera inevitable con el tamaño del mapa. Se descompone para ello el problema global en la construcción de distintos mapas de tamaño reducido, y se aprovecha el modelado geométrico para extraer información que permita relacionar todos los mapas entre sí y actualizar en consecuencia su estructura global.

Todos los resultados presentados son validados experimentalmente, tanto con datos simulados como extraídos en entornos reales. Se comprueba que las propiedades

de convergencia y consistencia de los algoritmos son equivalentes a los de cualquier otra solución SLAM-EKF. Pero ahora es posible construir mapas geométricos en entornos donde esto antes no era posible, con la ventaja añadida de no depender de una geometría concreta a ser detectada por los sensores del robot.

Abstract

One of the fundamental challenges of today's robotics is to obtain robust and efficient mechanisms for modeling increasingly complex environments, using mobile robots for their exploration. This is known as the Simultaneous Localization and Mapping problem (SLAM), where the robot concurrently uses the map to determine its own position.

Probabilistic solutions are the most popular for the SLAM problem; specially those based on the use of an Extended Kalman Filter (EKF) to estimate the map. Despite the large number of SLAM-EKF solutions in the literature, still significant barriers arise in the implementation of the algorithms. Perhaps one of the most important is the geometric modeling of arbitrarily complex environments. This obstacle is intended to be solved here.

In this Thesis it is proposed, for the first time, the use of spline curves as modeling tool. This takes advantage of the efficiency and versatility of these tools —cultivated during decades in the field of computer graphics, design and architecture—, in the complex problem of obtaining a model of the environment when it is not possible to extract simpler features such as points or segments. Thereby, the scope of popular existing solutions is significantly extended, and the way is open to other solutions and new ways of thinking and learning about the environment.

The necessary mechanisms to extract information from data supplied by a widely used sensor in mobile robotics, as it is the laser rangefinder, are developed. Equations that allow to effectively merge the mathematical theory of spline curves with the framework of the SLAM-EKF are presented, and appropriate forms of data association, an observation model, and algorithms for progressively extending map objects as new areas are explored, are provided.

It is also presented a method to reduce the computational cost of the algorithm, which inevitably increases with the size of the map. This breaks down to the overall problem in the construction of various maps of small size, and uses geometric modeling to extract information that allows all maps to relate to each other and update, accordingly, their overall structure.

All the presented results are experimentally validated, both with simulated and real data. It is found that the properties of convergence and consistency of the algorithms are equivalent to those of any other SLAM-EKF solution. But now it is possible to build geometric maps of environments where it was not possible before, with the added advantage of not relying on a specific geometry to be detected by

the sensors of the robot.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

El objetivo de este primer capítulo es el de poner en situación y guiar al lector de la presente tesis, que tiene por título **"SLAM Geométrico con Modelado Basado en Curvas Spline"**. Del propio título ya se desprenden algunas ideas fundamentales. *"SLAM"* es el acrónimo comúnmente utilizado por la comunidad científica para referirse a las técnicas empleadas en robótica e inteligencia artificial que buscan modelar el mundo que nos rodea aprovechando los recursos que proporcionan robots móviles, u otros vehículos dotados de los sensores y capacidad de cómputo necesarios. Procede, como no podía ser de otra manera, de su denominación en inglés: *"Simultaneous Localizacion and Mapping"* (Localización y Modelado Simultáneos).

Así pues, se espera que esta tesis presente algunas contribuciones novedosas al complejo problema, como veremos, del SLAM. Además se tratará un tipo particular de SLAM: aquel que busca describir los entornos a partir de características geométricas presentes en el mismo, y descriptibles matemáticamente. De ahí el adjetivo "geométrico", que diferencia las técnicas tratadas en este escrito de otras alternativas que se encuentran en la bibliografía y que son también comúnmente empleadas.

La parte final del título nos indica que la técnica de modelado elegida está basada en la utilización de curvas spline. Esta es precisamente la más importante y fundamental contribución de esta tesis: la de mostrar, por primera vez, cómo geometrías de creciente complejidad pueden ser efectivamente modeladas empleando herramientas matemáticas de gran versatilidad y potencia como son las curvas spline, y cómo esta nueva forma de razonar sobre el entorno puede ser empleada para obtener un mapa compacto del mismo. A lo largo del texto nos referiremos a las curvas spline de manera equivalente como *splines*.

Primeramente, la sección 1.1 supone una breve introducción, desde un punto de vista histórico, técnico, humano, y casi filosófico, a los motivos que llevan a emprender una tesis de estas características. Se pretende mostrar cómo el ser humano ha buscado, casi desde el origen de la Historia, imitar la vida, reproducir el movimiento, y percibir, manipular y conquistar su entorno utilizando dispositivos artificiales de la más variada índole. Todo ello siguiendo un instinto básico y primario como es la incansable búsqueda de el progreso y la mejora tecnológica.

La sección 1.2 se acerca aún más al tema de la tesis, anunciando algunas de las limitaciones del estado actual de la técnica que pondrán de relevancia la importancia de las mejoras que aquí se proponen. Estas limitaciones llevarán de manera natural a la presentación de los objetivos que con este trabajo se han querido alcanzar, y que se enumeran en la sección 1.3. Dichos objetivos pretenden aportar un pequeño granito de arena a los medios disponibles para modelar entornos cada vez más complejos, donde métodos existentes pierden su validez.

Finalmente, en la sección 1.4 se describe la estructura de la tesis, sirviendo de mapa de ruta para el lector que sigue estas líneas.

1.1. Motivación de la Tesis y Marco de Trabajo

La *robótica* es la ciencia que persigue la percepción y manipulación del mundo físico que nos rodea a través de dispositivos programables y controlables mediante computadores [192]. A tales dispositivos los conocemos como *robots*. Cuando los robots son capaces de desplazarse de manera autónoma a través de su entorno (*i.e.* sin intervención de operador humano alguno), decimos además que se trata de *robots móviles*.

Tal vez sean estas máquinas aquellas que a día de hoy se encuentran más cerca de materializar uno de los sueños más vehementemente anhelados por el género humano: la creación de nuevas formas de vida. No en vano, cualquier noción empírica acerca del concepto de *vida* viene ligada en mayor o menor medida al concepto de *movimiento*. De hecho, la biología considera al movimiento como una de las características sustanciales de los seres vivos, y que los diferencia de la materia inerte. El movimiento es aquí entendido como desplazamiento mecánico de alguna o todas las partes componentes de una entidad; esto incluye el crecimiento de los organismos o los tropismos de las plantas. También es intuitivo pensar en algo vivo como aquello que responde a estímulos externos. Esta respuesta se expresa frecuentemente en forma de movimiento; desde los girasoles que se orientan buscando los rayos del Sol, hasta el niño que corre tras una pelota, todos los seres vivos comparten una característica que los hace únicos: se mueven por sí mismos.

Esta idea de vida como movimiento también ha sido adoptada por muchas escuelas filosóficas a lo largo de la Historia. Pensemos por ejemplo en Aristóteles, el filósofo naturalista para quien el alma era un principio vital. Todos los seres vivos (seres *animados*) comparten la característica común de tener movimiento por sí mismos (*motus a se ipso*). Para el sabio estagirita Dios sería el primer motor inmóvil de todo el Universo.

Sin entrar en demasiadas disquisiciones filosóficas, lo cierto es que el ser humano siempre ha buscado la manera de reproducir por medios artificiales uno de los misterios más fascinantes de la realidad que nos rodea: la propia vida. Sin duda alguna, uno de los caminos que más y mejor se han acercado a este objetivo es la robótica.

1.1.1. La motivación histórica

Pero, ¿de dónde surge esta disciplina científica, y cuáles son sus motivaciones y objetivos? Para encontrar las raíces de la robótica, tal y como hoy la conocemos, debemos remontarnos muchos siglos atrás en el tiempo. Ya en el año 1500 a.C. se sabe que el ingeniero Amenhotep construyó una estatua del rey Memón de Etiopía, que emitía sonidos al ser iluminada por los primeros rayos solares al amanecer. Así, en el Antiguo Egipto se construyeron gran cantidad de artefactos mecánicos, la mayoría de ellos asociados al culto religioso, que adoptaban la forma de estatuas de reyes o dioses que despedían fuego por sus ojos, o estaban dotadas de articulaciones móviles que, al ser accionadas por los sacerdotes y servidumbre de los templos, buscaban asombrar y causar temor a todo aquel que las contemplara.

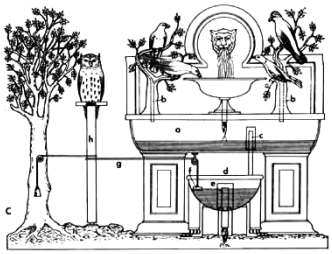
En el sigo I, Herón de Alejandría (*aprox.* 10 d.C-70 d.C.) diseñó numerosos ingenios mecánicos que aplicaban los conocimientos disponibles en la época en materia de hidráulica, neumática, mecánica... y dejó constancia escrita del saber acumulado por los reyes Ptolomeos en sus bibliotecas. Contribuyó al saber antiguo con numerosas obras como *"Métrica"* (dedicada al estudio de las áreas y los volúmenes y a la división del espacio en general), *"Mecánica"* (que describe máquinas simples, como tornos o palancas, y sus aplicaciones prácticas), *"Neumática"* (donde recoge conocimientos sobre mecánica de fluidos) o *"Catóptrico"* (en el cual describe los principios de funcionamiento de espejos planos y curvos, y las leyes fundamentales de la reflexión).

En su obra *"Autómata"*, Herón describe algunos de los mecanismos e invenciones, propios y ajenos, creados hasta el momento con propósitos religiosos y, cada vez con más frecuencia, lúdicos. Toda suerte de piezas mecánicas, engranajes, palancas y sistemas de conducción de agua o de vapor, dan vida a detalladas reproducciones de pájaros que aletean, gorjean y beben agua de una fuente, grupos escultóricos animados, o que permiten la apertura automática de las puertas de un templo al encender un fuego a su entrada. A Herón también se le atribuye la invención de la *Eolípila*, dispositivo capaz de transformar energía térmica en energía mecánica, y primer antepasado conocido de la máquina de vapor que tanta relevancia tuvo durante la Revolución Industrial.

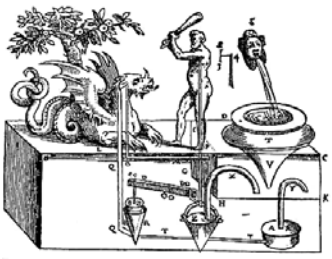
En el capítulo 34 de su libro *"Dioptra"*, Herón describe el funcionamiento de un rudimentario odómetro[1]. Este dispositivo ya había sido descrito por Vitruvio hacia el año 25 a.C., aunque su invención se atribuye a Arquímedes (287 a.C.-212 a.C.) durante la Primera Guerra Púnica. Consistía en un sistema de engranajes acoplado a una rueda de diámetro tal que al dar 400 revoluciones el recorrido lineal fuera exactamente de una milla romana (unos 1480 metros). Cada vez que esta distancia era completada, un mecanismo liberaba una piedra o esfera metálica que era depositada en una caja. Al final del recorrido no había más que contar las piedras acumuladas para conocer la distancia total recorrida. Leonardo da Vinci intentó sin éxito construir este odómetro (aunque sí fue capaz de construir su propia versión, tal y como aparece dibujada en el *Codex Atlanticus*, hoja 1a, hacia 1503), y hubo que esperar a que el ingeniero André Sleeswyk lo consiguiera en 1981 basándose en

[1]Odómetro. Del griego *hodós* (camino) y *métron* (medida)

(a) *Eolípila*



(b) Pájaros de Herón



(c) Hércules y el Dragón

Figura 1.1: Algunas de las invenciones descritas por Herón de Alejandría. (a) *Eolípila* de Herón de Alejandría, antepasado de la máquina de vapor. (b) Pájaros de Herón. (c) Autómata descrito por Herón que representa a Hércules luchando con un dragón. Cuando Hércules golpea la cabeza del dragón, este último arroja agua contra la cara del héroe.

los planos originales [176].

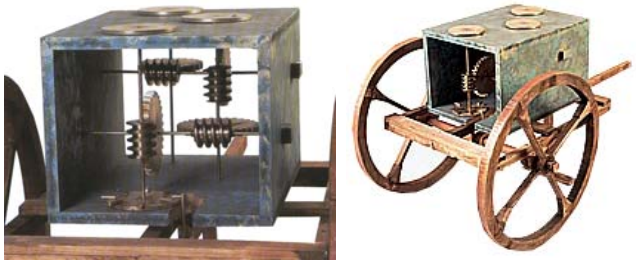
El famoso científico e inventor chino Zhang Heng (78-139 d.C.) también construyó un odómetro ciertamente original. El dispositivo, denominado "carruaje con tambor contador de *lis*" [2], consistiría en dos figuras humanas de madera accionadas mecánicamente. Cada vez que el carruaje recorría un *li*, la primera de ellas golpeaba un tambor. Cuando se acumulaba una distancia de 10 *lis*, la segunda de las figuras hacía sonar un gong o una campana con su brazo articulado.

El gran genio renacentista Leonardo da Vinci no pudo tampoco resistirse a la tentación de crear por sí mismo alguna forma de vida artificial o semejanza de esta [162]. Un interesante ejemplo de ello es el león mecánico que Francisco I le encargó construir, hacia 1515, y cuyo objetivo no está del todo claro. Según unas fuentes habría sido construido con motivo de las conversaciones de paz con el papa León X, en Bologna, mientras que según otras su función habría sido la de homenajear al rey francés a su entrada triunfal en Lyon. Lo cierto es que el león era capaz de caminar en línea recta para luego detenerse y abrir su pecho, que mostraba estar repleto de lirios y otras flores.

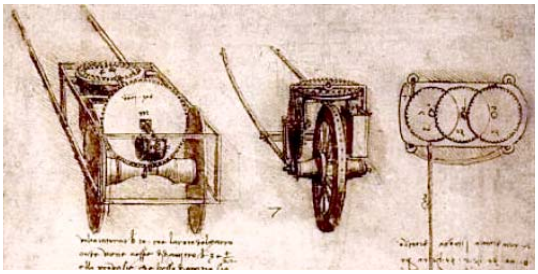
Leonardo pasa también por ser el inventor de uno de los primeros autómatas humanoides de la Historia y, posiblemente, el que mejor imitaba los movimientos propios de un ser humano. El *"Caballero de Leonardo"*, es un autómata diseñado alrededor del año 1495 y que posiblemente nunca se llegara a construir. Parece que el artista, científico e ingeniero, lo diseñó para probar que el cuerpo del ser humano podía ser imitado en su funcionamiento, y para ello partió de los estudios iniciales sobre anatomía que había realizado en Florencia. Así pues, es parcialmente fruto de la investigación anatómica del Canon de las Proporciones, que se detallan en el famoso dibujo del *Hombre de Vitruvio* (hacia 1487) conservado en la *Gallerie dell'Accademia*, en Venecia. Así, esta creación es una extensión de su hipótesis de que el cuerpo humano es una máquina en su estructura, y que sus movimientos pueden ser imitados utilizando piezas mecánicas como palancas y poleas.

[2] El *li* es una medida de longitud china, equivalente a 500 metros

(a) *Odómetro de Zhang Heng*



(b) Odómetro de Herón



(c) Odómetro de Leonardo

Figura 1.2: Odómetros en la antigüedad. (a) Dibujo del odómetro de Zhang Heng encontrado en una tumba de la dinastía Han, hacia el año 125. (b) Reconstrucción del odómetro de Herón de Alejandría y vista de detalle de sus engranajes. (c) Bocetos autógrafos del odómetro de Leonardo da Vinci.

El caballero, enfundado en una armadura de diseño germano-italiano, sería capaz de realizar movimientos humanos como sentarse y levantarse, o mover los brazos y el cuello. A pesar de que se ha especulado con la posibilidad de que Leonardo intentara construir un guerrero mecánico, parece más lógico pensar que el objetivo del caballero era impresionar y entretener a público y cortesanos. Sus diseños aparecen detallados en el *Codex Huygens*, descubierto en la década de los 50 por el estudioso Carlo Pedretti si bien su existencia ya era conocidadesde antes [140]. Precisamente fue un libro de este último lo que llamó la atención del ingeniero norteamericano Mark Rosheim para dedicar 5 años de su vida al estudio de los bocetos de Leonardo. Estos le inspiraron para crear una serie de robots llamados *"Arthrobots"*,que están siendo construidos por la empresa Ross-Hime Designs, Inc.[3] para la NASA con el ambicioso objetivo de servir de asistencia en la Estación Espacial Internacional, o la posible futura colonización del planeta Marte [160, 161].

La prolífica actividad creativa de Leonardo le sirvió también para convertirse en el diseñador del artilugio más parecido hasta entonces a lo que hoy consideramos un robot móvil. Su *"carro autopropulsado"* era un vehículo de tres ruedas capaz de moverse de manera autónoma, en el que se podía programar su trayectoria de antemano. La fuerza motriz necesaria se obtenía a partir de un complejo sistema de resortes capaces de almacenar la energía mecánica suficiente para desplazar el móvil a lo largo de unos cuantos metros. Los bocetos del artefacto se encuentran repartidos en varios documentos, pero los más claros pueden observarse en el folio 812r del *"Codex Atlanticus"*.

[3] *http://www.anthrobot.com*

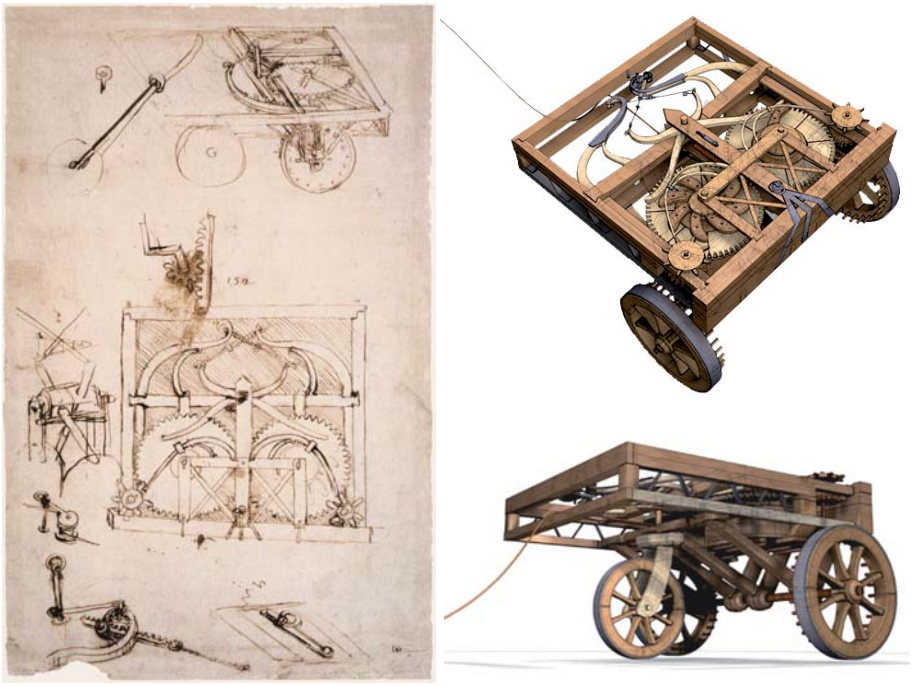


Figura 1.3: Carro autopropulsado de Leonardo da Vinci. Bocetos del artefacto tal y como puede observarse en el folio 812r del *Codex Atlanticus* (izquierda). Reconstrucción tridimensional del carro según estos bocetos (derecha).

La falta de documentación hizo imposible durante siglos su reproducción. Pero, una vez más, la colaboración entre Pedretti (considerado el más importante experto vivo en la figura y obra de Leonardo) y Rosheim hizo posible en 2004 su construcción. A este ingenio mecánico se refería Girolamo Calvi en 1936 como "el FIAT de Leonardo". Sin embargo es poco probable que el carro autopropulsado fuera pensado para transportar objetos o personas, sino más bien para impresionar al público en espéctaculos teatrales o como entretenimiento de la nobleza.

Pero los autómatas de Leonardo no son los únicos antepasados de los modernos robots que podemos encontrar en el Renacimiento. Especialmente curioso es también el monje mecánico móvil que se conserva en uno de los museos de la *Institución Smithsonian*, en Washington[4]. Se trata del "Hombre de Palo" (Fig. 1.4.(a)), la figura de un monje de uno 40 cm de alto, fabricada en madera y metal, y cuya construcción se atribuye a Juanelo Turriano (1501-1585), relojero, matemático, inventor e ingeniero del emperador Carlos V, y más tarde nombrado matemático mayor por Felipe II. Accionado por un mecanismo de relojería, el monje camina a lo largo de un cuadrado mientras golpea su pecho con el brazo derecho. Al mismo tiempo que se desplaza mueve una pequeña cruz de madera y un rosario con la mano izquierda, así como la cabeza, los ojos, y la boca. De vez en cuando, el monje acerca la cruz a su boca para besarla. Sorprendentemente, más de 400 años después de su construcción, datada hacia 1560, el ingenio aún funciona.

Se conoce la existencia de otras dos figuras de monjes autómatas, también datadas hacia la segunda mitad del S. XVI, y que comparten características similares en

[4] *http://www.si.edu*

(a) (b) (c)

Figura 1.4: Monjes mecánicos renacentistas

cuanto a diseño y accionamiento. Una se conserva en el *Museo de Artes Aplicadas* de Budapest[5] (Fig. 1.4.(b)), mientras que la otra puede ser observada en el *Deutsches Museum* de Munich[6] desde 1985 (Fig. 1.4.(c)).

La lista de mecanismos que podemos encontrar a lo largo de la historia, a caballo entre el meticuloso trabajo de relojería y el complejo automatismo hidráulico o mecánico es casi interminable. Todos ellos han buscado de una u otra manera reproducir el movimiento característico de las formas de vida, con el objetivo de asombrar, entretener, divertir, o como mera prueba y puesta en práctica de teorías científicas y avances tecnológicos acumulados en cada época.

Sin embargo, hubo que esperar a la llegada de la revolución industrial, a partir de la segunda mitad del siglo XVIII, para que las ideas que se habían estado gestando a lo largo de los siglos encontraran verdadera aplicación práctica con fines productivos [195]. No se trataba ya simplemente de imitar la vida. Las nuevos desarrollos permitían por vez primera ayudar o sustituir al ser humano en una infinidad de tareas, especialmente cuando estas eran de carácter repetitivo o peligroso. La máquina de vapor de James Watt (1774), diseñada como mejora de la máquina de vapor atmosférica de Thomas Newcomen (1712), o las diferentes versiones aparecidas del motor de combustión interna que culminarían con la invención del motor de cuatro tiempos por Karl Otto (1876), propiciarían un conjunto de cambios socioeconómicos, tecnológicos y culturales como no se habían visto desde el Neolítico.

Pero a las vetustas máquinas de la revolución industrial aún les faltaba mucho para convertirse en algo parecido a los modernos robots. Fue con la llegada del siglo XX y tras el final de la Segunda Guerra Mundial cuando los desarrollos tecnológicos en electricidad y electrónica permitieron la aparición de las máquinas de control numérico primero y, más tarde, la llegada al mundo de los primeros manipuladores robóticos. El primer robot comercial de la historia vio la luz en 1956, y se llamó Unimate. Fue fabricado por la compañía Unimation, fundada por Joseph Engelberger y George Devol, y estaba basado en la patente de este último para un "manipulador programable" obtenida en 1954 [69].

Tal vez el primer ejemplo de un auténtico robot móvil sea la tortuga desarrollada

[5] *http://www.imm.hu*
[6] *http://www.deutsches-museum.de*

(a) iRobot Roomba 560

(b) Automower Solar Hybrid

(c) OFRO

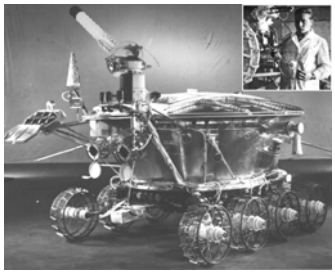
Figura 1.5: Robots móviles comerciales. (a) El robot de limpieza Roomba, dirigido en su operación por faros de luz infrarroja. (b) El robot cortacésped Automower, en su versión con paneles solares. (c) El robot de vigilancia para exteriores OFRO.

por Walter, en el año 1948, que era capaz de desplazarse exhibiendo un comportamiento aparentemente inteligente, al ser capaz de reaccionar ante la presencia de obstáculos en su entorno. Desde entonces, los intentos por crear máquinas móviles dotadas cada vez de mayor autonomía han ido en aumento, hasta el punto de que a día de hoy existen numerosas aplicaciones comerciales de robots móviles que asisten al ser humano en numerosas tareas. Basten como ejemplo los robots aspiradores *Trilobite* o *Roomba* (de los fabricantes Electrolux[7] e iRobot[8], respectivamente), los robots cortacésped como el *Automower*[9] desarrollado por la firma sueca Husqvarna[10] (cuya última versión, el *Automower Solar Hybrid*, es capaz de funcionar de manera híbrida con sus placas solares y su batería), los robots domésticos de vigilancia *Roborior*[11] de la empresa nipona Tmsuk[12], o los más avanzados de uso profesional *MOSRO* y *OFRO* de la alemana RoboWatch[13].

Pero no es sólo en la superficie terrestre donde estos nuevos habitantes de nuestro planeta pueden servirnos de ayuda. Hace tiempo que el ser humano se propuso la meta de conquistar otros mundos y, para ello, los robots móviles constituyen una herramienta de gran utilidad como mensajero previo o heraldo de la visita del hombre. El robot soviético *Ludokhov I* fue pionero en tareas de exploración extraterrestre. Llegó al *Mare Imbrium* (Mar de las Lluvias) lunar a bordo del Luna 17 el 17 de noviembre de 1970. En la figura 1.6.(a) puede verse el rover[14] junto con un detalle de la rueda odométrica que iba colocada en la parte posterior del vehículo.

Más recientemente todos recordamos al robot de exploración lunar *Sojourner*, que llegó a Marte a bordo de la misión *Mars Pathfinder* el 4 de julio de 1997 (Fig. 1.6.(c)), o al *Spirit* (*Mars Exploration Rover - A*), rover de exploración Marciana que llegó a su destino el 4 de enero de 2004 (Fig. (d)), tres semanas antes que su hermano gemelo *Opportunity*. Ambos robots aún se encuentran operativos, y a fecha

[7] *http://trilobite.electrolux.com*
[8] *http://www.irobot.com*
[9] *http://www.automower.es*
[10] *http://www.husqvarna.es*
[11] *http://www.roborior.com*
[12] *http://www.tmsuk.co.jp*
[13] *http://www.robowatch.de*
[14] Vehículo diseñado especialmente para la exploración espacial.

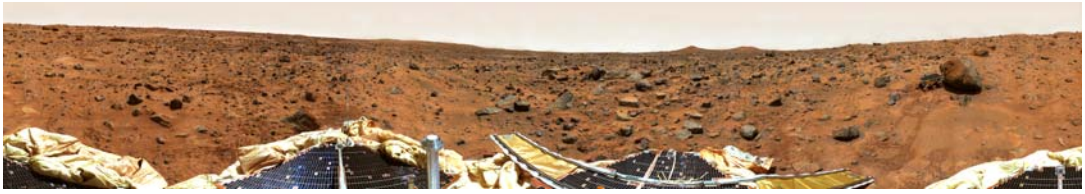


(a) *El rover lunar Ludokhov I*

(b) Controlando el *Ludokhov I* desde la Tierra

(c) *El robot Sojourner*

(d) El rover de exploración marciana *Spirit*



(e) *Panorámica del Ares Vallis (Vaye de Ares), en la región Chryse Planitia (Planicies de Oro)*

Figura 1.6: Robots de exploración lunar y marciana. (a) El robot soviético Ludokhov I. (b) Un operario soviético controla el Ludokhov I desde la Tierra. (c) El robot de exploración Sojourner. (d) El robot Spirit. (e) Panorámica del suelo marciano enviada por la misión *Mars Pathfinder* a su llegada al planeta rojo.

del 30 de septiembre de 2008 el odómetro del *Spirit* había medido una distancia total recorrida de 7.528 metros.

1.1.2. Los robots interactivos

Otro tipo interesante de robot móvil, con grandes perspectivas de encontrar aplicación práctica de utilidad y popularizar su uso en el futuro más inmediato, es el robot interactivo. Se trata de máquinas que han de ser capaces de navegar, dialogar, razonar, aprender y sobrevivir en entornos adaptados por y para el ser humano. Para conseguir este objetivo, estos robots deben tener la habilidad de combinar su conocimiento sobre el entorno que les rodea con la información que adquieren a través de sus sistemas sensoriales. Se trata pues de robots capaces de operar en entornos humanos, llevando a cabo tareas sociales útiles como robots instructores (en museos y colegios), robots de entretenimiento (en parques de atracciones), o

robots de compañía y asistencia (en el hogar o en hospitales).

Existen algunos ejemplos funcionales de esta clase de robot, algunos de los cuales incorporan una cabeza capaz de mostrar expresiones faciales reconocibles por el ser humano, e incluso de interactuar verbalmente. El grupo de investigación en el que se ha desenvuelto esta tesis, desarrolló dentro del proyecto **Urbano** (DPI2001-3652C0201) un robot guía para ferias y museos. Esta plataforma sirve como base y punto de partida para incorporar en ella los comportamientos sociales y habilidades necesarios para interactuar adecuadamente con humanos en el desarrollo de aquellas misiones que le sean asignadas. Al proyecto Urbano le siguieron los proyectos **Robint** (DPI-2004-07907-C02) y **Robonauta** (DPI2007-66846-C02-01), dentro de los cuales se ha enmarcado el desarrollo de la presente tesis doctoral. Se trata, por tanto, de un campo de aplicación innovador, de gran actualidad, en el cual los avances científicos y tecnológicos alcanzados encontrarán aplicación inmediata en beneficio de la sociedad.

Robots similares a **Urbano**, que combinen capacidades de navegación con un cierto modelado del comportamiento y capacidades de interacción con personas, existen pocos en la actualidad. El robot **Rhino**, desarrollado por el Computer Science Department III de la Universidad de Bonn fue concebido como un robot móvil con capacidades de adaptación y aprendizaje [37,190], y puede ofrecer recorridos guiados a los visitantes del *"Deutsches Museum Bonn"*[15] (Museo Alemán de Bonn, principal museo tecnológico de Alemania). **Minerva** funciona también como robot guía en el Museo Nacional de Historia Americana de la Institución Smithsonian[16], en Washington, y fue desarrollado por el mismo grupo creador de RHINO en colaboración con el Robot Learning Laboratory de la Universidad Carnegie Mellon.

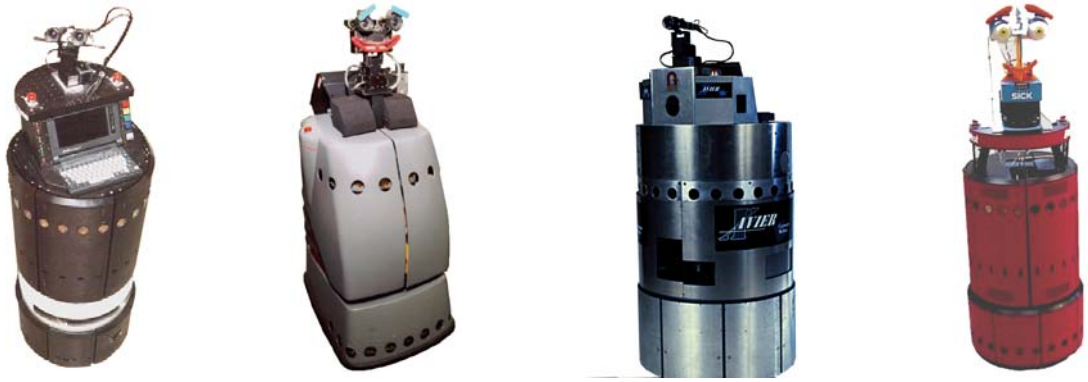
Otros ejemplos de robot interactivo son **Xavier** [110,174], construido como plataforma de investigación en la Universidad Carnegie Mellon, y para que esta participase en la AAAI Robotics Competition de 1993, o **Tourbot** (Interactive Museum Tele-Presence Through Robotic Avatars), proyecto lanzado en Enero de 2000, cuyo objetivo era el desarrollo de un robot guía para museos, haciendo énfasis en facilitar acceso individual remoto a través de Internet.

1.1.3. El problema de la navegación

Vemos por tanto que los robots en general, los robots móviles en particular y, más concretamente los robots interactivos, parecen decididos a quedarse en nuestras vidas [80]. Pero además se les exige cada día un mayor grado de autonomía —y no solamente desde el punto de vista de su consumo energético—, y mayor inteligencia a la hora de tomar decisiones. Un robot móvil, para ser verdaderamente autónomo (independiente de la intervención humana en su funcionamiento), deberá ser capaz de responder a las siguientes tres preguntas que definen el problema básico y primario de su propia navegación [115]:

1. *¿Dónde estoy?*

[15] *http://www.deutsches-museum.de/bonn*
[16] *http://americanhistory.si.edu*



(a) Rhino (b) Minerva (c) Xavier (d) Urbano

Figura 1.7: Algunos robots interactivos.

2. *¿A dónde voy?*

3. *¿Cómo llego a ese destino?*

La respuesta a estas preguntas resulta para nosotros los humanos, como *animalia*[17] que somos, trivial en la mayoría de los casos. Así, para solucionar el problema de nuestra propia navegación disponemos de nuestra percepción y conocimiento del entorno, y de la capacidad de decisión que nos otorga nuestro libre albedrío para responder a nuestras necesidades, apetitos y emociones, y dependemos de los condicionamientos externos sociales, culturales y geográficos adaptados al modo de razonar del ser humano. Sin embargo, para máquinas tan tecnológicamente avanzadas como son los modernos robots del sigo XXI, que combinan cientos de años de conocimiento acumulado a través de generaciones, la respuesta se vuelve aún un problema de considerable envergadura.

A la resolución de la primera y la tercera de las anteriores cuestiones se ha dedicado ya mucho esfuerzo por parte de la comunidad científica. La primera pregunta alude al problema de la localización del robot. Su respuesta determinará en gran medida la que obtendrán las dos siguientes; la posición actual dará opción a visitar un abanico de posibles destinos, mientras que la manera de llegar a ellos también vendrá condicionada en parte por la situación de partida. Para obtener una respuesta, el robot deberá emplear la información capturada por sus sensores y el conocimiento del que disponga acerca de su entorno.

La tercera pregunta intenta ser resuelta por los algoritmos existentes dedicados a planificación de trayectorias adaptados a las condiciones particulares y funcionamiento de un robot móvil. En la solución de este problema también intervienen las diversas técnicas de control reactivo de bajo nivel y, una vez más, toma gran importancia la información disponible en cada instante acerca del entorno y la manera en que el robot es capaz de percibirlo y razonar sobre el mismo.

Finalmente, la segunda de las preguntas permanece como un problema abierto, o cuya resolución viene impuesta por la acción humana exterior que define el destino y

[17]Seres vivientes o animados.

objetivo de la tarea a ser realizada por el robot. La elección autónoma de un destino por parte de la máquina encuentra aún pocas aplicaciones prácticas, quedando relegada mayoritariamente a casos de exploración de entornos desconocidos [33,130,168], y realizándose aún a un nivel muy rudimentario.

1.1.4. El modelado del entorno

Resulta evidente que cualquier ser dotado de capacidad de movimiento, sea humano o robot, necesita disponer de algún tipo de modelo de su entorno para resolver el problema básico de su navegación; ya sea para determinar su propia posición, definir un punto de destino, o planificar la trayectoria a seguir.

Este objetivo ha sido perseguido a lo largo de las últimas décadas. Si bien en la década de 1980, las primeras soluciones propuestas mantenían desacoplados el problema de la construcción de un mapa, y el de la localización del robot, pronto se descubrió que una solución rigurosa del problema no sería posible sin considerar ambos aspectos de manera simultánea. Ello se debe al hecho de que el robot, en su exploración, realiza medidas del entorno que sitúa en el espacio teniendo en cuenta su propia posición, al mismo tiempo que utiliza los objetos previamente detectados para determinar su localización.

Puesto que los sensores no son perfectos —siempre proporcionan medidas contaminadas, en mayor o menor medida, por un cierto nivel de ruido—, y los modelos matemáticos empleados a menudo no son sino aproximaciones de la realidad —introduciendo simplificaciones que permiten la aplicabilidad de los algoritmos—, se entiende que el proceso de construcción del mapa deba considerar, como parte fundamental e integrante de su estructura, todas estas fuentes de incertidumbre.

En el artículo seminal de Smith, Self y Cheeseman [178] se sentaron las bases para resolver ambos problemas simultáneamente. Otros se apoyaron en esas ideas, y desde 1995 el problema de la construcción de un mapa y localización simultánea del robot se ha venido conociendo como SLAM (*Simultaneous Localizacion and Mapping*) [65]. Las soluciones que mejor han sabido lidiar con las particularidades de este problema son sin duda aquellas que emplean técnicas probabilísticas y, de entre estas, una de las más populares es la que utiliza un Filtro Extendido de Kalman (*Extended Kalman Filter*, EKF) para fusionar toda la información disponible. A este tipo de soluciones nos referiremos en lo sucesivo bajo la denominación común de SLAM-EKF.

También se han explorado diferentes estrategias a la hora de describir el mundo físico. Algunos métodos dependen de la adaptación previa del entorno. En este caso se distribuyen previamente una serie de hitos o balizas fácilmente identificables; por su color (si el sensor es una cámara), por su reflectividad (si se trata de un telémetro láser)... Puesto que las posiciones de estos elementos suelen ser conocidas, estos métodos se emplean a menudo en tareas de localización y navegación [23,157], siendo especialmente adecuados en entornos industriales [105], y no es preciso considerar el problema de la construcción del mapa como tal.

La necesidad de adaptar el entorno va en detrimento del objetivo perseguido de lograr mayor autonomía para los modernos agentes móviles artificiales. Por esta

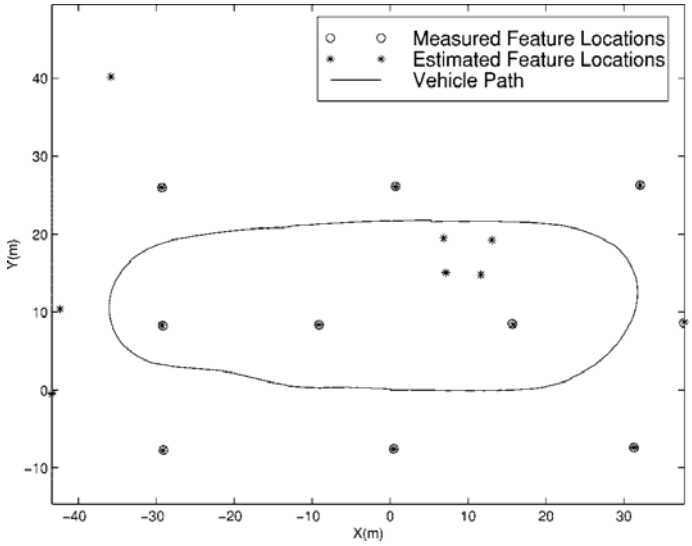


Figura 1.8: Mapa de balizas o hitos puntuales [61]

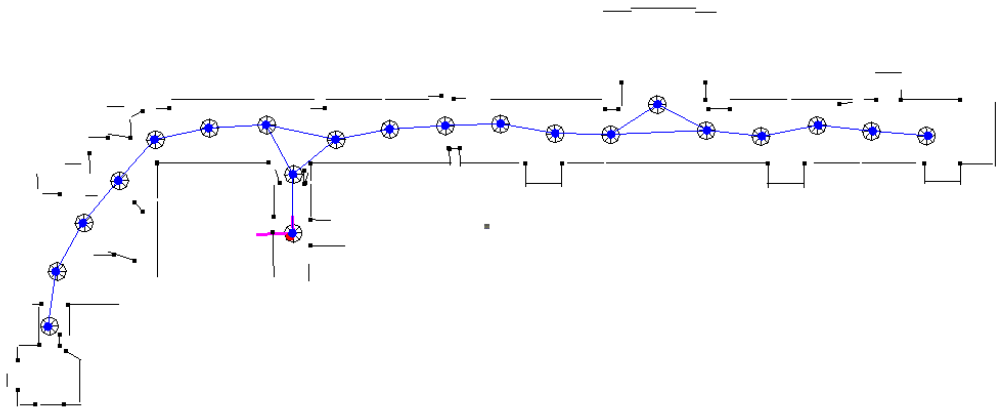


Figura 1.9: Mapa de segmentos [155]

razón se buscan métodos que intenten describir y modelar el entorno tal cual se presenta, e intentar obtener del mismo la información necesaria para su representación. Mientras que unas soluciones intentan extraer características geométricas y representar sus posiciones en un mapa, otras prefieren discretizar el espacio en retículas cuadrangulares, y clasificar cada celdilla individual como vacía u ocupada. La primera opción parece razonable cuando la estructura del entorno y los sensores empleados permiten extraer con comodidad elementos cuya posición es fácilmente descriptible. Así, no resulta excesivamente complicado detectar troncos de árboles o columnas, y representarlos como si de balizas puntuales se tratara [61, 208], o aproximar las paredes de corredores y habitaciones mediante segmentos rectilíneos [155, 187].

Estas últimas soluciones, que atacan el problema del modelado desde un punto de vista geométrico, permiten obtener mapas más próximos a la visión que el ser humano tiene del mundo que le rodea. Esto es beneficioso, por una parte, desde el punto de vista de la interpretación humana del mapa. Por otro lado, también desde la perspectiva del razonamiento artificial y de la extracción de información por parte de una máquina, la disponibilidad de mapas geométricos parece más atractiva.

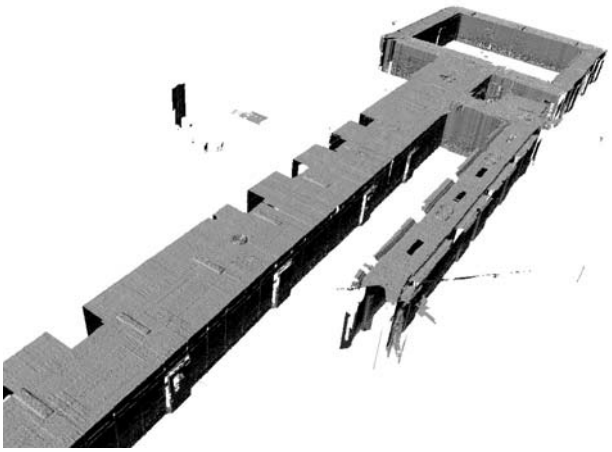


Figura 1.10: Modelo obtenido por alineación de las medidas obtenidas con un escáner láser [95].

Existe una última categoría de métodos, capaces de representar el entorno sin depender de ninguna suposición acerca de las características geométricas del mismo. Cuando se dispone de un sensor láser capaz de tomar medidas precisas del entorno, los métodos de *scan-matching* buscan la manera de alinear entre sí dos medidas consecutivas, de manera que la discrepancia obtenida entre ambas sirva para corregir la posición del robot [121,129]. Posteriormente estas medidas brutas son almacenadas en el mapa, de manera que se consiguen representaciones de muy alta resolución.

Sin embargo, estos modelos que a la vista del ojo humano adquieren pleno significado, para una máquina aún requieren de un procesamiento posterior que permita obtener características más compactas como planos [95]. Así, para que los modelos obtenidos permitan su máximo aprovechamiento y utilidad práctica en labores de navegación y conocimiento del entorno, siempre terminamos llegando al modelado geométrico como manera más sencilla de representar el mundo en términos matemáticos, y de un modo compacto.

En esta tesis se pretende abordar el problema de la representación geométrica de entornos de interior, en los cuales los modelos geométricos actuales, basados en primitivas sencillas como puntos o segmentos, pierden su validez.

1.2. Tema e Importancia de la Tesis

Como ya se ha comentado, esta tesis se enmarca dentro de los proyectos **Robint** y **Robonauta**, cuyos objetivos a grandes rasgos persiguen dotar de mayor autonomía e inteligencia a un robot autónomo, en su función principal de servir de guía en ferias, museos y exposiciones. Uno de los participantes en el proyecto Robint era la Ciudad de las Artes y las Ciencias de Valencia [18], un gran complejo de cultural y lúdico diseñado por los arquitectos Santiago Calatrava y Félix Candela. Fue inaugurado en 1998 y está formado por cinco espacios diferenciados:

[18] *http://www.cac.es*

- L'Hemisfèric
- Museo de las Ciencias Príncipe Felipe
- L'Umbracle
- L'Oceanogràfic
- Palau de les Arts Reina Sofía

El Museo de las Ciencias Príncipe Felipe, inaugurado el 13 de noviembre de 2000, es un edificio de enormes proporciones que le permiten albergar distintos tipos de actividades simultáneamente. Se trata de un museo predominantemente interactivo, de estilo abierto y participativo. Obra del arquitecto valenciano Santiago Calatrava, es único en el mundo por la geometría del edificio, su estructura, y los materiales que lo conforman. Algunos datos técnicos sobre el museo:

- 42.000 m^2 construidos, de los cuales 26.000 m^2 son expositivos, distribuidos en tres plantas. Es el más grande de España en superficie total.
- En unos 7.000 m^2 alberga la mayor exposición acerca del genoma humano.
- Mide 220 metros de largo, 80 metros de ancho y 55 metros de altura.

Lo primero que llama la atención al observar el interior del museo, es la práctica ausencia de paredes planas (Fig. 1.11). Las distintas zonas se encuentran distribuidas a lo largo de todo el recinto, separadas mediante paneles curvilíneos de longitud variable. Además se da la circunstancia de que estas formas caprichosas no pueden ser aproximadas mediante primitivas geométricas sencillas. Aún en el caso de que lo fueran, sería necesario concatenar diferentes tramos de tales primitivas, con la consiguiente necesidad de determinar con precisión sus puntos de unión.

Este no es el único ejemplo de edificios integrados por geometrías curvas que podemos encontrar en el mundo. El diseño de nuevos museos, estaciones de ferrocarril, edificios de oficinas, universidades... pone a prueba la imaginación de los arquitectos, que cada vez cuentan con medios más sofisticados para hacer realidad sus ideas. En la figura 1.12 se muestran algunos ejemplos.

Ninguno de los métodos de modelado geométrico existentes son capaces de enfrentarse con este tipo de geometrías. En esta tesis se muestra cómo es posible construir mapas de estas características, utilizando un punto de vista geométrico en el proceso de construcción del mapa. Para ello se explotará la potencia y versatilidad de las curvas spline, de amplia aplicación en el mundo de diseño gráfico asistido por computador, y en aquellas situaciones donde es necesario aproximar conjuntos de datos contaminados por ruido.

1.3. Objetivos de la Tesis

En este apartado se describen los propósitos de la tesis, basados en las motivaciones y necesidades anteriormente expuestas, indicando la originalidad y aportaciones

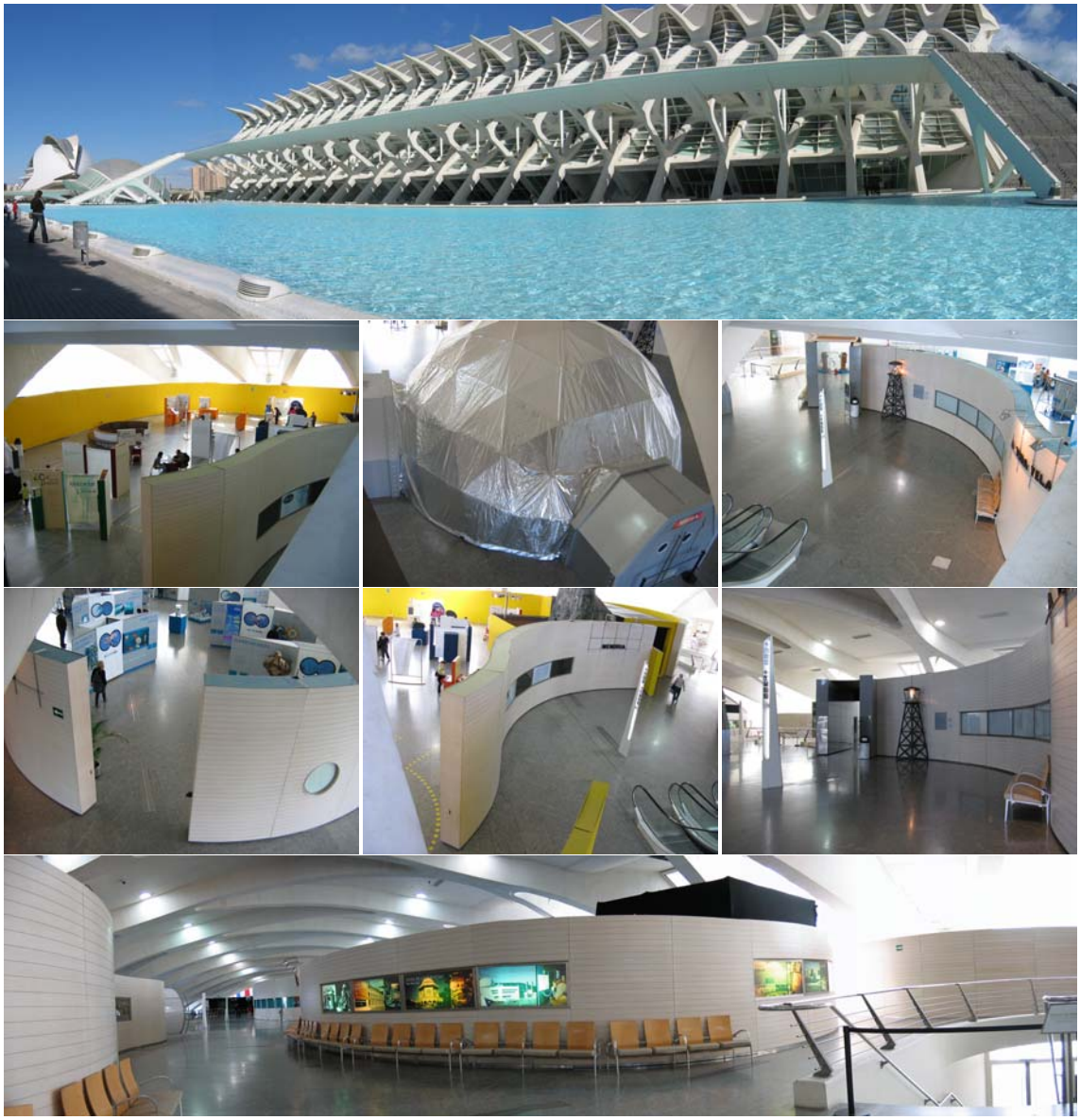
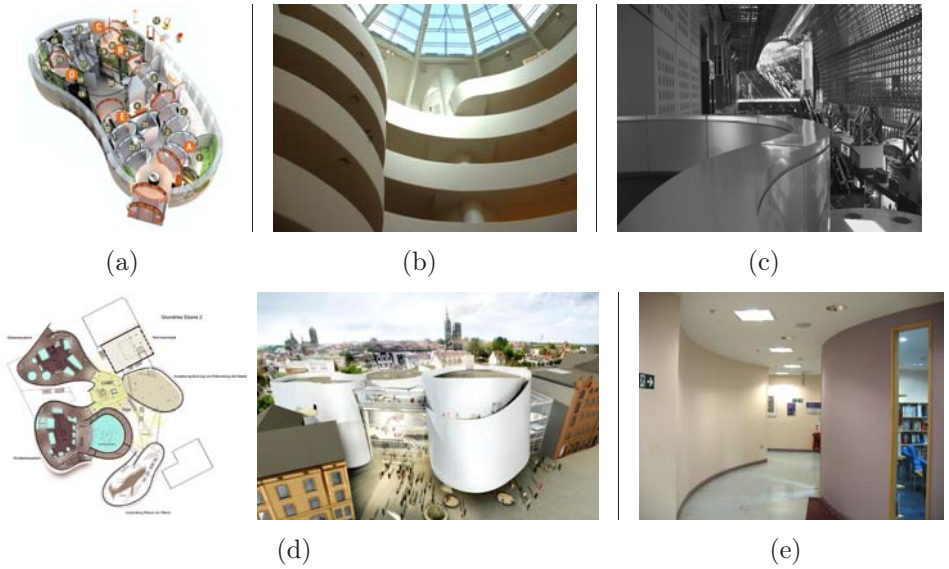


Figura 1.11: Museo de las Ciencias "Príncipe Felipe" (Valencia).



(a) (b) (c)

(d) (e)

Figura 1.12: Algunos ejemplos de edificios con paredes curvas. (a) Exposición *CVIDA*[19]. (b) Museo Guggenheim de Nueva York[20]. (c) Estación de Kyoto. (d) Museo Oceanográfico Alemán (Stralsund). (e) Interior del South Devon College[21].

innovadoras que se derivan de la misma.

El objetivo fundamental de esta tesis es **extender los límites actuales del SLAM geométrico, desarrollando herramientas que permitan la construcción de mapas de geometrías de creciente complejidad**. Como ya se ha comentado, una alternativa para la construcción de mapas consiste en extraer características fácilmente identificables del entorno, tales como balizas colocadas previamente, esquinas, paredes planas o segmentos rectilíneos de los elementos decorativos, funcionales, o arquitectónicos, y establecer una parametrización que permita su representación matemática en un mapa. Esta forma de representación geométrica es de especialidad utilidad a la hora de establecer relaciones entre los elementos que conforman el entorno del robot.

El objetivo último y fundamental de la presente tesis persigue mejorar sustancialmente los actuales métodos de localización y modelado simultáneos allá donde se encuentran las **fronteras actuales de la teoría y de la técnica**:

1. **Mejora de los métodos de representación actuales mediante la adecuada utilización de curvas spline para representar las características geométricas de los elementos que conforman el entorno del robot**. Así, se pretende ampliar el ámbito de aplicación de técnicas existentes de SLAM geométrico en los siguientes términos:

   - Los elementos geométricos del entorno no necesariamente han de ser paredes planas u otras primitivas geométricas sencillas. Se pretende explotar así la versatilidad y potencia de los splines a la hora de representar formas arbitrariamente complejas.
   - Los elementos del entorno no necesariamente han de seguir un patrón geométrico determinado a ser identificado. Mediante la adecuada seg-

mentación y preprocesado de las medidas suministradas por el sensor se identificarán los diferentes elementos susceptibles de ser aproximados mediante curvas spline.

- Será necesario desarrollar las ecuaciones necesarias para conseguir que la teoría e implementación de las curvas spline encaje en un marco de trabajo como el del SLAM-EKF. En concreto, será imprescindible (i) definir metodologías para segmentación y extracción de características de las medidas suministradas por un sensor láser, (ii) definir algún método de aproximación o ajuste de los datos obtenidos mediante curvas spline, (iii) establecer la morfología de los Jacobianos involucrados en el filtro del algoritmo de SLAM, (iv) sintetizar las ecuaciones del modelo de observación para un robot que se desplaza en un mapa modelado mediante splines.

2. **Mejora de los métodos de asociación de datos actuales explotando la ganancia de información que representa la descripción paramétrica de geometrías arbitrarias**. Parece razonable que, una vez disponible un mapa modelado mediante entidades matemáticas en forma de curvas paramétricas, sea posible utilizar información como pendientes, curvaturas, dimensiones, orientaciones relativas, puntos de inflexión... con objeto de mejorar y robustecer metodologías de asociación de datos existentes, o desarrollar otras nuevas. Se pretende así enriquecer la información disponible en el mapa construido, asunto que no ha sido tratado con la adecuada atención en la literatura existente.

3. **Estudio de alternativas de construcción de mapas de grandes dimensiones, de manera que el coste computacional del algoritmo permita su utilización en tiempo real**. Para ello se consideran especialmente adecuadas las técnicas basadas en submapas, de creciente popularidad en la literatura y en la práctica. La ventaja fundamental de estos métodos es la posibilidad de emplear cualquier algoritmo de SLAM disponible en la construcción de los mapas locales, mientras que el ajuste relativo de los diferentes submapas se realiza mediante técnicas de optimización bien conocidas y estudiadas. A esto se añadiría la potencial ventaja que supondría la utilización de curvas spline en la asociación de datos, que permitiría una más robusta detección de cierres de trayectoria comparando entre sí los diferentes mapas locales construidos.

1.4. Estructura de la Tesis

La presente tesis consta de 6 capítulos, un apéndice y bibliografía. Con este primer capítulo se ha querido transmitir al lector la motivación que ha llevado a emprender esta tesis, y mostrar el marco de trabajo en el que se encuadra su desarrollo. Desde la búsqueda histórica por parte del hombre de formas de vida artificiales que le entretuvieran primero, para más tarde servirle de eficaz ayuda, hasta los actuales retos que pretenden dotar de mayor autonomía a los modernos robots móviles, haciéndoles útiles en tareas cada vez más complejas, se ha visto cómo la obtención de

detallados modelos de los entornos es un objetivo ineludible de la robótica actual.

Los dos siguientes capítulos tienen un carácter introductorio, y con ellos se pretende guiar al lector en la problemática que aborda la tesis, así como mostrar los fundamentos que más tarde serán de utilidad para obtener las soluciones que con ella se proponen. Los tres últimos capítulos constituyen el núcleo de la tesis, recogiendo las principales contribuciones y resultados experimentales.

- En el capítulo 2 se realiza un repaso del actual estado del arte en el campo de la localización y modelado simultáneos. Se muestran las principales estrategias empleadas para modelar entornos explorados mediante robots móviles, de entre las que destacan aquellas basadas en técnicas probabilísticas. Se consigue de este modo dar cabida y modelar en la medida de lo posible, no sólo el propio entorno, sino también las incertidumbres y errores presentes en cualquier medida adquirida por los sensores disponibles a día de hoy. Se hace especial hincapié en las tres fronteras detectadas en el desarrollo de la técnica: el coste computacional de los algoritmos empleados, el problema del modelado eficiente y detallado del entorno, y los problemas que plantea la asociación de datos, todos ellos abordados en esta tesis.

- En el capítulo 3 se presenta una introducción a la herramienta de modelado escogida en este trabajo: las curvas spline y, más concretamente, su representación en forma de combinación lineal de un tipo especial de funciones básicas. Se mostrará el origen histórico de esta herramienta, comenzando en la industria naviera para terminar en los complejos algoritmos matemáticos de los modernos programas de CAD/CAM. Se justificará que esta herramienta, que actualmente sirve para modelar complejas estructuras arquitectónicas y escultóricas, sea empleada en el proceso inverso de modelar la realidad física que nos rodea. Aquí se introducen los fundamentes matemáticos de las curvas B-spline, sus principales propiedades, y la manera de aproximar geometrías detectadas por sensores tan populares en la robótica móvil como son los telémetros láser.

- El capítulo 4 constituye el capítulo central de esta tesis. En él se muestra la manera de procesar la información proporcionada por los sensores de a bordo del robot para:

  - Extraer información sobre el entorno circundante en forma de objetos independientes.
  - Modelar cada uno de estos objetos mediante una curva paramétrica.
  - Construir incrementalmente mapas de entornos complejos empleando estas curvas como herramienta de modelado.

  Aquí se recogen los principales algoritmos matemáticos que permiten fusionar la solución del modelado de entornos basada en la utilización de un filtro extendido de Kalman, con la teoría y complejidad, pero también flexibilidad y potencia, que significa el uso de curvas spline el proceso de construcción del mapa.

- El capítulo 5 presenta una solución para reducir el coste computacional de la construcción de mapas de entornos de grandes dimensiones. Para ello se emplea una técnica popular, que reduce el problema a la construcción de unidades de tamaño reducido, que se combinan para obtener la representación global del entorno. Sin embargo, aquí se muestra uno de los beneficios de la representación paramétrica de geometrías complejas: la interpretación geométrica de las estructuras contenidas en os mapas. Así, las curvas spline se revelan no solo como herramienta eficaz y eficiente de modelados, sino como una representación de la que es posible extraer información sin más que analizar la forma de los objetos representados.

- El capítulo 6 recoge los resultados experimentales que demuestran, sustentan y muestran la aplicación práctica de todos los procedimientos y algoritmos presentados en los capítulos anteriores. Los resultados incluyen tanto experimentos con datos simulados, como experimentos con datos extraídos de entornos reales, con plataformas disponibles en el Grupo de Control Inteligente de la UPM, o por otros grupos investigadores.

- Finalmente, en el capítulo 7 se resumen las principales aportaciones de esta tesis, y las conclusiones que se pueden extraer de ellas, así como se indican algunas líneas de futuros desarrollos.

El anexo final de la tesis recoge algunos desarrollos que son útiles para tender algunos aspectos de la tesis, como el método de Newton-Raphson o las ecuaciones involucradas en el marco de estimación que proporciona el filtro extendido de Kalman, junto con algún ejemplo de aplicación.

Capítulo 2

SLAM-EKF: Estado del Arte

2.1. Introducción

El problema de la Localización y Modelado Simultáneos (*Simultaneous Localisation and Mapping*, SLAM), investiga los problemas que plantea la construcción de modelos matemáticos, geométricos o lógicos de entornos físicos, empleando como herramienta un robot móvil —en ocasiones varios de ellos— y el conjunto de sensores y actuadores que lo conforman. Dicho de otra manera, el SLAM busca resolver los problemas que plantea el colocar un robot móvil en un entorno y posición desconocidos, y que él mismo sea capaz de construir incrementalmente un mapa consistente del entorno al tiempo que utiliza dicho mapa para determinar su propia localización. La solución de dicho problema, objetivo insoslayable si se desea disponer de robots móviles verdaderamente autónomos, ha sido objeto de estudio por parte de la comunidad científica durante los últimos 20 años.

El ruido presente en los sistemas sensoriales, los inevitables errores y aproximaciones cometidos en los modelos empleados, y la dificultad representativa de los entornos a medida que éstos aumentan en complejidad, hacen que la tarea de resolver el mencionado problema sea ardua. Tal complejidad tiene una doble vertiente:

- Desde un punto de vista **conceptual** se impone la necesidad de razonar en un mundo a veces confuso, en ocasiones dinámico y cambiante, aprehendido mediante sensores que distan mucho de ser perfectos. En estas condiciones se busca la manera de obtener y manipular datos acerca del entorno, extraer aquel conocimiento que sea sustancial para la tarea de su representación, e integrar la información así obtenida del modo más conveniente. Muchas preguntas surgen en este nivel de complejidad: *¿Qué es el entorno? ¿Qué geometrías cabe esperar encontrar en él? ¿Existen objetos móviles? ¿Con qué precisión es necesario representarlo? ¿Qué nivel de conocimiento permitirá obtener el mapa generado?...*
- La otra vertiente de la complejidad del problema tiene que ver con el aspecto **computacional** de las soluciones planteadas al problema del SLAM, y está indisolublemente ligada a la anterior. El modo en que el robot perciba su en-

> torno, la cantidad de información disponible así como las técnicas empleadas en su procesamiento, interpretación y combinación, determinarán los recursos computacionales necesarios para la construcción del mapa. Estos recursos no son ilimitados; menos aún si el objetivo es ceñirse a los disponibles a bordo de la máquina. Aquí los interrogantes tienen que ver con la idoneidad de los algoritmos utilizados y la posibilidad de obtener soluciones cuya implementación sea posible en tiempo real.

Así pues, en la base de cualquier solución al problema del SLAM nos encontramos siempre con la necesidad de trabajar con cantidades progresivamente crecientes de información —contaminada en mayor o menor medida por ruido—, y manipulada mediante modelos que, la mayoría de las veces, no son sino meras aproximaciones a la realidad. No es de extrañar, por lo tanto, que las soluciones más exitosas hasta el momento hayan estado basadas en la utilización de técnicas probabilísticas. Dado que es imposible obtener mediciones del entorno y del propio estado del robot con precisión arbitrariamente pequeña y certidumbre ilimitada, ¿por qué no tratar de emplear aquellas técnicas y algoritmos mejor preparados para lidiar con la incertidumbre? El propio robot y su estado no son ya sino un conjunto de variables estocásticas cuyo valor se pretende estimar con la mayor precisión posible.

Los trabajos de Smith y Cheeseman [181] y Durrant-Whyte [64] establecieron las bases estadísticas para describir relaciones espaciales entre los diferentes elementos del entorno y manipular las incertidumbres geométricas asociadas a sus respectivas posiciones. Poco después, el artículo de Smith, Self y Cheeseman [179] mostró cómo a medida que un robot móvil se desplaza por un entorno desconocido, realizando observaciones de objetos relativas a su propio sistema de referencia, las estimaciones de las posiciones de dichos objetos quedan necesariamente correladas. Esto se debe a la parte del error común a todas las medidas que tiene su origen en la propia incertidumbre de la estimación de la posición del robot.

Las implicaciones de este hecho son claras; cualquier solución completa y consistente al problema de localización y modelado simultáneos requiere de un vector de estado compuesto tanto por la pose[1] del vehículo como por cada uno de los elementos presentes en el mapa. La necesidad de actualizar tal vector de estado (del orden del número de elementos contenidos en el mapa) a medida que se realiza nuevas observaciones, hace que el coste computacional de los algoritmos empleados crezca cuadráticamente con el tamaño del mapa.

A pesar de que inicialmente la mayoría de la investigación se centró en aligerar la carga computacional de los algoritmos, pronto se demostró que el problema de localización y modelado, formulado como un único problema de estimación, era convergente [61]. Esto es, la estimación del mapa, lejos de presentar un comportamiento errático en el estadio permanente, tendía a converger asintóticamente. Una vez presentada la estructura del problema, sus propiedades de convergencia y acuñarse el acrónimo "SLAM", una pléyade de grupos investigadores de todo el mundo centró

[1] En robótica móvil, se habla generalmente de "pose" para referirse a la posición y orientación del robot conjuntamente, valores que determinan completamente la situación de la máquina en el espacio de 2 ó 3 dimensiones.

sus esfuerzos en participar en una carrera por alcanzar lo que es considerado *el Santo Grial de la robótica móvil* [41, 42, 116, 209].

Los trabajos más recientes se centran en resolver los principales caballos de batalla con los que aún se encuentra los algoritmos existentes: el coste computacional, la representación de entornos de complejidad creciente, y el modo de conseguir una correcta asociación de datos. En las siguientes secciones se enumeran algunas de las soluciones propuestas hasta la actualidad en cada uno de los tres frentes mencionados.

2.2. SLAM con Técnicas Probabilísticas

Las soluciones que mejores resultados han obtenido a la hora de abordar el problema del SLAM son aquellas basadas en técnicas probabilísticas. Consiguen así hacer frente a todas las fuentes de incertidumbre involucradas en el proceso, ya comentadas anteriormente. Este tipo de algoritmos tienen su base en el teorema de Bayes, que relaciona entre sí las probabilidades marginal y condicional de dos variables aleatorias.

Veremos primero una introducción a la formulación del SLAM basada en el teorema de Bayes, para más tarde revisar brevemente algunas de las soluciones más populares que la utilizan.

2.2.1. La formulación Bayesiana del SLAM

El principal obstáculo a la hora de modelar un entorno desconocido, utilizando un robot móvil para su exploración es, como sabemos, la inevitable incertidumbre que se deriva de la imperfección de los sensores y modelos empleados. Si los sensores fueran perfectos, proporcionarían medidas absolutamente precisas de los objetos detectados, que podrían ser insertados en un mapa en su posición exacta respecto a un sistema de referencia ligado al robot. Del mismo modo, al desplazarse la máquina, una medida exacta del giro de sus ruedas —o del avance de sus patas—, combinada con un modelo exacto de su movimiento, nos permitirían determinar exactamente la posición del robot cuando este realizase una nueva medida. Podríamos así construir incrementalmente un modelo perfecto del entorno.

Desafortunadamente tanta perfección no existe en este mundo; o al menos no en el mundo que nos ofrece nuestro actual desarrollo tecnológico. Se impone por tanto la necesidad de acomodar estas incertidumbres en las soluciones obtenidas. La manera evidente de hacerlo es considerar que tanto la posición del robot como los elementos que modelan su entorno son variables aleatorias. Así, los algoritmos existentes modelan ambos de manera probabilística, y utilizan métodos de inferencia para determinar aquella configuración que es más probable teniendo en cuenta las medidas que se van obteniendo.

El principio básico que subyace en cualquier solución exitosa del SLAM es la regla de Bayes. Para dos variables aleatorias, x y d, esta regla establece de manera

muy compacta lo siguiente [189]:

$$p\left(x|d\right) = \frac{p\left(d|x\right)p\left(x\right)}{p\left(d\right)} \tag{2.1}$$

La anterior ecuación constituye un mecanismo básico de inferencia, cuya simplicidad no resta un ápice a la potencia de su significado. Supongamos que queremos obtener información acerca de la variable x —por ejemplo, el estado de un sistema compuesto por un mapa y un robot— basándonos en la información contenida en otra variable d —que bien podría ser un conjunto de medidas adquiridas por un sensor—. La anterior regla indica que este problema se puede resolver simplemente multiplicando dos términos:

- El modelo generativo $p\left(d|x\right)$, que expresa la probabilidad de obtener la medida d bajo la hipótesis expresada por el estado x, y
- el grado de confianza que damos a que x sea precisamente el caso antes de recibir los datos, $p\left(x\right)$.

Es importante el hecho de que el denominador de la ecuación (2.1) no depende de la variable que pretendemos estimar, x. Por esta razón $p\left(d\right)^{-1}$ se suele escribir como un factor de normalización en la regla de Bayes, generalmente expresado mediante la letra η. Obtenemos así una expresión algo más compacta para la famosa regla:

$$p\left(x|d\right) = \eta p(d|x)p(x) \tag{2.2}$$

El problema del mapeado de un entorno está sujeto a una variable adicional no contemplada por la formulación clásica del teorema de Bayes: el tiempo. Durante la labor exploratoria de un robot móvil, los datos se adquieren secuencialmente en el tiempo. Tanto las medidas propioceptivas, a las que denominaremos con la letra u —las cuales miden el desplazamiento del robot—, como las medidas exteroceptivas obtenidas por los sensores disponibles a bordo de la máquina, z, son adquiridas a lo largo de todo el proceso. Sin pérdida de generalidad podemos asumir que esta información se recibe alternativamente según la secuencia:

$$z_1, u_1, z_2, u_2, \ldots, z_t, u_t, \ldots \tag{2.3}$$

donde los subíndices expresan un instante temporal concreto. La variable u puede representar desde la señal de control enviada a los actuadores de la máquina, hasta las medidas proporcionadas por los encóders acoplados a las ruedas —o articulaciones— del robot o las suministradas por sensores de navegación inercial (IMUs) como acelerómetros o giróscopos. En el primer caso, será el modelo de la máquina el que determine el desplazamiento realizado, mientras que en el segundo y tercer casos

será la propia medida, o su integración a lo largo del tiempo, la que proporcione esta estimación.

La generalización temporal del teorema de Bayes, en la cual se fundamentan todas las soluciones probabilísticas del SLAM, recibe el nombre de *Filtro de Bayes*. Se trata de un estimador recursivo que calcula la secuencia de distribuciones de probabilidad *a posteriori* (tras recibir los datos) de magnitudes que no pueden ser directamente observadas, en función de otras que sí lo son. Así, la formulación genérica del filtro de Bayes calcula la probabilidad del estado en un instante dado, x_t, utilizando la siguiente ecuación recursiva:

$$p\left(x_t|z^t,u^t\right) = \eta p\left(z_t|x_t\right) \int p\left(x_t|u_t,x_{t-1}\right) p\left(x_{t-1}|z^{t-1},u^{t-1}\right) dx_{t-1} \tag{2.4}$$

En este caso se utiliza el superíndice t para indicar el conjunto de todas las medidas acumuladas hasta ese preciso instante:

$$z^t = \{z_1, z_2, \ldots, z_t\} \tag{2.5}$$
$$u^t = \{u_1, u_2, \ldots, u_t\} \tag{2.6}$$

La anterior ecuación (2.4) expresa una relación recursiva, puesto que la probabilidad $p\left(x_t|z^t,u^t\right)$ se calcula en cada momento haciendo uso de la misma probabilidad obtenida un instante temporal anterior, $p\left(x_{t-1}|z^{t-1},u^{t-1}\right)$. De esta manera se evita la necesidad de mantener almacenadas todas las medidas $\{z_i, i = 1,\ldots,t\}$ y $\{u_i, i = 1,\ldots,t\}$ a lo largo del proceso que, según la anterior expresión, se puede mantener indefinidamente en el tiempo.

En el problema de la localización y modelado simultáneos, el estado del sistema x_t está compuesto en cada instante por la posición del robot, a la que denominaremos s_t, y por las posiciones del conjunto de objetos incluidos en el mapa, m_t.

$$x_t \equiv \{s_t, m_t\} \tag{2.7}$$

Por lo tanto, el filtro de Bayes adaptado al problema que nos ocupa se puede escribir como:

$$p\left(s_t, m_t|z^t,u^t\right) = \eta p(z_t|s_t,m_t) \iint p(s_t,m_t|u_t,s_{t-1},m_{t-1}) p\left(s_{t-1},m_{t-1}|z^{t-1},u^{t-1}\right) ds_{t-1} dm_{t-1} \tag{2.8}$$

Asumiendo que el mundo es estático —o, al menos, la parte de él que se pretende modelar— podemos omitir el subíndice temporal al referirnos al mapa, ($m_t \equiv m$).

Por otra parte, si asumimos también que el movimiento del robot es independiente del mapa (algo perfectamente lógico), podemos escribir la siguiente expresión para el filtro de Bayes adaptado al problema del SLAM:

$$p\left(s_t, m|z^t, u^t\right) = \eta p\left(z_t|s_t, m\right) \int p\left(s_t|u_t, s_{t-1}\right) p\left(s_{t-1}, m|z^{t-1}, u^{t-1}\right) ds_{t-1} \quad (2.9)$$

Esta expresión, para servir de utilidad práctica, requiere de dos elementos fundamentales. Se trata de dos modelos generativos que dependen del propio robot, y de la relación perceptual que este establece con su entorno:

- El **modelo de percepción** $p\left(z_t|s_t, m\right)$, que describe en términos probabilísticos cómo se generan las medidas z_t en función de la posición del robot y la configuración del mapa.
- El **modelo de movimiento** $p\left(s_t|u_t, s_{t-1}\right)$, que determina la probabilidad de que el robot llegue a la posición s_t desde otra s_{t-1}, cuando se aplica la acción u_t (o, equivalentemente, cuando se mida un desplazamiento dado por esta variable).

Ahora bien, el mecanismo de inferencia propuesto por la ecuación (2.9), si bien es cierto que se adapta a nuestras necesidades desde el punto de vista lógico, presenta el grave inconveniente de no admitir una solución cerrada, o que sea fácilmente calculable utilizando un computador. Así pues, es preciso realizar simplificaciones o suposiciones que faciliten su uso práctico.

La manera de establecer estas hipótesis adicionales a la hora de implementar el filtro de Bayes es el rasgo que diferencia a los múltiples algoritmos de construcción de mapas, que a continuación se describen brevemente.

2.2.2. Principales algoritmos

El filtro extendido de Kalman

Se trata de una de las soluciones más extendidas a la ecuación (2.9), y también una de las que mejores resultados ha proporcionado en la práctica. A esta categoría de soluciones, cuyo fundamento enraíza en los trabajos introductorios realizados por Randall Smith, Matthew Self y Peter Cheeseman a finales de la década de 1980 [177, 178, 180, 181], se la conoce generalmente como SLAM-EKF.

El punto de partida es la suposición (no siempre próxima a la realidad) de que la probabilidad del estado $p\left(s_t, m|z^t, u^t\right)$ se puede modelar mediante una distribución unimodal gaussiana multidimensional. Así, dicha probabilidad puede ser completamente descrita utilizando únicamente su valor esperado (media de la distribución) y su matriz de varianzas y covarianzas (a la que en lo sucesivo denominaremos, simplemente, matriz de covarianzas):

$$x_t \equiv \{s_t, m\} \equiv \mathbf{x}(t) \tag{2.10}$$
$$\mathbf{x}(t) \sim N(\hat{\mathbf{x}}(t|t), \mathbf{P}(t|t)) \tag{2.11}$$

siendo la media de la distribución y la matriz de covarianzas:

$$\hat{\mathbf{x}}(t|t) = E[\mathbf{x}(t)] \tag{2.12}$$
$$\mathbf{P}(t|t) = E\left[(\mathbf{x}(t) - \hat{\mathbf{x}}(t|t))(\mathbf{x}(t) - \hat{\mathbf{x}}(t|t))^T\right] \tag{2.13}$$

Por su propia naturaleza, el SLAM-EKF requiere disponer de un mapa en el cual las entidades que lo componen sean fácilmente parametrizables. Esto es, los elementos que componen el mapa deben poder ser descritos utilizando un conjunto de parámetros que encajen de forma sencilla en el vector de estado del sistema.

A la vista de las hipótesis realizadas, cabe enumerar las siguientes desventajas de esta clase de algoritmos:

- La premisa de partida del algoritmo, que supone una distribución gaussiana para el estado del sistema, puede no corresponderse con la realidad. En el mejor de los casos, las linealizaciones introducidas en los modelos harán que que las estimaciones de los momentos de esta distribución (eucaciones (2.12) y (2.13)) degeneren a lo largo del tiempo no correspondiéndose con sus auténticos valores.
- El punto anterior indica además un grave problema de consistencia del algoritmo, que hace que el nivel de confianza de la estimación obtenida no se corresponda con el auténtico error cometido. Esto origina que la exactitud de los resultados obtenidos por el filtro sean a menudo impredecibles, observándose saltos bruscos en la estimación sin causa aparente alguna. Este hecho ha sido estudiado en profundidad en la literatura más reciente [12, 44, 158].
- El coste computacional crece al cuadrado con el número de objetos contenidos en el mapa. Este hecho limita su aplicación en tiempo real a mapas formados por unos pocos cientos de objetos.
- No siempre es sencillo o inmediato extraer características del entorno asimilables a una clase particular de objeto. En ocasiones ni siquiera es preciso extraer información que pueda describirse empleando primitivas geométricas simples como puntos, segmentos, arcos de circunferencia o planos (por ejemplo, en el interior de una mina).
- Es preciso disponer de un método de asociación de datos robusto que permita emparejar las observaciones realizadas con los elementos contenidos en el mapa. Una asociación de datos errónea provocará casi con total seguridad la divergencia irrecuperable de la estimación del filtro.

A pesar de los inconvenientes que surgen al emplear esta solución, se trata de la más extendida en la literatura y presenta interesantes propiedades que resultan muy atractivas desde el punto de vista de los objetivos perseguidos en esta tesis:

- El hecho de describir el entorno a partir de entidades geométricas descriptibles de manera compacta, se corresponden más con una visión antropomórfica del mundo, que representa este último a través del concepto del "objeto" y sus relaciones.
- Estas soluciones cuentan con una larga tradición [60, 91, 203], lo cual hace que su estructura, ventajas e inconvenientes sean bien conocidos. Históricamente se han planteado múltiples soluciones que intentan paliar algunos de los problemas anteriormente mencionados, como veremos.
- Al mantenerse la matriz de covarianzas del sistema completa, es capaz de cerrar bucles exitosamente (con las limitaciones que suponen la inconsistencia del algoritmo o el coste computacional de la actualización de la estimación en mapas de grandes dimensiones).

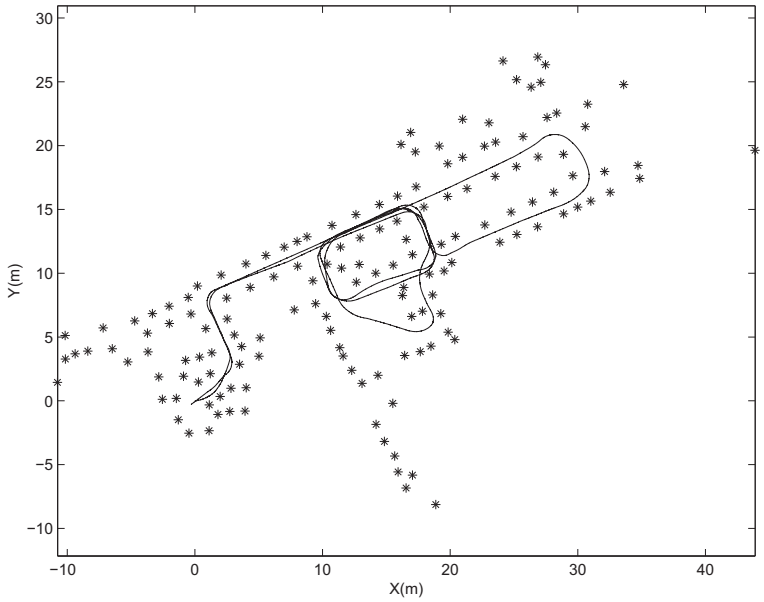


Figura 2.1: Mapa de 148 objetos puntuales obtenido con la solución EKF en [60].

Existen múltiples implementaciones de la solución SLAM-EKF. Tal vez una de las más populares es el marco de trabajo que propone el SPmap [43], que modela el vector de estado teniendo en cuenta las simetrías y empleando vectores de error. Consigue así una mayor robustez e inmunidad ante el problema de la parcialidad de la incertidumbre, que aparece cuando el número de parámetros que es capaz de estimar el filtro, es inferior al de parámetros que definen cada objeto en su totalidad (por ejemplo, a la hora de determinar no sólo la posición y orientación de una pared, sino también sus límites). Una excelente implementación real de este método se puede encontrar en [155].

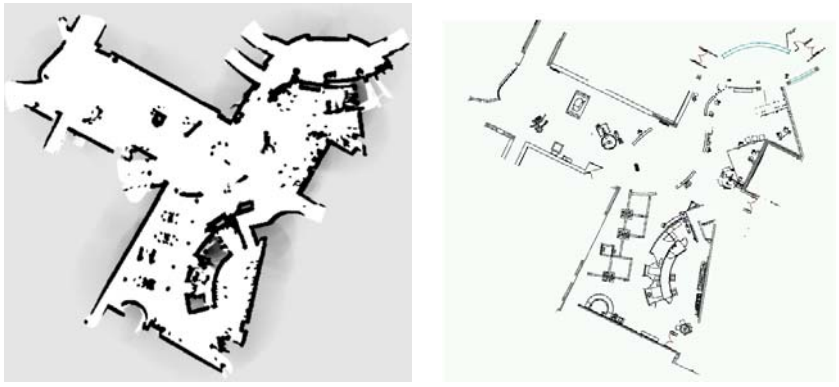
Quizá la mayor desventaja del SLAM geométrico basado en el EKF es su dependencia en la existencia de geometrías concretas en el entorno. Así, cada método existente es capaz de trabajar con objetos puntuales o características modelables mediante segmentos, como es el caso de paredes en entornos de interior [113, 156]. Los hay que incluso intentar representar el entorno utilizando polilíneas [199]. Sin embargo, no existe en la actualidad ningún método de representación geométrica que tenga cabida universal en cualquiera de los métodos de modelado existentes.

Con esta tesis se pretender llenar este vacío, proporcionando métodos de modelado más generales, sin perder la eficacia y eficiencia que supone la representación compacta del entorno mediante características geométricas.

Mapa de ocupación de celdillas

El algoritmo de ocupación de celdillas (*Occupancy Grid Mapping*) fue introducido por los trabajos de Hans Moravec [131] y Alberto Elfes [66] a mediados de la década de 1980. Se basa en discretizar el espacio, dividiéndolo en unidades de tamaño predefinido, que se clasifican como ocupadas o vacías con un determinado nivel de confianza o probabilidad.

Estas soluciones parten de la hipótesis de que la posición del robot es conocida. En la práctica, se necesita de algún método de localización que estime la posición del robot en cada instante que, en este caso, no es considerada una variable estocástica. En la práctica, la precisión que alcanzan estos mapas en la descripción del entorno (tanto mayor cuanto más fina es la división del espacio), permite que el algoritmo de localización empleado acumule errores reducidos a lo largo de intervalos prolongados de tiempo. Así pues, la mayor desventaja de estos métodos es la pérdida de potencia que se deriva de no tener en cuenta la incertidumbre asociada a la posición del robot, lo cual origina que su capacidad para cerrar bucles correctamente se vea mermada.



(a) (b)

Figura 2.2: Mapa de ocupación de celdillas del Museo Tecnológico de San José[2] (a) y dibujo CAD de la planta del mismo (b) [188].

Entre sus ventajas cabe destacar las siguientes:

- El algoritmo es robusto y su implementación sencilla.

- No hace suposiciones acerca de la naturaleza geométrica de los elementos presentes en el entorno.
- Distingue entre zonas ocupadas y vacías, consiguiendo una partición y descripción completa del espacio explorado. Por esto motivo es popular en tareas de navegación, al facilitar la planificación y generación de trayectorias empleando métodos convencionales.
- Permite descripciones arbitrariamente densas o precisas del mundo, simplemente aumentando la resolución de la rejilla que lo divide (*i.e.* disminuyendo el tamaño de las celdillas individuales). Como es lógico, esto va en detrimento del rendimiento computacional del algoritmo.
- Permite una extensión conceptualmente simple al espacio tridimensional.

Recientemente han aparecido variantes de este método que mejoran sustancialmente algunos aspectos. Por ejemplo, los mapas de cobertura (*coverage maps*) [184, 185], que mantienen una probabilidad de ocupación para cada celdilla que se ajusta más correctamente a casos en los que esta se encuentra sólo parcialmente ocupada, o el DP-SLAM [67,68], que mantiene varias hipótesis sobre el mapa construido y utiliza un filtro de partículas para estimar la que más se ajusta a la realidad.

Figura 2.3: Ejemplo de mapa de cobertura en [184].

Máxima probabilidad incremental

Los buenos resultados obtenidos con los mapas de celdillas, así como su necesidad de disponer de una estimación para la posición del robot (aunque esta no fuera modelada probabilísticamente), animaron a los investigadores a explorar técnicas simplificadas en las que algunos de los elementos involucrados en la ecuación (2.9) no son modelados de manera estocástica.

Así, la idea básica de los algoritmos de máxima probabilidad incremental (*Incremental ML*) [96, 189] es la de obviar toda noción de incertidumbre asociada a los elementos del mapa. Simplemente optan por mantener la estimación del mapa m^* y de la posición del robot s_t^* más probables en cada instante, maximizando la siguiente expresión:

$$\{s_T^*, m^*\} = argmax_{s_t,m} p\left(z_t|s_t, m\right) p\left(s_t, m|u_t, s_{t-1}^*, m^*\right) \tag{2.14}$$

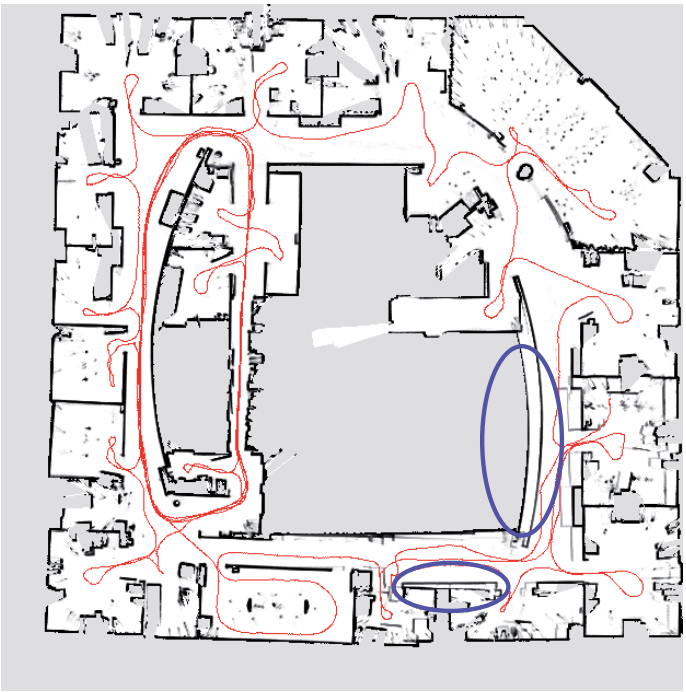


Figura 2.4: Mapa construido mediante un algoritmo de máxima probabilidad incremental en [96].

La anterior maximización se suele limitar al cálculo de la posición del robot más probable a la vista de las medidas sensoriales obtenidas y el modelo no estocástico del mapa disponible. A esta clase de algoritmos pertenecen las técnicas de *scan matching* [59, 121, 129], que se limitan a calcular la posición más probable del robot como resultado de alinear las medidas brutas adquiridas por un sensor del láser, y construir incrementalmente el mapa sobre esta localización.

Su principal ventaja es su sencillez conceptual y su ligereza computacional, pero encuentra serios problemas a la hora de cerrar bucles exitosamente, puesto que no mantiene ninguna información acerca de las correlaciones entre los diferentes objetos. Un ejemplo de este hecho puede observarse en la figura 2.4, donde se han resaltado las zonas donde la inconsistencia de la estimación obtenida resulta evidente.

Solución factorizada del filtro de Bayes

En la tesis de Michael Montemerlo [127] se realiza un detallado análisis de la formulación Bayesiana del problema SLAM, y se llega a una conclusión que resulta muy intuitiva y casi evidente. Si el camino verdadero que recorre el robot fuera perfectamente conocido, la estimación del mapa sería un problema de solución casi inmediata. Además supondría una ventaja computacional, puesto que la estimación de los objetos que componen el mapa sería independiente, ya que no estarían relacionados por la incertidumbre común asociada a la posición del sensor del robot. Por tanto, las correlaciones entre los objetos serían nulas. Es algo parecido a lo que sucede con los mapas de ocupación de celdillas, descritos en la sección 2.2.2, que también suponen conocida la localización del robot en todo momento.

Así, la solución factorizada del SLAM (*Factored Solution to SLAM*), también conocida como FastSLAM, se apoya en la suposición de esta independencia condicional. Se encarga de calcular una probabilidad ligeramente diferente a la expresada en la ecuación (2.9). Si allí se estimaba la distribución de un mapa y la pose del robot s_t, aquí se estima sobre el mapa y el camino completo del robot, s^t. Bajo la hipótesis de que la asociación de datos entre las observaciones adquiridas y los objetos del entorno es conocida, se puede factorizar el filtro de Bayes para este problema del siguiente modo: [128]:

$$p\left(s^t, m|z^t, u^t\right) = p\left(s^t|z^t, u^t\right) \prod_{i=1}^{n} p\left(m_i|s^t, z^t, u^t\right) \tag{2.15}$$

La anterior ecuación indica que, para un mapa en el que existen n objetos cuyas posiciones se desea determinar, y un robot explorador, el problema se puede descomponer en $n+1$ estimaciones independientes: una estimación del camino s^t trazado por el robot, y n problemas de estimación de las posiciones de los n, condicionados a la estimación de la trayectoria.

En la práctica, la estimación de la trayectoria del robot se realiza mediante un filtro de partículas [62, 119] en el que cada partícula mantiene una estimación de los n objetos del mapa mediante n filtros extendidos de Kalman independientes. Entre sus ventajas cabe mencionar la mayor inmunidad a las no linealidades del modelo odométrico, gracias precisamente a la utilización del filtro de partículas. También se comporta mejor ante errores en la asociación de datos, puesto que aquellas partículas erróneas tenderán a desaparecer a lo largo del tiempo (resultado mucho más deseable que la divergencia de un único filtro).

El coste computacional de la solución así formulada es proporcional al número de partículas p y al número n de objetos en el mapa: $O\left(pn\right)$. La capacidad de estas soluciones para trabajar con mapas que contienen un elevado número de objetos, ha animado a algunos autores a combinarla con representaciones basadas en almacenar las medidas brutas del láser, obteniéndose representaciones muy densas y detalladas de los entornos [94] (véase la Fig. 2.5).

En la figura 2.5 se muestra el mismo mapa que aparece en la figura 2.4, pero construido utilizando una solución factorizada al filtro de Bayes planteado en su forma completa. Se aprecia la evidente mejora en la calidad del resultado.

2.3. Fronteras del Estado del Arte

A la vista de las anteriores soluciones, se pueden identificar tres límites o fronteras fundamentales en el estado de la técnica relacionada con el problema del SLAM. Una frontera tiene que ver con las **limitaciones computacionales** de los algoritmos, que no siempre permiten su aplicación en tiempo real. En particular, cuanto mayor es la precisión y calidad de los resultados obtenidos, mayor es la cantidad de

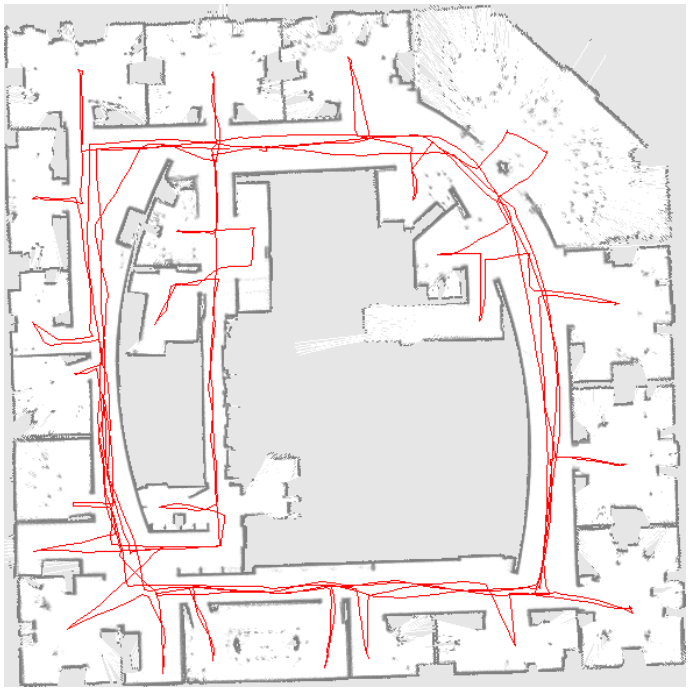


Figura 2.5: Mapa obtenido con FastSLAM y técnicas de *scan matching* en [94]

información que es preciso manejar, con el consiguiente aumento de requerimientos computacionales.

La segunda está relacionada con el problema de la representación del entorno. Si bien la representación geométrica es deseable, puesto que confiere mayor compacidad y sencillez de interpretación a los modelos obtenidos, no siempre los entornos presentan estructura suficiente como para extraer de ellos características simples como puntos o segmentos. Como hemos visto, la totalidad de técnicas geométricas presentadas, limitan su aplicación a entidades simples, cuya detección y representación es sencilla. Entornos con presencia de paredes curvas sólo pueden ser modelados empleando técnicas de *scan matching* o mapas de ocupación de celdillas.

Finalmente existe un problema aún abierto relacionado con la asociación de datos. Es necesario disponer de mecanismos robustos que sean capaces de asociar de manera confiable las observaciones obtenidas con aquellos objetos correspondientes en el entorno. Una asociación de datos incorrecta provocará divergencias irrecuperables en la estimación, a la que sólo algunos algoritmos (como el FastSLAM) presentan cierta inmunidad.

2.3.1. Coste computacional

Las técnicas que afrontan el problema de mejorar el rendimiento computacional de los algoritmos empleados en SLAM pueden ser clasificadas en dos grandes grupos:

- Técnicas **óptimas**: reducen el coste computacional manteniendo resultados equivalentes al algoritmo del que se derivan.

- Técnicas **conservativas**: resultan en estimaciones menos precisas, o con mayor incertidumbre que las del algoritmo del que se derivan.

A pesar de que la solución basada en el filtro de Kalman es óptima para el caso lineal, y que la solución que supone la aplicación EKF es la mejor posible en caso de existir no linealidades, su aplicación es computacionalmente impracticable para entornos de grandes dimensiones. Esto se debe a que la matriz de covarianzas almacena las correlaciones entre cada dos objetos cualesquiera, de manera que el mero hecho de incluir un nuevo objeto repercute en el estado del resto de variables. Sin embargo, la inclusión de un nuevo objeto tiene típicamente un efecto reducido sobre elementos distantes geométricamente, por lo cual no es descabellado atacar el problema global descomponiéndolo en mapas de dimensiones más reducidas.

El Filtro Extendido de Kalman Comprinido (CEKF) [89], por ejemplo, se encarga de retrasar la incorporación de información local en el mapa global, mientras el robot se encuentra explorando una cierta área del entorno. Se trata de una técnica óptima en el sentido de que genera resultados equivalentes a los del EKF completo. Sin embargo resulta computacionalmente mucho más atractiva, puesto que las actualizaciones del mapa completo se realizan poco frecuentemente.

Por la especial popularidad que han adquirido recientemente, se comentan a continuación los métodos basados en submapas, que pretenden mejorar la carga computacional que suponen los algoritmos de SLAM dividiendo el problema global en otros computacionalmente más económicos. Todos ellos producen submapas óptimos de tamaño reducido, que son integrados para conformar un mapa global conservativo.

Submapas Globales

Estos métodos intentan minimizar el impacto que supone el incremento de elementos contenidos en el mapa, que afecta cuadráticamente al coste computacional del algoritmo. La idea fundamental es definir una serie de sistemas de referencia a lo largo de la trayectoria del robot, construyendo sobre cada uno de ellos mapas con las características geométricas del entorno que le son próximas. Los mapas son organizados entre sí siguiendo algún tipo de estructura jerárquica que es reorganizada cuando es posible relacionar geométricamente dos sistemas de referencia locales mediante alguna observación común. Ejemplo de esta estrategia son RLR (*Relative Landmark Representation*) [90] y CTS (*Constant Time SLAM*) [135, 136].

Submapas Relativos

Se diferencian de los anteriores en que no existe un sistema de referencia común a todos los submapas. Se establece un grafo que relaciona a cada submapa con un conjunto de vecinos próximos, y la posición de cada uno de ellos respecto a otro cualquiera se puede calcular componiendo transformaciones relativas [179] a lo largo de la ruta que los conecta.

Ejemplos de esta modalidad son el *Atlas framework* [34,35] y los NCFM (*Network Coupled Feature Maps*) [8]. Otros algoritmos, como el SLAM jerárquico (*Hierarchical* SLAM) [71] son capaces de aumentar la velocidad de convergencia imponiendo restricciones cada vez que se detecta que el vehículo cierra un bucle. Todos estos métodos producen submapas óptimos con coste computacional independiente del tamaño del mapa global, son numéricamente muy estables (al restringir las actualizaciones a áreas reducidas), permiten asociación de datos en lote entre los distintos submapas construidos y minimizan en gran medida los problemas que a lo largo del tiempo surgen como consecuencia de las linealizaciones introducidas [12], siendo esta la ventaja fundamental respecto a los submapas globales.

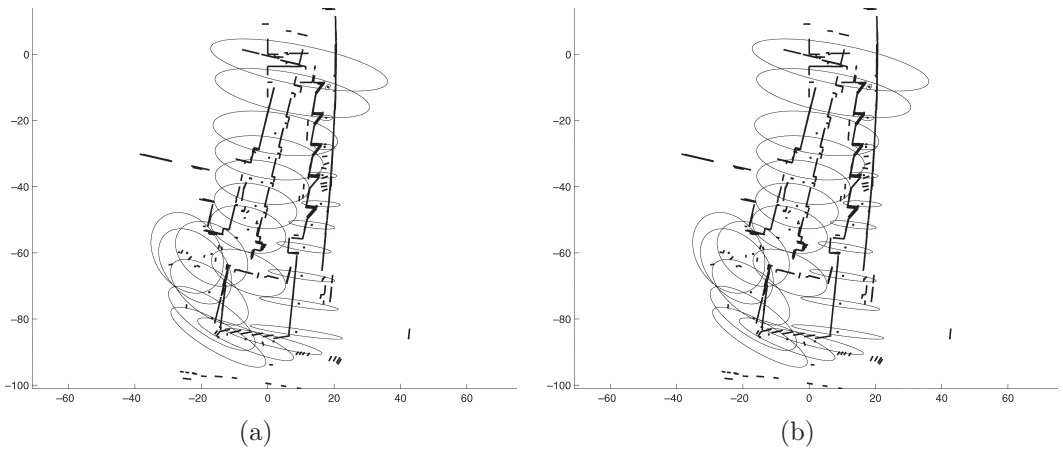


Figura 2.6: SLAM jerárquico en [71]. Configuración antes (a) y después (b) de cerrar el primer bucle.

Más recientemente, se han presentado soluciones que recurren a métodos tradicionales de optimización para reorganizar los submapas contenidos en el grafo con impresionantes resultados tanto desde el punto de vista de la calidad del mapa resultante como del tiempo de cómputo requerido [88, 139].

2.3.2. Representación del entorno

Los primeros trabajos sobre SLAM asumían un mundo que podía ser representado mediante un conjunto de primitivas geométricas sencillas, tales como puntos, líneas o círculos. Poco ha cambiado el panorama la hora de mejorar la representación del entorno si hablamos de mapas geométricos [43, 155, 156]. Sin embargo, en entornos complejos y poco estructurados (tales como entornos de exterior, subterráneos o submarinos) esta suposición parece limitar sustancialmente la aplicación de las técnicas desarrolladas.

Para abordar entornos de creciente complejidad, han aparecido en la literatura diferentes soluciones:

SCAN-SLAM

La mayoría de las implementaciones de EKF-SLAM asumen un mundo modelando mediante hitos puntuales. Recientemente, algunos autores han sugerido la posibilidad asociar un sistema de referencia a un objeto de geometría arbitraria, de manera que metodologías ya existentes pueden ser aplicadas en entornos de características más generales.

El trabajo de Nieto *et al.* [137] propone un acercamiento entre la solución SLAM-EKF, atractiva por sus propiedades de convergencia [61], y las técnicas de *scan matching* [129], capaces de lidiar con las medidas brutas suministradas por los sensores. Como los mismos autores señalan, el hecho de limitar los elementos del mapa a aquéllos que satisfacen determinada propiedad geométrica, tiende a rechazar ingentes cantidades de datos de potencial utilidad.

3-D SLAM

A pesar de que podría parecer tarea sencilla extender los resultados del SLAM bidimensional al escenario en tres dimensiones, la mayor complejidad del modelo de movimiento del vehículo y sobre todo, del modelado del entorno, complican el proceso. Existen tres variantes básicas de SLAM en tres dimensiones:

- SLAM convencional en dos dimensiones, con capacidades de representación añadidas en la tercera dimensión [123, 191]. Esto sólo es posible cuando el movimiento del vehículo está realmente confinado en un plano.

- Extensión directa de algoritmos convencionales a las tres dimensiones, utilizando características fácilmente identificables y modelables del entorno para su representación en el mapa. Ejemplo de ello es el trabajo de Davison *et al.* con visión monocular [51].

- La tercera alternativa consiste en una formulación completamente diferente, en la que el estado que se pretende estimar viene definido por el historial de todas las poses del robot y sus medidas asociadas en cada instante [72, 134]. Esta modalidad de SLAM basado en trayectoria es especialmente útil en entornos donde la extracción de características es inviable. El mapa deja de formar parte del estado, y es reconstruido fusionando los conjuntos individuales de datos obtenidos en cada pose del robot, tal y como se muestra en el ejemplo de la Ilustración 1. La principal desventaja de este método es que tanto la longitud del vector de estado, como la cantidad de datos que es preciso almacenar, crecen ilimitadamente con el tiempo. Así pues, sería deseable disponer de métodos que permitan fusionar la información adquirida para limitar el coste de almacenamiento.

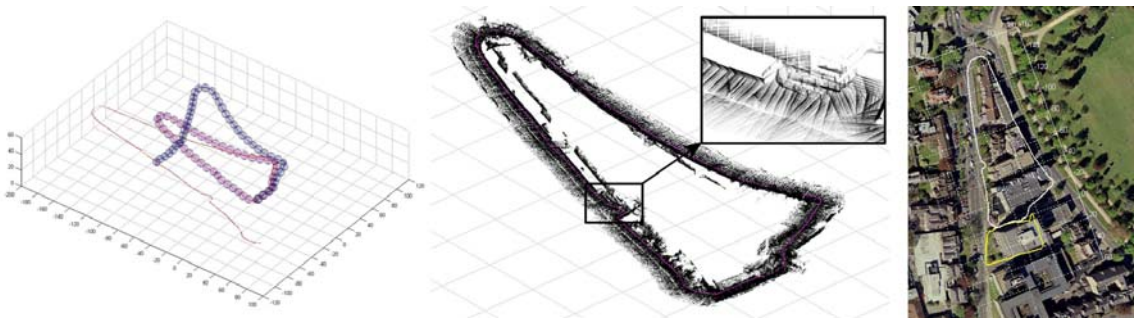


Figura 2.7: SLAM basado en trayectorias [134]. La parte izquierda de la figura muestra el conjunto de poses del robot antes y después de cerrar el bucle. A la derecha se muestra la apariencia final del mapa, donde se han representado todas las medidas almacenadas en cada posición del robot.

Incrustación de Información Adicional

El tradicional marco de trabajo de SLAM no es adecuado para la representación de información espacial de muy alta densidad. Cualquier característica del entorno cuya posición espacial se quiera estimar dentro del vector de estado del sistema ha de ser cuidadosamente seleccionada (el crecimiento de dicho vector afecta de manera muy negativa al desempeño computacional de los algoritmos).

La solución propuesta por Nieto et al. [138] resulta especialmente atractiva para afrontar este reto. Discretizando el espacio en regiones triangulares definidas por un subconjunto de elementos del mapa, es posible almacenar, representar y actualizar cualquier tipo de información adicional que queda así embebida en el mapa. Un ejemplo de estos Mapas Métricos Híbridos (HYbrid Metric Maps) se puede apreciar en la figura 2.8. En ella se muestra un conjunto de hitos puntuales que son utilizados para actualizar probabilísticamente la estructura global del mapa. Estos, a su vez, dividen el espacio en regiones triangulares sobre las que se representa información de más alta densidad; en este caso en la forma de un mapa de ocupación de celdillas.

2.3.3. Asociación de datos

Una correcta asociación de datos es un aspecto crítico en cualquier implementación de SLAM. Antes de fusionar una nueva observación con el mapa, esta es comparada con los elementos ya descritos en el mapa. Ocurre que una sola asociación errónea puede introducir divergencias considerables en la estimación del entorno. Con frecuencia esto provoca fallos catastróficos en el algoritmo de localización y, lo que es peor aún, un fallo irrecuperable de la estimación del modelo.

Posiblemente sea este uno de los aspectos más pobremente resueltos hasta el momento en la literatura. En muchas ocasiones, los algoritmos se validan asumiendo que la asociación de datos viene dada. En otros, las soluciones propuestas se limitan a entornos de geometrías demasiado simplistas. Se enumeran a continuación algunas de las soluciones más exitosas planteadas hasta el momento.

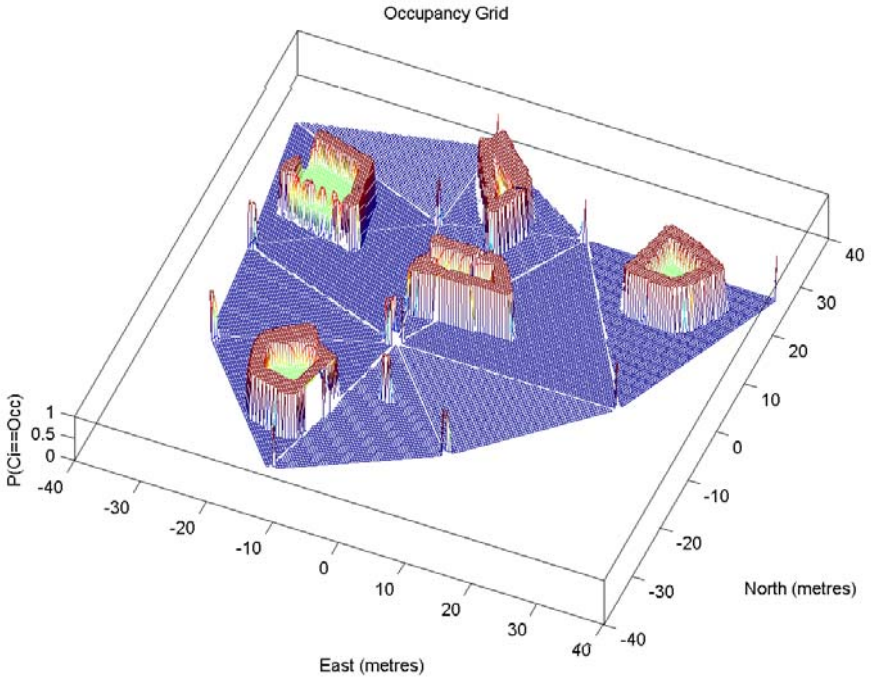


Figura 2.8: Mapa métrico híbrido que combina la solución geométrica estocástica con un mapa de ocupación de celdillas [138].

Asociación en Lote

Las primitivas implementaciones de SLAM consideraban aisladamente el problema de asociar cada observación con una localización prevista en el mapa. Esta solución es extremadamente peligrosa, especialmente cuando la pose del vehículo es incierta, y sólo conduce a resultados aceptables cuando los entornos están muy estructurados y la densidad de características representables es reducida.

Así, un importante avance es la búsqueda de asociaciones en lote, donde se consideran múltiples asociaciones simultáneamente. Estos métodos se aprovechan de las relaciones geométricas que pueden ser establecidas entre hitos del mapa. Dos ejemplos de ello son el Test de Compatibilidad Conjunta (*Joint Compatibility Branch and Bound*, JCBB) que realiza la búsqueda en un árbol [133], y la Asociación de Datos con Restricción Combinada (*Combined Constraint Data Association*, CCDA) que realiza la búsqueda sobre un grafo [8].

Estos métodos presentan gran robustez, ya que reducen el número de asociaciones incorrectas, y propicia que los errores se cometan confundiendo elementos correctos del mapa por otros físicamente próximos, lo que se traduce en una inconsistencia menor. Sin embargo, esto podría no ser cierto en entornos complejos de grandes dimensiones, siendo necesarios mecanismos alternativos.

Firmas de Apariencia

Cuando los sensores proporcionan información lo suficientemente rica, tal y como ocurre con la visión artificial, es posible explotar formas, colores o texturas en la búsqueda de correspondencias entre dos conjuntos de datos. Recientemente, este tipo de información ha sido empleado con gran éxito en la detección de cierre de trayectorias [92, 134].

Asociación de Datos Multi-Hipótesis

Es fundamental considerar este tipo de asociación de datos cuando se pretende mejorar la robustez en entornos con gran densidad de elementos observables [13]. Las ambigüedades surgidas durante la asociación de datos son resueltas generando un rastro de estimación para cada una de las hipótesis admisibles. De esta manera se crea un árbol de hipótesis que va creciendo con el tiempo, y que sólo es limitado por los recursos computacionales disponibles. Las hipótesis de menor probabilidad son sucesivamente eliminadas del árbol. Algoritmos como el FastSLAM [127] son inherentemente multi-hipótesis, al mantener para cada partícula su propia estimación del mapa.

2.4. Conclusiones

En este capítulo se ha realizado un repaso por los métodos más populares que abordan el problema del SLAM desde un punto de vista probabilístico. Esta es sin duda la manera más apropiada de hacerlo, pues la tarea exploratoria de un robot móvil está expuesta a fuentes de incertidumbre de la más variada procedencia; desde el ruido presente en los sensores hasta los modelos simplificados empleados.

Todas las soluciones probabilísticas intentan, de uno u otro modo, resolver la ecuación del filtro de Bayes (2.9), que permite estimar el estado del sistema condicionado a las medidas obtenidas, en el caso en que estas son adquiridas secuencialmente en el tiempo —como es el caso que nos ocupa—. Puesto que dicha ecuación es difícil de resolver en forma cerrada, múltiples técnicas han ido apareciendo a lo largo de los últimos años.

Las soluciones más populares son las basadas en la utilización de un Filtro Extendido de Kalman para estimar recursiva y concurrentemente el estado del mapa y la localización del robot. Presenta el indudable atractivo de modelar el mapa desde un punto de vista antropomórfico; describiéndolo como agregación de objetos individuales cuya posición en el entorno se intenta determinar. Sin embargo estos métodos, a los que se conoce generalmente como SLAM-EKF, pierden su validez cuando no es posible extraer información geométrica de los obstáculos encontrados por el robot, que resulte fácilmente parametrizable.

Así, todas las soluciones basadas en la utilización de un EKF, dependen de la existencia física de una determinada geometría a ser detectada: troncos de árbo-

les que puedan ser asimilados a objetos puntuales (por la posición de su centro o centroide), paredes planas modelables mediante segmentos...

En esta tesis se pretende aprovechar la eficacia y versatilidad de la solución EKF al problema del SLAM, dotándola de un nuevo método de representación capaz de modelar el entorno de manera tan ajustada a la realidad como sea posible. Se pretende evitar así la necesidad de depender de una geometría específica a ser detectada.

Cabe resaltar el hecho de que la nueva parametrización propuesta no será incompatible con los métodos señalados que palían determinados inconvenientes de las solución general del problema. En concreto, la descripción propuesta en esta tesis, basada en la utilización de curvas spline, se reduce en última instancia a la estimación en el vector de estado del sistema de las posiciones de un conjunto de puntos de control.

Así, en esta tesis se aborda el problema de la construcción de mapas geométricos en el marco de trabajo proporcionado por el SLAM-EKF, con las siguientes características que constituyen contribuciones originales de este trabajo:

- Se aborda la frontera existente en los **métodos de representación** del entorno proponiendo un método de representación generalista basado en la utilización de curvas spline.
- Se defiende que este tipo de representación contribuye a facilitar el problema de la **asociación de datos**. La existencia de curvas paramétricas en el modelo del sistema permite así extraer valiosa y rica información en forma de longitudes, distancias, curvaturas...
- Se adopta como solución para paliar el coste computacional del algoritmo una basada en la **utilización de mapas locales**, cuyas posiciones relativas se actualizan para formar un mapa global del entorno. Aquí cobrará especial importancia y significado el anterior punto, ya que el análisis geométricos de los diferentes submapas permitirá establecer relaciones entre ellos que permitan la actualización de su estructura global.

Capítulo 3

Una Introducción a las Curvas Spline

3.1. Introducción

En este capítulo se presenta una introducción a los conceptos fundamentales de la teoría de las curvas spline. El término *spline* se utiliza para referirse a una amplia clase de funciones que se emplean en aplicaciones donde se requiere la interpolación o aproximación de conjuntos de datos arbitrariamente grandes, de una manera flexible y computacionalmente eficiente.

Un spline de grado κ (orden $\kappa - 1$) es una curva polinomial por tramos o, lo que es lo mismo, una curva formada por diferentes secciones, cada una de las cuales es un polinomio de grado κ. Los diferentes fragmentos se unen entre sí en puntos a los que se denomina *nodos*. Su representación más común está basada en la utilización de combinaciones lineales de un cierto tipo de funciones básicas. A esta forma de expresar las curvas spline se la conoce como curvas B-spline (donde la legra B significa *básico*).

En los problemas de interpolación, se utilizan a menudo splines porque dan lugar a resultados similares a los de la interpolación polinomial, incluso utilizando polinomios de grado muy reducido. Además se evitan las oscilaciones —indeseables en la mayoría de las aplicaciones—, que aparecen al interpolar mediante polinomios de grado elevado (fenómeno de Runge) [196,197]. Los splines se utilizan para aproximar formas de arbitraria complejidad. La simplicidad de su manipulación y su facilidad de cómputo los convierten en la más popular de las herramientas para la representación de curvas en informática, particularmente en el terreno de los gráficos por computador.

En la sección 3.2 se hace una breve revisión de los orígenes históricos de las curvas spline, citándose a continuación, algunos de los campos en los que han sido ampliamente utilizadas. En concreto, se hace referencia a la utilidad de estas funciones en la aproximación de datos adquiridos por un sensor. Con esta sección introductoria se pretende mostrar los beneficios de su potencial utilización en el campo de la construcción de mapas geométricos de entornos complejos, donde son aplicadas por vez primera en esta tesis. El resto del capítulo se distribuye del siguiente modo:

En la sección 3.3 se comentan algunos conceptos fundamentales sobre las curvas como entidad matemática. A continuación se formula la definición de B-spline como combinación lineal de funciones básicas. Esta manera de manipular las curvas polinomiales por tramos es especialmente apropiada para su integración en el marco de trabajo del SLAM-EKF, y por esta razón ha sido la elegidas en esta tesis. A continuación se hace una revisión de las particularidades e influencia del vector de nodos (sección 3.5), y un repaso de las propiedades fundamentales de los B-splines, que cobrarán importancia en desarrollos sucesivos (sección 3.6). Finalmente se muestra la manera de aproximar conjuntos de datos obteniendo representaciones compactas de las geometrías que estos representan (sección 3.7).

El capítulo se cierra con la sección 3.8, que enumera las conclusiones que pueden extraerse a la vista del contenido expuesto a lo largo del mismo.

3.2. Orígenes y Aplicaciones

La noticia más antigua sobre la utilización de curvas (en el sentido más abstracto del término) en entornos fabriles de manera, podríamos decir, artesanal e intuitiva, se remonta a la época del Imperio Romano y su poderosa industria naviera. Por aquel entonces, las costillas de los barcos (estructuras de madera que se insertan en la quilla, como si de un esternón animal se tratara) eran construidas empleando plantillas que podían ser reutilizadas tantas veces como fuera necesario. Estas técnicas fueron perfeccionadas por los venecianos entre los siglos XIII y XVI. Sin embargo, no se realizaba ningún tipo de dibujo a lo largo del proceso constructivo de las naves.

No fue hasta el siglo XVII cuando esta práctica se popularizó en Inglaterra. El spline clásico, una barra larga y flexible empleada para dibujar curvas de trazado suave, fue probablemente inventado en esta época. La más antigua aparición de la palabra *spline* se encuentra en un artículo del S. XVIII [63]. Sin embargo, se acepta generalmente que la primera referencia formal matemática a los splines como entidades matemáticas es un artículo de Schoenberg [169], ya en el siglo pasado, donde por primera vez la palabra *spline* es utilizada en conexión con la idea de aproximación polinomial por tramos.

Las soluciones encaminadas a encontrar métodos que permitiesen trazar de manera sencilla curvas de geometría complicada hunden sus raíces en las industrias naviera, y más tarde en la aeronáutica. Pero antes de la aparición de los modernos computadores, cualquier tipo de cálculo numérico era realizado empleando métodos más tradicionales. En concreto, un spline era inicialmente una tira de madera o metal, delgada y flexible, que era curvada y adaptada a la forma deseada para realizar dibujos sobre papel. En [18] se describe el *lofting*[1], una técnica utilizada por

[1] Aún hoy en día se conoce como *lofting* al diseño y/o fabricación de componentes complejos de la estructura de aviones y barcos, utilizando planos o incluso moldes de tamaño real, generalmente colocados sobre el suelo de naves de grandes dimensiones. Por extensión, en el diseño asistido por computador, también se aplica a cualquier modelado tridimensional de un objeto, fundamentalmente durante el diseño de barcos y de aviones [20].

la industria aeronáutica británica durante la Segunda Guerra Mundial que permitía fabricar plantillas para aeronaves en forma de armazones de grandes dimensiones.

Otros desarrollos importantes encontraron caldo de cultivo en el campo de la aeronáutica. En 1944, Roy Liming escribió un libro titulado *"Practical Analytical Geometry with Application to Aircraft"* [117]. Liming trabajó para la NAA (*North American Aviation*) durante la Segunda Guerra Mundial, y su empresa fabricó aviones de combate como el legendario Mustang. En el mencionado libro, los métodos de esbozo clásicos fueron por primera vez combinados con técnicas de cómputo modernas. Liming se percató de que era mucho más eficiente conservar los diseños en forma de datos numéricos que almacenar físicamente las plantillas con curvas trazadas, a partir de las cuales se fabricaban las piezas de sus aviones. Llegó a afirmar orgulloso que la razón por la cual el Mustang era capaz de volar a Berlín y volver sin repostar, era la precisión conseguida al trasladar las ecuaciones matemáticas empleadas en la etapa de diseño, al tablero de dibujo. En palabras del propio Liming:

> *"La aplicación de métodos matemáticos revierte en un control más completo sobre los desarrollos [...]. Consecuentemente, las características de desempeño deseadas o esperadas se consiguen con mayor precisión."*

Poco a poco, iba tomando forma y cuerpo una nueva disciplina dentro del mundo de la ingeniería y las matemáticas: el diseño gráfico asistido por computador (*Computer Aided Graphic Design* o *CAGD*). En la década de 1950 otro hecho fundamental impulsó su desarrollo: la aparición de las máquinas de control numérico. Los primeros computadores eran capaces de generar instrucciones numéricas que eran interpretadas por máquinas mecanizadoras de metales. En el MIT, se desarrolló un lenguaje de programación específico para este propósito: el APT [2].

Pero quedaba por resolver el problema de cómo trasladar la información geométrica relevante, almacenada en planos, a algo que pudiera entender el computador que manejaba la máquina. Inicialmente se intentó extraer ciertos puntos significativos de los planos, y ajustar los contornos mediante técnicas familiares como la interpolación de Lagrange. Pero aquellos primeros intentos demostraron pronto no ser suficientes ante los nuevos retos y crecientes demandas que planteaban los diferentes procesos de fabricación industrial. En Francia, de Casteljau y Bézier contribuyeron de manera decisiva a resolver estos problemas, como veremos.

Mientras, en Estados Unidos, J. Ferguson en Boeing y S. Coons en el MIT proporcionaban técnicas alternativas [47, 49]. General Motors desarrolló el primer sistema CAD/CAM denominado *DAC-I* (*Design Augmented by Computer*), que utilizaba técnicas de generación de curvas y superficies desarrolladas por C. de Boor y W. Gordon [112].

La utilización de los splines en el modelado de las carrocerías de automóviles parece tener diversos orígenes independientes. El mérito es compartido por Casteljau

[2] El *CAM* (*Coumputer-Aided Manufacturing*) comenzó en la década de 1950 con la aparición del lenguaje de programación *APT* (siglas de *Automatic Programmed Tool*) desarrollado en el MIT. El programador trabajaba a partir de los planos y generaba manualmente el programa que era introducido en la máquina de control numérico. El *CAD* (*Computer-Aided Design*) realmente no apareció hasta 20 años más tarde, con la aparición de computadores con capacidades gráficas.

en Citroën, Bézier en Renault y Birkhoff, Garabedian y de Boor en General Motors, todos los cuales desarrollaron sus trabajos desde finales de la década de 1950 hasta comienzos de la de 1960. Los resultados de de Boor fueron publicados en varios artículos, incluyendo algunos estudios fundamentales sobre los B-splines. No fue hasta 1967 cuando apareció la primera obra monográfica sobre estas interesantes curvas [2].

Todos estos trabajos tenían lugar durante la década de 1960. Durante mucho tiempo se desarrollaron en paralelo, aisladamente, hasta que en los años setenta comenzó a materializarse una convergencia entre las diferentes aproximaciones al problema, que culminó en la creación de una nueva disciplina: el Diseño Gráfico Asistido por Computador (CAGD).

El término CAGD fue acuñado por R. Barnhill y R. Riesenfeld en 1974 cuando organizaron una conferencia sobre el tema en la Universidad de Utah. Aquella conferencia puede ser considerada el germen de este campo de la ingeniería [15]. El primer libro de texto, *"Computational Geometry for Design and Manufacture"* apareció en 1979, obra de I. Faux y M. Pratt [74], y la revista *Computer Aided Geometric Design* fue fundada en 1984.

3.2.1. El origen del término *spline*

Hemos visto que a lo largo de la historia ha sido necesario representar curvas en numerosas aplicaciones de la artesanía y de la técnica. Al principio la mayoría de estas curvas eran arcos de circunferencia, pero pronto surgió la necesidad de dibujar curvas de forma libre en campos como la construcción naval o la arquitectura. Cuando era preciso trazar la curva con cierta exactitud, la herramienta más común empleada por los delineantes era un conjunto de plantillas conocido como *curvas francesas*. Éstas eran fabricadas inicialmente en madera, a partir de curvas cónicas y espirales. Así, era posible dibujar curvas a tramos utilizando adecuadamente una o varias de estas plantillas.

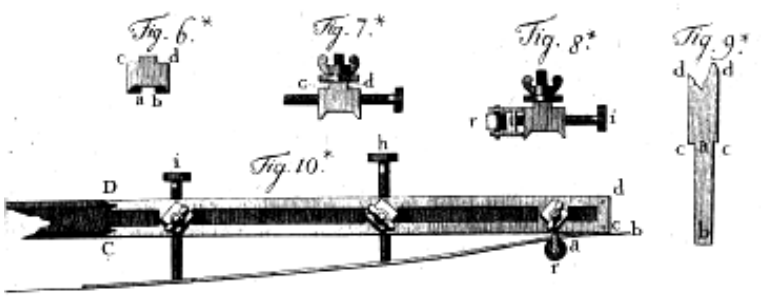
También se utilizaba por los dibujantes otra herramienta mecánica llamada *"spline"*. Esta era una tira flexible de madera a la que se daba forma utilizando una serie de pesos metálicos conocidos como *patos* (*"ducks"*). Cuando era necesario realizar los dibujos a gran escala se utilizaban las buhardillas (*"lofts"*) de los edificios para realizar y almacenar los dibujos (de ahí el origen de la palabra *"lofting"*). La correspondiente representación matemática de las curvas así obtenidas toman el nombre de la herramienta utilizada para su trazado, y son uno de los tipos fundamentales de curvas paramétricas.

La geometría diferencial de las curvas paramétricas es bien conocida desde finales del siglo XIX tras los trabajos de Serret y Frenet. Por otro lado, la investigación sobre teoría de la aproximación y análisis numérico se había centrado fundamentalmente en el estudio de las funciones no paramétricas. Ambas ramas se fusionaron gracias al desarrollo del CAGD.

Desde mediados de la década de 1950, la compañía americana Boeing utilizaba la tecnología desarrollada por Liming, basada en la utilización de curvas y superficies

(a)

(b)



(c)

Figura 3.1: Algunos ejemplos de splines mecánicos como herramientas para el trazado de curvas: (a) Spline de madera flexible acompañado de los pesos metálicos o *"ducks"* que ayudan a fijar su posición. (b) Juego de curvas francesas. (c) Spline mecánico utilizado en el siglo XVIII.

cónicas, en el diseño de los fuselajes de sus aeroplanos. En la misma compañía, J. Ferguson y D. MacLaren desarrollaron un nuevo tipo de curva para el diseño de las alas. Se les ocurrió la idea de unir varias curvas cónicas de manera que la curva compuesta resultante fuera dos veces diferenciable [75, 122]). Mediante estas curvas, era sencillo interpolar un conjunto de puntos dados, y las llamaron *splines* porque su forma resultaba de minimizar un funcional similar a las ecuaciones físicas relacionadas con los splines mecánicos.

Así, la expresión "curva spline" ha sufrido un ligero cambio de significado; en vez de referirnos por ese nombre a curvas que minimizan ciertos funcionales, son ahora aquellas curvas constituidas a tramos por polinomios (o relaciones entre polinomios), compartiendo con las anteriores la propiedad de presentar formas redondeadas y trazado suave.

3.2.2. De la industria automovilística al arte y la arquitectura

En 1959, la compañía francesa Citroën se puso en contacto con un joven matemático para resolver los problemas teóricos que entrañaba el ya mencionado proceso de transformar la información contenida en planos a datos numéricos. Este matemático era Paul de Faget de Casteljau, que acababa de terminar su tesis doctoral. Comenzó a desarrollar un sistema enfocado fundamentalmente al diseño, partiendo de cero, de curvas y superficies, más que a la mera reproducción de diseños existen-

tes. Desde un comienzo, adoptó la utilización de los polinomios de Bernstein para definir sus curvas y superficies, unido a lo que hoy es conocido como el *algoritmo de de Casteljau.*

Su principal adelanto fue la utilización de los *polígonos de control* (*courbes à pôles*), una técnica no empleada hasta la fecha. En vez de definir una curva o superficie a partir de puntos contenidos *en* ella, el polígono de control que da origen a la curva utiliza puntos que están *cerca* de ella. En vez de cambiar la curva en sí, se modifica el polígono de control de manera que se puede cambiar la apariencia de la curva de manera muy intuitiva.

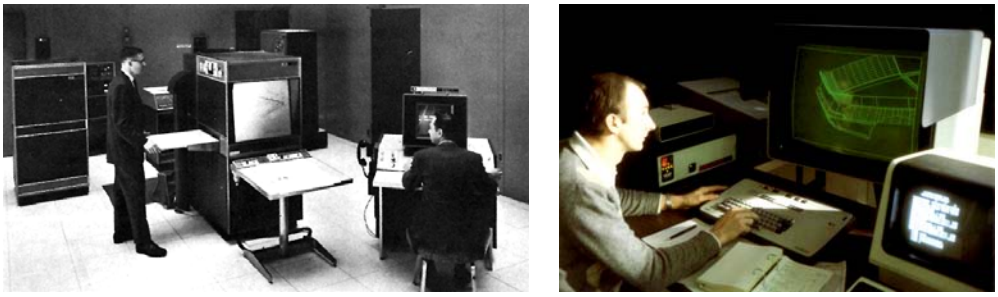
Pierre Étienne Bézier, a comienzos de los años sesenta, se encontraba trabajando para uno de los principales competidores de Citroën, la también francesa Renault. Allí también se dieron cuenta de la necesidad de obtener representaciones geométricas de las piezas mecánicas en términos comprensibles para los computadores. La idea inicial de Bézier era representar una curva básica como la intersección entre dos cilindros elípticos definidos en el interior de un paralelepípedo; así, transformaciones afines de este paralelepípedo resultaban en correspondientes transformaciones afines de la curva que generaban. Más tarde, Bézier procedió a formulaciones polinomiales de este concepto inicial. Los resultados obtenidos fueron idénticos a las curvas obtenidas por de Casteljau, con la diferencia de que los razonamientos matemáticos empleados en su obtención eran diferentes.

Los trabajos de Bézier y su equipo [24–28, 28, 201]) fueron muy comentados en la época, llamando pronto la atención de A. R. Forrest. Este se dio cuenta de que las curvas de Bézier podían ser expresadas en términos de los polinomios de Bernstein (como ya lo había hecho de Casteljau desde finales de los cincuenta). El artículo de Forrest sobre las curvas de Bézier [78] tuvo mucha influencia, y contribuyó a la popularización de la utilización de estas curvas. UNISURF [29], el sistema CAD/CAM de Renault cuya primera versión apareció en 1971, estaba basado enteramente en curvas y superficies de Bézier. Este sistema influenció los desarrollos en el seno de la compañía de construcción aeronáutica francesa Dassault, que construyó el sistema EVE. Más tarde, este último evolucionó hasta convertirse en CATIA (*Computer Aided Three-dimensional Interactive Application*) [3].

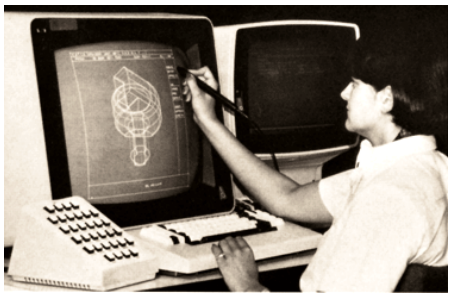
Los sistemas CAD/CAM han evolucionado enormemente a lo largo de los últimos años. El arquitecto norteamericano Frank Gehry fue pionero en su utilización en la arquitectura. Así, se sirvió de CATIA para diseñar obras como el Museo Guggenheim de Bilbao, o el Pez Dorado del Puerto Olímpico barcelonés. También el escultor Richard Serra se sirvió de este programa para modelar su obra "La materia del Tiempo", que se exhibe en el mencionado museo bilbaíno. En todos los casos, las complejas geometrías son modeladas empleando un tipo especial de curva spline denominado *NURBS* (*Non-Uniform Ration B-Splines*, B-Splines Racionales No Uniformes). En la figura 3.3 se pueden observar algunos ejemplos de estas creaciones.

Las inquietudes artísticas que Bézier tuvo a lo largo de su vida también fueron plasmadas en numerosos diseños de flores y esculturas utilizando las curvas de su invención. En la figura 3.4 se muestran un par de ejemplos. En la introducción del

[3] *http://www.catia.com*



(a) (b)



(c)

Figura 3.2: (a) El sistema DAC-I, desarrollado por IBM para General Motors, considerado por muchos el primer sistema CAD. (b) El sistema UNISURF de Renault, empleado en el diseño y fabricación del modelo GTA hacia 1985. (c) El sistema CATIA de Dassault, en su versión de comienzos de la década de los 80.

artículo de Jordi Alberich titulado "Las flores de Bézier. Elasticidad e inestabilidad en el grafismo digital interactivo" [3] puede leerse la siguiente frase, refiriéndose a la flor que aparece en la figura 3.4(a):

> *"En la flor representada se corporeiza de forma singular cómo una técnica matemática nacida originalmente en el contexto de unas determinadas necesidades de diseño industrial permitiría con el tiempo expandir su uso y utilidad mucho más allá de su finalidad inicial, hasta alcanzar los vastos campos del grafismo, el arte y el expresionismo digital."*

Estas palabras recogen también el espíritu y objetivo último de la presente tesis: expandir el uso de técnicas de representación nacidas al amparo de las industrias naval, aeronáutica y automovilística, y mostrar su aplicabilidad, utilidad y eficacia en un campo tan lejano y vasto como es el SLAM.

3.2.3. B-splines y NURBS

El concepto de los *B-splines* fue introducido por I. Schoenberg en 1946 [169] para el caso particular de presentarse los nodos uniformemente distribuidos. En 1960 C. de Boor comenzó a trabajar en los laboratorios de investigación de General Motors y comenzó a usar B-splines como una herramienta de representación geométrica. Más tarde se convirtió en uno de los más importantes defensores de la utilización

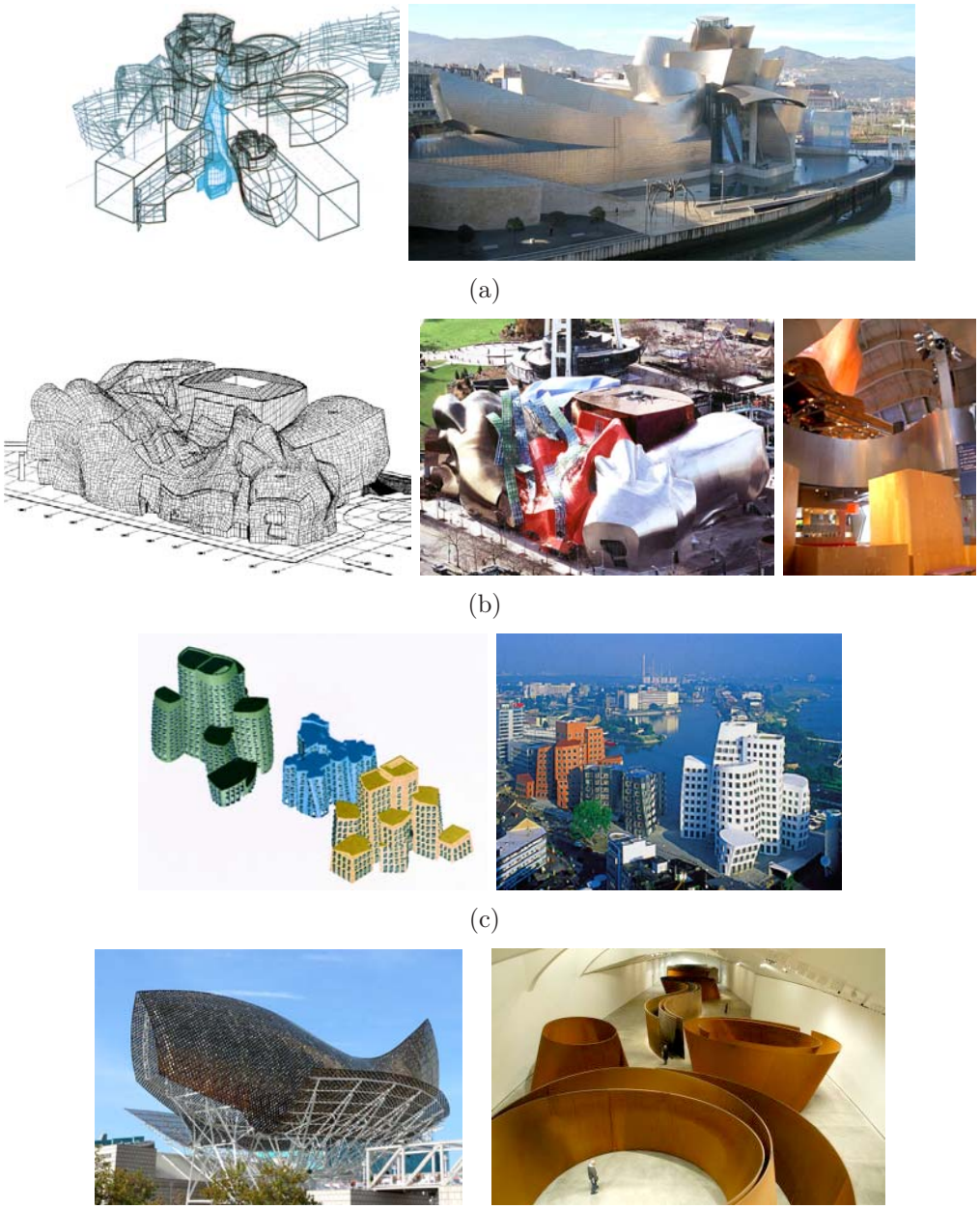
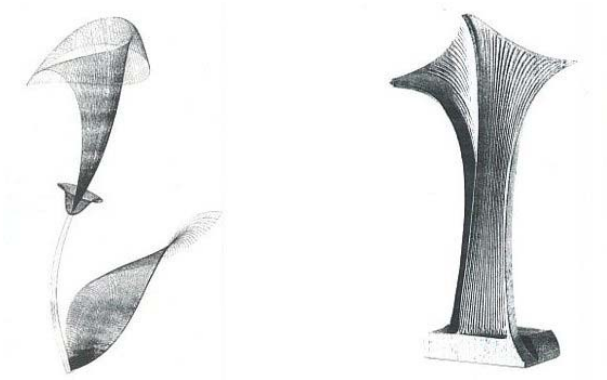


Figura 3.3: Algunas obras de Frank Gehry modeladas con NURBS: (a) Museo Guggenheim de Bilbao[4], (b) el EMP|SFM[5] de Seattle, (c) y el complejo de oficinas *Der Neue Zollhof*, en Dusseldorf. (e) "La materia del Tiempo", de Richard Serra.



(a) (b)

Figura 3.4: Creaciones artísticas de Pierre Bézier. (a) Diseño de una flor asistido por computador. (b) *"Notre Dame de la Commande Numérique"*: diseño de una escultura asistido por computador, utilizando las curvas que llevan su nombre.

de estas curvas en la teoría de la aproximación. A él se debe la evaluación recursiva de las funciones básicas, conocida como algoritmo de de Boor [52]. Este algoritmo hizo viable la aplicación práctica de estas curvas, ya que antes de su aparición los B-splines se definían utilizando una aproximación en diferencias divididas que era numéricamente muy inestable [53].

Las funciones spline tienen una importancia capital en la teoría de la aproximación, pero en el CAGD, los splines paramétricos son mucho más importantes aún. Fueron introducidos por R. Riensfeld y W. Gordon en [84] (este artículo es un extracto de la tesis de Riensfeld [154]), quienes se dieron cuenta de que la evaluación recursiva propuesta por de Boor era la generalización natural del algoritmo de de Casteljau. Por tanto, las curvas B-spline incluían a las de Bézier como un caso particular, y pronto fueron implementadas en todos los sistemas CAD. El primer esquema de conversión B-spline a Bézier fue encontrado por W. Boehm [31]. A partir de entonces se desarrollaron numerosos algoritmos que simplificaban el tratamiento matemático de los B-splines; ejemplo de ello son el algoritmo de inserción de nodos de Boehm [32], el algoritmo Oslo de E. Cohen, T. Lyche y R. Riensfeld [46], y la introducción del principio de *"blossoming"* por L. Ramshaw [153] y P. de Casteljau [56].

La generalización de las curvas B-spline a las *NURBS* (*"Non Uiform Rational B-Splines"*) se ha convertido en la representación estándar de curvas y superficies en el CAD/CAM industrial. Ofrecen una representación unificada de las geometrías spline y cónica, ya que cualquier sección cónica es susceptible de ser representada mediante polinomios racionales por tramos. El primer tratamiento sistemático de los NURBS fue realizado por K. Versprille en su tesis doctoral [202]. Versprille fue alumno de S. Coons, que comenzó a trabajar con curvas racionales en la década de 1960 [48].

Un caso especial de NURBS son las curvas racionales de Bézier. Más especial aún es el caso de las secciones cónicas, o curvas racionales cuadráticas de Bézier. Por último, G. Farin encontró la generalización racional del algoritmo de de Casteljau

en 1983 [73].

3.2.4. Aplicaciones científicas

Son innumerables las áreas de la ciencia y de la técnica que se han beneficiado en el pasado y en el presente de las múltiples ventajas que supone la utilización de las curvas spline. Muchas disciplinas precisan de modelos que describan fenómenos, para representar los cuales sólo se dispone de un conjunto discreto de medidas. Un ejemplo de ello serían los mapas meteorológicos, de isobaras o isotermas, ya que para construirlos se dispone de los datos adquiridos en determinadas estaciones meteorológicas, pero se desea disponer de un modelo continuo de temperaturas, presiones, etc. Dado que a menudo —y siguiendo con el símil meteorológico— estas estaciones no están localizadas de manera estructurada (por ejemplo, dispuestas en forma de rejilla sobre el mapa), se habla a menudo de *datos dispersos*.

Es por tanto necesario disponer de funciones que sean capaces, al menos, de interpolar los datos disponibles, así como de proporcionar estimaciones de los valores en localizaciones intermedias. Uno de los primeros intentos en este sentido lo constituye el método de Shepard [14, 85, 170].

El estadístico británico R. Sibson desarrolló un esquema de interpolación que denominó "interpolación del vecino más próximo" (*"nearest neighbour interpolation"*). Está basado en el concepto de los diagramas de Voronoi, también conocidos como teselaciones de Dirichlet. Sibson mostró que cualquier punto contenido en la envolvente convexa de un conjunto dado de puntos podía ser expresado como combinación lineal de sus vecinos [172, 173]. Otros ejemplos de aplicación en el campo del modelado estadístico y análisis de datos pueden encontrarse en [39] y [104].

Otros campos de la ciencia y la tecnología en los que la teoría de los splines ha encontrado campo de cultivo son la teoría de control [109, 213], el reconocimiento y tratamiento de formas [38, 151], el procesamiento de señal [198], el procesamiento de imagen [210], la mecánica de fluidos [114], la síntesis de voz [57], tratamientos tipográficos [100], o la robótica y la inteligencia artificial al servicio de la medicina [126, 198].

Kostková y Halíř utilizan en [111] splines de interpolación y de aproximación para representar los contornos de fragmentos de cerámica procedentes de excavaciones arqueológicas. El objetivo es automatizar su clasificación, y en su trabajo hacen especial énfasis en los beneficios que supone la utilización de estas curvas a la hora de preservar y describir adecuadamente las características geométricas de los fragmentos.

Dentro de la robótica móvil también se han utilizado curvas spline, pero su aplicación se ha visto limitada fundamentalmente al problema de la generación de trayectorias [45,106]. Piccolo *et. al* intentan en [147] modelar mediante curvas spline los obstáculos que encuentra un robot móvil en su exploración. Para ello utilizan un método de ajuste recursivo a medida que se reciben datos de un sensor láser, y realizan experimentos con tres geometrías concretas, cuyos resultados se muestran en la figura 3.6. Además utilizan un marco de trabajo no probabilístico, lo cual les obliga

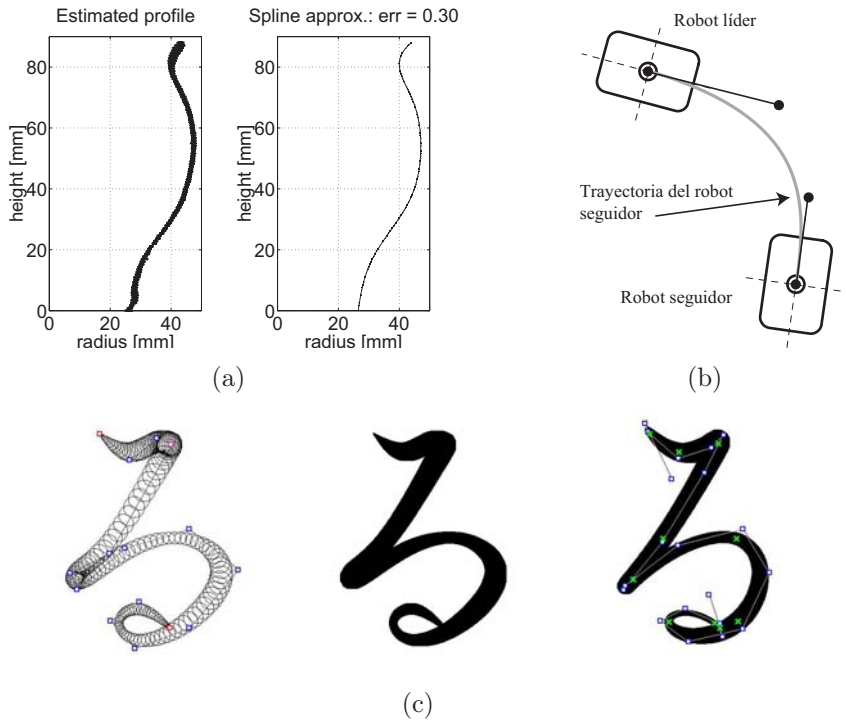


Figura 3.5: Ejemplos de aplicación de los splines. (a) Descripción del borde fragmentos arqueológicos de cerámica [111]. (b) Generación de la trayectoria que ha de trazar un robot móvil seguidor de otro guía [45]. (c) Generación del carácter japonés *"Ru"* utilizando splines cúbicos [109].

a mantener el robot en trayectorias circulares que minimizan el desplazamiento, y (cito) "permiten que el robot retorne a la posición de partida tras cada revolución" sin tener que utilizar referencias externas.

Así pues, en el mencionado trabajo el problema se reduce a el ajuste incremental de datos, cuyos valores asumen contaminados por el ruido del sensor, pero centrados en la posición correcta al obviar el error de localización del robot. Además, a pesar de realizar los experimentos en un ambiente controlado (sin presencia de dinámicos como personas), los resultados obtenidos no son aceptables cuando en el entorno aparecen esquinas.

3.3. Conceptos Sobre Curvas

Dado que las curvas spline constituyen un caso particular de curva, se ha considerado necesario dedicar algunos párrafos de esta tesis a considerar, al menos someramente, algunos aspectos fundamentales relativos a esta categoría geométrica. Así, en esta sección se presentan algunos conceptos e ideas fundamentales sobre las curvas, consideradas como entidades geométricas que capturan la idea abstracta de línea continua, y la concretan en formulaciones matemáticas más o menos complejas. También se comenta alguna propiedad de estas entidades, cuyo conocimiento será de utilidad tener presente en el desarrollo posterior de los resultados que presenta esta tesis.

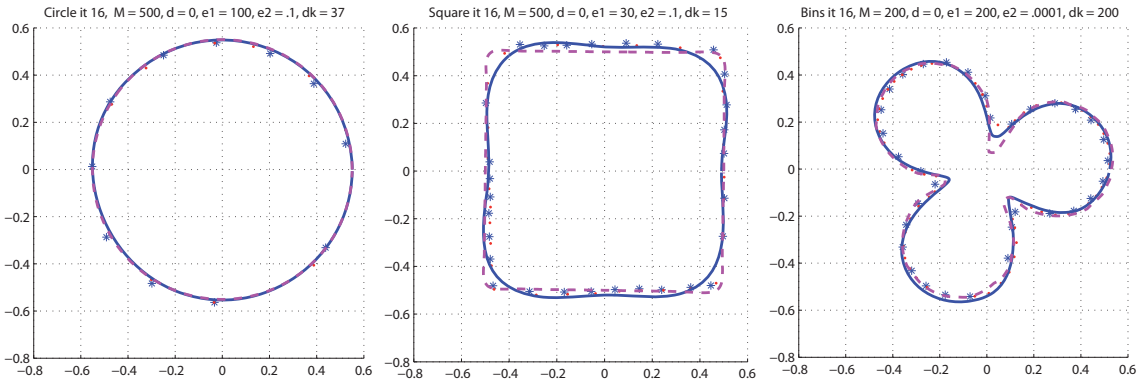


Figura 3.6: Trabajo de Piccolo *et. al* [147] en el que intentan construir mapas utilizando curvas spline. Se observa que la falta de una segmentación previa adecuada, y el método de ajuste empleado, hacen que no se consiga aproximar determinadas geometrías adecuadamente.

3.3.1. ¿Qué es una curva?

Según el diccionario de la Real Academia Española de la Lengua, en una de sus acepciones, una curva es una *línea que representa gráficamente la magnitud de un fenómeno según los valores que va tomando una de sus variables.* Matemáticamente, podemos considerar una curva como *una relación funcional continua entre un espacio unidimensional, y otro n-dimensional.*

Más formalmente, una curva en el espacio euclídeo de n dimensiones $\Re^n$ es un conjunto de puntos $C \subset \Re^n$ obtenido como imagen de un intervalo $I \subset \Re$ (recordemos que un intervalo es un subconjunto conexo de $\Re$) bajo una aplicación continua $\mathbf{x}$:

$$\mathbf{x} \quad : \quad I \longrightarrow \Re^n \tag{3.1}$$

$$C \quad = \quad \{\mathbf{x}(t) \in \Re^n : t \in I\} \tag{3.2}$$

Intuitivamente, podríamos pensar que una curva es algo que se puede dibujar con un lápiz sobre una hoja de papel. También es cierto que una curva es un conjunto infinitamente grande de puntos. Los puntos de una curva tienen la propiedad de que cualquiera de ellos tiene dos vecinos, excepto a lo sumo dos, que sólo tienen un vecino; estos son los *puntos extremos* o *terminaciones* de la curva. Algunas curvas no tienen terminaciones, bien porque su longitud es infinita, o porque son cerradas.

El problema fundamental que nos encontramos al intentar estudiar y analizar los componentes de esta categoría geométrica es el de *cómo podemos describir matemáticamente una curva.* Dicho con otras palabras: el primer problema que se plantea, antes de proceder a la interpretación o el análisis de la geometría que efectivamente muestra una curva, es el de obtener representaciones para todas las curvas, sin importar su complejidad, de manera que puedan ser tratadas, entendidas y analizadas por una mente humana o artificial.

Para algunas curvas, el problema de caracterizarlas es sencillo, ya que presentan formas bien conocidas y estudiadas históricamente por la ciencia matemática. Tal es el caso de la línea recta (caso límite de una cruva, cuya curvatura tiende al infinito), un segmento (fragmento de recta comprendido entre dos puntos de la misma), una circunferencia, los arcos de elipse, etc. Así, nos encontramos con casos particulares de curvas a las que podemos dar un nombre, y cuya representación es inmediata y fácilmente parametrizable.

Sin embargo, una curva general, que no presenta una forma a la que podamos dar un nombre es conocida como una *curva de forma libre*. Dado que estas curvas pueden tomar cualquier forma imaginable, son mucho más difíciles de describir.

Se puede decir que existen tres métodos fundamentales para describir matemáticamente una curva:

- **Método Implícito**. Se define el conjunto de puntos que conforman la curva proporcionando un procedimiento que permite comprobar que el punto pertenece a la curva. Generalmente, una curva implícita en el espacio $\Re^2$ viene definida por una *función implícita* de la forma:

$$f(x, y) = 0 \tag{3.3}$$

  $x, y \in \Re$, de manera que la curva es precisamente el conjunto de puntos para los cuales la ecuación se verifica. Nótese que la función implícita es una función escalar (devuelve un único valor real).

- **Método Explícito o Paramétrico**. Se establece una relación entre un parámetro independiente y el conjunto de puntos que forman la curva. Este parámetro independiente (un escalar) se puede entender como un *índice* de los puntos de la curva. Intuitivamente, el parámetro independiente podría ser el tiempo que tarda un móvil en alcanzar una posición sobre la curva, o la distancia recorrida sobre la misma (Fig. 3.7). La *función paramétrica* de esta curva nos indicaría la posición de la punta del lapicero en cada instante de tiempo:

$$\begin{bmatrix} x(t) & y(t) \end{bmatrix}^T = \mathbf{f}(t) \tag{3.4}$$

  En este caso la función paramétrica devuelve la posición de un punto de la curva en el espacio $\Re^2$.

- **Método Generativo o Procedural**. En este caso, la descripción de la curva es un procedimiento que es capaz de generar los puntos de la curva, y que no pertenece a ninguna de las dos categorías anteriores. Se trata más bien de métodos basados en reglas, y ejemplos de estas curvas son los esquemas de subdivisión, los fractales, o las simetrías.

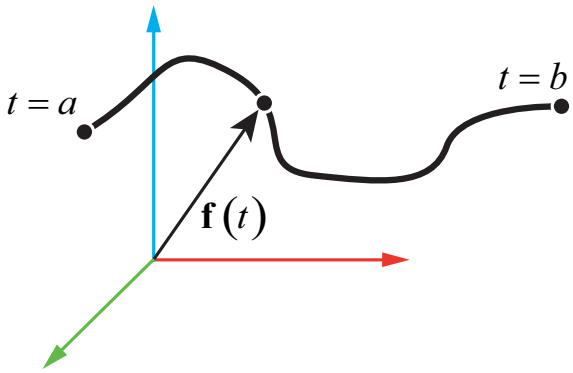


Figura 3.7: Ejemplo de curva paramétrica. El vector calculado como resultado de aplicar la función **f** sobre el rango de definición del parámetro $[a, b] \in Re$ da como resultado una curva en el espacio $\Re^3$. El parámetro independiente puede ser entendido como el instante temporal t en el que podríamos encontrar un móvil que se desplazar sobre la curva en la posición $\mathbf{f}(t)$, o como la distancia que separa al móvil sobre la curva de un origen predefinido, dependiendo de la parametrización escogida.

Es fácil representar algunas curvas tanto de manera explícita como implícita. Por ejemplo, la circunferencia de radio unidad centrada en el origen admite representación implícita como:

$$f(x, y) = x^2 + y^2 = 1$$

o la equivalente formulación paramétrica:

$$\begin{pmatrix} x \\ y \end{pmatrix} = \mathbf{g}(u) = \begin{pmatrix} cos(u) \\ sin(u) \end{pmatrix}$$

Cada uno de los tipos de representación tiene sus propias ventajas y desventajas. Por ejemplo, las curvas paramétricas son mucho más sencillas de dibujar, ya que sólo hay que determinar los puntos $(x, y) = \mathbf{f}(t)$ para diferentes valores del parámetro independiente t. Por ello, las formulaciones paramétricas son las más comúnmente utilizadas en los gráficos por computador. Los splines son un claro ejemplo de curva paramétrica.

3.3.2. Representaciones paramétricas por tramos

Para algunas curvas, encontrar una función paramétrica que represente su forma es sencillo. Por ejemplo, líneas, círculos, elipses, parábolas... son diferentes curvas que disponen de funciones paramétricas relativamente simples que nos dan los puntos que las conforman como función de un único parámetro. Pero para otras muchas curvas (la mayoría, a decir verdad), encontrar una única función que describa su forma puede ser realmente difícil.

La principal estrategia empleada históricamente a la hora de caracterizar curvas de forma complicada es la de *divide y vencerás*: se trocea la curva en un número arbitrario de fragmentos más pequeños, cada uno de los cuales es susceptible de ser descrito de manera simple.

Por ejemplo, consideremos la curvas representadas en la figura 3.8. Las dos primeras curvas son fácilmente representables matemáticamente una vez descompuestas en tramos cortos. Para crear una representación paramétrica de una curva como la (b) de la figura 3.8 es preciso hacerlo a partir de los tramos de curva elemental que la componen.

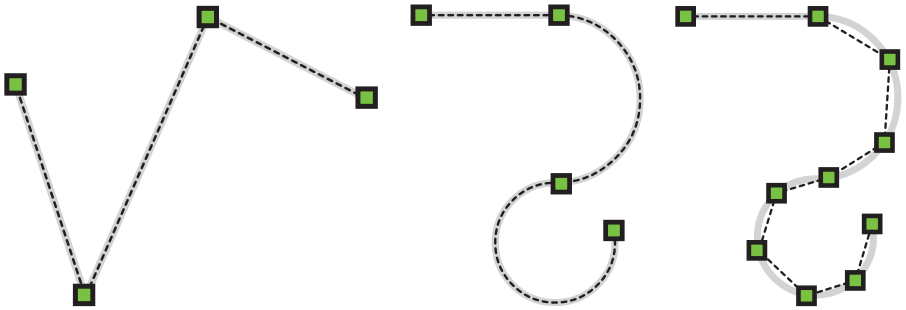


Figura 3.8: (a) Una curva que puede ser representada por tres líneas. (b) Una curva que puede ser representada fácilmente como una línea y dos arcos de circunferencia. (c) Aproximación de la curva (b) mediante segmentos.

Para representar adecuadamente la curva (b) de la figura 3.8 es preciso utilizar dos tipos de curva elemental como piezas: un segmento y dos arcos circulares. Por simplicidad, podríamos preferir utilizar un único tipo de curva como elemento constructivo. Si intentáramos reproducir la curva (b) utilizando, por ejemplo, únicamente segmentos, no podríamos aproximar la curva original (a menos que empleáramos para ello un número infinito de fragmentos rectilíneos).

A pesar de que la curva (c), construida a partir de segmentos, no tiene exactamente la misma forma que la curva original, podría ser lo suficientemente aproximada en determinadas aplicaciones. En tal caso, podría ser preferible la simplicidad que proporciona utilizar funciones paramétricas sencillas aproximando los diferentes tramos de la curva, que utilizar funciones complicadas que aproximen más fidedignamente la curva original.

También es importante resaltar que, a medida que incrementamos el número de piezas, obtenemos una mejor aproximación a la curva original utilizando funciones sencillas. En el límite (descomponiendo la curva en un número infinito de tramos) obtendríamos exactamente la forma original. Con todo ello, una ventaja fundamental de utilizar la representación por tramos de curvas complicadas es que permite establecer compromisos entre distintos criterios:

- Cómo de bien representa nuestra aproximación a la forma original.
- Cómo de complicadas son las piezas que utilizamos para construir la curva.
- Cuántas piezas empleamos o, lo que es lo mismo, en cuántos fragmentos dividimos la curva original.

En el mundo de los gráficos por computador, la tendencia habitual es la de

utilizar tramos de curva relativamente sencillos, bien sean estos segmentos lineales o tramos de polinomio. Tal es el caso de las curvas spline.

3.3.3. Propiedades de las curvas

Para describir una curva, es preciso definir ciertos parámetros que describen sus propiedades. Para las curvas "con nombre", que son las más conocidas y descritas en la bibliografía matemática, estas propiedades son generalmente específicas del tipo de curva que estemos tratando. Por ejemplo, para describir un círculo, podríamos establecer su radio y la posición de su centro. En el caso de una elipse, también serían precisos como datos la orientación del eje mayor y la relación entre las longitudes de los ejes. Pero para las curvas de forma libre necesitamos un conjunto más general de propiedades a la hora de describir las curvas individuales a partir de las cuales las construiremos.

Algunas propiedades describen sólo una parte pequeña de la curva, mientras que otras requieren conocer la curva completa. Supongamos que nuestra curva es un tren de mercancías. Si estamos situados en uno de los vagones en un día con mucha niebla, podremos decir si nuestro vagón se desplaza en línea recta o está trazando una curva, o si estamos o no en uno de los vagones extremos del tren. Éstas serían *propiedades locales*. Pero no podríamos decir si el tren se encuentra dando vueltas en círculo, sobre una vía en bucle cerrado, o cuál es su longitud total. Estas segundas serían *propiedades globales*.

El estudio de las propiedades locales de las entidades geométricas (curvas y superficies) se conoce como *geometría diferencial*. Las propiedades locales son importantes herramientas para describir las curvas, ya que no requieren un conocimiento de toda la curva. Algunas propiedades locales son:

- Continuidad.
- Posición en un punto de la curva.
- Dirección en un punto de la curva.
- Curvatura.

Algunos ejemplos de propiedades globales interesantes podrían ser:

- Saber si la curva es abierta o cerrada.
- Saber si la curva pasa por un punto determinado, o atraviesa cierta región.
- Saber si la curva en algún tramo apunta en una determinada dirección.

Si una curva es definida utilizando diferentes tramos descritos por diferentes funciones, hay que ser muy cuidadosos a la hora de definir los tramos. En general será necesario que se cumpla la condición de continuidad, lo cual obliga a que cada tramo

comience donde terminó el anterior. En ocasiones también es precisa la continuidad en la primera derivada y sucesivas.

En general, diremos que una curva presenta continuidad C^n si sus n primeras derivadas son continuas en todos sus puntos. La figura 3.9 muestra gráficamente algunos tipos de continuidad. Una discontinuidad en la primera derivada es generalmente perceptible, puesto que da lugar a puntos angulosos en las curvas. Las discontinuidades en la segunda derivada también son a menudo apreciables. Discontinuidades en derivadas de orden superior pueden ser importantes o no, dependiendo de la aplicación. Por ejemplo, si la curva representa el movimiento de un sólido, una discontinuidad en la segunda derivada (aceleración) puede tener repercusiones importantes, y si la curva representa un perfil que va a estar en contacto con un fluido (si es por ejemplo la forma del ala de un avión), una discontinuidad en la cuarta o quinta derivadas podría causar turbulencias.

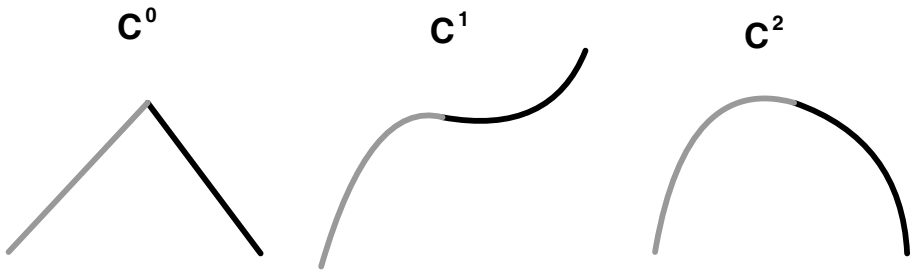


Figura 3.9: Varios tipos de continuidad entre dos tramos curvos.

En ocasiones se define la *continuidad geométrica*, como la condición por la cual la diferencia entre las derivadas de dos tramos de curva adyacentes difiere sólo en una cierta magnitud. Es decir, mientras una continuidad C^1 exigiría

$$\mathbf{f}_\mathbf{i}'(1) = \mathbf{f}_{\mathbf{i+1}}'(0) \tag{3.5}$$

la continuidad geométrica, G^1 requiere:

$$\mathbf{f}_\mathbf{i}'(1) = k \cdot \mathbf{f}_{\mathbf{i+1}}'(0) \tag{3.6}$$

para algún $k \in \Re$. La continuidad geométrica es menos restrictiva que la continuidad paramétrica.

3.3.4. Una propiedad interesante: la curvatura

Intuitivamente, la curvatura es la "cantidad" que una forma geométrica se "aleja" de ser *plana*. El término *plana* puede tener diferentes significados, dependiendo de la entidad geométrica considerada. En el caso de las curvas, algo plano sería la línea recta, mientras que en el caso de las superficies la referencia de planitud sería el plano euclídeo.

En el caso de una curva plana, la curvatura en un punto dado P se define como la magnitud igual al inverso del radio del círculo osculador en el punto (el círculo

que "besa", o toca tangencialmente a la curva en el punto), y se representa mediante un vector apuntando hacia el centro de este círculo. Su unidad es la *dioptría*, que a su vez tiene dimensión de m^{-1}.

Cuanto más pequeño sea el radio ρ de curvatura, mayor será la magnitud de la curvatura, $1/\rho$. Como ejemplo, una línea recta tendrá curvatura nula en todos sus puntos, mientras que una circunferencia de radio R tendrá una curvatura constante de valor $1/R$.

Para una curva plana dada paramétricamente por una ecuación del tipo de la 3.4, el valor de la curvatura con su signo es:

$$\kappa = \frac{x'y'' - y'x''}{(x'^2 + y'^2)^{3/2}} \tag{3.7}$$

Tomando x como parámetro queda una curva expresada en la forma: $y = f(x)$. En este caso, la ecuación de la curvatura se puede escribir como sigue:

$$\kappa = \frac{\frac{d^2y}{dx^2}}{\left[1 + \left(\frac{dy}{dx}\right)^2\right]^{3/2}} \tag{3.8}$$

La curvatura es una interesante propiedad de las curvas planas. No depende de ningún sistema de referencia. Se trata de una propiedad puntual, que puede ser evaluada de manera muy sencilla. Obsérvese que, al venir expresado su valor en función de las derivadas primera y segunda, para cualquier curva $y = f(x) \in C^2$ la función curvatura será una función continua. Para más información sobre el tema se pueden consultar [50] y [86]. En la figura 3.10 se puede observar una representación de la función $y = sin(x)$ y de su curvatura, en el intervalo $[0, 2\pi]$.

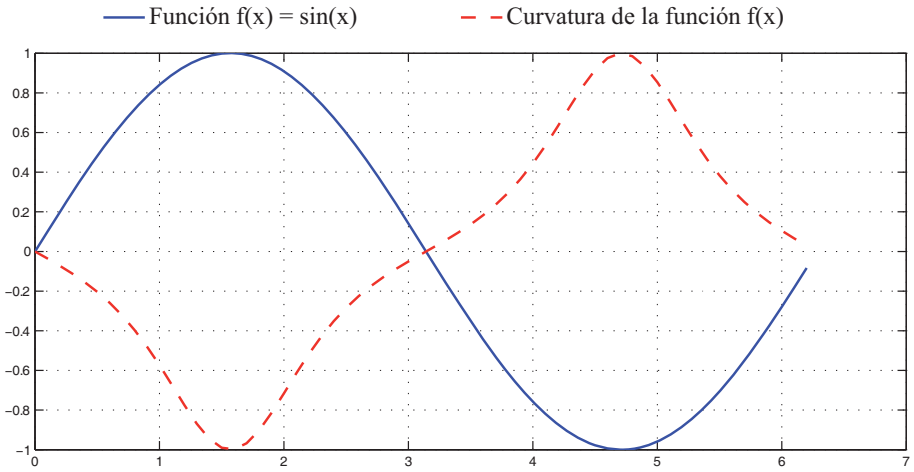


Figura 3.10: Representación de la función $y = \sin(x)$ y de su curvatura, $\kappa = \frac{-\sin(x)}{[1+\cos^2(x)]^{(3/2)}}$

3.4. Definición de B-spline

Sea $\mathbf{s}(t)$ el vector de posición a lo largo de una curva en el espacio $\Re^n$ expresado como función del parámetro real t. Una curva spline de orden κ (grado $\kappa - 1$), con puntos de control $\mathbf{x}_i \in \Re^n, (i = 0 \ldots n)$ y vector de nodos $\Xi = \{\xi_0, \ldots, \xi_{n+k}\} \subset \Re$ se puede expresar como:

$$\mathbf{s}(t) = \sum_{i=0}^{n} \mathbf{x}_i \beta_{i,\kappa}(t) \tag{3.9}$$

donde cada uno de los términos $\beta_{i,\kappa}(t)$ es una función base B-spline normalizada de orden κ. Cada una de estas funciones base puede ser calculada empleando las fórmulas recursivas de Cox-de Boor [53, 159]:

$$\beta_{i,1}(t) = \begin{cases} 1 & \xi_i \leq t < \xi_{i+1} \\ 0 & \text{en otro caso} \end{cases} \tag{3.10}$$

y

$$\beta_{i,\kappa}(t) = \frac{(t - \xi_i)}{\xi_{i+\kappa-1} - \xi_i} \beta_{i,\kappa-1}(t) + \frac{(\xi_{i+\kappa} - t)}{\xi_{i+\kappa} - \xi_{i+1}} \beta_{i+1,\kappa-1}(t) \tag{3.11}$$

El vector de nodos es cualquier secuencia no decreciente de números reales ($\xi_i \leq \xi_{i+1}$ para $i = 0, \ldots, n + \kappa$), y su misión fundamental es generar el conjunto de funciones de base que darán origen a la curva B-spline. Sus valores definen sobre la curva spline las posiciones en las que dos tramos polinomiales de orden κ se unen, y cuya concatenación genera la curva global representada. Nótese que la única condición es que la secuencia de nodos sea no decreciente, pudiendo existir nodos múltiples en cualquier punto de la cadena. Lo mismo sucede con los puntos de control, cuyo conjunto ordenado define el polígono de control de la curva: no se exige para ellos ninguna condición especial, pudiendo presentar cualquier configuración espacial imaginable.

La figura 3.11 muestra cuatro ejemplos con distintas configuraciones tanto del vector de nodos como del polígono de control. En las dos gráficas inferiores puede apreciarse cuan sencilla es la representación de esquinas en el interior de la curva, sin más que incrementar suficientemente la multiplicidad de un nodo interior (figura 3.11.c), o la multiplicidad de un vértice del polígono de control (figura 3.11.d).

Cabe realizar las siguientes consideraciones respecto a la anterior definición de curva B-spline:

- En la ecuación 3.11 se adopta la convención $0/0 = 0$ para evitar singularidades.
- El cómputo del conjunto de funciones básicas requiere de la definición de un vector de nodos y del orden κ de la curva.

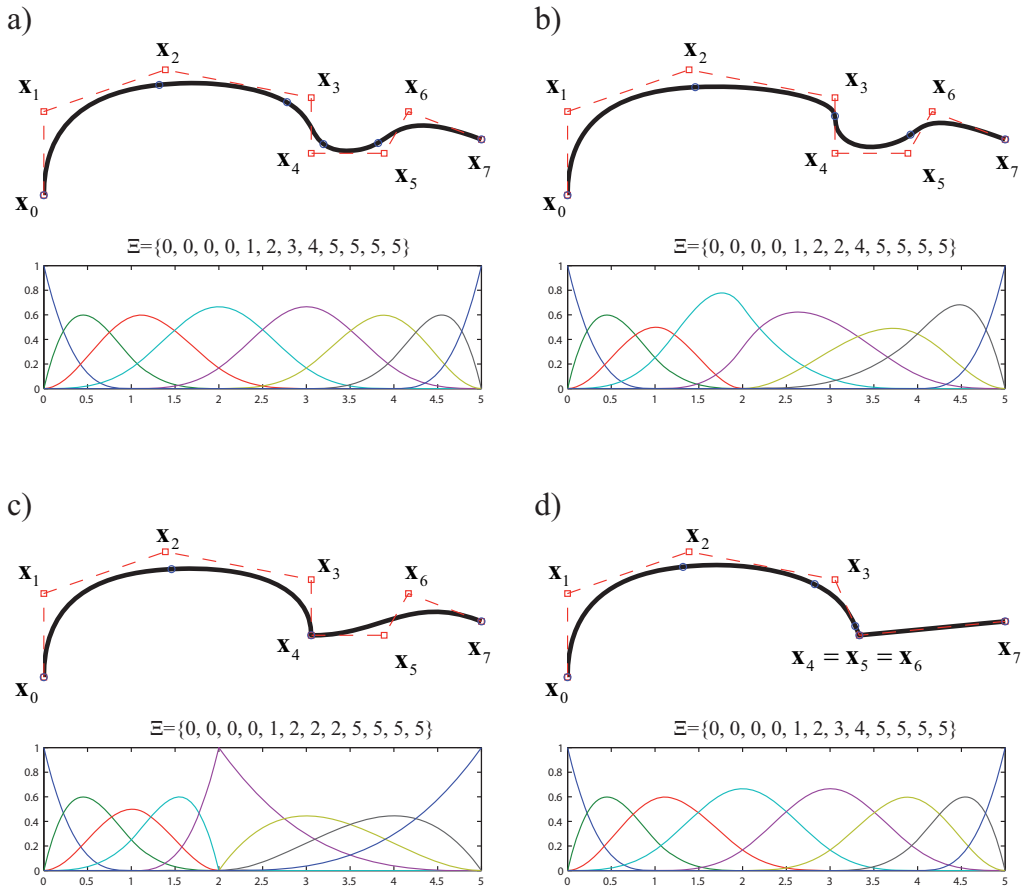


Figura 3.11: Cuatro ejemplos de curvas spline, con diferentes configuraciones del vector de nodos y del polígono de control.

- Cada una de las funciones básicas de orden 1 , $\beta_{i,1}(t)$, es una función escalón, idénticamente igual a cero excepto en el intervalo abierto $t \in [\xi_i, \xi_{i+1})$.
- Cada uno de los intervalos semiabiertos $[\xi_i, \xi_{i+1})$ es conocido como i-ésimo tramo nodal, y puede tener longitud nula dado que no es preciso que todos los nodos sean diferentes. Este hecho se aprecia en las figuras 3.11.b y 3.11.c.
- Para cualquier orden $\kappa > 1$, cada una de las funciones básicas $\beta_{i,\kappa}(t)$ se obtiene como combinación lineal de dos funciones básicas de orden $\kappa - 1$. Esta dependencia funcional se suele representar esquemáticamente mediante tablas triangulares truncadas como la que puede observarse en la figura 3.12.
- Según se observa en la ecuación (3.9), cada función básica tiene asociado un punto de control, que define su peso o importancia en el conjunto de la curva.
- La diferencia entre el número de puntos de control y el número de nodos es siempre igual al orden κ de la curva. Así, una curva con $n+1$ puntos de control $\mathbf{x}_i, i = 0, \ldots, n$ tendrá asociado siempre un vector de nodos de $n+\kappa+1$ elementos $\xi_i, i = 0, \ldots, n+\kappa$

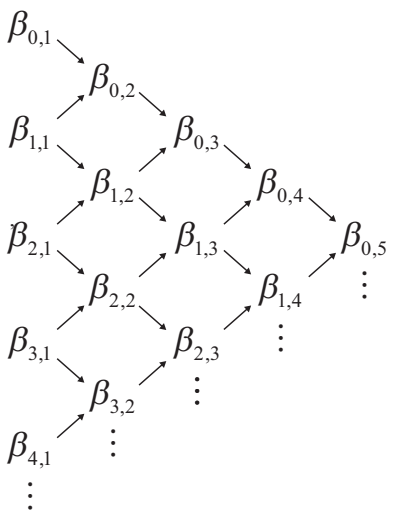


Figura 3.12: Representación esquemática de las relaciones funcionales entre funciones básicas B-spline de órdenes crecientes.

3.5. El Vector de Nodos

Ya se ha comentado que el vector de nodos es cualquier secuencia no decreciente de valores reales, que es empleada para generar el conjunto de funciones básicas B-spline, cuya combinación lineal dará lugar a la curva final. Los elementos de esta serie definen las localizaciones sobre la curva en la que se unen dos tramos polinomiales diferentes. Dicho vector de nodos puede ser definido de múltiples maneras, y la elección de su configuración determinará en gran medida la apariencia y propiedades de la curva a la que da origen.

Existen diferentes clasificaciones y nomenclaturas para referirse a los vectores de nodos según su tipología. El objetivo de esta tesis no es realizar una disquisición profunda sobre la casuística de este elemento generador de las curvas spline, sino más bien mantener las herramientas en un nivel lo más básico posible, que permita alcanzar de una manera cómoda el objetivo final de aplicar las curvas paramétricas definidas por tramos polinomiales al complejo problema del modelado de entornos. Sin embargo, se considera importante realizar un breve repaso por las posibilidades que las diferentes versiones de este elemento ofrecen, justificando los motivos por los cuáles se ha elegido una modalidad concreta para ser empleada en futuros desarrollos. Al mismo tiempo, la presentación de otras alternativas existentes, servirá al lector interesado para hacerse una idea de las múltiples posibilidades, ventajas e inconvenientes, que presenta el empleo de esta nueva manera de modelar geometrías.

En una primera aproximación, se pueden establecer dos clasificaciones binarias, no excluyentes entre sí, atendiendo a dos características fundamentales del vector de nodos: la multiplicidad de los nodos extremos, y el espaciado internodal.

La **multiplicidad de los nodos extremos** se refiere al número de veces que se repiten los valores del primer nodo y del último dentro del vector. Esta característica permite establecer una distinción entre los siguientes dos grupos:

1. Vectores de nodos **enclavados**. En ellos la multiplicidad de los nodos extremos es igual al orden κ de la curva. Así, para un spline de orden κ definido por un polígono de control con $n+1$ vértices el vector de nodos enclavado tendrá la siguiente forma:

$$\Xi_e = \{\xi_0 = \ldots = \xi_{\kappa-1} \leq \xi_\kappa \leq \ldots \leq \xi_n \leq \xi_{n+1} = \ldots = \xi_{n+\kappa}\} \tag{3.12}$$

2. Vectores de nodos **desenclavados**. Son todos aquellos que no verifican la anterior condición.

En esta tesis se han escogido los términos *enclavado/desenclavado* como las traducciones más apropiadas de los términos ingleses empleados para referirse a las mismas categorías: *clamped/unclamped*. En inglés, *clamped* significa mordaza o tornillo de banco; aquel dispositivo metálico o de madera empleado para sujetar dos elementos juntos de manera firme. Así, un vector de nodos *clamped* sería aquel que está sujeto o afianzado mediante mordazas.

La explicación de esta curiosa denominación es fácil de entender observando las figuras 3.13 y 3.14. En la primera de ellas se observan cuatro ejemplos de curvas construidas empleando vectores de nodos con multiplicidades de los nodos extremos iguales al orden de la curva. Se observa que los extremos de la curva obtenida coinciden exactamente con el primer y último vértices del polígono de control. Los extremos quedan *sujetos*, *amordazados* o *enclavados* precisamente en esos lugares. Esto no sucede cuando empleamos otros vectores de nodos, como se aprecia en la figura 3.14, que muestra distintos ejemplos de curvas con vectores de nodos desenclavados.

La segunda clasificación atiende, como se ha avanzado, al espaciado entre cada dos nodos consecutivos. Así, dependiendo de si esta distancia es constante o no, se establece la siguiente clasificación para los vectores de nodos:

1. Vectores de nodos **uniformes**, cuando el espaciado entre cada dos nodos consecutivos es constante. En el caso de vectores de nodos enclavados, como es obvio, se exigirá esta característica sólo en los nodos interiores del vector: $\xi_i, \kappa - 1 \leq i \leq n + 1$.
2. Vectores de nodos **no uniformes**, cuando el espaciado internodal no es constante.

Las diferentes combinaciones de estas clasificaciones binarias, darán lugar a una variedad de configuraciones que se corresponden con los vectores de nodos más frecuentemente empleados en la práctica. A continuación se muestran algunos ejemplos:

- enclavado uniforme para curva de orden 4:

$$\{\ 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 4 \quad 4 \quad 4\ \}$$

- enclavado no uniforme para curva de orden 3:

$$\{\ -1 \quad -1 \quad -1 \quad 0 \quad 1{,}2 \quad 1{,}5 \quad 3 \quad 3 \quad 3\ \}$$

- desenclavado uniforme:

$$\{\ -1 \quad 0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 7 \quad 8\ \}$$

- desenclavado no uniforme:

$$\{\ 0 \quad 0 \quad 1 \quad 2 \quad 3{,}5 \quad 4 \quad 4{,}5 \quad 5 \quad 8\ \}$$

Las figuras 3.13 y 3.14 ilustran gráficamente el amplio escenario de posibilidades al que nos enfrentamos. La figura 3.13 muestra cuatro ejemplos de curvas con vectores de nodos enclavados. En la columna izquierda se muestran dos splines cúbicos (orden $\kappa = 4$), en las dos variedades posibles (vector de nodos uniforme y no uniforme). Lo mismo sucede en la columna derecha, en este caso con splines cuadráticos (orden $\kappa = 3$). La figura 3.14 muestra ejemplos similares, pero empleando en este caso vectores de nodos cuya multiplicidad de nodos terminales es diferente al orden de la curva representada en cada caso. En todos los casos, acompañan a la representación de cada curva el conjunto de funciones básicas que las generan.

La distinción entre vectores de nodos enclavados y no enclavados es bastante reciente históricamente [148]. En los comienzos del desarrollo de los B-splines (antes de la década de 1980), el uso de vectores de nodos no uniformes no estaba aún generalizado. En aquellos tiempos, era común establecer una clasificación de los vectores de nodos entre **periódicos** y **no periódicos**. Decir que un vector de nodos era no periódico era equivalente a hablar de un vector de nodos enclavado, y este

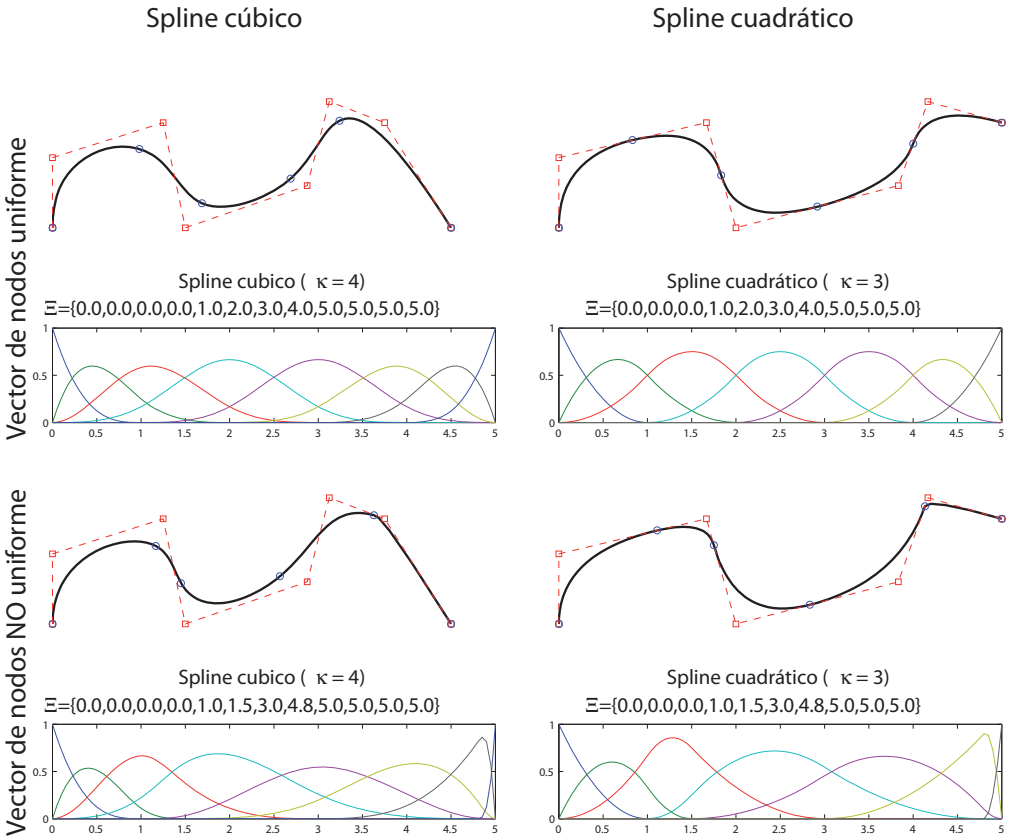


Figura 3.13: Vectores de nodos enclavado.

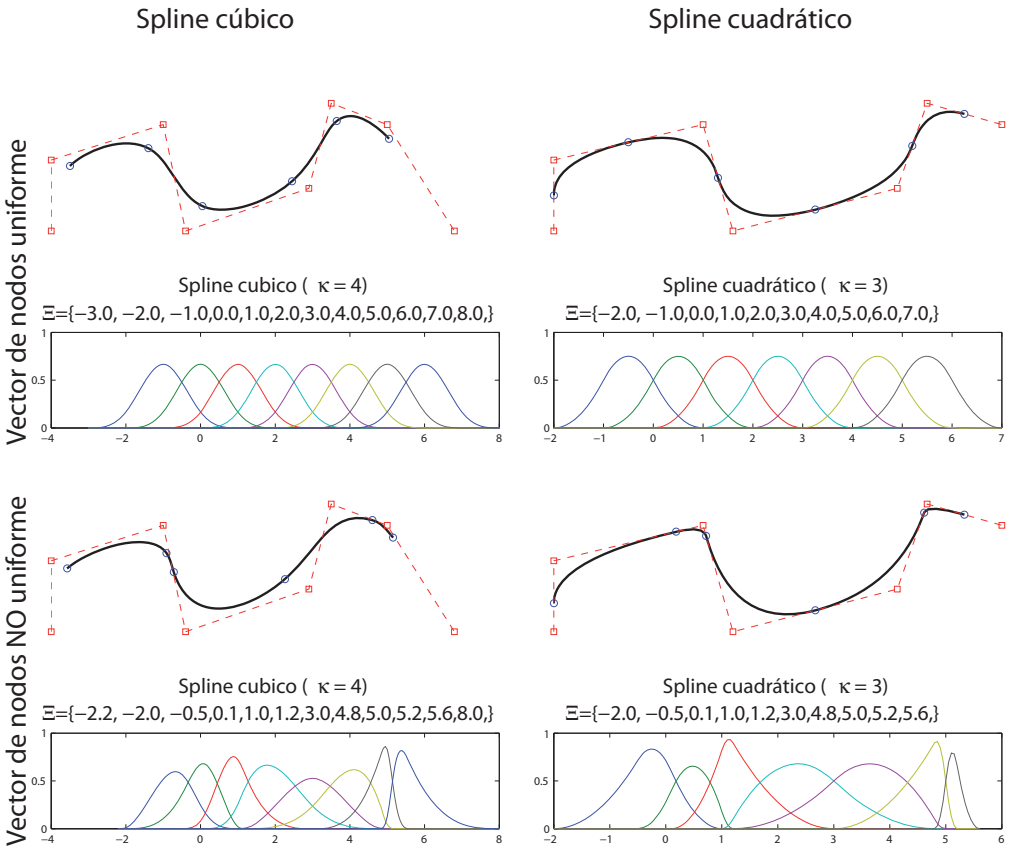


Figura 3.14: Vectores de nodos desenclavado.

tipo era usado con frecuencia para generar curvas y superficies abiertas. Por esta razón, a los vectores de nodos no periódicos (enclavados) también se les llamaba **abiertos**. Por otra parte, los vectores de nodos periódicos eran lo que aquí hemos denominado vectores de nodos no enclavados. Como puede observarse en la figura 3.14, los vectores de nodos desenclavados (o periódicos) uniformes, dan origen a funciones básicas que son traslaciones unas respecto de otras. De ahí el origen del término. Al extenderse el uso de vectores de nodos no uniformes, la denominación se mantuvo.

Por otra parte, cabe resaltar el hecho de que no es preciso el empleo de una configuración específica a la hora de definir curvas cerradas. En la figura 3.15 se presentan dos ejemplos de splines cúbicos cerrados. En la figura 3.15.a se ha escogido utilizar un vector de nodos desenclavado que da origen a un conjunto de funciones básicas periódicas. En este caso, la obtención de una curva cerrada es posible sin más que hacer coincidir los $\kappa - 1$ primeros puntos de control con los $\kappa - 1$ últimos, tal y como se observa, de manera que el polígono de control se envuelve de alguna manera alrededor de la curva.

Del mismo modo, en la figura 3.15.b se muestra cómo es posible conseguir un resultado similar empleando un vector de nodos enclavado. En este caso, la continuidad geométrica se consigue sin más que hacer coincidir el primer punto de control con el último ya que, como sabemos, estos puntos coinciden para este tipo de vector de nodos con los puntos inicial y final de la curva. Para conseguir continuidad de la primera derivada y sucesivas en el punto de cierre de la curva, es preciso imponer condiciones adicionales a la localización de otros vértices del polígono de control.

3.6. Propiedades de los B-splines

Por su especial importancia en el desarrollo de futuros algoritmos presentados en esta tesis, a continuación se enumeran algunas de las propiedades que exhiben las curvas spline cuando son formuladas como combinación lineal de funciones base B-spline.

1. El máximo orden de un spline es igual al número de puntos con forman su polígono de control.

2. La forma general de la curva sigue la geometría definida por el polígono de control.

3. Cualquier transformación afín se puede aplicar sobre la curva aplicándola a los puntos que conforman el polígono de control.

4. Cada función base $\beta_{i,\kappa}(t)$ es una función polinomial por tramos de orden κ, con puntos de unión definidos por los valores del parámetro $\xi_i, \ldots, \xi_{i+\kappa}$.

5. Propiedad de soporte local: Cada función base $\beta_{i,\kappa}(t)$ se anula fuera del intervalo $[\xi_i, \xi_{i+\kappa})$ y es positiva en el interior de dicho intervalo. Esta propiedad es

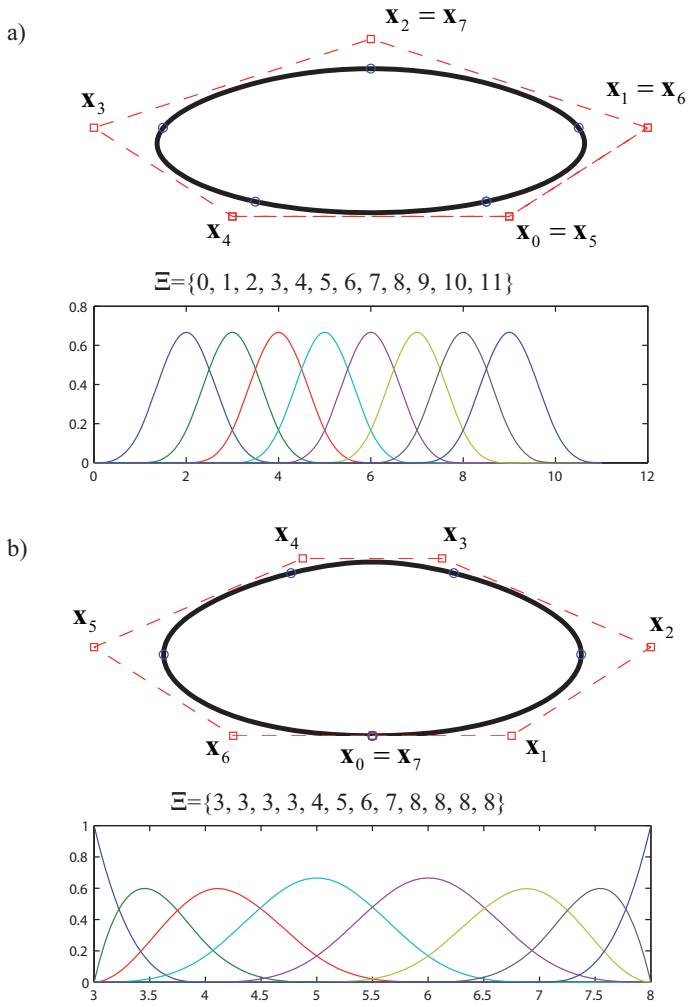


Figura 3.15: Ejemplos de curvas B-spline cerradas. **a)** Spline cúbico cerrado con vector de nodos desenclavado, o periódico. **b)** Spline cúbico con vector de nodos enclavado, o abierto.

fácil de comprender mirando a la figura 3.12, y recordando que cada función básica $\beta_{i,1}(t)$ de orden 1 se anula fuera del intervalo $[\xi_i, \xi_{i+1})$.

$$\beta_{i,\kappa}(t) > 0 \quad \forall \quad \xi_i \leq t < \xi_{i+\kappa} \tag{3.13}$$

6. Como consecuencia de la propiedad 4, el valor de $\mathbf{s}(t)$ en una posición determinada por el valor del parámetro $\xi_j \leq t \leq \xi_{j+1}$ para algún $j \in \{\kappa - 1, \ldots, n\}$ depende sólo de κ coeficientes:

$$\mathbf{s}(t) = \sum_{i=j-\kappa+1}^{j} \mathbf{x}_i \beta_{i,\kappa}(t) \tag{3.14}$$

7. Partición de la unidad: la suma de las funciones base para cualquier valor del parámetro $\xi_{\kappa-1} \leq t \leq \xi_{n+1}$ es 1. En el caso de vectores de nodos enclavados, esta propiedad se verifica para cualquier valor posible del parámetro $\xi_0 \leq t \leq \xi_{n+\kappa}$:

$$\sum_{i=0}^{n} \beta_{i,\kappa}(t) = 1 \tag{3.15}$$

8. La derivada de un spline de orden κ es otro spline de un orden inferior $\kappa - 1$, cuyos coeficientes pueden ser obtenidos diferenciando los originales [53].

$$\frac{d\mathbf{s}(t)}{dt} = \mathbf{s}'(t) = (\kappa - 1) \sum_{i=0}^{n-1} \frac{\mathbf{x}_{i+1} - \mathbf{x}_i}{\xi_{i+\kappa} - \xi_{i+1}} \beta_{i,\kappa-1}(t) \tag{3.16}$$

9. Propiedad de la envolvente convexa. Para una curva B-spline de orden κ, esta propiedad establece que el tramo de curva comprendido entre dos nodos consecutivos $[\xi_i, \xi_{i+1})$ permanece confinado en el interior del polígono convexo formado por κ puntos de control consecutivos: $\mathbf{x}_{i-\kappa+1}, \ldots, \mathbf{x}_i$. Dicha propiedad aparece ilustrada en la figura 3.16.

Para más información y justificación de las anteriores propiedades, mírese [2, 53, 148, 159].

3.7. Aproximación Mediante B-splines

En este apartado se presenta un método para aproximar mediante B-splines colecciones de puntos obtenidos mediante el muestreo sensorial de los contornos presentes en el entorno. La representación obtenida debería ser de formas suaves, pero preservando las características significativas presentes en los datos obtenidos.

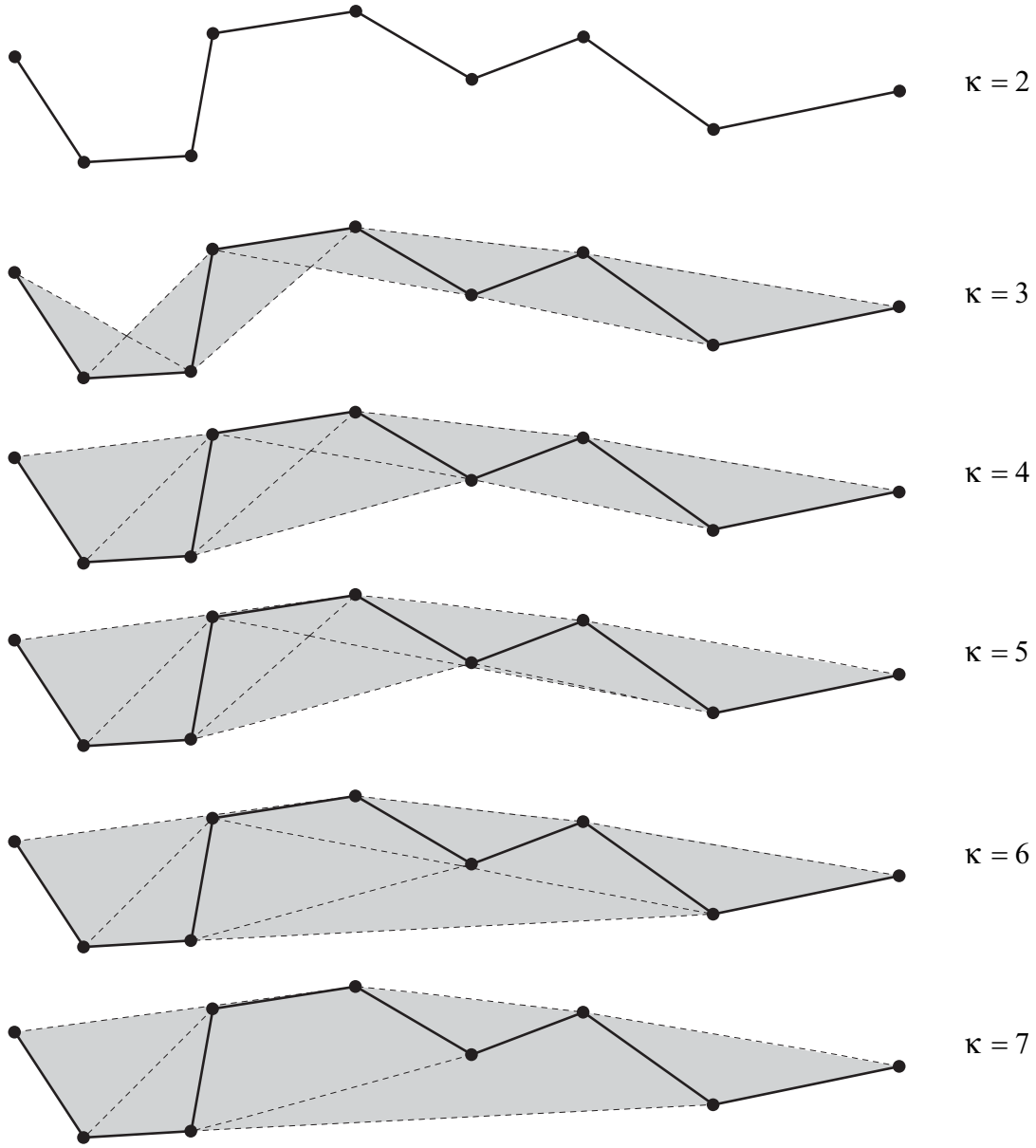


Figura 3.16: Propiedad de envolvente convexa. Para cada orden de la curva spline que define el polígono de control, la totalidad de la curva queda confinada en el interior de la zona sombreada.

Dada la gran cantidad de datos obtenida, en esta y otras muchas aplicaciones, la simple interpolación de los datos no es en general de utilidad.

Los métodos más comunes se basan en la minimización, por el método de mínimos cuadrados, de las distancias de los puntos de partida a la función spline de aproximación, sujeta a las condiciones de suavidad en el trazado de la representación. El método propuesto en esta tesis, ampliamente utilizado, a pesar de su simplicidad proporciona resultados excelentes. Además presenta la ventaja adicional de que proporciona una relación funcional lineal entre los datos brutos y los parámetros de la curva obtenida.

3.7.1. ¿Por qué aproximar?

El problema que nos ocupa es el de aproximar un conjunto de puntos, obtenidos por un sensor como muestras de un perfil o contorno físico de forma arbitraria y desconocida, mediante una curva o función, manejable de manera eficiente por un computador, y constituida por formas geométricas sencillas. En este caso utilizaremos un spline o curva polinomial por tramos. La representación de estos contornos mediante el conjunto de puntos obtenidos no es viable por diversos motivos:

- El número de datos es en ocasiones demasiado elevado para su procesamiento en tiempo real. Así, es conveniente reducir el número de puntos significativos, pero preservando todas las características geométricas proporcionadas por la muestra original.
- La representación mediante expresiones funcionales es a menudo más apta para procesamientos ulteriores.
- Las medidas obtenidas presentan generalmente una cierta cantidad de ruido, obteniéndose contornos que no se corresponden completamente con la realidad.

Cuando el número de puntos es pequeño, son preferibles los métodos de interpolación, mientras que, cuando el número de puntos es más grande o los datos presentan ruido (inherente a todo proceso de medida con un sensor real) los métodos de aproximación proporcionan mejores resultados.

3.7.2. Splines de aproximación

El método de representación mediante una función spline sustituye el conjunto de puntos de partida mediante una curva descrita matemáticamente por un spline. Como ya se ha comentado, un spline es una función polinomial a tramos, con ciertas condiciones de continuidad impuestas en las uniones entre los diferentes polinomios. Los métodos de aproximación dividen la muestra inicial en un número suficiente de intervalos. En cada intervalo, los puntos son aproximados por un polinomio. En aplicaciones típicas basta con utilizar splines de segundo o tercer grado [53]. Grados

más elevados sólo añaden complejidad computacional. La exactitud de la representación puede ser caracterizada por el error de aproximación, que viene dado por la distancia media de los puntos a la función aproximadora. En [152], por ejemplo, se utiliza otro criterio para cuantificar la bondad de la aproximación: el número de puntos de inflexión en la curva resultante.

Para poder realizar la aproximación es preciso resolver las siguientes tareas:

- Definir un número suficiente de intervalos.
- Calcular los valores de los parámetros que definen el spline.
- Estimar el error de la aproximación propuesta.

Muchos investigadores han realizado trabajos dedicados a buscar la mejor manera de aproximar tanto una función, como un conjunto de datos muestreados, utilizando estas interesantes curvas que son los splines. Algunos métodos mantienen el vector de nodos como un elemento constante, mientras que otros permiten cierta flexibilidad en su estructura, adaptándola para lograr el mejor ajuste posible. La mayoría de estos métodos son iterativos. Para una comprensión más detallada del tema se pueden consultar [54, 55, 83, 111, 150].

En esta tesis, se ha escogido utilizar el método de aproximación más simple, que permite obtener una relación lineal entre los datos de partida y los puntos de control de la curva aproximadora. El método se detalla en el siguiente apartado.

3.7.3. Ajuste de datos con curvas spline

En esta sección se considera el problema de obtener una curva spline que aproxime un conjunto de datos $\mathbf{d}_j$, $j = 0 \ldots m$. Si cada uno de estos puntos pertenece a la misma curva, entonces se ha de satisfacer la ecuación 3.9:

$$\mathbf{d}_j = \beta_{0,k}(t_j)\,\mathbf{x}_0 + \ldots + \beta_{n,k}(t_j)\,\mathbf{x}_n, \quad j = 0 \ldots m$$

Este sistema de ecuaciones se puede escribir de manera más compacta como:

$$\mathbf{d} = \mathbf{B}\mathbf{x} \tag{3.17}$$

con

$$\begin{cases} \mathbf{d} = \begin{bmatrix} \mathbf{d}_0 & \mathbf{d}_1 & \ldots & \mathbf{d}_m \end{bmatrix}^T \\ \mathbf{x} = \begin{bmatrix} \mathbf{x}_0 & \mathbf{x}_1 & \ldots & \mathbf{x}_n \end{bmatrix}^T \\ \mathbf{B} = \begin{bmatrix} \beta_{0,k}(t_0) & \ldots & \beta_{n,k}(t_0) \\ \vdots & \ddots & \vdots \\ \beta_{0,k}(t_m) & \ldots & \beta_{n,k}(t_m) \end{bmatrix} \end{cases} \tag{3.18}$$

La matriz $\mathbf{B}$, generalmente conocida como *matriz de colocación*, tiene por cada una de sus filas a lo sumo κ elementos no nulos. El valor del parámetro t_j define la posición de cada uno de los puntos $\mathbf{d}_j$ a lo largo de la curva spline. Este valor puede ser aproximado por el valor de la suma acumulada de las cuerdas que unen dos puntos consecutivos:

$$\left.\begin{array}{rcl} t_0 & = & 0 \\ t_j & = & \sum_{s=1}^{j} |\mathbf{d}_s - \mathbf{d}_{s-1}|, \quad j \geq 1 \end{array}\right\} \tag{3.19}$$

siendo la longitud total de la curva

$$\ell = \sum_{s=1}^{m} |\mathbf{d}_s - \mathbf{d}_{s-1}| \tag{3.20}$$

que es tomado como el valor máximo del vector de nodos.

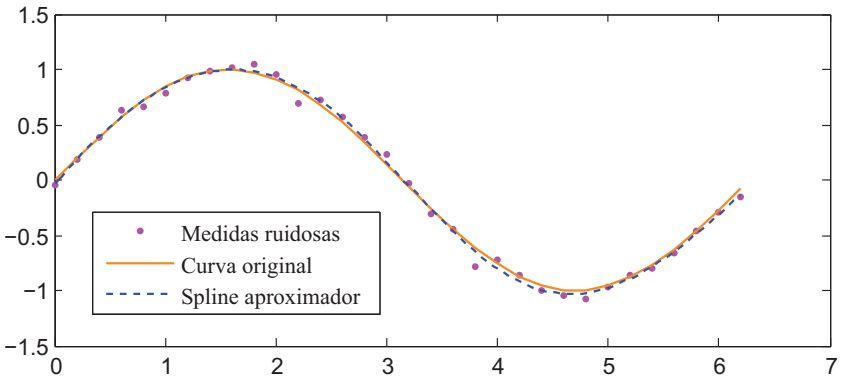
El caso más general sucede cuando $2 \leq k \leq n+1 < m+1$; el problema está sobredeterminado y sólo puede resolverse de una manera aproximada. Se puede obtener la solución de mínimos cuadrados utilizando el método de la psuedo inversa de $\mathbf{B}$:

$$\mathbf{x} = \left[\mathbf{B}^T\mathbf{B}\right]^{-1}\mathbf{B}^T\mathbf{d} = \mathbf{\Phi}\mathbf{d} \tag{3.21}$$

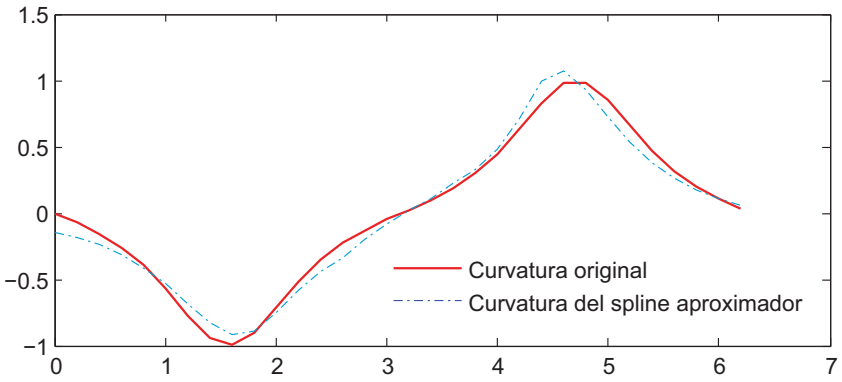
Si el orden de las funciones B-spline κ está predefinido, el número de puntos de control $n+1$ y los valores del parámetro para cada uno de los datos es conocido (calculado utilizando la ecuación 3.19), las funciones base $\beta_{i,k}(t_j)$ y, por lo tanto, la matriz $\mathbf{B}$, pueden obtenerse fácilmente.

En la figura 3.17 se muestra un ejemplo de cómo una función dada se puede aproximar mediante un spline a partir de un conjunto de muestras afectadas por cierta cantidad de ruido, junto con una comparación entre las curvaturas de la función original y de la función que la aproxima.

En el trabajo presentado en esta tesis, se utilizan vectores de nodos enclavados, generados tomando la longitud total de la curva ℓ (ecuación 3.20), y definiendo una densidad de nodos que depende de la complejidad del entorno. Recuérdese que los nodos definen los puntos en los que dos tramos polinomiales toman contacto, de manera que entornos complejos precisarán de una densidad de nodos más elevada (tramos polinomiales más cortos), mientras que tramos más parecidos a segmentos podrán ser apropiadamente representados utilizando tramos polinomiales de mayor longitud. El lector interesado puede consultar en [7] y [54] más información sobre métodos de ajuste mediante curvas spline.



(a)



(b)

Figura 3.17: Ejemplo de aproximacion de datos mediante curvas spline. (a) Aproximación de la función $y = sin\,(x)$ sobre el intervalo $[0, 2\pi]$ a partir de un conjunto de medidas tomadas cada 0,2 unidades sobre el eje de abscisas y afectadas por un ruido de distribución normal de media 0 y varianza $\sigma^2 = 0{,}005$. Para la aproximación se ha utilizado un spline cúbico. (b) Comparación entre las curvaturas de la función original y del spline aproximador obtenido a partir de las medidas ruidosas.

3.8. Conclusiones

En este capítulo se ha presentado una breve introducción a los orígenes históricos, conceptos fundamentales y formulaciones matemáticas relacionados con la tecnología, utilización y aplicaciones de las curvas B-spline.

El somero repaso histórico ha servido para constatar cómo la necesidad del disponer de métodos de diseño, representación y fabricación de geometrías arbitrariamente complejas ha sido una constante a lo largo de la historia de la humanidad. Ingeniosos artilugios mecánicos primero, y sofisticadas representaciones matemáticas más tarde, han permitido al hombre materializar ideas y geometrías que de otro modo sólo habrían tenido cabida en la mente de sus creadores. En la cúspide de las tecnologías desarrolladas en torno a las curvas spline y sus aplicaciones se encuentra su implementación mediante sofisticados programas para computador, que han dado origen a las modernas disciplinas de diseño y fabricación asistidas por computador.

También se ha visto cómo estas técnicas se han comenzado a aplicar recientemente al diseño de edificios y esculturas. No es descabellado por tanto intentar utilizar las curvas polinomiales por tramos en la descripción de estos elementos en un mapa. Del mismo modo que los splines sirvieron para generar la realidad física, se pretende ahora utilizarlos para modelarla.

Se ha mostrado cómo la representación de geometrías basada en la concatenación de tramos polinomiales resulta una solución idónea al problema de aproximar o modelar contornos y morfologías de complejidad creciente. también se ha visto cómo es posible describir tales curvas mediante la combinación lineal de un tipo específico funciones básicas B-spline. En concreto, la descripción matemática de la realidad física mediante la utilización de curvas B-spline reduce el problema a la generación de un vector de nodos que engendra el conjunto de funciones básicas utilizadas, y un conjunto de puntos de control. Estos puntos dan una idea general de la apariencia de la curva a la que dan lugar, y por otro lado son empleados para combinar linealmente las funciones base de modo que cualquier posición sobre la curva puede ser calculada eficientemente de manera paramétrica. Más aún, es posible dotar de significado al parámetro independiente de modo que venga a representar, por ejemplo, la longitud medida sobre la propia curva.

La formulación de curvas spline como combinación lineal de funciones básicas presenta una serie de interesantes propiedades. Por ejemplo, la propiedad de soporte local hace que la modificación de cada punto de control afecte sólo localmente a la geometría de la curva. Esto es interesante si se quiere obtener un refinamiento local de su apariencia, sin que este cambio afecte a la curva globalmente, lo cual es atractivo en la tarea de modelado que nos ocupa.

Del mismo modo que las curvas spline son de utilidad para materializar, o trasladar al mundo físico, conceptos surgidos de la mente de un diseñador, arquitecto o ingeniero, se ha mostrado cómo su potencia puede ser aplicada también en el proceso inverso: el de describir analíticamente y trasladar a lenguaje matemático realidades físicas, señales o cualquier otra magnitud mensurable, permitiendo su posterior interpretación y análisis. Es este segundo aspecto el que nos interesará en esta tesis.

Se mostrará cómo es posible expandir el ámbito de aplicación de estas tecnologías que vienen siendo desarrolladas y estudiadas desde hace medio siglo, y fusionarlas con modernas técnicas de localización y modelado simultáneos que vienen siendo empleadas en el campo de la robótica móvil.

Así pues, en capítulos sucesivos se mostrará cómo los conceptos, matemáticas y propiedades de las curvas B-spline presentados en este capítulo encontrarán utilidad práctica y ensamblaje matemático con las técnicas de generación de mapas geométricos mediante filtro extendido de Kalman en entornos geométricamente complejos. ¿Qué entendemos aquí por complejo? Algo tan simple (o no, según se mire) como el hecho de que no será preciso que los entornos físicos en los que se desenvuelve en robot contengan una geometría predefinida a ser detectada, tal y como sucede en métodos tradicionales de SLAM geométrico.

Será posible representar contornos tan alejados de la mera linealidad de un segmento como sea necesario, pero sin tener que limitar la representación matemática de las formas encontradas a entidades fácilmente parametrizables, tales como simples circunferencias o secciones cónicas. Se abordará el problema fundamental de representar el entorno analíticamente, pero de forma compacta; tal y como es detectado por los sensores del robot.

Capítulo 4

SLAM con Modelado Basado en Curvas Spline

4.1. Introducción

En este capítulo se presenta una metodología que combina exitosamente los siguientes dos elementos:

- La solución al problema del SLAM basada en el filtro extendido de Kalman: SLAM-EKF.
- La teoría de las curvas spline como herramienta capaz de modelar geometrías complejas.

Así, en este capítulo se desarrollarán los razonamientos, conceptos y algoritmos necesarios para lograr una simbiosis entre los dos mencionados marcos teóricos que permitirá (y este es el objetivo fundamental de esta tesis) la construcción de mapas en los que todas las entidades son modeladas mediante curvas spline.

Más concretamente, se presentan los siguientes resultados:

En la sección 4.2 se presenta la forma de operar para, partiendo de la información bruta suministrada por un sensor típico en robótica móvil como es el escáner láser, obtener:

- el conjunto de objetos que se ocultan tras dichas medidas,
- el conjunto de curvas spline que modelan los objetos del entorno y
- la manera de realizar una correcta asociación entre las entidades detectadas y las contenidas en el mapa.

En la sección 4.3 se presenta el modelo de estado utilizado en el marco de trabajo propuesto. Como se verá, el estado del sistema vendrá dado por la información

geométrica que caracteriza al robot y al mapa, y la información estocástica que relaciona espacialmente todas estas variables.

En la sección 4.4 se explica minuciosamente el modelo de observación propuesto. Este modelo permitirá la eficiente aplicación de un filtro extendido de Kalman para estimar el estado del sistema bajo estudio. Se comentan los principales inconvenientes de la utilización de curvas spline a la hora de ser empleadas en en el marco de trabajo SLAM-EKF, y se muestra cómo estos inconvenientes pueden ser ingeniosamente solventados.

La sección 4.5 combina los resultados obtenidos hasta este punto del capítulo, presentando el conjunto de algoritmos matemáticos que permiten la obtención de mapas geométricos de entornos complejos, modelados mediante curvas spline, donde la estimación de la pose del robot y las posiciones de los puntos de control que definen la geometría de los elementos del mapa son estimados utilizando un filtro extendido de Kalman.

Finalmente, la sección 4.6 presenta los razonamientos y la algorítmica necesarios para:

- inicializar nuevos objetos en el mapa y
- extender objetos ya contenidos en el mapa; esto es, inicializar puntos de control adicionales que extienden geometrías ya presentes en el vector de estado del sistema.

Todos estos resultados constituyen aportaciones originales de esta tesis. Juntos, permiten por primera vez modelar geometrías complejas en mapas formados exclusivamente por curvas paramétricas.

Esta forma de modelar geométricamente los elementos que conforman el entorno de un robot móvil constituye una gran novedad dentro del campo del SLAM, extendiendo significativamente resultados anteriores basados en características geométricas más simples como puntos o segmentos. Se entiende por tanto la originalidad de los resultados presentados, y la interesante aportación que constituyen dentro del área de conocimiento del modelado de entornos complejos mediante la utilización de robots móviles.

4.2. Gestión de los Datos del Láser

El sensor exteroceptivo más profusamente empleado en robótica móvil, por sus características de buena precisión y velocidad en la adquisición de datos, es el escáner láser. Cuando el robot realiza una observación de su entorno con un dispositivo de este tipo, se obtiene un conjunto de m puntos $\mathbf{d}_i \in \Re^2$ (consideraremos sólo el escenario bidimensional), tal y como puede observarse en la figura 4.1.

En esta sección, se describen métodos para extraer un conjunto de splines paramétricos que representen de manera precisa el mundo físico que rodea al robot.

También se muestra la manera de realizar la asociación de datos, estableciendo correspondencias entre los splines detectados y los que ya se encuentran contenidos en el mapa.

4.2.1. Segmentación de los datos del láser

Cuando el robot obtiene un nuevo conjunto de medidas de su entorno a través de un sensor como es el escáner láser, el mundo que le rodea tiene para él el mismo sentido que un conjunto de puntos desconectados, y ligados entre sí por la única lógica de la mera ordenación que proporciona el sensor al barrer su entorno (ver figura 4.1). Es nuestro objetivo ahora obtener un conjunto de splines que representen la realidad física observada. Para ello será necesario lograr los siguientes objetivos:

- Diferenciar claramente a qué objeto pertenece cada una de las medidas obtenidas, agrupándolas en subconjuntos.
- Obtener la curva spline que represente la porción detectada para cada objeto de la manera más aproximada posible.

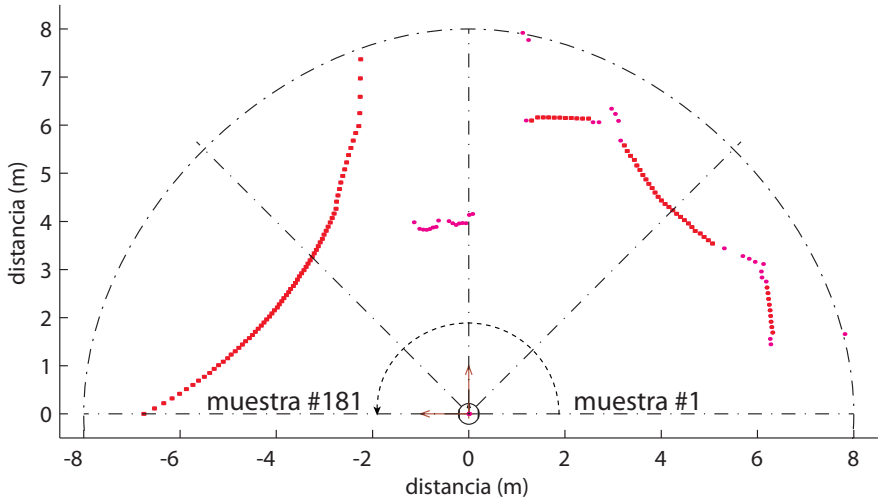


Figura 4.1: Ejemplo del conjunto de datos adquiridos por un escáner láser en un experimento real.

Al primero de estos procedimientos lo llamaremos **segmentación**. Dado el conjunto bruto de m medidas suministradas por el sensor (en la figura 4.1 $m = 181$), el objetivo es agrupar aquellos puntos que pertenecen al mismo objeto del entorno.

La principal diferencia entre el método presentado en esta tesis, y algoritmos tradicionales de SLAM geométrico, es que ahora no dependemos de una geometría específica a ser detectada. Nuestra meta es representar el entorno tan detalladamente como sea posible, sin necesidad de tener que confiar en la existencia de una particular característica geométrica. No en vano, el intento de aproximar o interpretar un

conjunto de datos empleando un modelo equivocado, es una de las causas que puede ocasionar el fallo estrepitoso e irreversible de cualquier algoritmo de estimación.

La segmentación propuesta en esta tesis se basa en el análisis de las posiciones relativas de cada tres puntos consecutivos, tal y como se indica detalladamente en la Fig. 4.2. Así, se define primeramente un conjunto de $m-1$ vectores; aquellos que conectan entre sí cada dos puntos consecutivos obtenidos por el sensor láser:

$$\mathbf{p}_i = \mathbf{d}_i - \mathbf{d_{i-1}} \tag{4.1}$$

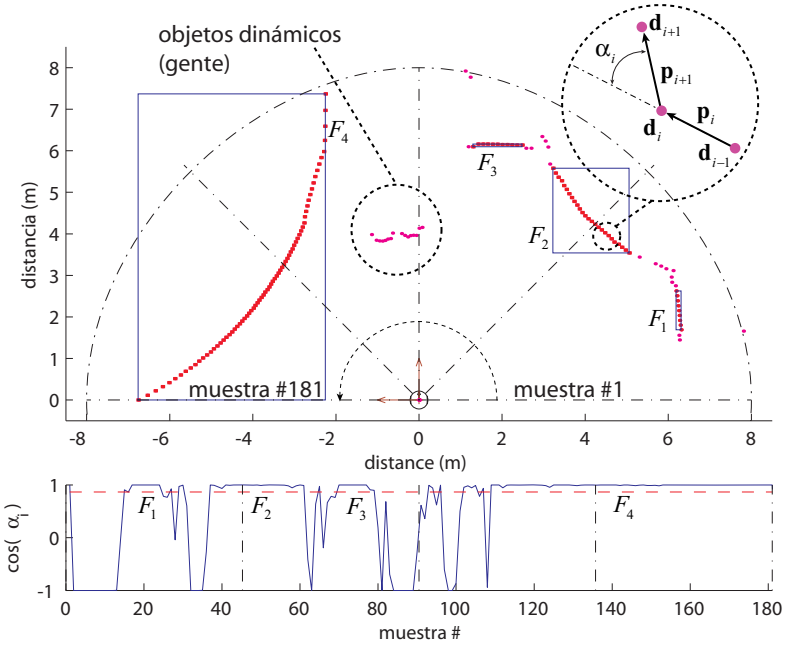


Figura 4.2: Segmentación de un conjunto de datos. La mayor parte de la información adquirida por el sensor láser cobra significado, y puede ser involucrada en el proceso de localización y modelado del entorno.

A continuación se realizan las siguientes comparaciones:

$$|\alpha_i| \leq \alpha_{max} \tag{4.2}$$

$$max\left(\|\mathbf{p}_i\|, \|\mathbf{p}_{i+1}\|\right) \leq \eta \cdot min\left(\|\mathbf{p}_i\|, \|\mathbf{p}_{i+1}\|\right) \tag{4.3}$$

La Fig. 4.2 muestra el proceso descrito utilizando medidas extraídas de un conjunto de datos real. En este caso, el número total de objetos detectados es 4. Nótese que los elementos F_2 y F_4 serían difícilmente descriptibles utilizando métodos tradicionales de representación geométrica, tales como segmentos, o incluso arcos de circunferencia. También se aprecia que los objetos dinámicos presentes en el entorno

han sido adecuadamente descartados. Esto es posible puesto que la variabilidad geométrica en el perímetro de estos elementos (personas, fundamentalmente), hace posible que el algoritmo de segmentación los rechace.

Como puede observarse en la Fig. 4.2, α_i es precisamente el ángulo formado entre los vectores $\mathbf{p}_i$ y $\mathbf{p}_{i+1}$, y es utilizado como una medida de lo "alineados" o "no alineados" que están tres puntos consecutivos. Así, el objetivo de la verificación dada por la ecuación (4.2) es determinar qué puntos próximos están lo suficientemente alineados como para ser considerados pertenecientes a un mismo objeto, pero sin restringirlos geométricamente a estar situados sobre la misma línea recta (como sucedería en el caso de un método basado en segmentos), o sobre cualquier otra geometría predefinida. Dicha comprobación angular tolera este alejamiento geométrico de la linealidad de una recta, pero se impone un cierto límite o umbral para cortar el conjunto de datos en los puntos donde se encuentran esquinas.

Por lo tanto es importante el límite angular elegido:

- **Valores excesivamente pequeños** provocarán una segmentación excesiva, multiplicándose innecesariamente el número de objetos estrictamente necesarios para describir una característica geométrica (especialmente cuando el nivel de ruido que afecta a las medidas del sensor es elevado).
- **Valores excesivamente grandes** del límite angular, provocarán que el algoritmo de segmentación sea incapaz de detectar adecuadamente localizaciones de gran importancia, como las esquinas.

A pesar de que las curvas spline son capaces de representar adecuadamente esquinas (puntos en los que se produce una discontinuidad en la primera derivada) dentro de su propia geometría (sin más que aumentar adecuadamente la multiplicidad de un nodo interior), en esta tesis se ha optado por no incluir esta característica dentro de la propia definición de la curva. Así, la presencia de una esquina provoca la terminación del objeto que representa una superficie con continuidad geométrica.

Las esquinas son utilizadas para delimitar el spline; pero la esquina en sí no es empleada en la construcción del mapa. Nótese que la inclusión de esta información en el proceso de modelado, empleando los algoritmos propuestos en esta tesis, sería trivial: teniendo en cuenta que la esquina representa el punto en el que dos splines (superficies de un elemento del entorno) toman contacto, y teniendo en cuenta que con los vectores de nodos empleados, el primer y el último punto de control coinciden, respectivamente, con el primer y el último punto de la curva, una esquina representaría un punto de control que es exactamente el mismo en dos curvas del mapa. La inclusión de esta información en el modelado del entorno sería considerada como una restricción geométrica adicional.

Los efectos de la variación de este parámetro umbral α_{max} sobre el desempeño del algoritmo de segmentación propuesto pueden observarse con claridad en la figura 4.3. Como se puede apreciar, valores excesivamente altos hacen que localizaciones significativas como son las esquinas no sean detectadas adecuadamente, obteniéndose subconjuntos de datos difíciles de aproximar mediante el simple esquema propuesto

en la sección 3.7.3. A pesar de que la presencia de esquinas es sencilla de modelar mediante una única curva spline, ello requiere de la existencia de nodos interiores de multiplicidad mayor que uno, escenario que se ha dejado fuera del alcance de esta tesis, al tiempo que incrementa innecesariamente el número de puntos de control necesarios para describir el entorno.

Por otra parte, a medida que disminuye excesivamente el valor de este parámetro α_{max}, puede apreciarse cómo el simple ruido presente en las medidas obtenidas por el sensor láser hace que la condición (4.2) no se verifique con demasiada frecuencia. Este hecho se traduce en una segmentación excesiva del conjunto de datos de partida. Peor aún, puede darse el caso de que demasiados puntos sean descartados y no incluidos en ninguno de los objetos segmentados, con la consiguiente pérdida de información acerca del entorno. Este efecto es precisamente el opuesto a lo que persigue el modelado mediante curvas spline, que es el aprovechamiento máximo de la información suministrada por el sensor.

Típicamente $\alpha_{max} \in [0, \pi/4]$ y $\eta \in [1{,}5, 2]$ son valores que proporcionan resultados excelentes en la práctica.

Desde el punto de vista de la implementación de la comprobación dada por la ecuación (4.2), es importante resaltar que no es preciso calcular el ángulo:

$$\alpha_i = acos\left(\frac{\mathbf{p}_i \cdot \mathbf{p}_{i+1}}{|\mathbf{p}_i \cdot \mathbf{p}_{i+1}|}\right) \tag{4.4}$$

dado que esta ecuación es equivalente la presentada a continuación, mucho más eficiente desde un punto de vista computacional:

$$cos\left(\alpha_i\right) \geq cos\left(\alpha_{max}\right) \tag{4.5}$$

o, lo que es lo mismo:

$$\frac{\mathbf{p}_i \cdot \mathbf{p}_{i+1}}{|\mathbf{p}_i \cdot \mathbf{p}_{i+1}|} \geq cos\left(\alpha_{max}\right) \tag{4.6}$$

La verificación de la ecuación (4.3) se ha introducido para evitar situaciones como la representada en la figura 4.4. En ella se ha segmentado un conjunto de datos reales sin tener en consideración la condición dada por la ecuación (4.3). Ello provoca un serio problema en la detección del objeto F_5. Como puede apreciarse, a pesar de que el ángulo α_i verifica la condición (4.2), la excesiva diferencia entre los módulos de los vectores que definen este ángulo haría no verificarse la ecuación (4.3).

Esta situación provoca no sólo la incorrecta extracción de objetos del entornos. Aún más grave es el hecho de que intentar aproximar conjuntos de datos con esta configuración, provoca resultados inesperados y muy perniciosos al intentar se ajustados mediante una única curva spline, tal y como se observa en el detalle 4.4.c.

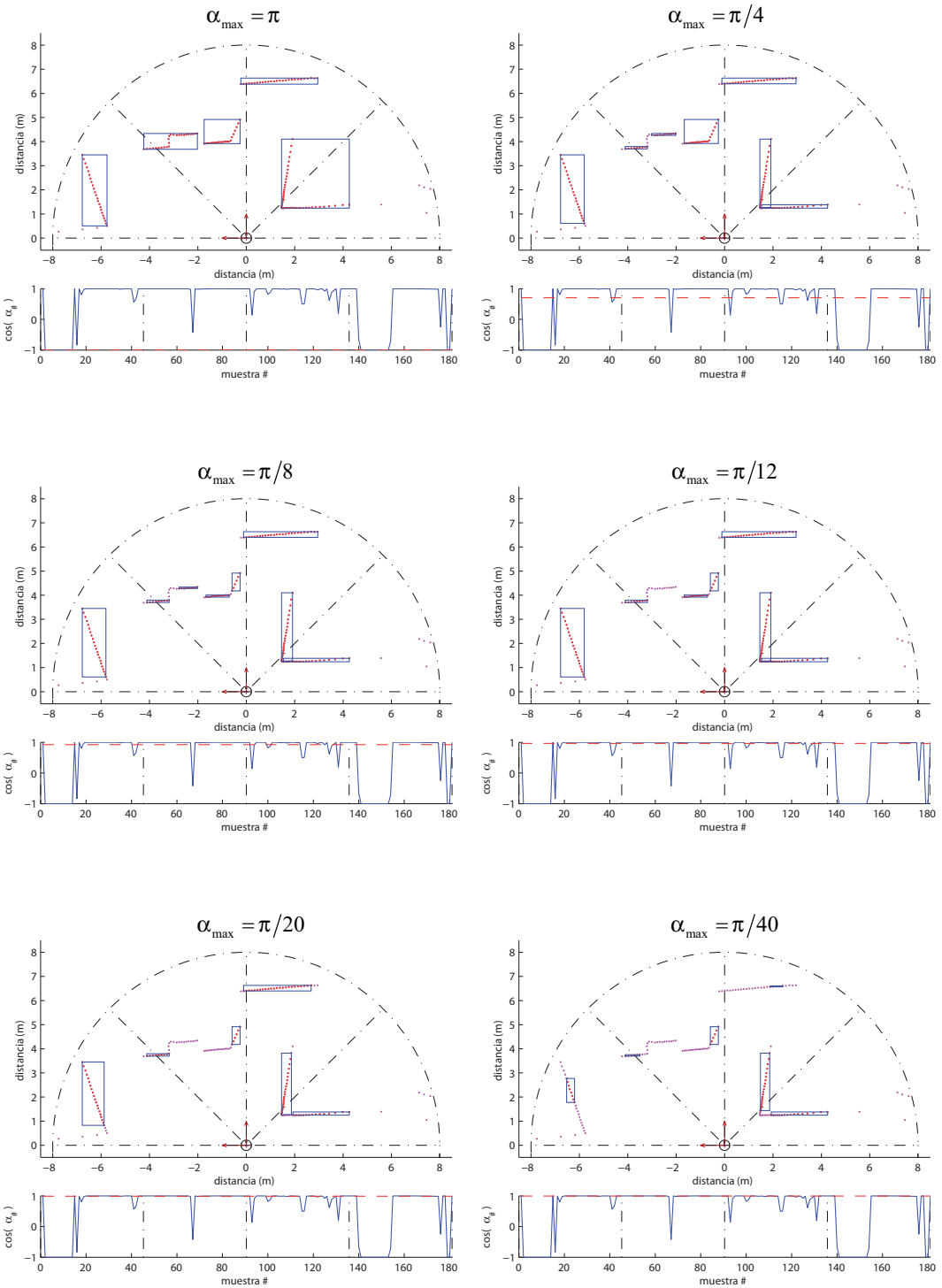


Figura 4.3: Efecto de la variación del valor de α_{max} sobre la segmentación de los datos del láser. Valores elevados provocan la incorrecta detección de características significativas del entorno, mientras que valores demasiado pequeños se traducen en una excesiva segmentación y en pérdida de información.

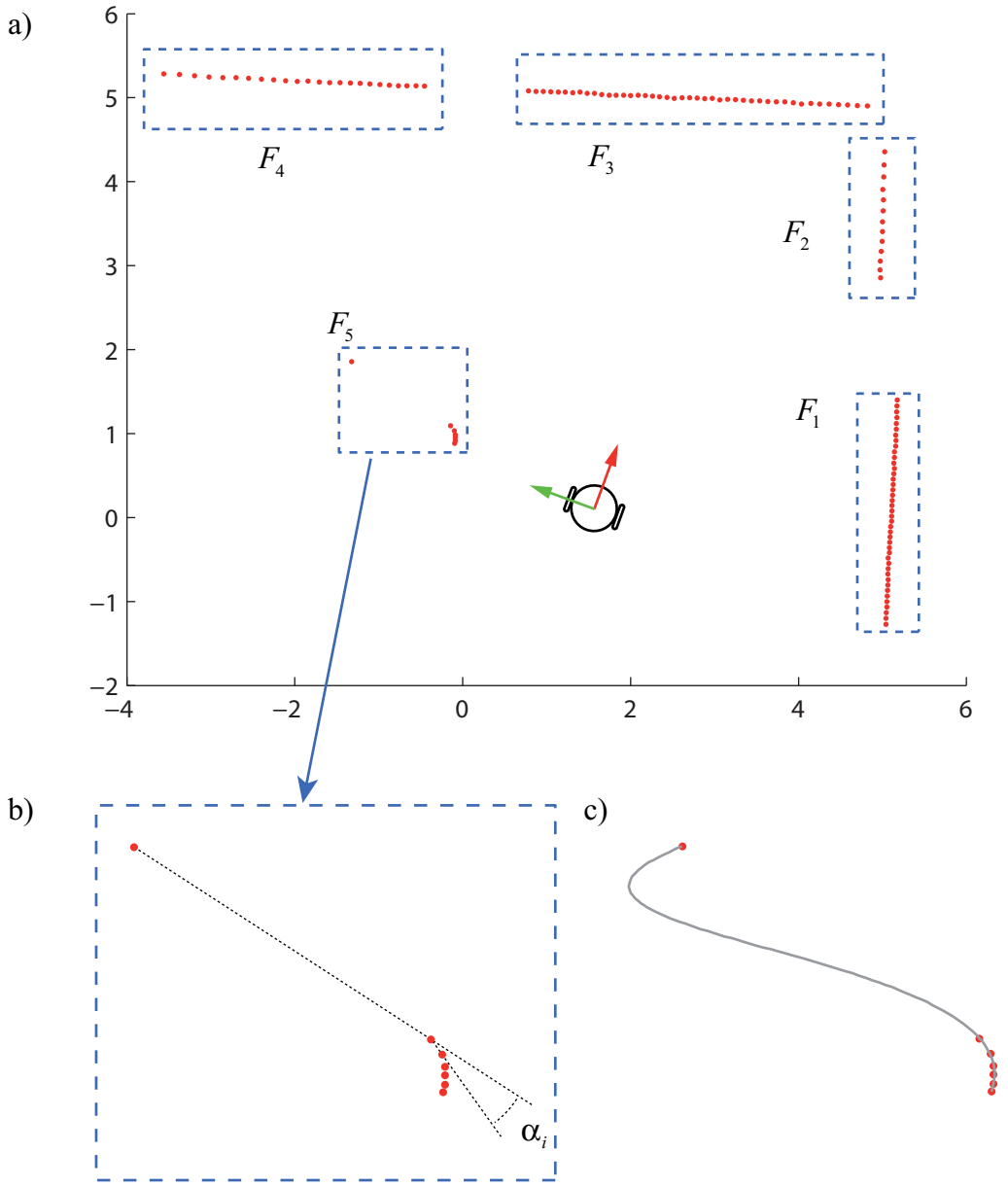


Figura 4.4: Ejemplo de error de segmentación. a) Conjunto de datos de partida, en el que se han detectado 5 objetos únicos sin verificar la condición dada por la ecuación 4.3. b) Detalle del objeto F_5. A pesar de que el ángulo α_i verifica la condición de la ecuación 4.2, la excesiva diferencia de módulos de los vectores que definen este ángulo haría no verificarse la ecuación 4.3. c) Curva resultante de aproximar el subconjunto de datos contenido en el objeto F_5.

Así pues, la simple verificación (4.3) se encarga de comprobar que la diferencia entre la distancia entre los puntos $\mathbf{d}_i$ y $\mathbf{d}_{i-1}$, y la distancia entre los puntos $\mathbf{d}_{i+1}$ y $\mathbf{d}_i$ no sea demasiado grande. Al menos no tan grande como para que quepa la posibilidad de que alguno de los tres puntos que intervienen en las comparaciones pertenezca a un objeto diferente.

Cuando un conjunto de m_F datos consecutivos verifican las anteriores relaciones, se asume que pertenecen al mismo objeto. Adicionalmente, se puede imponer otro tipo de restricciones para considerar un objeto como válido. En esta tesis se proponen las siguientes restricciones que, si bien son muy básicas e intuitivas, han proporcionado excelentes resultados prácticos en cuanto a eficiencia y robustez experimental.

- Exigencia de **mínimo número de puntos para cada objeto**. Podría suceder que durante el proceso de segmentación, apareciesen objetos con un número muy reducido de puntos. Esto puede indicar que se trata de un objeto muy pequeño, o de una persona. Es interesante incluir en el mapa sólo objetos de un tamaño significativo, especialmente teniendo en cuenta que no han de ser confundidos con personas y otros elementos móviles presentes en el entorno.
- Exigencia de una **longitud mínima para el objeto** detectado. De esta sencilla manera se pueden filtrar en cierta medida objetos dinámicos como personas. Hay que tener en cuenta que el algoritmo de segmentación, tal como se ha planteado, no hace discriminaciones en cuanto a la geometría de los objetos detectados. Así, con la sola información que proporciona el láser, es difícil distinguir una persona de pie, de una columna u otro elemento arquitectónico de sustentación o decoración. Se ha optado por lo tanto por utilizar las dimensiones de los objetos como elemento fundamental para distinguir aquellos objetos dinámicos de los que no lo son, asumiendo que las dimensiones relativas de los primeros son reducidas, comparadas con las de los segundos.

En este momento realizar un ajuste de los datos m_F datos correspondientes a cada uno de los objetos detectados por el sensor mediante una curva spline es un paso trivial, tal como fue descrito en la sección 3.7.3.

Es importante hacer especial hincapié en el hecho de que el simple proceso de segmentación descrito, es capaz de extraer de manera eficiente y, a juzgar por los experimentos realizados, robusta, información relevante acerca del entorno del robot. Las simples comprobaciones dadas por las ecuaciones (4.2) y (4.3) permiten segmentar puntos consecutivos pertenecientes a la misma entidad física, y determinar los puntos en los que se localizan las esquinas.

A pesar de que estas comprobaciones (fundamentalmente la primera de ellas) son capaces de descartar por sí mismas objetos dinámicos tales como personas (las cuales presentan una variabilidad geométrica más acusada en el perímetro de su sección transversal o, al menos, mayor que los elementos que conforman el entorno estático del robot), las comprobaciones adicionales referentes al tamaño (tanto en número de puntos como en longitud total) de los objetos detectados, permiten filtrar exitosamente aquellos elementos dinámicos que no se desea incluir en el mapa.

4.2.2. Asociación de datos entre splines

En esta sección se describe el proceso de asociación de datos entre splines detectados por el robot y splines contenidos en el mapa, que representan objetos detectados en instantes temporales anteriores y ya se encuentran presentes en el modelo del entorno.

En cada instante k, el robot adquiere a través de sus sensores un nuevo conjunto de observaciones, que se traducen tras los procesos de segmentación y de ajuste, descritos en las secciones 4.2.1 y 3.7.3, en el oportuno juego de curvas spline que describen la realidad física que rodea al robot en su entorno más próximo. Es necesario obtener una correcta asociación entre las N curvas contenidas en el mapa $(\mathbf{s}_{m,1}, \ldots, \mathbf{s}_{m,N})$, y los N_o objetos que acaban de ser detectados $(\mathbf{s}_{o,1}, \ldots, \mathbf{s}_{o,N_o})$.

Cada uno de los objetos observados pertenecerá a una, y sólo una, de las siguientes categorías:

1. **Objetos nuevos** que nunca antes han sido detectados por el láser. Es de esperar que ninguna de estas curvas pueda ser asociada con las ya contenidas en el mapa. Los objetos pertenecientes a esta categoría deberán ser agregados al mapa, y los puntos de control que definen su geometría correctamente inicializados en el vector de estado, como se explicará en el apartado 4.6.1.

2. **Objetos ya presentes en el mapa**, que son nuevamente detectados total o parcialmente. Dentro de esta categoría cabe establecer una distinción entre dos situaciones diferentes:

   a) **Que la región del objeto detectada ya se encuentre total o parcialmente contenida en el mapa**. Dicho con otras palabras, que al menos un conjuntos de los puntos adquiridos por el sensor, se correspondan con regiones del objeto ya detectadas anteriormente. A esta situación nos referiremos como de "solapamiento", y la información que proporcione será de utilidad para (i) actualizar el estado del sistema como se verá en el apartado 4.5.2, y (ii) alargar objetos existentes en el mapa con la nueva información adquirida, como se explicará en el apartado 4.6.2.

   b) **Que la región del objeto detectada no se encuentre aún contenida en el mapa**. En este caso no existirá solapamiento entre los nuevos datos adquiridos y el objeto del mapa. Será por tanto imposible asociar la observación con ninguno de los objetos conocidos hasta el instante y el objeto detectado será considerado como nuevo. Se tratará por tanto de una situación equivalente a la mencionada en el punto 1 de esta lista.

El concepto de "solapamiento" anteriormente enunciado es de especial importancia a la hora de entender la asociación de datos propuesta. Sólo será posible entender que un objeto detectado ya está presente en el mapa, cuando al menos un subconjunto de los puntos que lo definen se correspondan con al menos una de las curvas presentes el modelo disponible del entorno hasta ese instante.

Muy posiblemente, la mayor desventaja en la utilización de splines como herramienta de modelado, es que los puntos de control que las definen no son directamente observables. Para una geometría arbitraria dada, existen infinitas maneras de describir su forma dependiendo del vector de nodos específico empleado, y de la región particular del objeto que ha sido capturada por los sensores del robot. Sobre este engorroso problema se incidirá de nuevo más adelante, al estudiar el modelo de observación propuesto, en el apartado 4.4.

El proceso de asociación de datos propuesto en esta tesis se explica con ayuda de las figuras 4.5, 4.6 y 4.7, y el pseudocódigo presentado en la Fig. 4.8. Tómese la situación mostrada en la figura 4.5. La imagen superior de esta figura (4.5.a) muestra la configuración del entorno simulado que servirá para ilustrar los conceptos fundamentales de la asociación de datos empleada: una pared curvilínea, y un elemento de sección cuadrangular.

En la imagen 4.5.b se muestra la configuración inicial: en el instante k se dispone de un modelo del entorno que contiene un único objeto $F_1 \equiv \mathbf{s}_{m,1}(t)$.

En la siguiente figura correspondiente al instante $k+1$ (detalle 4.5.c) el robot ha avanzado a la nueva posición $\mathbf{x}_R(k+1|k)$, y acaba de detectar tres objetos presentes en sus proximidades: $O_{k+1,1} \equiv \mathbf{s}_{k+1,1}(u)$, $O_{k+1,2} \equiv \mathbf{s}_{k+1,2}(v)$ y $O_{k+1,3} \equiv \mathbf{s}_{k+1,3}(w)$. Se llama aquí la atención sobre el hecho de que se ha empleado una letra diferente para designar el parámetro de cada una de las tres curvas involucradas.

¿Cómo determinar cuáles de los objetos detectados se corresponden con elementos físicos ya contenidos en el mapa (al menos parcialmente), y cuáles son características del entorno detectadas por vez primera? Como puede apreciarse en la figura 4.5.c, la posición obtenida para el robot aplicando el modelo de movimiento disponible para el mismo, es una mera aproximación de la posición real; los errores y simplificaciones en el modelo, y el ruido presente en las medidas sensoriales hacen que la posición $\mathbf{x}_R(k+1|k)$ sea una mera estimación *a priori* de la verdadera localización de la máquina. Por este motivo las observaciones $O_{k+1,1}, O_{k+1,2}$ y $O_{k+1,3}$ aparecen deslocalizadas, y no se superponen extactamente con los objetos a los cuales representan (dibujados en línea discontinua) por estar sus posiciones calculadas utilizando información relativa a la posición del robot que será, en general, errónea.

En una primera aproximación, se puede obtener una medida de las distancias entre los diferentes objetos (los detectados, y los contenidos en el mapa) simplemente comparando las distancias que separan sus respectivos puntos de control (recuérdese la propiedad 2 de las curvas spline, comentada en la sección 3.6). Así, a pesar de que las curvas observadas no están en su verdadera posición global, y de que aunque lo estuvieran los puntos de control que las definen no tienen por qué corresponderse con su representación hipotéticamente ya contenida en el mapa, es cierto que los polígonos de control de objetos repetidos han de ser similares o, al menos, estar próximos. Con el objetivo de mejorar la velocidad de esta búsqueda, no es necesario comparar todas las observaciones con todos los objetos contenidos en el mapa: bastará simplificar el modelo disponible tomando sólo aquellas curvas del mapa próximas a la estimación actual de la posición del robot.

Así pues, se calculan las distancias entre los puntos de control de cada uno de los

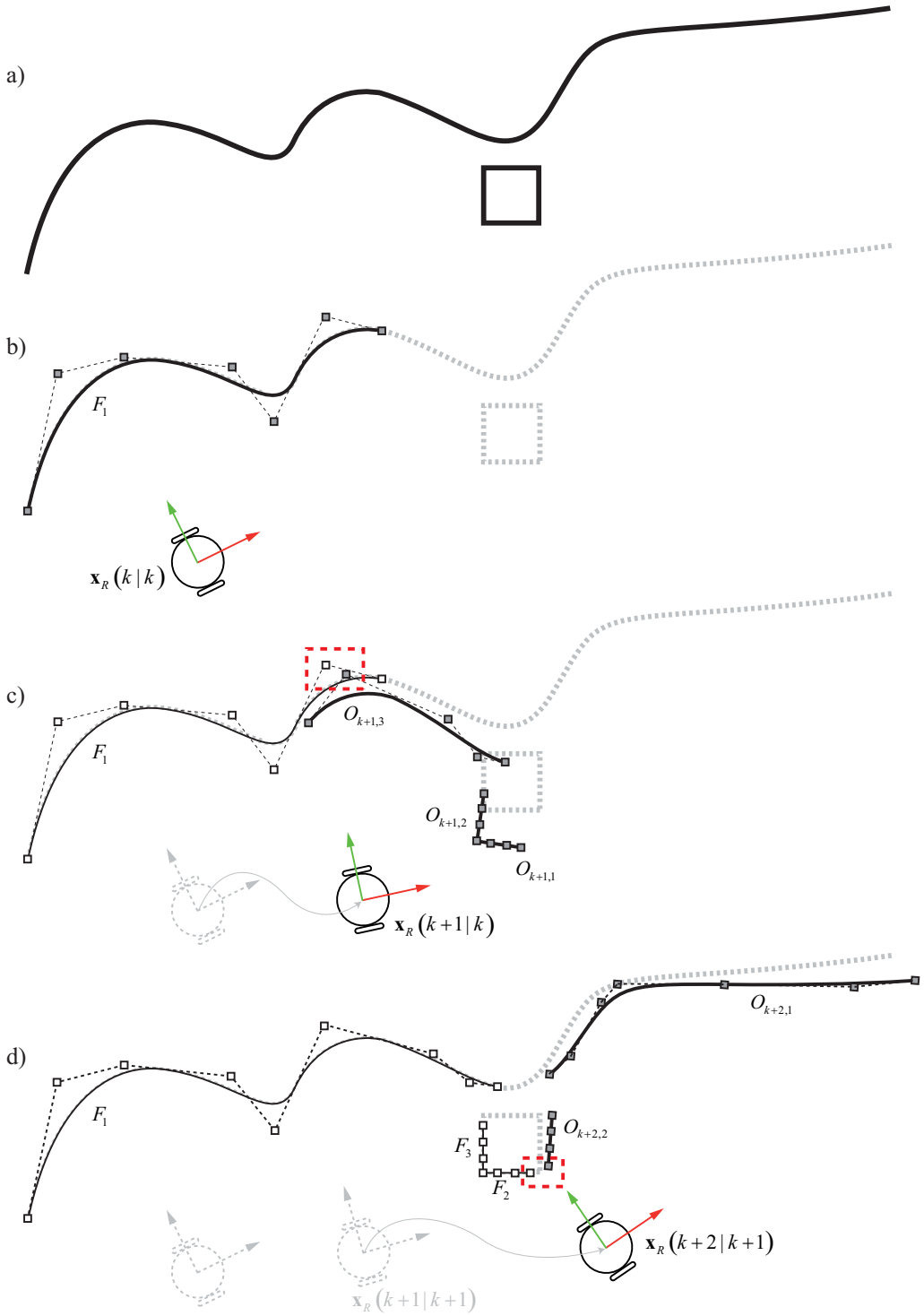


Figura 4.5: Proceso de asociación de datos. a) Configuración del entorno a ser modelado. b) Estado del modelo en el instante k. El mapa contiene un único objeto. c) El robot se desplaza y detecta tres objetos. d) El robot se desplaza nuevamente, detectando dos objetos.

objetos detectados y los puntos de control de los splines contenidos en el mapa. Cada spline observado es preliminarmente asociado con la curva del mapa más próxima, siguiendo el siguiente criterio:

$$min\left(dist\left(\mathbf{x}_{m,i},\mathbf{x}_{o,j}\right)\right) \leq d_{min}, \begin{cases} i = 1 \ldots n_m \\ j = 1 \ldots n_o \end{cases} \tag{4.7}$$

donde $\mathbf{x}_{m,i}$ son los puntos de control de un spline del mapa, n_m y n_o son, respectivamente, el número de puntos de control del spline del mapa y del spline detectado, y $dist\left(\mathbf{x}_{m,i},\mathbf{x}_{o,j}\right)$ representa la distancia euclídea entre dos puntos de control.

Cuando ningún spline observado se encuentra lo suficientemente cerca de algún spline del mapa (tal como ocurre con $O_{k+1,1}$ y $O_{k+1,2}$ en la figura 4.5.c), el objeto correspondiente es considerado como detectado por primera vez, y es agregado al mapa (proceso descrito en el apartado 4.6.1) una vez que la posición del robot ha sido actualizada en la etapa de corrección del filtro (como veremos en la sección 4.5.2). Esto es, cuando la simple comprobación de proximidad de los polígonos de control no es superada, se considera que los objetos pertenecen inequívocamente a la categoría 1 anteriormente enumerada.

Por el contrario, en el caso más habitual de que un spline sea preliminarmente asociado con algún elemento del mapa (como ocurre entre los objetos F_1 y $O_{k+1,3}$, se hace necesario establecer una correspondencia entre los puntos de ambas curvas, tal y como se ilustra detalladamente en la figura 4.6. En la misma, se observa claramente el concepto de solapamiento anteriormente expuesto: existe una zona común a ambos objetos, identificable a pesar del evidente error en el posicionamiento estimado del objeto detectado.

A continuación se indicará un procedimiento para comparar ambas curvas de manera que, a pesar de la no coincidencia de la región de solapamiento, sea posible detectar que esta situación se produce. Esta comparación proporcionará también información acerca de la correspondencia entre las parametrizaciones de ambas curvas, conocimiento que será de gran utilidad cuando sea necesaria la extensión de una de las curvas del mapa con nuevos datos, tal como se comentará en la sección 4.6.2.

El proceso de asociación de datos propuesto se desarrolla del siguiente modo:

- Se comienza considerando uno de los puntos extremos de la curva observada (recordemos que los puntos extremos coinciden con los puntos extremos del polígono de control). Este es el punto A.
- Se busca el punto más cercano de la curva del mapa al punto A:punto B.
- En el caso en que B se corresponda con uno de los puntos extremos de la curva del mapa, entonces se repite el proceso a la inversa, calculándose el punto más cercano a B sobre la curva observada; el punto C en nuestro ejemplo. En caso contrario, los puntos A y B serían asociados.

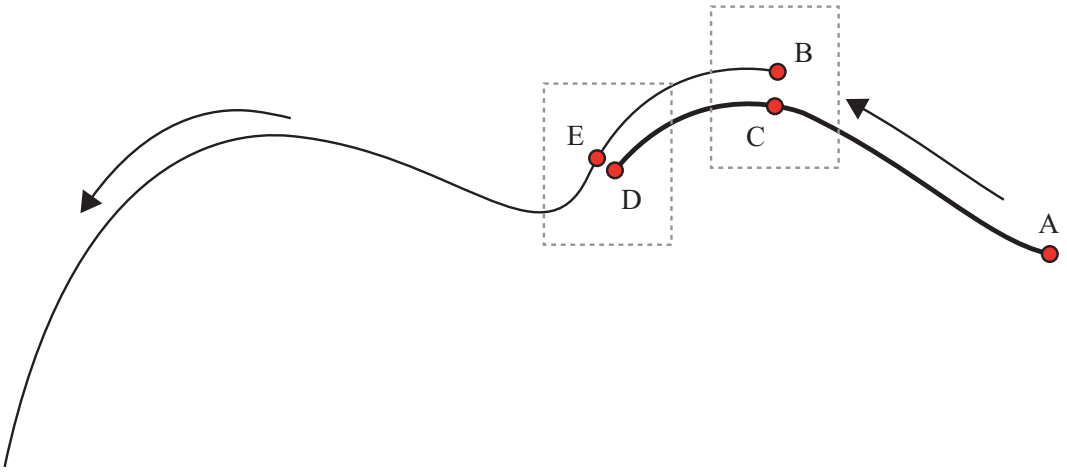


Figura 4.6: Detalle del proceso de asociación de datos mostrado en la figura 4.5.c. Dos objetos solapados (uno en el mapa, y otro observado) son asociados, y se establece una correspondencia paramétrica entre ambos.

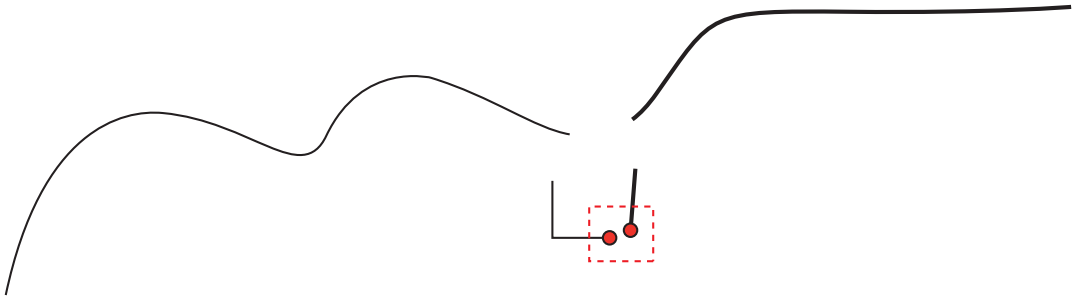


Figura 4.7: Detalle del proceso de asociación de datos mostrado en la figura 4.5.d. No es posible realizar una asociación correcta entre ninguno de los objetos detectados, y ninguno de los objetos contenidos en el mapa.

- El proceso se repite tomando como punto de partida el otro extremo de la curva observada (punto D en la figura) que, en el ejemplo mostrado, es asociado con con el punto E sobre la curva del mapa.

Al final de este proceso, no sólo se ha obtenido una correspondencia entre splines observados y splines del mapa, sino también entre puntos de las distintas curvas (C, B) y (D, E) y, algo más importante aún como veremos: una correspondencia entre los valores de los distintos parámetros que las definen, puesto que

$$\begin{array}{ll} C = \mathbf{s}_{o,1}\left(u_{ini}\right) & B = \mathbf{s}_{m,1}\left(t_{ini}\right) \\ D = \mathbf{s}_{o,1}\left(u_{fin}\right) & E = \mathbf{s}_{m,1}\left(t_{fin}\right) \end{array} \tag{4.8}$$

En el caso de que los puntos extremos asociados en alguna de las curvas sean coincidentes, tal y como ocurre en la figura 4.7, la asociación es rechazada por considerarse no válida.

Todos los razonamientos anteriormente expuestos aparecen recogidos en el pseudocódigo mostrado en la figura 4.8.

```
for (i = 1 → N_o)      % para todas las observaciones
  for (j = 1 → N_m)      % para todas las curvas del mapa
                            % (próximas a la posición estimada del robot)
    if (min(dist(x_o^i, x_m^j),) < d_mincp)    % comparación puntos de control
      % las curvas están próximas
      % búsqueda de correspondencia de puntos iniciales
      u_ini = first(Ξ_o,i);              % primer nodo de la curva observada s_o,i(u)
      s_m,j(t_ini) = nearest in s_m,j(t) to s_o,i(u_ini);
      if ((t_ini == first(Ξ_m,j)) || (t_ini == last(Ξ_m,j)))
        s_o,i(u_ini) = nearest in s_o,i(u) to s_m,j(t_ini);   % actualización de u_ini
      end;
      % búsqueda de correspondencia de puntos finales
      u_fin = last(Ξ_o,i);               % ultimo nodo de la curva observada s_o,i(u)
      s_m,j(t_fin) = nearest in s_m,j(t) to s_o,i(u_fin);
      if ((t_ini == first(Ξ_m,j)) || (t_ini == last(Ξ_m,j)))
        s_o,i(u_fin) = nearest in s_o,i(u) to s_m,j(t_fin);   % actualización de u_fin
      end;
      % comprobación final:
      if ((dist(s_o,i(u_ini), s_m,j(t_ini)) < d_match) &&
        (dist(s_o,i(u_ini), s_m,j(t_ini)) < d_match) &&
        (t_fin > t_ini) && (u_fin > u_ini))
        % ¡¡ MATCHING CORRECTO !!
      end;
    end;
  end;
end ;
```

Figura 4.8: Pseudocódigo que ilustra el proceso de asociación de datos propuesto.

El simple proceso de asociación de datos descrito, a pesar de ser muy sencillo y estar basado únicamente en el cálculo de distancias euclídeas, se ha comportado de manera muy robusta durante los experimentos. Sin embargo, los beneficios que proporciona la representación paramétrica del entorno pueden ser explotados aún más, como veremos en capítulos siguientes.

4.3. El Modelo de Estado

El estado del sistema que nos ocupa está compuesto, en cualquier instante k, por el propio robot (considerado como el único objeto móvil del entorno bajo estudio), y por todos los elementos físicos que rodean al robot y son estáticos, y que formarán parte del mapa cuya configuración se pretende estimar. Es importante resaltar el hecho de que los únicos elementos que pasarán a formar parte del mapa son los estáticos. Elementos dinámicos como personas que se desplazan en el mismo entorno que el robot, u otros elementos móviles tales como puertas y ventanas que se abren o se cierran, no son de interés para ser incluidos en el mapa. Aquellos de estos últimos elementos que sean capturados por los sensores del robot, habrán de ser oportunamente identificados y eliminados de la observación, y no incluidos en el proceso de estimación recursiva del filtro.

En esta tesis, mientras no se diga lo contrario, los elementos que conforman el mapa son modelados mediante splines cúbicos ($\kappa = 4$). Esto es así por tratarse del tipo de splines más comúnmente utilizado en la práctica, y el más apropiado en términos generales en tareas de aproximación. La utilización de splines de orden más reducido no son sino casos particulares de los algoritmos propuestos (en general las ecuaciones aquí presentadas seguirán siendo válidas tal cual), y órdenes más elevados no son necesarios.

Cuando estos splines son expresados como combinación lineal de funciones básicas B-spline, el estado de cada uno de ellos viene determinado por las posiciones de los puntos de sus polígonos de control (o puntos de control), dado un vector de nodos conocido y fijo que genera la correspondiente base de funciones B-spline para cada elemento del mapa.

Refiriendo todas las posiciones y orientaciones a un sistema de referencia global $\{\mathbf{u}_W, \mathbf{v}_W\}$, y asumiendo que el robot es el primer elemento del mapa (F_0), en un cierto instante temporal k las siguientes expresiones describen el estado del sistema compuesto por la pose del robot, $\mathbf{x}_R$, y las posiciones de los puntos de control de cada uno de los elementos del mapa S_i, representadas por los vectores $\mathbf{x}_{S_i}$:

$$\mathbf{x}_{F_0} = \mathbf{x}_R = [x_R, y_R, \phi_R]^T \tag{4.9}$$

$$\mathbf{x}_{F_i} = \mathbf{x}_{S_i} = [x_{i,1}, \ldots, x_{i,n_i}, y_{i,1}, \ldots, y_{i,n_i}]^T \quad i = 1, \ldots, N \tag{4.10}$$

Así, el estado del sistema puede ser escrito como sigue:

$$\mathbf{x} = \left[\mathbf{x}_r^T, \mathbf{x}_{s_1}^T, \ldots, \mathbf{x}_{s_N}^T\right]^T \tag{4.11}$$

En las anteriores ecuaciones, N es el número de elementos estáticos del mapa (características del mapa) y n_i es el número de puntos de control que definen la posición y geometría de cada uno de ellos. Debemos aquí hacer notar que dicho número de puntos de control puede ser variable, dado que en principio muchos elementos del mapa serán extendidos a medida que se descubran nuevas zonas de los mismos al avanzar el robot móvil en su exploración.

El punto de partida para la formulación probabilística del problema de estimación, es la asunción de que el estado real del sistema así formulado en el instante genérico k es desconocido, pero puede ser modelado mediante una distribución gaussiana que contiene toda la información adquirida hasta ese instante.

$$\mathbf{x}(k) \sim N\left(\hat{\mathbf{x}}(k|k), \mathbf{P}(k|k)\right) \tag{4.12}$$

donde

$$\hat{\mathbf{x}}(k|k) = \left[\begin{array}{cccc} \hat{\mathbf{x}}_R(k|k) & \hat{\mathbf{x}}_{S_1}(k|k) & \ldots & \hat{\mathbf{x}}_{S_N}(k|k) \end{array}\right] \tag{4.13}$$

$$\mathbf{P}(k|k) = \left[\begin{array}{cccc} \mathbf{P}_{RR}(k|k) & \mathbf{P}_{RS_1}(k|k) & \ldots & \mathbf{P}_{RS_N}(k|k) \\ \mathbf{P}_{S_1R}(k|k) & \mathbf{P}_{S_1S_2}(k|k) & \ldots & \mathbf{P}_{S_1S_N}(k|k) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{P}_{S_NS_1}(k|k) & \mathbf{P}_{S_NS_2}(k|k) & \ldots & \mathbf{P}_{S_NS_N}(k|k) \end{array}\right] \tag{4.14}$$

4.4. El Modelo de Observación

La implementación de un algoritmo de SLAM basado en el filtro extendido de Kalman requiere de un modelo de observación. Es necesaria alguna expresión matemática que permita predecir las medidas que se espera obtener de los sensores del robot, dada la pose actual de este y el conocimiento que se tiene del mapa hasta ese instante.

La primera idea que podría acudir a nuestra mente, como es lógico, es la utilización directa de los puntos de control en este modelo de predicción. Dado que el mapa se ve representado en el vector de estado del sistema mediante los puntos de control que definen las geometrías presentes en el entorno, parece razonable intentar predecir la posición de estos puntos de control en la medida obtenida por el robot. Sin embargo, esta solución presenta inconvenientes insoslayables. Esto se debe a que la descripción de una geometría arbitraria mediante una curva B-spline, depende directamente del vector de nodos empleado, y de la parametrización escogida para modelar el objeto.

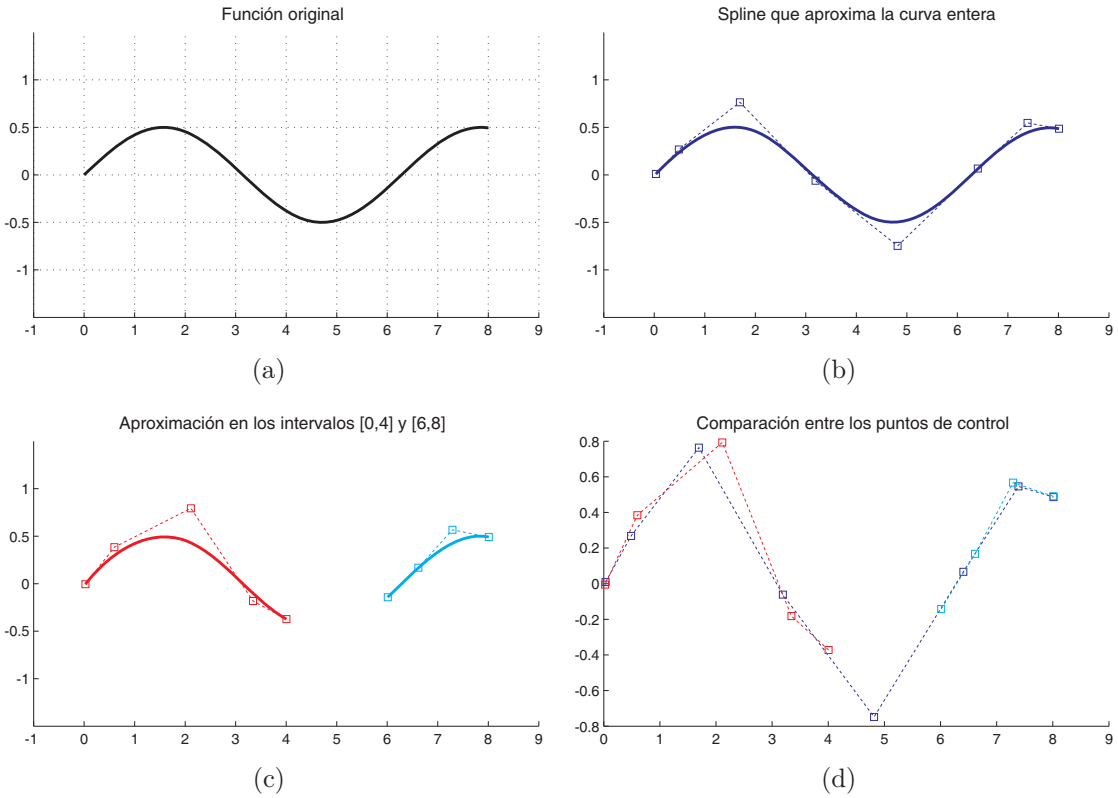


Figura 4.9: La misma geometría (a) aproximada como un todo (b) y en presencia de una oclusión (c). Como puede apreciarse, las posiciones de los correspondientes puntos de control son diferentes en cada caso, lo cual puede apreciarse con claridad en (d).

Como vimos en el capítulo 3, cuando estudiamos la manera de aproximar un conjunto de datos puntuales (sección 3.7.3, página 70), el vector de nodos empleado para construir el spline que modela un objeto cualquiera del entorno, es generado usando como punto de partida las posiciones relativas de los datos obtenidos por el sensor láser. Esta metodología —que presenta múltiples ventajas desde el punto de vista práctico— provoca que, en general, los puntos de control que definen la apariencia de un tramo de geometría dado, sean muy diferentes de los que definen esa misma geometría, pero han sido sintetizados a partir de datos capturados en un instante temporal diferente, desde una posición distinta, o en presencia de oclusiones. Las figuras 4.9 y 4.10 ilustran algunas de estas situaciones.

En ambos casos se ha tomado como geometría de prueba la definida por la función

$$y\left(x\right) = \frac{sin\left(x\right)}{2}$$

definida dentro del intervalo cerrado $x \in [0, 8]$, que se representa para mayor comodidad del lector en las figuras 4.9(a) y 4.10(a). En cada experimento la función ha sido muestreada cada 0,1 unidades del parámetro independiente x, y las medidas obtenidas han sido alteradas con ruido de distribución normal, de media 0 y desviación típica $\sigma = 0{,}02$ tanto en su posición horizontal como en la vertical.

En la figura 4.9(b) se observa el resultado de aproximar la función original cuan-

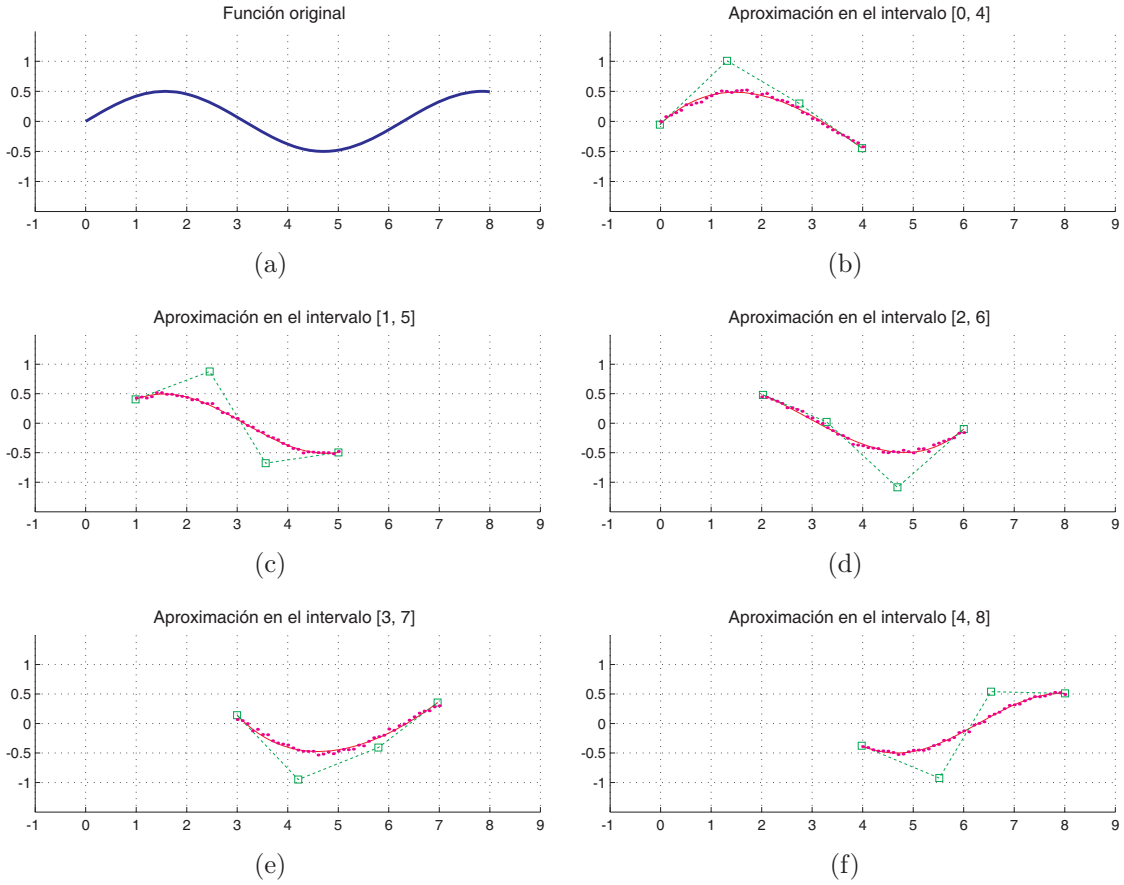


Figura 4.10: La misma geometría (a) aproximada desde distintas posiciones, y detectando diferentes secciones en cada caso. Se aprecia que los puntos de control de de cada una de las curvas dependen del conjunto de datos capturado.

do es posible observarla por completo, mientras que en la figura 4.9(c) se ha hecho lo mismo en presencia de una hipotética oclusión. Esto provoca que la función sea aproximada en dos tramos diferentes. Finalmente el detalle de la figura 4.9(d) reproduce los polígonos de control de cada una de las curvas obtenidas, y puede apreciarse con claridad la discrepancia en las posiciones sintetizadas en cada caso para el estado —*i.e.* los puntos de control— del objeto o fragmentos del objeto modelado.

Por otra parte, la figura 4.10 reproduce el experimento en el caso de que la geometría sea demasiado larga como para que un sensor la detecte en toda su longitud, y sea preciso desplazarse a lo largo del eje de abscisas observando un tramo de 4 unidades de largo en cada caso. Una vez más, se aprecia que las descripciones obtenidas en cada observación no mantienen información en común que sirva para relacionar unas con otras desde el punto de vista de su descripción geométrica.

Podría parecer llegados a este punto, que dada la impredecibilidad de la localización del polígono de control de un spline cualquiera del mapa, el modelado planteado en esta tesis presentará más inconvenientes que ventajas. Puesto que la descripción de los elementos que conforman el entorno del robot depende fuertemente del instante en que han sido detectados, y del conjunto particular de medidas obtenidas, la descripción propuesta plantea serios inconvenientes respecto a soluciones tradicionales como los mapas geométricos de puntos o segmentos. Siempre es posible predecir la posición de una baliza puntual, dada por sus coordenadas x e y en un sistema de referencia dado. Del mismo modo, tampoco entraña gran dificultad la predicción de la medida que se espera obtener tanto de la distancia del robot a un segmento del mapa, como de la orientación relativa del mismo.

Afortunadamente, en esta tesis se presentan mecanismos de razonamiento geométrico y matemático que dotan a la solución planteada de ecuaciones efectivas y computacionalmente eficientes para, no sólo construir un modelo de observación apropiado sino, y esto es importante, obtener las matrices Jacobianas involucradas en la etapa de actualización del filtro. Estos mecanismos son presentados en los dos siguientes apartados.

4.4.1. Predicción de la observación

Ya hemos visto que predecir la posición del polígono de control de un spline contenido en el mapa, y que ha sido nuevamente detectado total o parcialmente por el robot en su exploración es, en la práctica, imposible. Sin embargo no todo está perdido. Como alternativa, se propone en esta tesis como modelo de observación aquel que permite predecir la medida bruta que se obtendrá a través del sensor láser, dado el conocimiento del estado del sistema. Esto es, nos planteamos ahora el problema de determinar, para un instante k, el conjunto de medidas individuales $\mathbf{z}_k = \{z_i, i = 0, \ldots, m\}$ que obtendrá el sensor láser ligado al robot, dada la mejor estimación disponible para la posición de este último, y el conocimiento geométrico del mapa hasta ese instante.

Así, el modelo de observación se puede entender, en este caso, como el cálculo de la intersección de la línea recta definida por un rayo láser —para cada una de

las posiciones a lo largo del recorrido angular del sensor— con los splines contenidos en el mapa. Más concretamente, con aquella curva del mapa que haya sido asociada con un objeto observado, al cual pertenece la medida particular considerada. Este modelo propuesto aparece representado gráficamente en la parte izquierda de la figura 4.11.

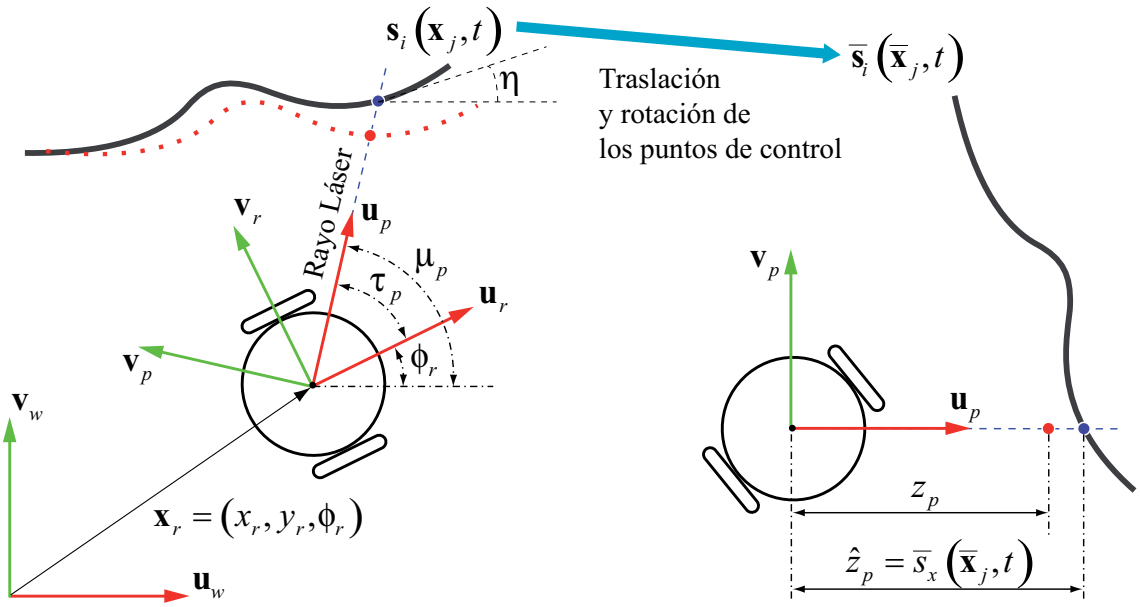


Figura 4.11: El modelo de observación propuesto.

Por desgracia, encontramos un nuevo obstáculo al tomar esta nueva senda: el cálculo de la intersección entre una línea recta y una curva paramétrica, dada en la forma $\mathbf{s}(t) = [s_x(t), s_y(t)]^T$ no es susceptible de ser resuelto mediante una expresión analítica explícita. También es cierto que no se trata de un problema nuevo. Existe un campo de investigación entero dedicado al estudio de este tipo de intersecciones, conocido en la literatura como trazado de rayos (o *"ray tracing"*, según su denominación en inglés) [82], [186].

En el área tecnológica de los gráficos por computador, el trazado de rayos es un método de iluminación global empleado en el renderizado de objetos. Permite pasar de modelos tridimensionales de una escena, a representaciones bidimensionales en la pantalla de un dispositivo electrónico. Dicho con otras palabras: el trazado de rayos es una técnica de síntesis, que permite pasar de un modelo tridimensional a una imagen en dos dimensiones. El algoritmo, en su forma original, fue propuesto en 1980 por Turner Whitted [207], quien se basó en el algoritmo de determinación de superficies visibles de Arthur Appel denominado *"ray casting"* (proyección de rayos) [4–6, 163].

Resulta curioso que esta técnica no fue creada originalmente en el ámbito de los gráficos por computador. Los conceptos del trazado de rayos fueron publicados por primera vez por René Descartes en su tratado *"Les Météores"*, que forma parte de su famoso *"Discours de la Méthode"* ("El Discurso del Método") [58], publicado por primera vez en 1637. En él, Descartes se pregunta por el origen de la forma del arco iris, y utiliza el trazado de rayos como un armazón teórico para la explicación de

ciertos fenómenos.

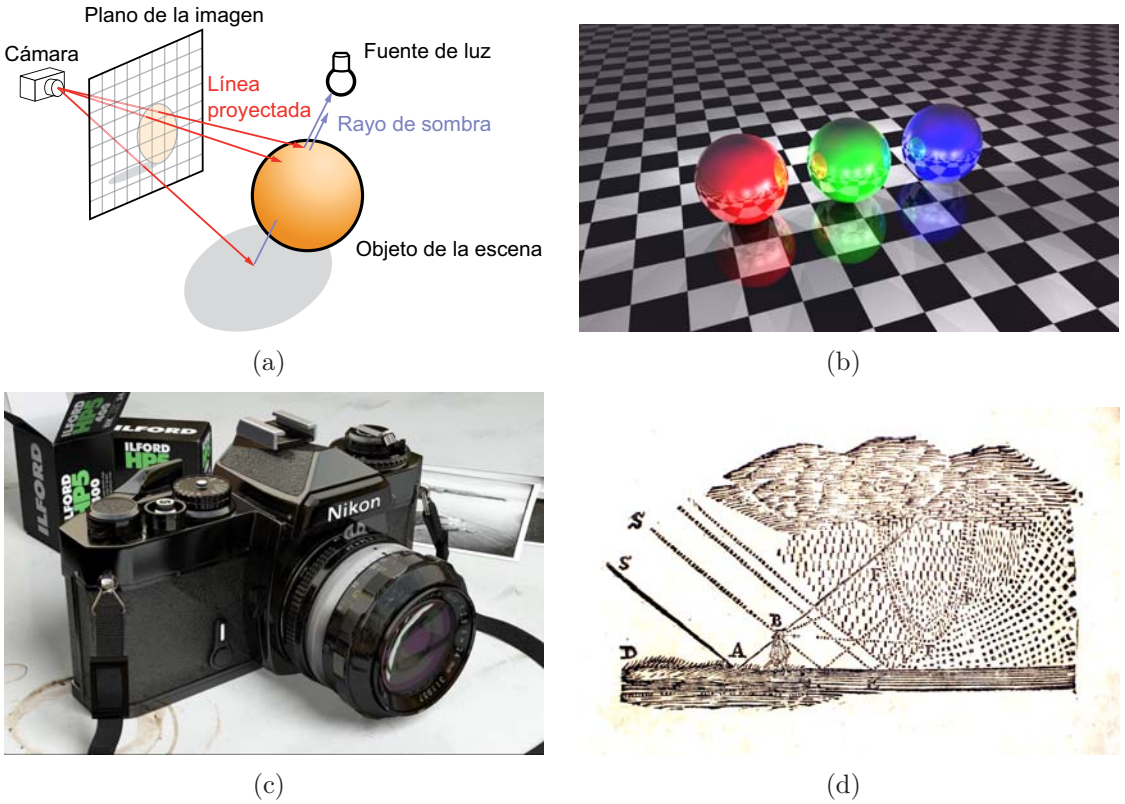


Figura 4.12: El trazado de rayos (a) permite pasar de modelos tridimensionales de objetos y escenas a representaciones bidimensionales como la del ejemplo (b), o imágenes fotorrealistas como la que se observa en la figura (c) (imagen creada con Yafray[1] y Blender[2] por Bert Buchholz). (d) Estudios geométricos de refracción de René Descartes [58].

Los métodos de trazado de rayos se encargan de enviar rayos de luz desde el punto desde el que es observada la escena (el ojo, o la cámara), a través del plano en el que se representará la imagen, hacia el interior de la escena. El problema consiste en determinar si cada uno de los rayos proyectados intersecta o no con algún objeto de la escena. Si esto no sucede, el píxel correspondiente en el plano de la imagen es coloreado con el color del fondo. Cuando se detectan intersecciones, es posible colorear el píxel correspondiente de la manera más oportuna para dotar del realismo deseado a la escena representada. Es una de las técnicas más populares para generar escenas fotorrealistas de universos modelados artificialmente, y es empleada por artistas gráficos, usuarios particulares, y empresas de animación como los estudios Pixar[3].

A pesar de que tradicionalmente la mayoría de las técnicas de trazado de rayos sólo emplean triángulos y otras primitivas geométricas sencillas [118], recientemente han aparecido nuevos métodos que trabajan con geometrías más sofisticadas como superficies paramétricas [1, 98, 124, 171, 204] e incluso implícitas [97, 175], o que aplican técnicas interactivas [81, 144]. En todos los casos las soluciones se obtienen de

[3]http://www.pixar.com

manera aproximada, empleando métodos iterativos.

En esta tesis se propone de manera original un método de trazado de rayos adaptado a las necesidades particulares del problema que nos ocupa. Así, la medida esperada se calcula mediante un procedimiento iterativo, haciendo uso de la propiedad 3 descrita en la sección 3.6, y el método de Newton-Raphson para hallar los ceros de una función. El primer paso es definir un sistema de referencia ortonormal $\{\mathbf{u}_p, \mathbf{v}_p\}$, centrado en el sistema de referencia ligado al robot $\{\mathbf{u}_r, \mathbf{v}_r\}$ y con el eje de abscisas $\mathbf{u}_p$ determinado por la dirección y orientación del correspondiente rayo láser (ver Fig. 4.11).

Sea $\bar{\mathbf{s}}\left(\bar{\mathbf{x}}_i\left(\mathbf{x}_i, \mathbf{x}_r\right), t\right)$ el vector de posición que recorre una curva del mapa expresado en el sistema de referencia así definido (nótese que se ha hecho aquí explícita la dependencia funcional del spline respecto de sus puntos de control). La relación entre los puntos de control $\mathbf{x}_i = [x_i, y_i]^T$ y $\bar{\mathbf{x}}_i = [\bar{x}_i, \bar{y}_i]^T$, $i = 0 \ldots n$ viene entonces dada por:

$$\begin{bmatrix} \bar{x}_i \\ \bar{y}_i \end{bmatrix} = \begin{bmatrix} cos\mu & sin\mu \\ -sin\mu & cos\mu \end{bmatrix} \begin{bmatrix} x_i - x_r \\ y_i - y_r \end{bmatrix} \tag{4.15}$$

siendo μ el ángulo que determina la orientación del rayo láser considerado en el sistema de referencia global. Es decir, dada la orientación del rayo láser en el sistema de referencia del robot, τ, se tiene que:

$$\mu = \phi_r + \tau \tag{4.16}$$

En este contexto, la predicción de la medida $\hat{z} = h\left(\mathbf{x}_i, \mathbf{x}_r\right)$ viene dada por el valor de $\bar{s}_x\left(\bar{x}_i\left(\mathbf{x}_i, \mathbf{x}_r\right), t^*\right)$, donde t^* es el valor del parámetro t que hace $\bar{s}_y\left(\bar{y}_i\left(\mathbf{x}_i, \mathbf{x}_r\right), t^*\right) = 0$. Dado que sólo un reducido número de κ puntos de control afectan a la forma de la curva para cada valor del parámetro t, sólo estos puntos necesitan ser rotados y trasladados al nuevo sistema de referencia.

Una buena aproximación del valor inicial en la iteración de Newton-Raphson para cada una de las curvas del mapa puede ser obtenida directamente del proceso de asociación de datos. Para cada una de las posiciones del rayo láser, el valor de partida utilizado es el valor solución de la posición anterior. Durante los experimentos realizados, se necesitó un máximo de dos iteraciones para realizar estos cálculos con una precisión de 0,1 mm.

4.4.2. Obtención de los Jacobianos

A pesar de la carencia de un modelo de observación explícito, aún es posible calcular las derivadas de estas predicciones respecto del estado del robot y de los elementos del mapa de una manera aproximada. Una vez calculado el valor t^* que hace $\bar{s}_y\left(t^*\right) = 0$, el modelo de predicción en las cercanías de esa localización del

parámetro, asumiendo pequeñas perturbaciones en el vector de estado del sistema, puede ser aproximado por la siguente expresión analítica:

$$h\left(\mathbf{x}_i, \mathbf{x}_r\right) = \bar{s}_x \left(\bar{x}_i\left(\mathbf{x}_i, \mathbf{x}_r\right), t^* - \frac{\bar{s}_y\left(\bar{y}_i\left(\mathbf{x}_i, \mathbf{x}_r\right), t^*\right)}{\bar{s}'_y\left(\bar{y}_i\left(\mathbf{x}_i, \mathbf{x}_r\right), t^*\right)} \right) \tag{4.17}$$

Para obtener este resultado se ha supuesto que, ante pequeñas variaciones del estado del sistema (reducido en este caso a los puntos de control del spline considerado $\Delta\mathbf{x}_j \rightarrow \mathbf{0}, j = 0 \ldots n_i$ y a la pose del robot $\Delta\mathbf{x}_r \rightarrow \mathbf{0}$), el comportamiento de la curva en las proximidades del punto de trabajo t^* puede ser aproximado linealmente por la tangente a la curva en ese punto. Por lo tanto, se precisaría una única nueva iteración del método de Newton-Raphson para obtener el nuevo valor del parámetro $t^{\circledast}$ que permite obtener la medida esperada bajo esa pequeña perturbación del sistema. Cabe hacer las siguientes aclaraciones respecto de la anterior ecuación:

- Se ha indicado explícitamente la dependencia funcional entre la medida esperada, y el estado del sistema en un instante k dado. Así, la medida esperada viene dada por la siguiente relación funcional

$$z = h\left(x_j, x_R\right) \tag{4.18}$$

- La medida esperada ante pequeñas variaciones del estado, viene dada por el valor de la coordenada x de la curva en el sistema de referencia $\{\mathbf{u}_p, \mathbf{v}_p\}$ definido en el apartado anterior, evaluado en el punto en el que la coordenada y se anula. Por lo tanto, llamando $t^{\circledast}$ a este nuevo valor del parámetro que verifica $s_y(t^{\circledast}) = 0$ podemos escribir:

$$h\left(\mathbf{x}_i, \mathbf{x}_r\right) = \bar{s}_x\left(\bar{x}_i, t^{\circledast}\right) \tag{4.19}$$

  donde se ha hecho explícita la dependencia funcional del valor de la coordenada x de la curva, respecto de las coordenadas en x de los puntos de control que la definen, y del valor del parámetro. Estas dependencias funcionales se indican ahora explícitamente, puesto que nuestro objetivo inmediato es derivar la expresión obtenida respecto del estado del sistema. Así, ahora los puntos de control de la curva no son algo fijo y estático que define su geometría, sino que se pretende estudiar cómo varían las medidas que el robot espera obtener de sus sensores, cuando la geometría de la curva del mapa experimenta pequeñas variaciones.

- Las posiciones de los puntos de control que definen la curva, expresadas en el sistema de referencia ligado a la posición definida por un rayo láser del sensores, depende a su vez tanto de las posiciones de los puntos de control en el sistema de referencia global, como de la posición y orientación del robot, tal como se indica en la ecuación (4.15):

$$\bar{x}_i = \bar{x}_i\left(\mathbf{x}_i, \mathbf{x}_r\right) \tag{4.20}$$

- Finalmente, el valor del parámetro que anula la coordenada y de la curva en el sistema de referencia $\{\mathbf{u}_p, \mathbf{v}_p\}$ se puede calcular realizando una única iteración de Newton-Raphson en las proximidades del valor solución para el punto de trabajo, t^*:

$$t^{\circledast} = t^* - \frac{\bar{s}_y\left(\bar{y}_i\left(\mathbf{x}_i, \mathbf{x}_r\right), t^*\right)}{\bar{s}'_y\left(\bar{y}_i\left(\mathbf{x}_i, \mathbf{x}_r\right), t^*\right)} \tag{4.21}$$

La combinación de estos resultados permite la obtención de la expresión (4.17). Aplicando la regla de la cadena para derivar respecto de las posiciones de los puntos de control que definen la curva, referidos al sistema de referencia global (que son los que definen el estado del sistema), podemos obtener:

$$\begin{aligned}
\frac{\partial h}{\partial \mathbf{x}_i} &= \frac{\partial \bar{s}_x}{\partial \bar{x}_i}\frac{\partial \bar{x}_i}{\partial \mathbf{x}_i} + \bar{s}'_x\left(t^*\right)\frac{\frac{\partial \bar{s}'_y}{\partial \bar{y}_i}\frac{\partial \bar{y}_i}{\partial \mathbf{x}_i}\bar{s}_y\left(t^*\right) - \frac{\partial \bar{s}_y}{\partial \bar{y}_i}\frac{\partial \bar{y}_i}{\partial \mathbf{x}_i}\bar{s}'_y\left(t^*\right)}{\left[\bar{s}'_y\left(t^*\right)\right]^2} \\
&= \frac{\partial \bar{s}_x}{\partial \bar{x}_i}\frac{\partial \bar{x}_i}{\partial \mathbf{x}_i} - \frac{\bar{s}'_x\left(t^*\right)}{\bar{s}'_y\left(t^*\right)}\frac{\partial \bar{s}_y}{\partial \bar{y}_i}\frac{\partial \bar{y}_i}{\partial \mathbf{x}_i} \\
&= \frac{\partial \bar{s}_x}{\partial \bar{x}_i}\frac{\partial \bar{x}_i}{\partial \mathbf{x}_i} - \frac{1}{\tan\left(\eta - \mu\right)}\frac{\partial \bar{s}_y}{\partial \bar{y}_i}\frac{\partial \bar{y}_i}{\partial \mathbf{x}_i}
\end{aligned} \tag{4.22}$$

Aquí se ha tenido en cuenta que la ordenada de la curva, expresada en el sistema de referencia $\{u_L, v_L\}$ es cero

$$\bar{s}_y\left(t^*\right) = 0 \tag{4.23}$$

y se ha dejado expresado el valor de la pendiente del spline en este mismo sistema de referencia, en función de los ángulos η y μ, lo cual permitirá comparar los resultados obtenidos por este método analítico, con los que se derivan un poco más adelante de un razonamiento puramente geométrico

$$\frac{\bar{s}'_y\left(t^*\right)}{\bar{s}'_x\left(t^*\right)} = \tan\left(\eta - \mu\right) \tag{4.24}$$

Las derivadas parciales que aparecen en la ecuación 4.22 se pueden obtener ahora fácilmente. Mirando a la ecuación 3.9 se tiene que

$$\frac{\partial \bar{s}_x}{\partial \bar{x}_i} = \frac{\partial \bar{s}_y}{\partial \bar{y}_i} = \beta_{i,k}(t) \tag{4.25}$$

Y de la ecuación 4.15 se deduce

$$\frac{\partial \bar{x}_i}{\partial \mathbf{x}_i} = cos\mu \tag{4.26}$$
$$\frac{\partial \bar{x}_i}{\partial \mathbf{y}_i} = sin\mu \tag{4.27}$$
$$\frac{\partial \bar{y}_i}{\partial \mathbf{x}_i} = -sin\mu \tag{4.28}$$
$$\frac{\partial \bar{y}_i}{\partial \mathbf{y}_i} = cos\mu \tag{4.29}$$

Así, finalmente se pueden escribir las siguientes ecuaciones, que son las derivadas del modelo de observación respecto de los puntos de control que definen el spline del mapa:

$$\frac{\partial h}{\partial x_i} = \beta_{i,k}(t^*)\left[cos\mu + \frac{sin\mu}{\tan(\eta-\mu)}\right] \tag{4.30}$$
$$\frac{\partial h}{\partial y_i} = \beta_{i,k}(t^*)\left[sin\mu - \frac{cos\mu}{\tan(\eta-\mu)}\right] \tag{4.31}$$

De manera similar, haciendo uso de la propiedad 7, y de las ecuaciones 3.9 y 4.15, se tiene que:

$$\begin{aligned}\frac{\partial h}{\partial x_r} &= \sum \frac{\partial \bar{s}_x}{\partial \bar{x}_i}\frac{\partial \bar{x}_i}{\partial x_r} - \frac{\bar{s}'_x(t^*)}{\bar{s}'_y(t^*)}\sum \frac{\partial \bar{s}_y}{\partial \bar{y}_i}\frac{\partial \bar{y}_i}{\partial x_r} \\ &= \left[-cos\mu - \frac{\bar{s}'_x(t^*)}{\bar{s}'_y(t^*)}sin\mu\right]\sum \beta_{i,k}(t^*) \\ &= -cos\mu - \frac{sin\mu}{\tan(\eta-\mu)}\end{aligned} \tag{4.32}$$

$$\begin{aligned}\frac{\partial h}{\partial y_r} &= \sum \frac{\partial \bar{s}_x}{\partial \bar{x}_i}\frac{\partial \bar{x}_i}{\partial y_r} - \frac{\bar{s}'_x(t^*)}{\bar{s}'_y(t^*)}\sum \frac{\partial \bar{s}_y}{\partial \bar{y}_i}\frac{\partial \bar{y}_i}{\partial y_r} \\ &= \left[-sin\mu - \frac{\bar{s}'_x(t^*)}{\bar{s}'_y(t^*)}cos\mu\right]\sum \beta_{i,k}(t^*) \\ &= -sin\mu + \frac{cos\mu}{\tan(\eta-\mu)}\end{aligned} \tag{4.33}$$

$$\begin{aligned}\frac{\partial h}{\partial \phi_r} &= \sum \frac{\partial \bar{s}_x}{\partial \bar{x}_i}\frac{\partial \bar{x}_i}{\partial \phi_r} - \frac{\bar{s}'_x(t^*)}{\bar{s}'_y(t^*)}\sum \frac{\partial \bar{s}_y}{\partial \bar{y}_i}\frac{\partial \bar{y}_i}{\partial \phi_r} \\ &= \sum \beta_{k,i}(t^*)\,\bar{y}_i + \frac{1}{\tan(\eta-\mu)}\sum \beta_{k,i}(t^*)\,\bar{x}_i \\ &= \frac{\hat{z}}{\tan(\eta-\mu)} \end{aligned} \tag{4.34}$$

Estas ecuaciones permitirán el cálculo eficiente de los Jacobianos involucrados en las ecuaciones del filtro extendido de Kalman, descritas en las siguientes secciones de este capítulo.

Alternativamente a la obtención analítica de los Jacobianos, es posible sintetizar algunas de las anteriores expresiones derivadas siguiendo un razonamiento puramente geométrico. No es descabellado asumir que el comportamiento de una curva se puede aproximar en las proximidades del punto t^* mediante la tangente a la curva en ese punto. En realidad, esto no es otra cosa que la misma suposición que acabamos de realizar en la deducción analítica de las ecuaciones; el método de Newton-Raphson se basa en la linealización de la función estudiada en las cercanías del punto de trabajo.

Las figuras 4.13, 4.14 y 4.15 ilustran este escenario simplificado, una vez linealizada la función $\mathbf{s}(t)$ en las proximidades del punto t^*. Así resulta más cómodo e intuitivo visualizar las variaciones que experimentaría la medida esperada $\hat{z}$ cuando la pose del robot experimenta pequeñas perturbaciones.

Fijémonos por ejemplo en la figura 4.13. En ella se representa esquemáticamente cuál sería la variación de la medida esperada h ante pequeñas variaciones en la coordenada x del robot. Así, ante una pequeña perturbación Δx_R, la medida esperada experimentaría una variación Δh. Geométricamente hablando: al pasar el robot de la posición marcada por el punto O a la señalada por el punto O', la intersección de la línea definida por una posición angular del escáner láser con la curva que linealiza a la curva del mapa en las proximidades del punto A, pasaría a ser el punto B, y por tanto el segmento $\overline{O'B}$ sería la nueva medida esperada:

$$h = \overline{OA} \tag{4.35}$$
$$h + \Delta h = \overline{O'B} \tag{4.36}$$
$$\Delta h = \overline{CB} \tag{4.37}$$
$$\Delta x_R = \overline{OO'} \tag{4.38}$$

Manteniendo la vista fija en la figura 4.13 podemos escribir las siguientes expresiones que permiten deducir cuál es la variación de la medida esperada (es decir, la distancia del sensor a la curva linealizada para una posición angular del escáner láser), cuando el robot experimenta pequeñas variaciones de su posición en la dirección de la coordenada x:

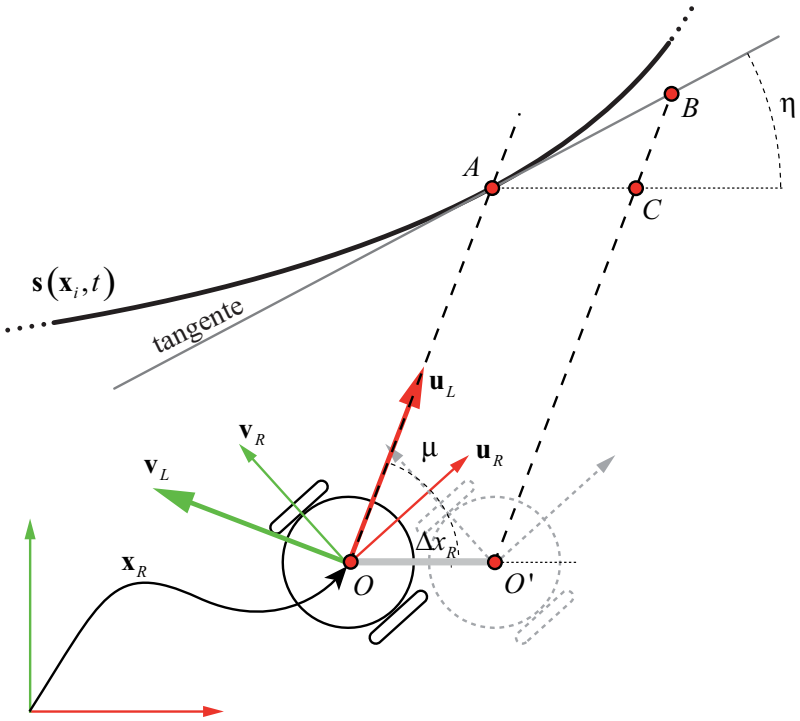


Figura 4.13: Representación esquemática de la relación $\frac{\Delta h}{\Delta x_R}$

$$\overline{AB} \cdot sin\,(\eta) = (h + \Delta h)\, sin\,(\mu) - h \cdot sin\,(\mu) \tag{4.39}$$
$$\overline{AB} \cdot cos\,(\eta) = \Delta x_R + (h + \Delta h)\, cos\,(\mu) - h \cdot cos\,(\mu) \tag{4.40}$$

Eliminando el término $\overline{AB}$ en las anteriores ecuaciones y simplificando, se llega a la siguiente expresión:

$$\frac{\Delta h}{\Delta x_R} = \frac{1}{sin\,(\mu)/tan\,(\eta) - cos\,(\mu)} \tag{4.41}$$

Se puede obtener una expresión similar para la variación de la medida esperada ante pequeñas variaciones en la coordenada y del robot. Mirando ahora a la figura 4.14, los puntos O, O', A, B y C tienen una interpretación análoga a la explicada para la figura 4.13, siendo las expresiones 4.35 a 4.38 igualmente válidas. Las relaciones que podemos escribir en este caso son las siguientes:

$$\overline{AB} \cdot sin\,(\eta) = h \cdot sin\,(\mu) - \Delta y_R - (h + \Delta h)\, sin\,(\mu) \tag{4.42}$$
$$\overline{AB} \cdot cos\,(\eta) = h \cdot cos\,(\mu) - (h + \Delta h)\, sin\,(\mu) \tag{4.43}$$

Eliminando nuevamente el término $\overline{AB}$ y simplificado, se obtiene la siguiente relación:

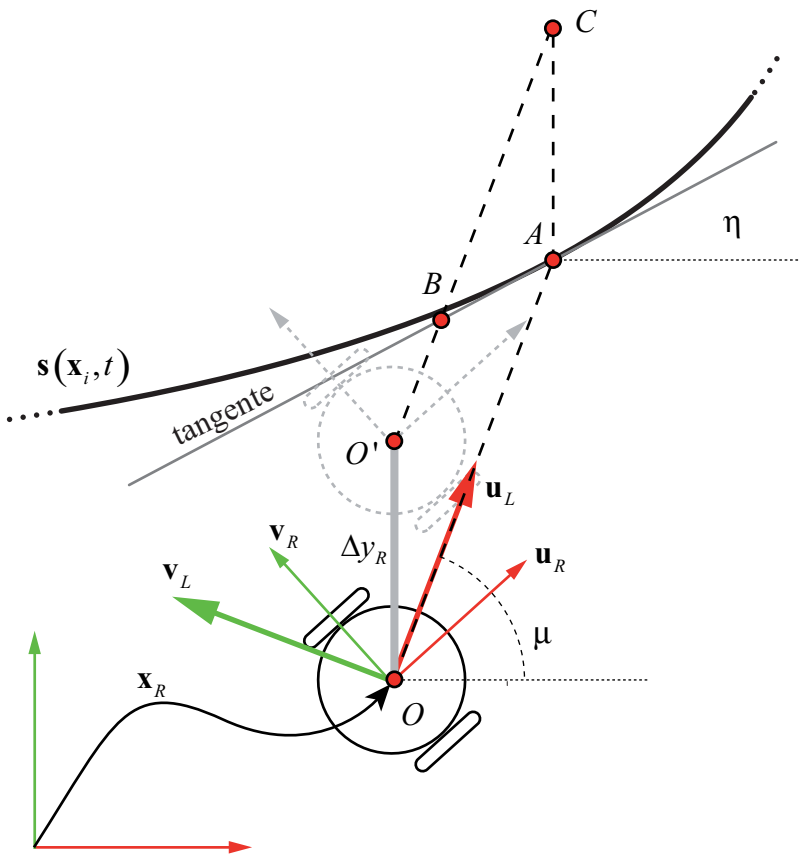


Figura 4.14: Representación esquemática de la relación $\frac{\Delta h}{\Delta y_R}$

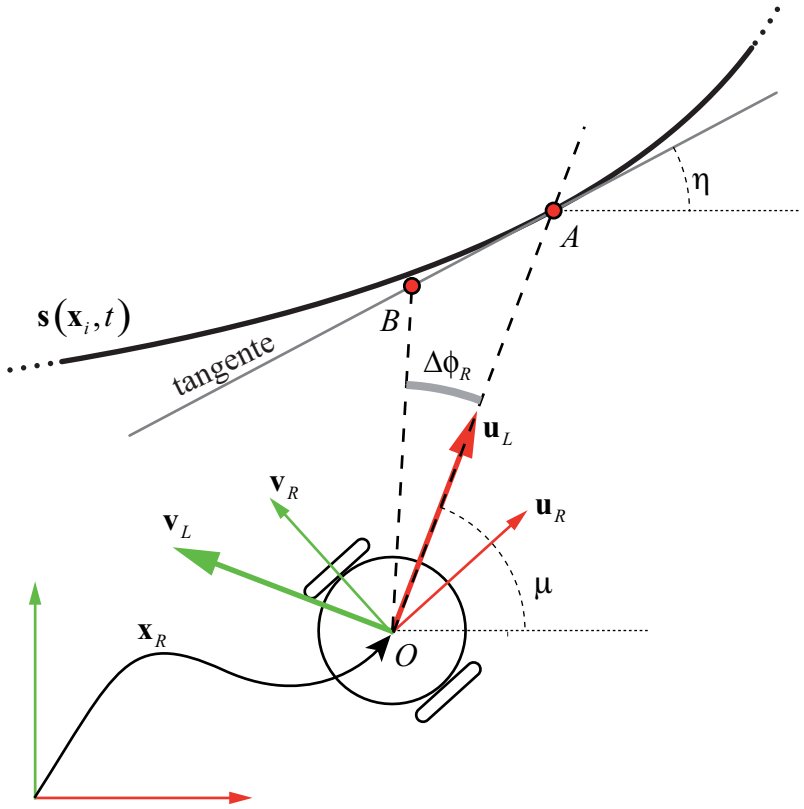


Figura 4.15: Representación esquemática de la relación $\frac{\Delta h}{\Delta \phi_R}$

$$\frac{\Delta h}{\Delta y_R} = \frac{1}{cos\,(\mu) \cdot tan\,(\eta) - sin\,(\mu)} \tag{4.44}$$

Finalmente, la figura 4.15 nos sirve para entender cómo variaría la medida obtenida ante pequeñas variaciones en la orientación del robot ϕ_R. En esta ocasión el robot no se mueve del punto O, pero cambia su orientación de manera que la medida esperada pasa a ser la definida por el segmento $\overline{OB}$. Podemos escribir las siguientes ecuaciones:

$$\overline{AB} \cdot sin\,(\eta) = h \cdot sin\,(\mu) - (h + \Delta h)\, sin\,(\mu + \Delta\phi_R) \tag{4.45}$$
$$\overline{AB} \cdot cos\,(\eta) = h \cdot cos\,(\mu) - (h + \Delta h)\, sin\,(\mu + \Delta\phi_R) \tag{4.46}$$

Teniendo en cuenta las siguientes relaciones

$$sin\,(\mu + \Delta\phi_R) = sin\,(\mu)\, cos\,(\Delta\phi_R) + cos\,(\mu)\, sin\,(\Delta\phi_R) \tag{4.47}$$
$$cos\,(\mu + \Delta\phi_R) = cos\,(\mu)\, cos\,(\Delta\phi_R) - sin\,(\mu)\, sin\,(\Delta\phi_R) \tag{4.48}$$
$$tan\,(\eta - \mu) = \frac{tan\,(\eta) - tan\,(\mu)}{1 + \tan\,(\eta)\, tan\,(\mu)} \tag{4.49}$$

y el hecho de que para $\Delta\phi_R$ lo suficientemente pequeño, $sin\,(\Delta\phi_R) \approx \Delta\phi_R$ y $cos\,(\Delta\phi_R) \approx 1$, podemos obtener

$$\frac{\Delta h}{\Delta\phi_R} \approx h \cdot \frac{1 + tan\,(\eta)\,tan\,(\mu)}{tan\,(\eta) - tan\,(\mu)} = \frac{h}{tan\,(\eta - \mu)} \tag{4.50}$$

Y finalmente, teniendo en cuenta cuál es el concepto básico de función derivada, podemos escribir:

$$\frac{\partial h}{\partial x_R} = \lim_{\Delta x_R \to 0} \frac{\Delta h}{\Delta x_R} = \frac{1}{sin\,(\mu)/tan\,(\eta) - cos\,(\mu)} \tag{4.51}$$

$$\frac{\partial h}{\partial y_R} = \lim_{\Delta y_R \to 0} \frac{\Delta h}{\Delta y_R} = \frac{1}{cos\,(\mu) \cdot tan\,(\eta) - sin\,(\mu)} \tag{4.52}$$

$$\frac{\partial h}{\partial \phi_R} = \lim_{\Delta \phi_R \to 0} \frac{\Delta h}{\Delta \phi_R} = \frac{\hat{z}}{tan\,(\eta - \mu)} \tag{4.53}$$

Las figura 4.16, 4.17 y 4.18 muestran comparativas entre las variaciones reales de la medida esperada ante diferentes perturbaciones del estado —tanto en la posición y orientación del robot como en los puntos de control que definen un spline de prueba—, y las variaciones que podrían predecirse utilizando las derivadas de las anteriores ecuaciones 4.30, 4.31, 4.51, 4.52 y 4.53. En los tres ejemplos, el B-spline viene dado por el vector de nodos $\Xi = \{0, 0, 0, 0, 1, 1, 1, 1\}$ y el polígono de control $\{\mathbf{x}_i, i = 0, \ldots, 3\}$ cuyas posiciones aparecen recogidas en la tabla 4.1. Para cada uno de los experimentos, el robot se ha colocado con diferentes orientaciones, de manera que el rayo láser intersecte a la curva en diferentes áreas. La zona sombreada en la figura 4.16 indica que no se produce intersección del rayo láser con la curva para esos valores de la orientación del robot.

$\mathbf{x}_0$	$\mathbf{y}_0$	$\mathbf{x}_1$	$\mathbf{y}_1$	$\mathbf{x}_2$	$\mathbf{y}_2$	$\mathbf{x}_3$	$\mathbf{y}_3$
-2	0	-2	3	2	1	3	4

Cuadro 4.1: Posiciones de los puntos de control de la curva empleada en las figuras 4.16, 4.17 y 4.18 (unidades en metros).

Experimento	ϕ_R	τ	$\mu = \phi_R + \tau$
4.16	30°	45°	75°
4.16	45°	45°	90°
4.16	60°	45°	105°

Cuadro 4.2: Posición angular del robot en las figuras 4.16, 4.17 y 4.18

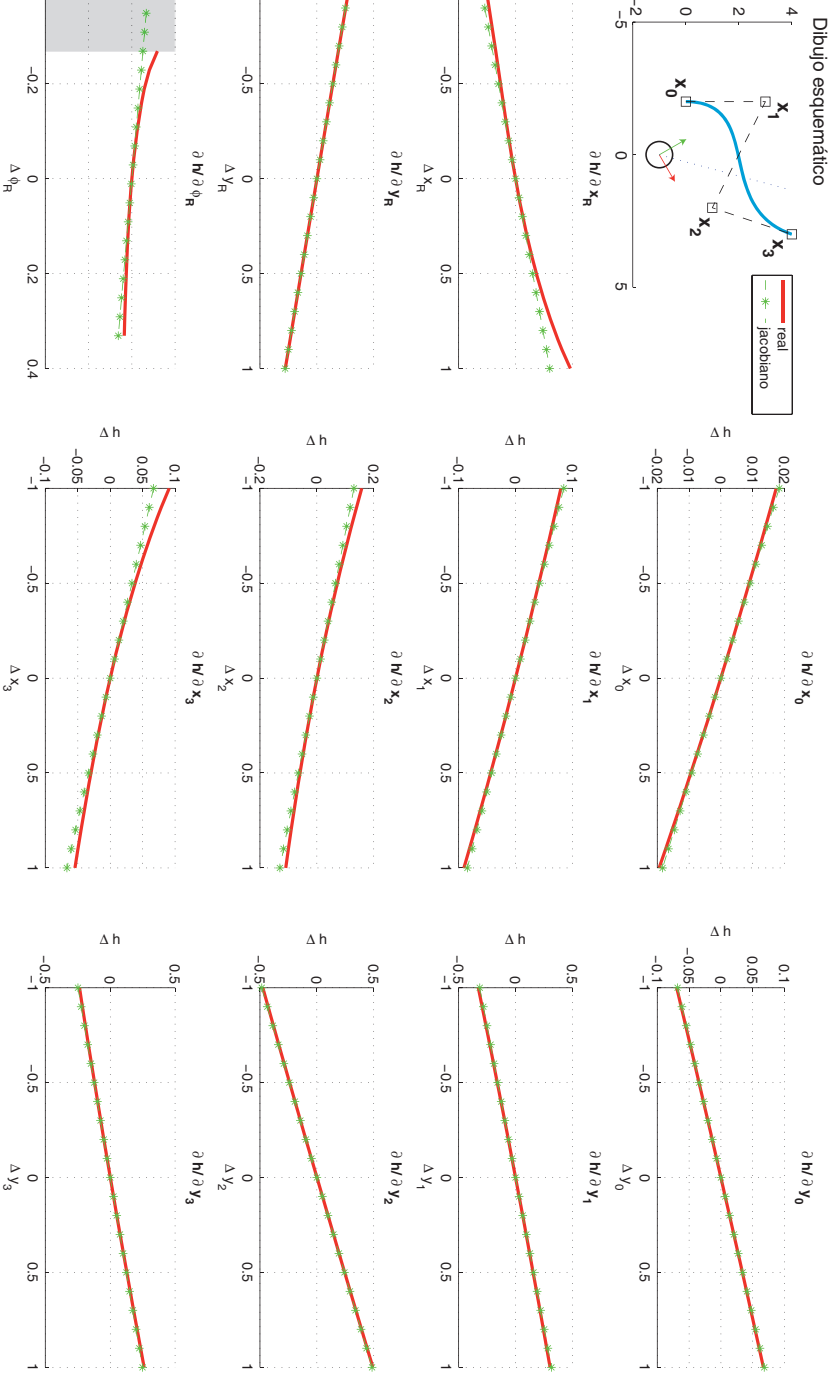


Figura 4.16: Comparativa entre las variaciones reales de la medida esperada, y las obtenidas utilizando los jacobianos sintetizados (I), para la configuración de prueba descrita en la tabla 4.1 y orientación del robot $\phi_R = 30^{\circ}$.

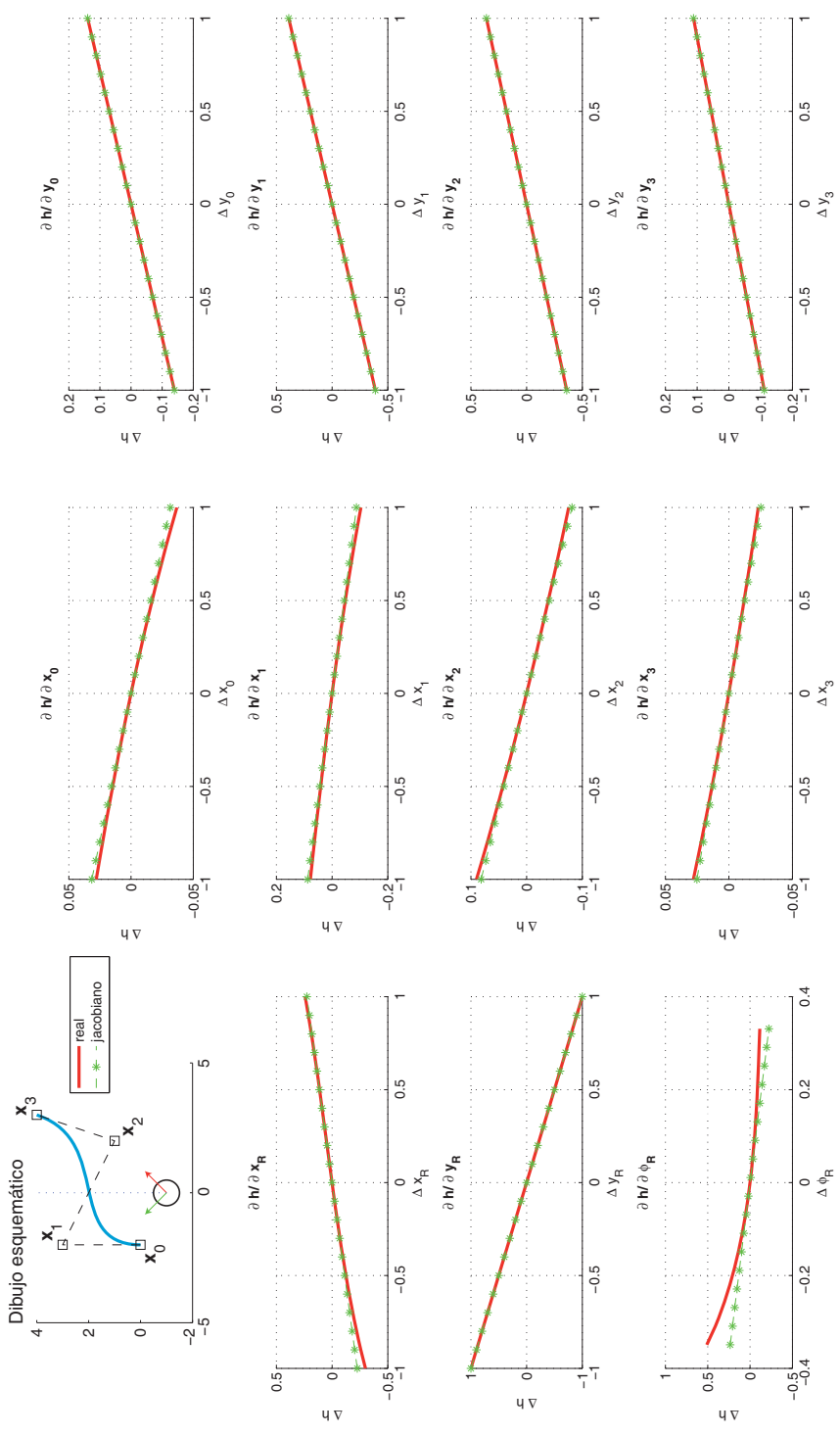


Figura 4.17: Comparativa entre las variaciones reales de la medida esperada, y las obtenidas utilizando los jacobianos sintetizados (II), para la configuración de prueba descrita en la tabla 4.1 y orientación del robot $\phi_R = 45°$.

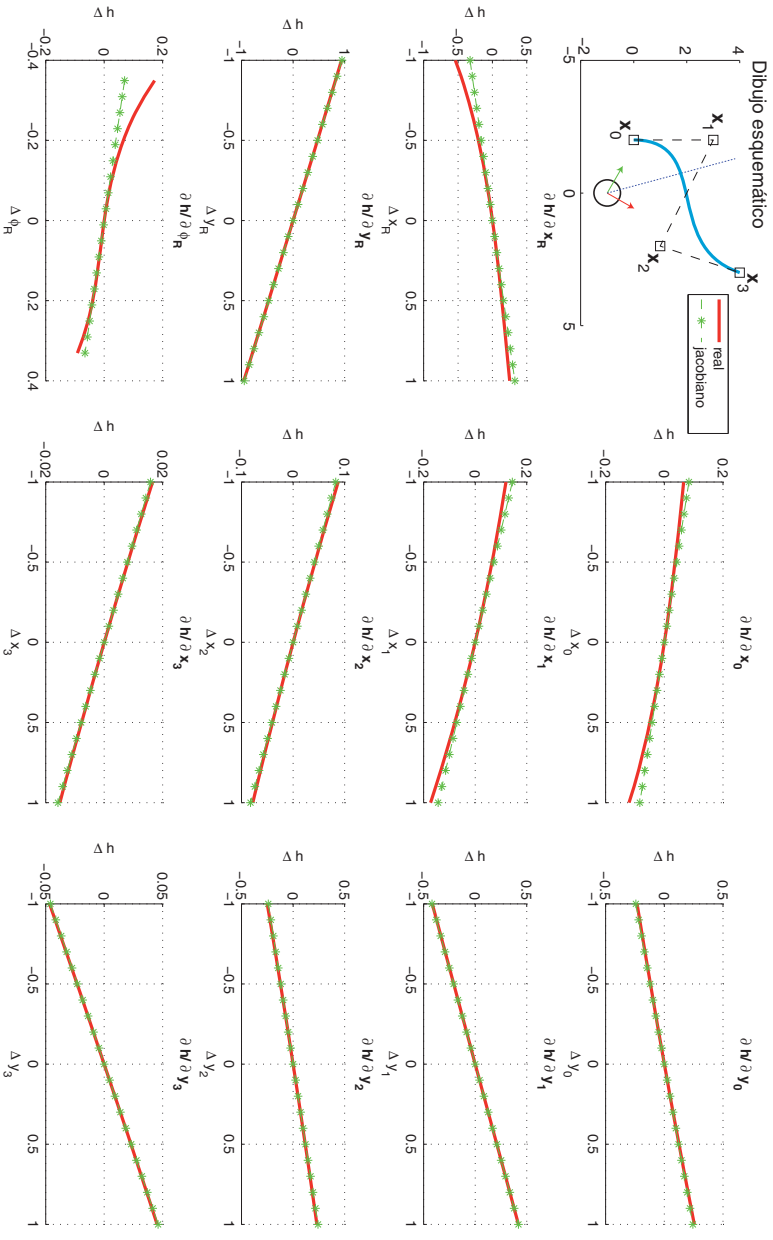


Figura 4.18: Comparativa entre las variaciones reales de la medida esperada, y las obtenidas utilizando los jacobianos sintetizados (III), para la configuración de prueba descrita en la tabla 4.1 y orientación del robot $\phi_R = 60°$.

4.5. Aplicando el Filtro de Kalman

En esta sección, se combinan todos los resultados obtenidos en los anteriores apartados en el marco de trabajo del filtro extendido de Kalman con el propósito de construir incrementalmente mapas de entornos modelados mediante splines cúbicos.

4.5.1. Etapa de predicción

Entre el instante k y el instante $k+1$ el robot realiza un movimiento dado por el vector

$$\mathbf{u}(k+1) = [u_x, u_y, u_\phi]^T \tag{4.54}$$

El vector $\mathbf{u}(k+1)$ expresa una variación en la posición y orientación del robot móvil, expresada en el sistema de referencia ligado al propio robot en el instante k. Lo más habitual es que en la práctica, dicho vector venga dado por las medidas odométricas del sistema sensorial propioceptivo de la máquina. Se tratará, por tanto, de medidas imperfectas, contaminadas por ruidos de distintas procedencias, y fuertemente dependientes de las características del sistema motriz del robot y de la superficie sobre la que se desenvuelve. Estas incertidumbres son modeladas asumiendo que el ruido que las afecta sigue una distribución gaussiana, cuya media es cero, y su varianza $\mathbf{Q}$ es ajustada empíricamente en la mayoría de los casos.

Teniendo esto en cuenta, la anterior transformación dada por la ecuación (4.54) se podrá descomponer como sigue:

$$\mathbf{u}(k+1) = \hat{\mathbf{u}}(k+1) + \mathbf{e}_u(k+1) \tag{4.55}$$
$$\mathbf{e}_u(k+1) \sim N(\mathbf{0}, \mathbf{Q}(k+1)) \tag{4.56}$$

Equivalentemente, la transformación que relaciona la nueva posición del robot con la que tenía en el instante inmediatamente anterior, seguirá la siguiente distribución:

$$\mathbf{u}(k+1) \sim N(\hat{\mathbf{u}}(k+1), \mathbf{Q}(k+1)) \tag{4.57}$$

La estimación a priori del estado en el instante $k+1$ viene dada por la siguiente expresión general:

$$\hat{\mathbf{x}}(k+1|k) = \mathbf{f}(\hat{\mathbf{x}}(k|k), \hat{\mathbf{u}}(k+1)) \tag{4.58}$$

Bajo la hipótesis de que el único objeto móvil es el robot (hipótesis de Markov), la anterior expresión se puede descomponer en las dos siguientes relaciones:

$$\hat{\mathbf{x}}_r\left(k+1|k\right) = \mathbf{f}_r\left(\hat{\mathbf{x}}_r\left(k|k\right), \hat{\mathbf{u}}\left(k+1\right)\right) \tag{4.59}$$

$$\hat{\mathbf{x}}_{s_i}\left(k+1|k\right) = \hat{\mathbf{x}}_{s_i}\left(k|k\right) \tag{4.60}$$

La covarianza del vector de estado del sistema en este momento, se puede calcular como sigue:

$$\begin{aligned}\mathbf{P}\left(k+1|k\right) = \mathbf{F}_x\left(k+1\right)\mathbf{P}\left(k|k\right)\mathbf{F}_x^T\left(k+1\right) + \\ +\mathbf{F}_u\left(k+1\right)\mathbf{Q}\left(k+1\right)\mathbf{F}_u^T\left(k+1\right)\end{aligned} \tag{4.61}$$

donde las matrices Jacobianas involucradas tienen las siguientes expresiones:

$$\mathbf{F}_x\left(k+1\right) = \begin{bmatrix} \frac{\partial \mathbf{f}_r}{\partial \mathbf{x}_r}\Big|_{\hat{\mathbf{x}}_r(k|k),\hat{\mathbf{u}}(k+1)} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{n_1} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{I}_{n_N} \end{bmatrix} \tag{4.62}$$

$$\mathbf{F}_u\left(k+1\right) = \begin{bmatrix} \frac{\partial \mathbf{f}_r}{\partial \mathbf{u}}\Big|_{\hat{\mathbf{x}}_r(k|k),\hat{\mathbf{u}}(k+1)} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix} \tag{4.63}$$

y la función $\mathbf{f}_r$ depende de la plataforma robótica móvil en consideración.

4.5.2. Etapa de actualización

Un vez que ha sido obtenida la medida esperada para cada una de las posiciones del láser a lo largo de su rango angular, la matriz de covarianza de la innovación de la medida viene dada por:

$$\begin{aligned}\mathbf{S}\left(k+1\right) = \mathbf{H}_{\mathbf{x}}\left(k+1\right)\mathbf{P}\left(k+1|k\right)\mathbf{H}_{\mathbf{x}}^T\left(k+1\right) + \\ +\mathbf{H}_{\mathbf{z}}\left(k+1\right)\mathbf{R}\left(k+1\right)\mathbf{H}_{\mathbf{z}}^T\left(k+1\right)\end{aligned} \tag{4.64}$$

$$\mathbf{S}\left(k+1\right) = \mathbf{H}_{\mathbf{x}}\left(k+1\right)\mathbf{P}\left(k+1|k\right)\mathbf{H}_{\mathbf{x}}^T\left(k+1\right) + \mathbf{R}\left(k+1\right) \tag{4.65}$$

donde el Jacobiano involucrado tiene la siguiente expresión:

$$\mathbf{H}_{\mathbf{x}}(k+1) = \left[\frac{\partial \mathbf{h}}{\partial \mathbf{x}_r} \; \mathbf{0} \ldots \mathbf{0} \; \frac{\partial \mathbf{h}}{\partial \mathbf{x}_{s_i}} \; \mathbf{0} \ldots \mathbf{0} \right] \tag{4.66}$$

En la ecuación anterior el término $\frac{\partial \mathbf{h}}{\partial \mathbf{x}_r}$ puede ser calculado haciendo uso de las ecuaciones 4.32, 4.33 y 4.34, y el término $\frac{\partial \mathbf{h}}{\partial \mathbf{x}_{s_i}}$ se calcula de 4.30 y 4.31.

La ganancia de Kalman tiene la siguiente expresión típica

$$\mathbf{W}(k+1) = \mathbf{P}(k+1|k)\,\mathbf{H}_{\mathbf{x}}^{T}(k+1)\,\mathbf{S}^{-1}(k+1) \tag{4.67}$$

Finalmente, la estimación actualizada del estado y su covarianza se calculan como sigue:

$$\hat{\mathbf{x}}(k+1|k+1) = \hat{\mathbf{x}}(k+1|k) + \mathbf{W}(k+1)\,\mathbf{h}(k+1) \tag{4.68}$$

$$\mathbf{P}(k+1|k+1) = [\mathbf{I} - \mathbf{W}(k+1)\,\mathbf{H}_x(k+1)]\,\mathbf{P}(k+1|k) \tag{4.69}$$

4.6. Extendiendo el Mapa

El mapa se construye incrementalmente según dos mecanismos claramente diferenciados:

1. **Agregando nuevos objetos**. Mediante este mecanismo, los objetos detectados por los sensores del robot, que no han sido asociados con ninguno de los objetos contenidos en el mapa, son inicializados dentro del vector de estado del sistema. Además de la extensión del vector de estado, es necesario ampliar la matriz de covarianzas para incluir la nueva información estocástica.

2. **Extendiendo objetos** que ya se encuentran contenidos en el mapa. Cuando un objeto detectado por el robot es asociado sólo parcialmente con uno o más elementos del mapa, es posible extender estos últimos con la nueva información adquirida. Serán necesarios mecanismos para:

   - insertar nuevos puntos de control si es preciso y
   - actualizar la matriz de covarianzas del sistema en consecuencia.

Cada uno de estos dos mecanismos es analizado en detalle en los dos siguientes apartados.

4.6.1. Inserción de nuevos objetos en el mapa

Cuando una nueva observación no es asociada con ninguno de las curvas contenidas en el mapa (tal y como sucede con los objetos $O_{k+1,1}$ y $O_{k+1,2}$ de la figura 4.5.c, o los objetos $O_{k+2,1}$ y $O_{k+2,2}$ de la figura 4.5.d), es oportuno considerarla como un nuevo elemento del mapa, y el spline que define su geometría ha de ser en consecuencia agregado al modelo. Por "agregar un spline al mapa" entenderemos en lo sucesivo "agregar los puntos de control del spline al vector de estado del sistema". Evidentemente, en el marco de trabajo del EKF esto también implica la necesidad de ampliar la matriz de covarianzas del sistema.

Dado un mapa que inicialmente contiene N objetos estáticos, y un conjunto de $q+1$ medidas $z_i \in \Re^2, i = p \dots p+q$ correspondientes al nuevo objeto detectado, el vector de estado aumentado del sistema tendrá la siguiente expresión:

$$\mathbf{x}^a = \mathbf{g}(\mathbf{x}, \mathbf{z}) \Leftrightarrow \begin{cases} \mathbf{x}_r^a & = \mathbf{x}_r \\ \mathbf{x}_{s_i}^a & = \mathbf{x}_{s_i}, \forall i = 1, \dots, N \\ \mathbf{x}_{s_{N+1}}^a & = \mathbf{g}_{s_{N+1}}(\mathbf{x}_r, \mathbf{z}) \end{cases} \tag{4.70}$$

Esto es, tanto el estado del robot como el de los N elementos contenidos en el mapa no se verá modificado por el hecho de incluir un nuevo elemento en este último. Por otra parte, el estado del nuevo elemento que se pretende insertar vendrá dado por las posiciones de los puntos de control de la curva que lo representa, calculadas como función del estado del robot $\mathbf{x}_r$ y el vector de medidas adquiridas, $\mathbf{z}$. Así, en la ecuación (4.70), la función $\mathbf{g}_{s_{N+1}}(\mathbf{x}_r, \mathbf{z})$ viene dada por la ecuación de ajuste de los $q+1$ nuevos puntos obtenidos por el sensor láser, tal como se describió en la sección 3.7.3:

$$\begin{bmatrix} x_{N+1,0} \\ \vdots \\ x_{N+1,n_N} \end{bmatrix} = \mathbf{\Phi} \begin{bmatrix} x_r + z_p \cos(\phi_r + \tau_p) \\ \vdots \\ x_r + z_{p+q} \cos(\phi_r + \tau_{p+q}) \end{bmatrix} \tag{4.71}$$

$$\begin{bmatrix} y_{N+1,0} \\ \vdots \\ y_{N+1,n_N} \end{bmatrix} = \mathbf{\Phi} \begin{bmatrix} y_r + z_p \sin(\phi_r + \tau_p) \\ \vdots \\ y_r + z_{p+q} \sin(\phi_r + \tau_{p+q}) \end{bmatrix} \tag{4.72}$$

La nueva matriz de covarianzas del vector de estado aumentado, una vez insertados los puntos de control correspondientes al nuevo objeto, será entonces:

$$\begin{aligned} \mathbf{P}^a(k+1|k+1) = {} & \mathbf{G}_{\mathbf{x}}(k+1|k+1)\, \mathbf{P}(k+1|k+1)\, \mathbf{G}_{\mathbf{x}}^T(k+1|k+1) + \\ & + \mathbf{G}_{\mathbf{z}}(k+1|k+1)\, \mathbf{R} \mathbf{G}_{\mathbf{z}}^T(k+1|k+1) \end{aligned} \tag{4.73}$$

donde la matriz $\mathbf{R}$ es la matriz de covarianzas de las medidas del láser, que será en general una matriz diagonal de la forma

$$\mathbf{R} = \sigma_L \cdot \mathbf{I}_{q+1} \tag{4.74}$$

Las matrices Jacobianas necesarias, $\mathbf{G_x} = \frac{\partial \mathbf{g}}{\partial \mathbf{x}}$ y $\mathbf{G_z} = \frac{\partial \mathbf{g}}{\partial \mathbf{z}}$, tendrán la siguiente comnfiguración:

$$\mathbf{G_x} = \begin{bmatrix} \mathbf{I}_r & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{n_1} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{I}_{n_N} \\ \hdashline \frac{\partial \mathbf{g}_{s_{N+1}}}{\partial \mathbf{x}_r} & \mathbf{0} & \dots & \mathbf{0} \end{bmatrix} \tag{4.75}$$

$$\mathbf{G_z} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \hdashline \frac{\partial \mathbf{g}_{S_{N+1}}}{\partial \mathbf{z}} \end{bmatrix} \tag{4.76}$$

siendo

$$\frac{\partial \mathbf{g}_{s_{N+1}}}{\partial \mathbf{x}_r} = \begin{bmatrix} \mathbf{\Phi} \begin{bmatrix} 1 & 0 & -z_p \sin \mu_p \\ \vdots & \vdots & \vdots \\ 1 & 0 & -z_{p+q} \sin \mu_{p+q} \end{bmatrix} \\ \hdashline \mathbf{\Phi} \begin{bmatrix} 0 & 1 & z_p \cos \mu_p \\ \vdots & \vdots & \vdots \\ 0 & 1 & z_{p+q} \sin \mu_{p+q} \end{bmatrix} \end{bmatrix} \tag{4.77}$$

$$\frac{\partial \mathbf{g}_{s_{N+1}}}{\partial \mathbf{z}} = \begin{bmatrix} \mathbf{\Phi} \begin{bmatrix} \cos \mu_p & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \cos \mu_{p+q} \end{bmatrix} \\ \hdashline \mathbf{\Phi} \begin{bmatrix} \sin \mu_p & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sin \mu_{p+q} \end{bmatrix} \end{bmatrix} \tag{4.78}$$

De esta manera, es posible obtener el vector de estado aumentado del sistema tras la inclusión de un nuevo objeto, como función del estado del sistema no aumentado —pero corregido tras la actualización del filtro extendido de Kalman—, y de las

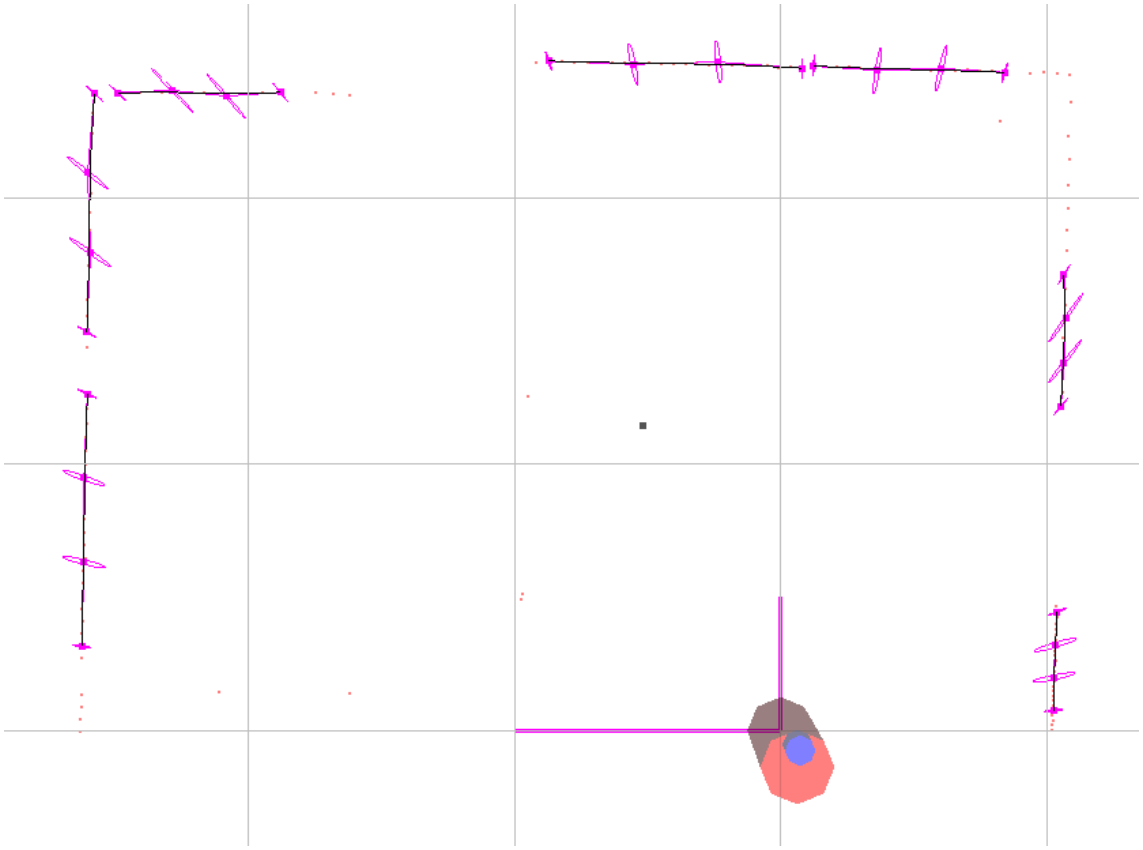


Figura 4.19: Inicialización de 7 objetos en el mapa. Las elipses representan las incertidumbre asociada a las posiciones de cada uno de los puntos de control que definen las nuevas curvas del mapa.

medidas obtenidas por el sensor láser para el nuevo objeto detectado. La sencilla relación lineal, escrita en forma matricial, entre las medidas adquiridas y los puntos de control del spline que las aproxima, permite la práctica y eficiente inclusión de la nueva información estocástica adquirida en la matriz de covarianzas global del sistema.

La figura 4.19 muestra un ejemplo con datos reales en el que 7 objetos son insertados en un mapa inicialmente vacío.

4.6.2. Extensión de objetos contenidos en el mapa

En el caso más frecuente, las observaciones obtenidas serán sólo parcialmente asociadas con algún objeto del mapa. Esto es precisamente lo que sucede con el objeto $O_{k+1,3}$ de la figura 4.5.c, caso que es reproducido en la figura 4.20(a) para mayor comodidad del lector. Esta situación indica que una nueva área inexplorada de un objeto del mapa está siendo detectada por los sensores del robot. Consecuentemente, el spline que modela este elemento ha de ser extendido con la nueva información adquirida por el sensor láser.

Así pues, tomemos como punto de partida el escenario mostrado en la figura 4.20. Este dibujo será empleado para explicar con precisión el proceso de extensión

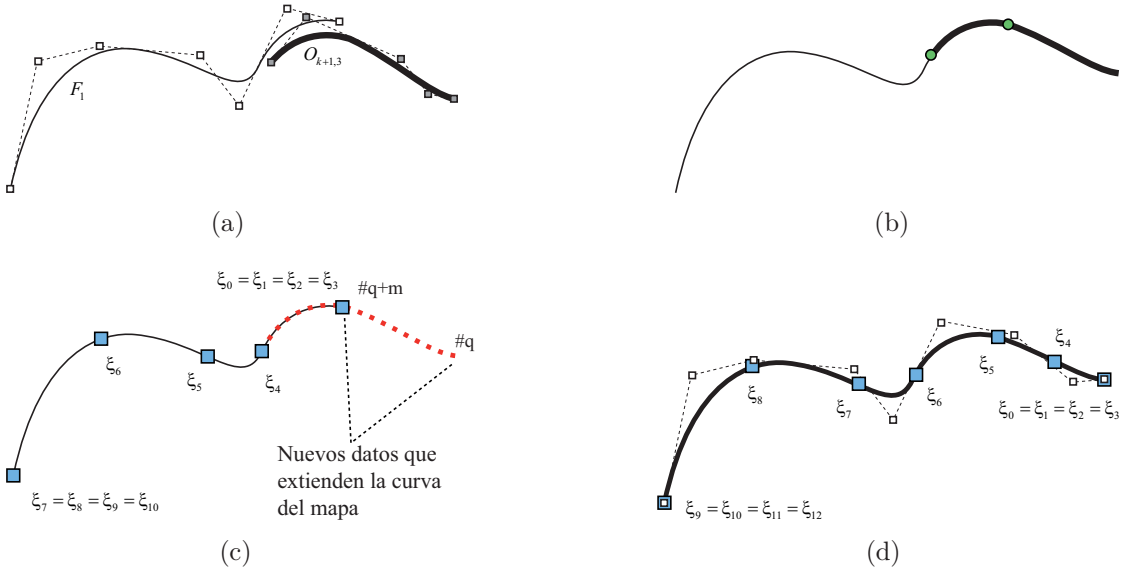


Figura 4.20: Extensión de un spline del mapa con nuevos datos. (a) Configuración inicial antes de la actualización del filtro. (b) Configuración tras la actualización del filtro y nueva correspondencia paramétrica. (c) Misma situación mostrada en la figura b), pero señalando las posiciones de los nodos de la curva del mapa y las medidas que dieron origen a la observación, $m + 1$ de las cuales se corresponden con la nueva zona detectada. (d) Configuración final de la curva del mapa, tras la extensión con la nueva información adquirida. Se aprecia la necesidad de extender el vector de nodos del spline original con nuevos elementos, para dar cabida a la nueva área explorada.

de un elemento del mapa con la nueva información adquirida por el sensor.

- En la figura 4.20(a) se muestra la situación de partida: El elemento del mapa F_1 ha sido parcialmente asociado con la observación $O_{k+1,3}$. Los datos correspondientes a la zona de solapamiento (apreciable en la figura 4.6) son empleados en la etapa de actualización del filtro extendido de Kalman para corregir el estado.

- La figura 4.20(b) muestra la situación una vez actualizado el estado del sistema. La posición relativa entre la observación y el mapa se ha corregido, y el solapamiento es más perfecto. Se procede a establecer una nueva correspondencia paramétrica, más precisa, del mismo modo que se explicó en la sección 4.2.2.

- La figura 4.20(c) muestra la misma configuración que la 4.20(b), pero en este caso se han marcado las posiciones de los nodos de la curva del mapa que se pretende extender, así como el conjunto de medidas del láser que dieron lugar a la observación correspondiente. Una fracción de estas medidas será empleada en el proceso de extensión de la curva.

- Finalmente, la figura 4.20(d) muestra la situación final, en la cual la curva del mapa ha sido extendida con la nueva información adquirida por el sensor. Cabe resaltar un par de aspectos para esta última figura:

- Ha sido necesario insertar dos puntos de control adicionales, con el objetivo de "hacer sitio" al nuevo tramo de curva detectado. Por consiguiente, se necesitan también dos nuevos puntos de control ya que la diferencia entre el número de nodos y el número de puntos de control de un spline ha de permanecer constante e igual al orden de la curva.
- Para el tipo de spline utilizado, los puntos de control inicial y final se corresponden, respectivamente, con el primer y último nodo de la curva.

Así pues, el problema de extender una entidad del mapa con nueva información, una vez finalizada la etapa de actualización del filtro, se reduce al de insertar la información geométrica y estocástica que contiene el conjunto de $m+1$ datos

$$\mathbf{d_i} = \begin{bmatrix} d_i^x \\ d_i^y \end{bmatrix} = \begin{bmatrix} x_r + z_i \cos \mu_i \\ y_r + z_i \sin \mu_i \end{bmatrix}, \quad \forall i = q, \ldots, q+m \tag{4.79}$$

correspondientes al nuevo tramo detectado, que no pertenecen a la zona de solapamiento con el objeto del mapa al que ha sido asociado. El vector de estado del sistema resultante de la extensión del objeto j-ésimo será obtenido del siguiente modo:

$$\mathbf{x}^e = \mathbf{g_e}\left(\mathbf{x}, \mathbf{z}\right) \Leftrightarrow \begin{cases} \mathbf{x}_r^e &= \mathbf{x}_r \\ \mathbf{x}_{s_i}^e &= \mathbf{x}_{s_i}, \quad \forall i \neq j \\ \mathbf{x}_{s_j}^e &= \mathbf{g}_{s_j}^e\left(\mathbf{x}_r, \mathbf{x}_j, \mathbf{z}_{q:q+m}\right) \end{cases} \tag{4.80}$$

Esto quiere decir que tanto el estado del robot como el de todos los elementos del mapa distintos al objeto extendido permanecen inalterados, mientras que los puntos de control de la curva extendida serán obtenidos como función $\mathbf{g}_{s_j}^e$ de los tres elementos siguientes:

- La posición y orientación del robot, $\mathbf{x}_r$.
- Los puntos de control del spline antes de ser extendido, $\mathbf{x}_{S_j}$.
- Las medidas obtenidas por el sensor láser, correspondientes a la nueva superficie detectada, $\mathbf{z}_{q:q+m}$.

Tanto la pose del robot, como los puntos de control del spline sin extender y las medidas del láser, son variables estocásticas. Será necesario por tanto integrar toda esta información, de manera que se actualice convenientemente no sólo la geometría del objeto elongado, sino también la matriz de covarianzas del sistema.

Con el objetivo de calcular la función $\mathbf{g}_{s_j}^e\left(\mathbf{x}_r, \mathbf{x}_j, \mathbf{z}\right)$, se seguirá un esquema de razonamiento similar al empleado para el ajuste de datos del láser en el caso de la inicialización de nuevos objetos en el mapa. Pero en este caso, se tendrán en cuenta los siguientes puntos:

1. **Que existe una descripción de partida** que representa geométricamente el tramo ya conocido del objeto. Dicha representación viene dada, como es sabido, por un conjunto de puntos de control y el vector de nodos correspondiente.

2. **Que será necesario modificar la configuración del vector de nodos original**, extendiendo el rango del parámetro independiente, de manera que la nueva información pueda ser acomodada en la curva resultante. Recordemos que para nosotros el parámetro independiente da una medida de la longitud de la curva, y que el vector de nodos define el rango de valores que puede tomar dicho parámetro; por tanto el hecho de elongar una curva con nueva información precisará de una cierta manipulación de dicho elemento definitorio del spline.

3. **Que la nueva información habrá de ser introducida** en la curva resultante, **estableciendo una parametrización para los nuevos puntos adquiridos por el sensor láser que sea congruente con la parametrización de la curva existente**. Esto es, la existencia de un modelo para el tramo ya conocido, comentada en el punto 1 de esta misma lista, define un cierto "sistema de referencia" que determina la posición de cualquier punto sobre la curva.

4. Finalmente, que los puntos de control que definen la curva extendida serán obtenidos como resultado de

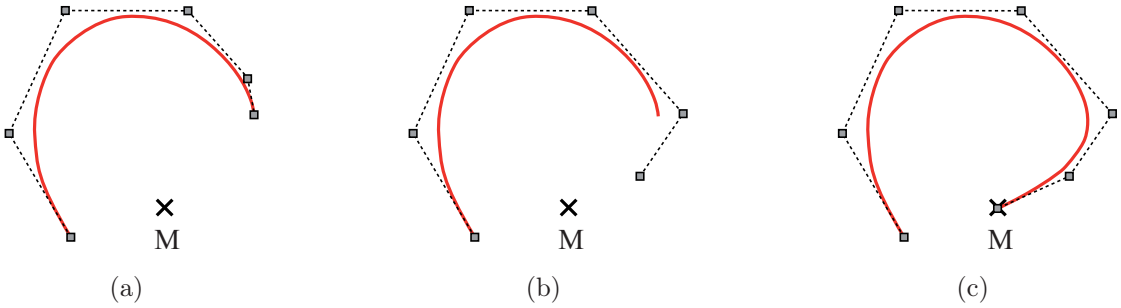
   a) haber modificado el vector de nodos original,

   b) haber introducido los nodos adicionales necesarios, con el fin de preservar la densidad de nodos predefinida para las curvas contenidas en el mapa, y

   c) aproximar los datos correspondientes al nuevo tramo detectado.

Como punto de partida se toma el trabajo de Shi-Min Hu *et al.* [103], el cual presenta un procedimiento de extensión de curvas B-spline utilizando el algoritmo de desenclavamiento del vector de nodos propuesto por Piegl y Tiller en [148]. Así, en [103] se plantea el problema de extender una curva definida por el conjunto de puntos de control $\{Q_i, i = 0 \dots n\}$ y el vector de nodos normalizado $\Xi = \{0, \dots, 0, \xi_k, \dots, \xi_n, 1, \dots, 1\}$ a uno o más puntos objetivo por los que se desea que pase la curva. La solución que allí se propone requiere la inserción de un nodo y un punto de control adicionales por cada uno de los nuevos puntos objetivo. Esta solución al problema de la extensión de una curva spline aparece representada esquemáticamente en la figura 4.21. Nótese que este esquema de extrapolación propuesto, consigue que las curvas resultantes pasen exactamente por los puntos objetivo seleccionados (punto M, en el caso de la figura 4.21).

Nuestro problema es obtener un spline elongado a partir un conjunto de $m + 1$ nuevos puntos —siendo m un número arbitrariamente grande— que representan el nuevo tramo descubierto por el sensor. Es evidente que la inserción de un punto de control adicional por cada nuevo dato individual obtenido va en contra de uno



(a) (b) (c)

Figura 4.21: Extensión de un spline cúbico a un único punto según la idea propuesta por Hu *et al.* en [103]. (a) Curva original y punto objetivo R hasta el cual se desea que esta se prolongue. (b) Curva original tras haber sido modificado su vector de nodos (desenclavamiento). (c) Curva resultante tras la extensión al punto objetivo, que pasa exactamente por R y contiene un nodo adicional y un punto de control adicional.

de los objetivos de esta tesis, que es precisamente la descripción de la complejidad del entorno con el mínimo número de elementos posible. Además deseamos que el vector de nodos de la curva extendida respete, en la medida de lo posible, el espaciado internodal prefijado. Así pues, el mecanismo de extensión que se presenta a continuación seguirá el mismo esquema empleado en el ajuste de un conjunto de puntos en el proceso de inicialización de un nuevo elemento del mapa (visto en la sección 4.6.1), resolviendo el sistema de ecuaciones planteado en el sentido de mínimos cuadrados.

Primeramente estudiaremos el algoritmo básico de desenclavamiento del vector de nodos tal y como proponen Piegl y Tiller en [148] y es aplicado por Hu *et al.* en [103].

El algoritmo básico de desenclavamiento

Se presenta a continuación el algoritmo recursivo básico que permite transformar una curva B-spline de orden κ definida por los $n+1$ puntos de control P_i y el vector de nodos enclavado Ξ:

$$P_i, \quad i = 0, \ldots, n \tag{4.81}$$

$$\Xi : \underbrace{\xi_0 = \ldots = \xi_{\kappa-1}}_{\kappa} \leq \xi_\kappa \leq \ldots \leq \xi_n \leq \underbrace{\xi_{n+1} = \ldots = \xi_{n+\kappa}}_{\kappa} \tag{4.82}$$

en otra geométricamente equivalente [148], pero definida por un vector de nodos desenclavado; *i.e.* con multiplicidad de sus nodos extremos diferente al orden κ de la curva. Así, hablaremos por convenio de *"desenclavamiento por la izquierda"* cuando esta multiplicidad se vea reducida en los nodos iniciales del vector Ξ $(\xi_0, \ldots, \xi_{\kappa-1})$, mientras que si esto sucede para los valores finales hablaremos de *"desenclavamiento por la derecha"* (nodos $\xi_{n+1} = \ldots = \xi_{k+\kappa}$). En el primer caso la curva resultante vendrá definida por los puntos de control L_i y el vector de nodos Ξ_l

$$L_i, \;\; i = 0, \ldots, n \tag{4.83}$$
$$\Xi_l : \;\; \bar{\xi}_0 \leq \ldots \leq \bar{\xi}_{\kappa-1} \leq \xi_\kappa \leq \ldots \leq \xi_n \leq \underbrace{\xi_{n+1} = \ldots = \xi_{n+\kappa}}_{\kappa} \tag{4.84}$$

mientras que en el segundo caso obtendremos los puntos de control R_i dado un vector de nodos Ξ_r

$$R_i, \;\; i = 0, \ldots, n \tag{4.85}$$
$$\Xi_r : \;\; \underbrace{\xi_0 = \ldots = \xi_{\kappa-1}}_{\kappa} \leq \xi_\kappa \leq \ldots \leq \xi_n \leq \bar{\xi}_{n+1} = \ldots = \bar{\xi}_{n+\kappa} \tag{4.86}$$

Cabe hacer especial hincapié en el hecho de que el algoritmo permite obtener los nuevos puntos de control (L_i ó R_i, $i = 0, \ldots, n$) a partir de la configuración inicial de la curva y dado el nuevo vector de nodos (Ξ_l ó Ξ_r, respectivamente) que se pretende utilizar. Este podrá ser definido como mejor convenga; en nuestro caso, intentando respetar el espaciado internodal escogido para los nodos que vayamos a separar.

El algoritmo recursivo que permite desenclavar un vector de nodos por la derecha, transformando una curva definida por las ecuaciones (4.81) y (4.82) en otra equivalente definida por las ecuaciones (4.85) y (4.86) es como sigue[4]:

1. Establecemos los valores iniciales para la iteración:

$$R_i^0 = P_i, \forall i = n-k+2, n-k+3, \ldots, n \tag{4.87}$$

2. Para $r = 1, 2, \ldots, \kappa - 2$ iteramos:

$$R_i^r = \begin{cases} R_i^{r-1} & \forall i = n-\kappa+2, \ldots, n-r \\ \frac{R_i^{r-1} - \left(1-\gamma_i^r\right) R_{i-1}^r}{\gamma_i^r} & i = n-r+1, \ldots, n \end{cases} \tag{4.88}$$

   siendo

$$\gamma_i^r = \frac{\xi_{n+1} - \xi_i}{\xi_{i+r+1} - \xi_i} \tag{4.89}$$

3. Finalmente, obtenemos los puntos de control de la curva modificada, definida por el vector de nodos Ξ_r del siguiente modo:

$$R_i = \begin{cases} P_i & \forall i = 0, \ldots, n-\kappa+1 \\ R_i^{\kappa-2} & i = n-\kappa+2, \ldots, n \end{cases} \tag{4.90}$$

[4]En [148] puede consultarse el algoritmo que consigue desenclavar la curva por la izquierda, obteniendo una representación dada por ecuaciones de la forma (4.83) y (4.84)

Adaptación del algoritmo

Una vez más nos encontramos frente a un engorroso algoritmo recursivo que, si bien no resulta computacionalmente prohibitivo, dista mucho de representar una solución atractiva. Son preferibles los algoritmos que proporcionan soluciones analíticas cerradas, computables en un sólo paso; especialmente cuando las relaciones funcionales que establecen entre las diferentes variables involucradas son lineales.

Esta linealidad se vuelve especialmente atractiva en el caso que nos ocupa. Nuestras variables son de carácter aleatorio, modeladas mediante distribuciones de probabilidad gaussianas, y es necesario propagar esta información estocástica desde las variables de entrada del algoritmo —los puntos de control de la curva original y los nuevos datos adquiridos por el sensor— a las variables de salida —los nuevos puntos de control de la curva extendida resultante—.

A continuación mostraremos cómo es posible manipular las ecuaciones del algoritmo propuesto en el apartado anterior, para obtener una relación funcional mucho más simple y fácil de implementar entre los puntos de control de la curva inicial, y los puntos de control de la curva resultante. Veamos, por ejemplo, cómo quedan las cosas en el caso de trabajar con splines cúbicos ($\kappa = 4$), particularizando el algoritmo recursivo para este orden concreto:

1. Establecemos los valores iniciales:

$$R_i^0 = P_i, \forall i = n-2, n-1, n \tag{4.91}$$

2. Para $r = 1$ tenemos:

$$R_{n-2}^1 = R_{n-2}^0 = P_{n-2} \tag{4.92}$$
$$R_{n-1}^1 = R_{n-1}^0 = P_{n-1} \tag{4.93}$$
$$\begin{aligned} R_n^1 &= \frac{R_n^0 - (1-\gamma_n^1)\, R_{n-1}^1}{\gamma_n^1} \\ &= \frac{P_n - (1-\gamma_n^1)\, P_{n-1}}{\gamma_n^1} \end{aligned} \tag{4.94}$$

siendo

$$\gamma_n^1 = \frac{\xi_{n+1} - \xi_n}{\xi_{n+2} - \xi_n} \tag{4.95}$$

3. Para $r = 2$ nos queda:

$$R^2_{n-2} = R^1_{n-2} = P_{n-2} \tag{4.96}$$

$$\begin{aligned} R^2_{n-1} &= \frac{R^1_{n-1} - \left(1 - \gamma^2_{n-1}\right) R^2_{n-2}}{\gamma^2_{n-1}} \\ &= \frac{P_{n-1} - \left(1 - \gamma^2_{n-1}\right) P_{n-2}}{\gamma^2_{n-1}} \end{aligned} \tag{4.97}$$

$$\begin{aligned} R^2_n &= \frac{R^1_n - (1 - \gamma^2_n) R^2_{n-1}}{\gamma^2_n} \\ &= \frac{1}{\gamma^2_n}\left[\frac{P_n - (1 - \gamma^1_n) P_{n-1}}{\gamma^1_n}\right] - \frac{1 - \gamma^2_n}{\gamma^2_n}\left[\frac{P_{n-1} - \left(1 - \gamma^2_{n-1}\right) P_{n-2}}{\gamma^2_{n-1}}\right] \\ &= \Gamma^2_n \Gamma^2_{n-1} P_{n-2} - \left[\frac{\Gamma^2_n}{\gamma^2_{n-1}} + \frac{\Gamma^1_n}{\gamma^2_n}\right] P_{n-1} + \frac{1}{\gamma^2_n \gamma^1_n} P_n \end{aligned} \tag{4.98}$$

siendo

$$\gamma^2_{n-1} = \frac{\xi_{n+1} - \xi_{n-1}}{\xi_{n+2} - \xi_{n-1}} \tag{4.99}$$

$$\gamma^2_n = \frac{\xi_{n+1} - \xi_n}{\xi_{n+3} - \xi_n} \tag{4.100}$$

y definiendo el parámetro Γ^r_i como:

$$\Gamma^r_i = \frac{1 - \gamma^r_i}{\gamma^r_i}, \forall i, \forall r \tag{4.101}$$

4. Finalmente, obtenemos los puntos de control que definen la curva generada con el nuevo vector de nodos Ξ_r como:

$$R_i = P_i, \quad \forall i = 0, \ldots, n-2 \tag{4.102}$$

$$R_{n-1} = -\Gamma^2_{n-1} P_{n-2} + \frac{1}{\gamma^2_{n-1}} P_{n-1} \tag{4.103}$$

$$R_n = \Gamma^2_n \Gamma^2_{n-1} P_{n-2} - \left(\frac{\Gamma^2_n}{\gamma^2_{n-1}} + \frac{\Gamma^1_n}{\gamma^2_n}\right) P_{n-1} + \frac{1}{\gamma^2_n \gamma^1_n} P_n \tag{4.104}$$

Las anteriores ecuaciones pueden escribirse de manera más compacta en forma matricial:

$$\begin{bmatrix} R_0 \\ \vdots \\ R_{n-1} \\ R_n \end{bmatrix} = \left[\begin{array}{c|ccc} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & -\Gamma^2_{n-1} & \frac{1}{\gamma^2_{n-1}} & 0 \\ \mathbf{0} & \Gamma^2_n \Gamma^2_{n-1} & -\left(\frac{\Gamma^2_n}{\gamma^2_{n-1}} + \frac{\Gamma^1_n}{\gamma^2_n}\right) & \frac{1}{\gamma^2_n \gamma^1_n} \end{array}\right] \begin{bmatrix} P_0 \\ \vdots \\ P_{n-2} \\ P_{n-1} \\ P_n \end{bmatrix} \tag{4.105}$$

Del mismo modo, si lo que se desea es desenclavar el vector de nodos por la izquierda, convirtiendo en vector de nodos Ξ (4.82)en el vector de nodos Ξ_l (4.84), se pueden obtener expresiones similares para los nuevos puntos de control L_i:

$$L_0 = \frac{1}{\omega_0^1 \omega_0^2} P_0 + \left(\frac{\Omega_0^2}{\omega_1^2} + \frac{\Omega_0^1}{\omega_0^2} \right) P_1 + \Omega_0^2 \Omega_0^1 P_2 \tag{4.106}$$

$$L_1 = \frac{1}{\omega_1^2} P_1 - \Omega_1^2 P_2 \tag{4.107}$$

$$L_i = P_i, \quad \forall i = 2, \ldots, n \tag{4.108}$$

o, equivalentemente, en forma matricial:

$$\begin{bmatrix} L_0 \\ L_1 \\ \vdots \\ L_n \end{bmatrix} = \left[\begin{array}{ccc|c} \frac{1}{\omega_0^1 \omega_0^2} & \left(\frac{\Omega_0^2}{\omega_1^2} + \frac{\Omega_0^1}{\omega_0^2} \right) & \Omega_0^2 \Omega_0^1 & \mathbf{0} \\ 0 & \frac{1}{\omega_1^2} & -\Omega_1^2 & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{array}\right] \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ \vdots \\ P_n \end{bmatrix} \tag{4.109}$$

siendo en este caso los coeficientes necesarios:

$$\omega_i^j = \frac{\bar{\xi}_{k-1} - \bar{\xi}_{i+k}}{\bar{\xi}_{i+k-j-1} - \bar{\xi}_{i+k}} \tag{4.110}$$

$$\Omega_i^j = \frac{1 - \omega_i^j}{\omega_i^j} \tag{4.111}$$

Extensión de objetos

Una vez obtenidas las ecuaciones que permiten el desenclavamiento del vector de nodos (por la derecha o por la izquierda, según convenga), es fácil obtener las ecuaciones para extender cualquier elemento del mapa a medida que se van adquiriendo nuevas medidas de áreas desconocidas previamente.

Así, los resultados obtenidos en la anterior sección se pueden combinar con la metodología propuesta en la sección 3.7.3 para obtener los splines extendidos a medida que se obtienen nuevos datos teniendo en cuenta los siguientes aspectos:

- Es necesario establecer una parametrización para los nuevos datos que sea coherente con el spline sobre el que se van a integrar. Esta información puede ser fácilmente obtenida de la etapa de asociación de datos previamente explicada.

- El vector de nodos necesita ser desenclavado, pero también puede ser necesario extenderlo con un número a determinar de nodos extra, de manera que se 'haga sitio' para los nuevos datos adquiridos y que corresponden a un nuevo tramo de objeto detectado. Tanto el número de nuevos nodos, como su espaciado, es determinado teniendo en cuenta la densidad de nodos predeterminada. Los extremos del vector de nodos han de ser dejados enclavados al final del proceso, repitiendo los nodos extremos hasta alcanzar la multiplicidad de las curvas utilizadas (κ). Hay que tener en cuenta que habrá que insertar tantos nuevos puntos de control como nodos nuevos se haya insertado.

De esta manera, el sistema de ecuaciones 3.17 es escrito para los nuevos datos, extendido utilizando las ecuaciones (4.105) y/o (4.109) según sea necesario, y su solución de mínimos cuadrados proporciona una relación lineal en forma matricial entre los antiguos puntos de control $\mathbf{x}_j$ y los nuevos datos obtenidos $\mathbf{d}_i$, y los nuevos puntos de control del spline extendido $\mathbf{x}_j^e$:

$$\begin{bmatrix} x^e_{j,0} \\ \vdots \\ x^e_{j,n_j+p} \end{bmatrix} = \mathbf{\Phi^e} \begin{bmatrix} d^x_q \\ \vdots \\ d^x_{q+m} \\ x_{j,0} \\ \vdots \\ x_{j,n_j} \end{bmatrix}, \begin{bmatrix} y^e_{j,0} \\ \vdots \\ y^e_{j,n_j+p} \end{bmatrix} = \mathbf{\Phi^e} \begin{bmatrix} d^y_q \\ \vdots \\ d^y_{q+m} \\ y_{j,0} \\ \vdots \\ y_{j,n_j} \end{bmatrix} \tag{4.112}$$

Nótese que una vez escogido un vector de nodos desenclavado para el spline extendido, la matriz Φ^e de extensión puede ser considerada constante. La nueva matriz de covarianzas después de extender el spline j-ésimo se podrá obtener por lo tanto como sigue:

$$\mathbf{P}^e = \mathbf{G}^e_{\mathbf{x}} \mathbf{P} \mathbf{G}^{eT}_{\mathbf{x}} + \mathbf{G}^e_{\mathbf{z}} \mathbf{R} \mathbf{G}^{eT}_{\mathbf{z}} \tag{4.113}$$

donde los jacobianos involucrados $\mathbf{G}^e_{\mathbf{x}} = \frac{\partial \mathbf{g}^e}{\partial \mathbf{x}}$ y $\mathbf{G}^e_{\mathbf{z}} = \frac{\partial \mathbf{g}^e}{\partial \mathbf{z}}$ tienen el siguiente aspecto:

$$\mathbf{G}^e_{\mathbf{x}} = \begin{bmatrix} \mathbf{I}_r & \mathbf{0} & \dots & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{n_1} & \dots & \mathbf{0} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\mathbf{g}^e_{s_j}}{\partial \mathbf{x}_r} & \vdots & \ddots & \frac{\partial \mathbf{g}^e_{s_j}}{\partial \mathbf{x}_{s_j}} & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} & \dots & \mathbf{I}_{n_N} \end{bmatrix}, \mathbf{G}^e_{\mathbf{z}} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \vdots \\ \frac{\partial \mathbf{g}^e_{s_j}}{\partial \mathbf{z}} \\ \vdots \\ \mathbf{0} \end{bmatrix} \tag{4.114}$$

siendo

$$\frac{\partial \mathbf{g}^e_{s_j}}{\partial \mathbf{x}_r} = \begin{bmatrix} \mathbf{\Phi}^e \begin{bmatrix} 1 & 0 & -z_p \sin \mu_p \\ \vdots & \vdots & \vdots \\ 1 & 0 & -z_{p+q} \sin \mu_{p+q} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \\ \mathbf{\Phi}^e \begin{bmatrix} 0 & 1 & z_p \cos \mu_p \\ \vdots & \vdots & \vdots \\ 0 & 1 & z_{p+q} \sin \mu_{p+q} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \end{bmatrix}, \frac{\partial \mathbf{g}^e_{s_j}}{\partial \mathbf{x}_{s_j}} = \begin{bmatrix} \mathbf{\Phi}^e \begin{bmatrix} \mathbf{0} \\ \hline \mathbf{I} \end{bmatrix} \\ \mathbf{\Phi}^e \begin{bmatrix} \mathbf{0} \\ \hline \mathbf{I} \end{bmatrix} \end{bmatrix}$$

y

$$\frac{\partial \mathbf{g}^e_{s_j}}{\partial \mathbf{z}} = \begin{bmatrix} \mathbf{\Phi}^e \begin{bmatrix} \cos \mu_p & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \cos \mu_{p+q} \\ \hline \mathbf{0} & \dots & \mathbf{0} \end{bmatrix} \\ \mathbf{\Phi}^e \begin{bmatrix} \sin \mu_p & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sin \mu_{p+q} \\ \hline \mathbf{0} & \dots & \mathbf{0} \end{bmatrix} \end{bmatrix}$$

La figura 4.22 muestra un ejemplo en el que la función definida como

$$y = 2t \cdot sin\,(t/2)$$

es ajustada secuencialmente dentro del intervalo $[-3, 10]$.

Primeramente la función es muestreada en el intervalo $[0, 4]$, obteniéndose el spline s_1. A continuación se muestrea la curva en el intervalo $[4, 10]$, de manera que la curva s_1 es extendida con la nueva información, obteniéndose una representación paramétrica para el intervalo $[-3, 10]$. Finalmente, se obtiene un tercer conjunto de muestras en el intervalo $[-3, 0]$, de modo que la curva s_2 es extendida, en este caso por la izquierda, para obtener la curva s_3 que representa a la función original en el intervalo $[-3, 10]$. En cada uno de los tres pasos, la función es muestreada cada 0,05 unidades, y contaminada con ruido normal de media 0 y desviación típica $\sigma = 0{,}2$ unidades:

$$y_s = 2t_s \cdot sin\,(t_s/2) + \epsilon_s$$
$$\epsilon_s \sim N\left(0, 2 \cdot 10^{-1}\right)$$

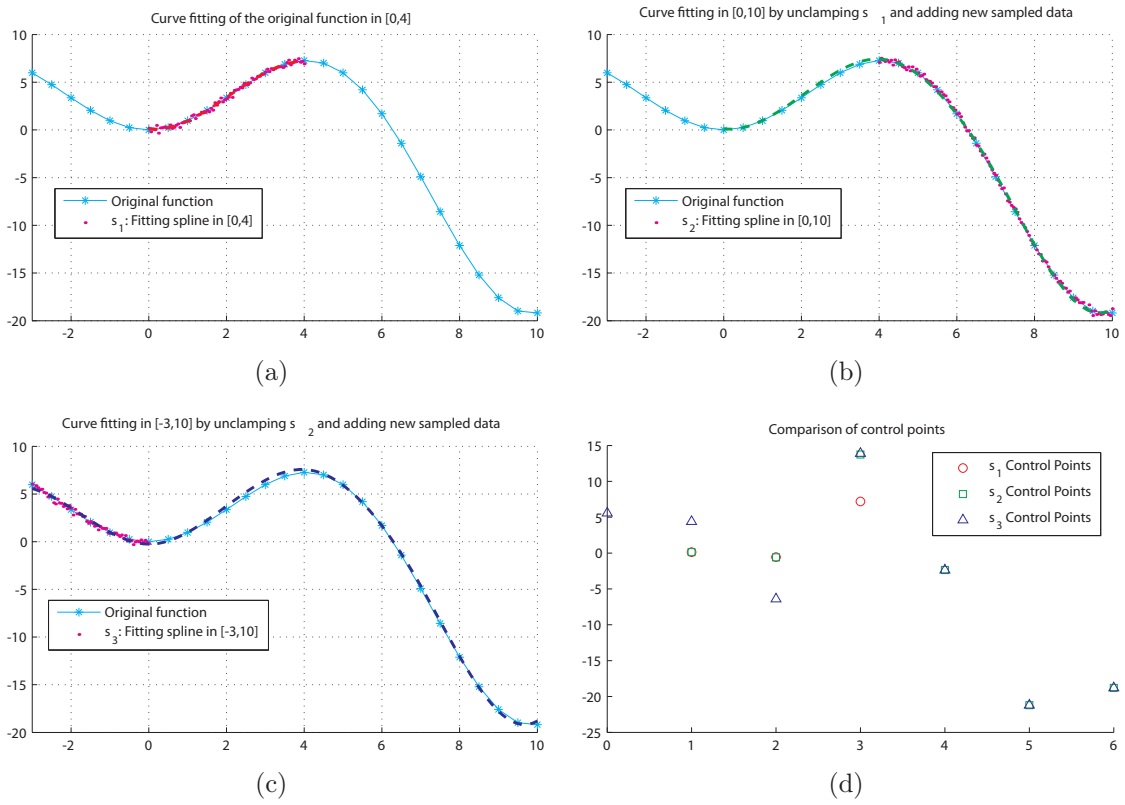


Figura 4.22: Ajuste secuencial de la función $y = 2t \cdot sin\,(t/2)$ en el intervalo $[-3, 10]$

4.7. Conclusiones

En este capítulo se han presentado las principales contribuciones de esta tesis, que permiten fusionar exitosamente la teoría de las curvas spline con el marco de trabajo SLAM-EKF. Se consigue así una nueva representación paramétrica de geometrías complicadas, que proporciona las siguientes ventajas respecto a algoritmos existentes:

- No es necesario depender de una geometría específica a ser detectada por los sensores.
- Los datos son sometidos a un preprocesamiento previo que elimina los dinámicos, y aproxima los fragmentos obtenidos mediante curvas spline. Esta representación permite razonar sobre los objetos contenidos en el mapa calculando distancias, curvaturas, puntos de inflexión...
- Los puntos de control que definen cada una de las curvas, encapsulan la información estocástica asociada a las medidas correspondientes, que le han dado origen.
- Esta formulación es idónea para ser distribuida en forma de un vector de estado y su correspondiente matriz de covarianzas, haciéndola apropiada para encajar en el marco de trabajo SLAM-EKF.

Al ser una manera completamente nueva de representar información espacial en el campo del SLAM y, más concretamente, en su solución basada en EKF, ha sido necesario desarrollar toda la algorítmica y procedimientos necesarios para poder manipular los datos y construir un mapa. En concreto, en este capítulo se han presentado las siguientes contribuciones originales:

- Se han presentado técnicas para manipular los datos brutos proporcionados por un sensor típico en robótica móvil como es el escáner láser. El sencillo mecanismo de segmentación permite obtener fragmentos individuales de superficies que se separan en función de la presencia de esquinas.

- Se ha mostrado la manera de realizar una asociación de datos robusta entre las medidas obtenidas y las curvas contenidas en el mapa. Esta asociación es crucial no sólo para el funcionamiento del algoritmo, sino también para establecer una correspondencia paramétrica entre ambos conjuntos de curvas que será de utilidad en la elongación de objetos contenidos en el mapa.

- Se ha formulado un adecuado modelo de estado, formado por la posición del robot y cada uno de los puntos de control de las curvas que componen el mapa. Dichos puntos de control encapsulan la incertidumbre asociada a las medidas obtenidas, y permiten establecer correlaciones entre el robot y todas las curvas del mapa, y entre las curvas entre sí.

- Se ha formulado un adecuado modelo de observación que permite explotar individualmente la información asociada a cada una de las medidas obtenidas por el sensor. Esto ha sido necesario por el hecho de que los puntos de control no son observables. El modelo de observación propuesto está inspirado conceptualmente en las técnicas de trazado de rayos utilizadas en el mundo de los gráficos por computador, pero adaptadas al caso particular del problema que nos ocupa.

- A partir del modelo de observación propuesto se ha mostrado cómo obtener las derivadas que permiten construir los jacobianos necesarios para utilizar un filtro extendido de Kalman en el proceso de estimación. De hecho, el modelo de observación ha sido pensado teniendo en cuenta la necesidad de obtener estos jacobianos.

- Se ha mostrado la estructura de las ecuaciones involucradas en el proceso de estimación recursiva del mapa mediante EKF.

- Se proporcionan los algoritmos necesarios para inicializar en el mapa con los nuevos objetos detectados, así como para prolongar objetos ya existentes a medida que nuevas áreas son descubiertas.

- Para conseguir el objetivo de elongar progresivamente objetos contenidos en el mapa, ha sido necesario adaptar un algoritmo de desenclavamiento que permite modificar el vector de nodos de una curva para dar cabida a nuevos datos. Las simples ecuaciones propuestas, permiten obtener relaciones lineales entre los antiguos puntos de control y los puntos de control de la curva alargada. La

combinación de estas ecuaciones con las del ajuste de datos básico presentado en el capítulo 3 permite, una vez más, obtener relaciones lineales entre los puntos de control antes y después de prolongar los objetos del mapa.

Capítulo 5

Modelado Basado en Descomposición en Mapas Locales

Como ya se vio en el capítulo 2, uno de los mayores inconvenientes de la solución del problema SLAM basada en la utilización de un filtro extendido de Kalman, es el coste computacional del algoritmo. Al crecer éste cuadráticamente con el número de objetos contenidos en el mapa (puntos de control, en nuestro caso), su aplicación se ve limitada a mapas formados por unos pocos cientos de ellos.

En este capítulo veremos cómo es posible solventar este inconveniente mediante la descomposición del problema global en unidades de tamaño limitado. En este marco de trabajo, cobrará especial importancia el análisis de las propiedades geométricas de los diferentes mapas construidos. Dicho análisis permitirá la extracción de características que serán empleadas a la hora de establecer relaciones entre los diferentes submapas generados, de manera que su estructura global pueda ser corregida adecuadamente.

5.1. Introducción

Uno de los principales inconvenientes de cualquier implementación práctica de una solución al problema del SLAM basada en la utilización del filtro extendido de Kalman —junto con la inevitable aparición de inconsistencia en la estimación del estado del sistema [44]— es su coste computacional. Dicho coste es cuadrático con el número de elementos contenidos en el mapa. En este sentido, se han realizado importantes trabajos para paliar este problema [60, 116].

Sin embargo, la solución más sencilla e inmediata parece ser la descomposición del mapa en diferentes submapas de tamaño limitado, de manera que la aplicación de un filtro extendido de Kalman a la construcción de cada uno de ellos sea un problema más fácilmente atacable [34, 71]. Siguiendo este enfoque, el mapa completo estaría formado por la combinación de las informaciones parciales contenidas en cada uno de los mapas construidos monolíticamente mediante un único filtro extendido de Kalman.

Por otra parte, la búsqueda de una correspondencia entre los diferentes mapas construidos durante el modelado de un entorno de grandes dimensiones, es aún un problema no resuelto. Así, sería deseable disponer de algún tipo de mecanismo que permitiera establecer correspondencias entre características contenidas en los diferentes submapas y que se corresponden, de hecho, con el mismo elemento del entorno. Dicho problema es conceptualmente equivalente al de la asociación de datos entre un conjunto de observaciones relativas al sistema de referencia del robot, y los objetos representados en un mapa del entorno. Este último problema es conocido como "localización global" o "problema del robot secuestrado", y puede ser enunciado del siguiente modo:

> "Dado un vehículo en una posición desconocida, y un mapa del entorno, utilizar un conjunto de medidas tomadas por los sensores de abordo para determinar la localización del robot en el mapa."

En SLAM, la solución de este problema también es necesaria para inicializar el robot en un mapa previamente construido, recuperarse de errores de localización o cerrar grandes bucles de manera robusta. En el caso que nos ocupa, considerando que cada uno de los submapas posee un sistema de referencia en el que están expresados los objetos contenidos en él, el objetivo es obtener un conjunto de asociaciones entre los elementos de ambos mapas que permitan determinar su posición relativa. Así, podríamos referirnos a este problema como el de el "submapa secuestrado", enunciándolo del siguiente modo:

> "Dada una representación local del entorno (submapa) en una posición desconocida, y otra representación local cuya posición sí se conoce (o se asume cierta, o simplemente se toma como referencia), utilizar el conjunto de objetos que son comunes a ambas representaciones para determinar su posición relativa."

El coste computacional de esta búsqueda es exponencial en el tiempo con el número de características consideradas. Además, existen numerosos factores que pueden complicar esta búsqueda, como el error o la incertidumbre en la localización de un objeto, observaciones espurias, o la presencia de simetrías en el entorno.

Para ilustrar estas ideas, observemos la situación representada en la figura 5.1, que reproduce un didáctico ejemplo extraído de [87]. En este caso se dispone de un modelo formado por 6 objetos (segmentos, en este caso, en la figura 5.1(a)) $F_1, \ldots, F_6$ y un conjunto de tres observaciones O_1, O_2, O_3 (figura 5.1(b)). Este segundo conjunto podría estar formado por objetos detectados en un instante por el robot, o por objetos pertenecientes a una segunda representación local del entorno obtenida independientemente de la otra (*i.e.* otro submapa).

El problema que se plantea es el de establecer alguna correspondencia entre ambas representaciones, que empareje los objetos contenidos en ellas como representantes de la misma entidad física del entorno. Más formalmente, el propósito de la asociación de datos no es otro que producir una hipótesis

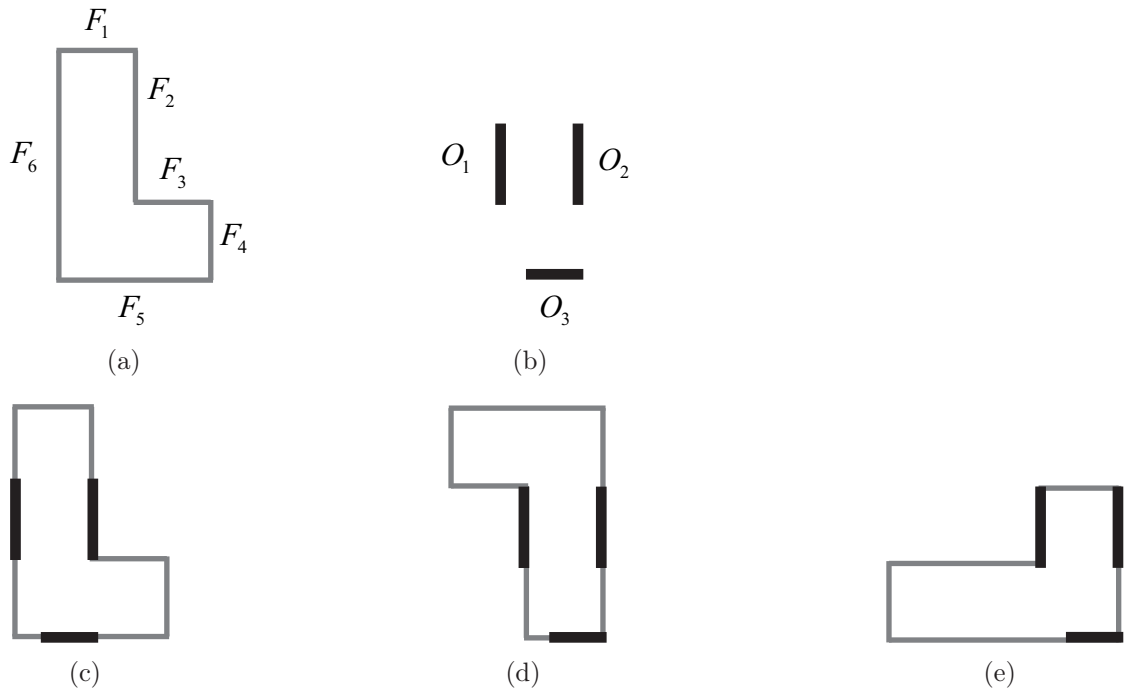


Figura 5.1: Múltiples soluciones válidas en el proceso de asociación de datos. La presencia de simetrías en el entorno hace que las tres soluciones de la parte inferior de la figura sean válidas. ¿Cuál es la correcta?

$$H = \{\zeta_1, \zeta_2, \zeta_3\}$$

que asocie cada observación $O_i, i = 1, 2, 3$ con un objeto del modelo $F_{\zeta_i}, i = 1, 2, 3$.

La presencia de simetrías hace que las tres soluciones presentadas en la parte inferior de la figura 5.1 sean correctas; $\{\zeta_1 = 6, \zeta_2 = 2, \zeta_3 = 5\}$ en el caso (c), $\{\zeta_1 = 2, \zeta_2 = 6, \zeta_3 = 1\}$ en (d) y $\{\zeta_1 = 3, \zeta_2 = 5, \zeta_3 = 6\}$ en (e). En el caso de que el objeto F_1, por ejemplo, no existiera en el modelo del entorno, la solución se complicaría aún más; o bien habría que considerar la solución de la figura 5.1(d) como incorrecta, o bien deberíamos suponer que la observación O_3 es errónea, o perteneciente a un nuevo objeto no contemplado en el actual modelo, en cuyo caso podría considerarse su inclusión en el mapa.

Se han realizado numerosos intentos para aliviar el coste computacional de esta búsqueda, asumiendo generalmente hipótesis simplificatorias, o tratando de buscar soluciones no óptimas. Por ejemplo, en [23] encuentran una solución en tiempo lineal con el número de objetos restringiendo la muestra a tres observaciones; número suficiente para localizar un robot móvil mediante triangulación.

Neira *et al.* [132] y Paz *et al.* [146] proponen un mecanismo de localización global que es también lineal en el tiempo. En este caso, emplean una representación aproximada del entorno, restricciones geométricas, y un algoritmo para muestrear aleatoriamente el espacio de posibles relaciones que son simultáneamente compatibles. Estos algoritmos son evaluados e un conjunto de datos adquirido en un parque [89] donde los árboles son modelados como objetos puntuales, utilizando el radio de

sus troncos como restricciones unarias, y la distancia entre estos como restricciones binarias.

A pesar de la robustez de estas soluciones, no son completas, ni garantizan el cálculo de todas las posiciones del robot que satisfacen de manera conjunta el máximo número de emparejamientos. Si se desea resolver este problema, el coste será siempre exponencial con el número de objetos considerados.

La búsqueda de correspondencia entre un conjunto de observaciones y otro conjunto de objetos almacenadas en el mapa puede ser formulado como la solución a un problema de máximo clique (MCP) [10] dentro del grafo de correspondencias [16] que codifica asociaciones válidas entre ambos conjuntos. Se trata de un problema NP-Duro [79], que encuentra una solución muy eficiente en [166] utilizando una formulación basada en codificar la información en cadenas de bits, explotando así el paralelismo a nivel de bit con el hardware para acelerar las operaciones de cálculo.

Esta solución codificada a nivel de bit para el problema del máximo clique (BE-MCP) fue utilizada con éxito en [167] para resolver el problema de localización global utilizando mapas compuestos por simples objetos puntuales, o balizas. Los resultados obtenidos mejoran sustancialmente algoritmos existentes como el *branch and bound* heurístico sobre el espacio de asociaciones (GCBB) [132] hasta en dos órdenes de magnitud. Más importante aún es el hecho de que la solución obtenida para el problema de máximo clique es exacta, y no requiere de ningún tipo de estimación inicial acerca de las posiciones relativas entre el conjunto de observaciones y el de elementos del mapa.

En este capítulo se propone la construcción de mapas limitados en tamaño, usando curvas spline como herramienta básica para modelar el entorno. El coste computacional de la construcción de estos mapas queda limitado restringiendo el número máximo de puntos de control que puede contener cada mapa. Las posiciones relativas de los mapas así construidos se actualizan utilizando un algoritmo BE-MCP para establecer correspondencias entre la constelación de submapas, en forma de transformaciones relativas entre ellos. Así es posible construir mapas de grandes dimensiones utilizando un *collage* de representaciones locales muy precisas.

5.2. Aligerando el Coste Computacional: Subdivisión del Mapa

Como ya se ha mencionado, una de las principales desventajas de la solución del problema del SLAM basada en EKF es su coste computacional, cuadrático con el número de elementos (puntos de control en nuestro caso) contenidos en el mapa. Dado que el crecimiento desmesurado del vector de estado del mapa es un problema para la implementación en tiempo real de los algoritmos presentados en el capítulo 4, recientemente han aparecido numerosas soluciones que pretenden atacar la tarea de construir mapas de grandes dimensiones, simplemente descomponiendo el problema global en trozos más pequeños; *i.e.*, una solución consiste en la construcción de mapas de tamaño limitado que describen diferentes áreas visitadas por el robot

durante su exploración [34, 71], para luego combinarlas o "coserlas" de algún modo más o menos ingenioso [145].

Otra alternativa es trabajar con un grafo de diferentes submapas, cuyas posiciones relativas se actualizan cuando es posible adquirir información espacial que las relacione. Se mantiene así una constelación de representaciones locales que, en conjunto, configuran el mapa del entorno completo. En la figura 5.2 se muestra una representación esquemática de este escenario, en el que existen 5 representaciones locales del entorno, cada una con su sistema de referencia asociado, y es posible establecer una nueva relación espacial entre el mapa M_5 y el mapa M_1 mediante algún mecanismo de asociación de datos.

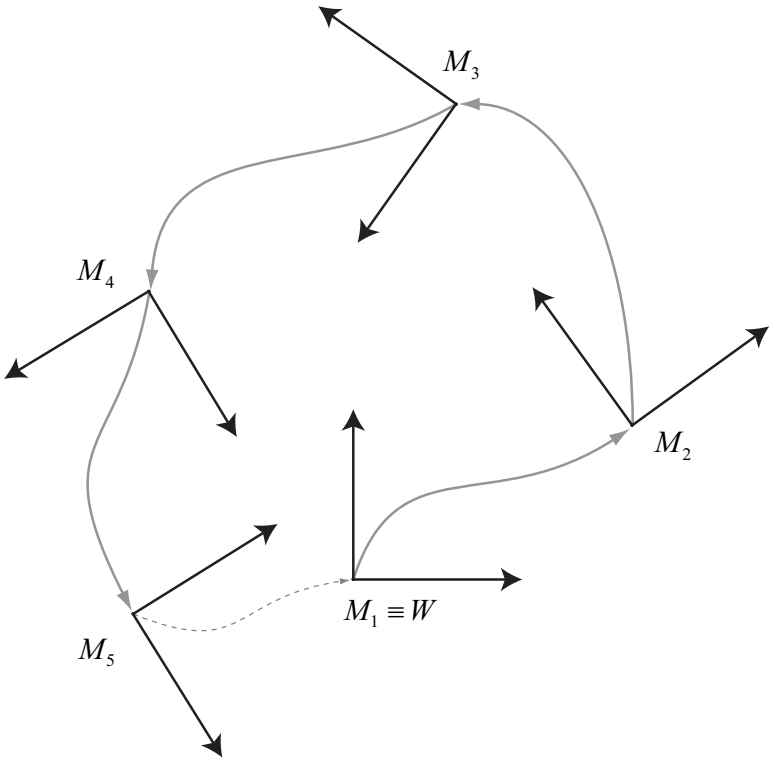


Figura 5.2: Diferentes submapas y sus relaciones relativas.

La solución que se propone en esta tesis consiste en utilizar diferentes pequeños submapas, limitados en tamaño con el número de puntos de control que contienen. Así, cada vez que un mapa alcanza un cierto número máximo de puntos de control, simplemente se comienza la construcción de uno nuevo que es inicializado utilizando la última observación del sensor, y situado en la mejor estimación para la localización del robot hasta el momento. Por lo tanto, la relación espacial [179] entre el nuevo mapa y su predecesor —al que denominaremos *padre*— es simplemente la situación del robot en el mapa progenitor en el instante de transición.

Así, volviendo al ejemplo de la figura 5.2, el primer mapa local que se construye, M_1, es inicializado de tal manera que su sistema de referencia coincida con el sistema de referencia global, W. Cuando este primer mapa alcanza un tamaño —*i.e.* número de puntos de control— predeterminado, que permita su construcción en tiempo real, se comienza a construir un nuevo mapa, M_2. Este segundo mapa se genera sobre un sistema de referencia coincidente con la posición y orientación del robot

en ese instante y, por tanto, afectado por la incertidumbre acumulada para dicha estimación. Así se van generando sucesivamente los mapas locales $M_3, M_4, \ldots$

El conjunto de submapas así construidos constituye un nuevo vector de estado que contiene las posiciones de todos ellos. El vector de estado de m submapas está formado por la posición absoluta de cada sistema de referencia local $M_i = \{\mathbf{u}_i, \mathbf{v}_i\}$ en el sistema de referencia global, que es coincidente con el primer submapa generado $W = \{\mathbf{u}_W, \mathbf{v}_W\} \equiv \{\mathbf{u}_0, \mathbf{v}_0\} = M_0$.

$$\mathbf{x} = \left[\mathbf{x}_0^T, \mathbf{x}_1^T, \ldots, \mathbf{x}_M^T\right]^T \tag{5.1}$$

Y la localización de cada submapa individual viene definida por:

$$\mathbf{x}_i = [x_i, y_i, \phi_i]^T, i = 0, \ldots, M \tag{5.2}$$

Ahora el problema es obtener relaciones espaciales entre los diferentes elementos de este conjunto de submapas, de modo que la información obtenida sirva para actualizar la estimación aplicando un filtro extendido de Kalman. Estas relaciones espaciales se pueden obtener cuando el robot revisita una zona previamente descrita por algún mapa local ya contenido en el estado. Por tanto, el problema se reduce a obtener correspondencias entre elementos comunes presentes en ambas representaciones.

Una vez obtenidas estas correspondencias, la transformación relativa entre ambos submapas puede ser calculada resolviendo un simple problema de minimización de los errores relativos entre cada par de objetos asociados, uno perteneciente a cada submapa. Esta situación aparece esquemáticamente representada en la figura 5.3.

En este caso existen dos mapas locales, M_i y M_j, que representan dos zonas distintas del entorno, pero contienen algunas características comunes. Si es posible identificar esta estructura común mediante algún mecanismo de asociación de datos, será posible obtener una observación de las posiciones relativas de ambos mapas. En la figura, $\mathbf{x}_{ij}$ es la estimación de la posición relativa entre ambos submapas antes de obtenerse esta información, mientras que $\mathbf{x}'_{ij}$ es la observación obtenida mediante el proceso de asociación de datos.

5.3. Asociación de Datos entre Submapas

La obtención de una correspondencia entre dos conjuntos de características contenidos en dos submapas diferentes, significa obtener una asociación entre un conjunto de objetos $L = \{L_1, L_2, \ldots, L_m\}$ contenidos en una representación local M_L y otro $O = \{O_1, O_2, \ldots, O_m\}$ pertenecientes a otro submapa M_O.

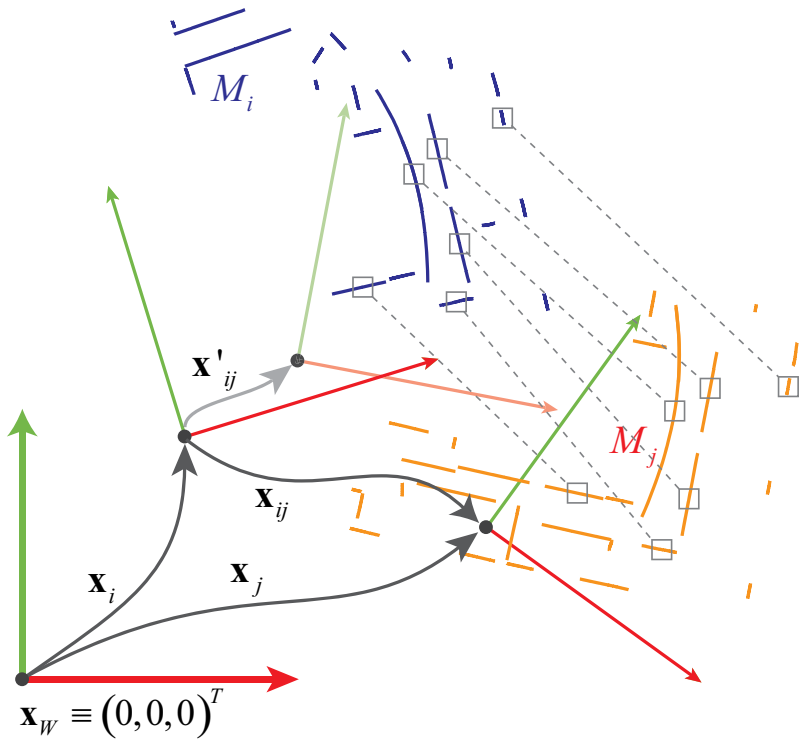


Figura 5.3: Observación relativa entre dos submapas asociados. Una vez obtenida la asociación entre características contenidas en ambos mapas, es posible obtener la observación $\mathbf{x}'_j$ que en general será diferente de la esperada por el conocimiento del entorno hasta ese momento $\mathbf{x}_j$. Esta información puede ser utilizada para actualizar el mapa global.

5.3.1. Resolución del problema con técnicas de grafos

Este problema ha encontrado interesantes soluciones cuando se interpreta mediante grafos que representan a los diferentes objetos, y a las relaciones geométricas que pueden establecerse entre ellos. Por ejemplo, el método del máximo subgrafo común (MCS, *maximum common subgraph*), empleado en la década de 1980 en técnicas de reconocimiento de objetos [101], ha sido más reciente aplicado al problema de la localización de robots móviles [9, 11].

El algoritmo de asociación de datos MCS describe cada conjunto de objetos mediante un grafo, de tal modo que cada uno de los vértices representa un objeto, y los bordes que los unen definen relaciones geométricas entre ellos. Cada vértice puede además contener información sobre el tipo de objeto y otras características. Es fácil comprender que los grafos así generados son completos (*i.e.*, cada uno de sus vértices está conectado con todos los demás), puesto que siempre es posible establecer una relación entre dos características geométricas del mismo mapa.

Ejemplos típicos de estas relaciones son la distancia entre dos puntos, el ángulo formado por dos rectas, la distancia de un punto a una recta... Como puede comprenderse, establecer este tipo de relaciones sólo es posible cuando las entidades que modelan el mapa son objetos geométricamente simples. Así pues, se entiende que el SLAM geométrico sea especialmente apropiado para aplicar estos métodos.

El MCS se ocupa de comparar ambos grafos y extraer de cada uno el máximo subgrafo completo cuyos nodos y bordes son compatibles con los del otro. Las figuras 5.4(a) y 5.4(b) ilustran esta idea mediante dos grafos de ejemplo, uno para cada submapa considerado. El grafo que aparece en la figura 5.4(a) muestra el grafo de relaciones para el mapa M_L, cuyos vértices son los objetos $L_1, \ldots, L_5$ y donde se han nombrado algunas de sus relaciones ($R_{i,j}$ indica una relación geométrica entre los objetos L_i y L_j). Lo mismo ocurre con el grafo de la figura 5.4(a), que en este caso muestra las relaciones entre los objetos $O_1, \ldots, O_4$ del mapa M_O. Como puede verse, ambos grafos son completos.

La comparación entre ambos grafos permite extraer de cada uno de ellos los subgrafos completos formados por los conjuntos de vértices $\{L_1, L_2, L, 4\}$ y $\{O_1, O_2, O_3\}$, cuyos bordes expresan relaciones compatibles dos a dos. Estos grafos se han representado con trazo grueso en el ejemplo. Así, las compatibilidades entre los pares de relaciones $\left\{R_{1,2}, R'_{1,3}\right\}$, $\left\{R_{2,4}, R'_{2,3}\right\}$ y $\left\{R_{1,4}, R'_{1,3}\right\}$ nos permitirían establecer una asociación entre los objetos $\{L_1, O_1\}$, $\{L_2, O_2\}$ y $\{L_4, O_3\}$, siempre y cuando los tipos de estos, y sus posibles restricciones unarias (por ejemplo, el radio de un objeto circular, o su color, si esta información estuviera disponible), fueran también compatibles.

Por compatibilidad entre dos relaciones se entiende que la posición relativa que se define entre los dos objetos considerados sea igual en ambos casos:

$$R_{1,2}\left(L_1, L_2\right) = R'_{1,3}\left(O_1, O_3\right) \Leftrightarrow d\left(L_1, L_2\right) = d\left(O_1, O_3\right) \tag{5.3}$$

donde la letra d indica una distancia que puede ser euclídea (por ejemplo, la distancia entre dos puntos) o estocástica, considerando en este último caso los posibles errores existentes en el modelo.

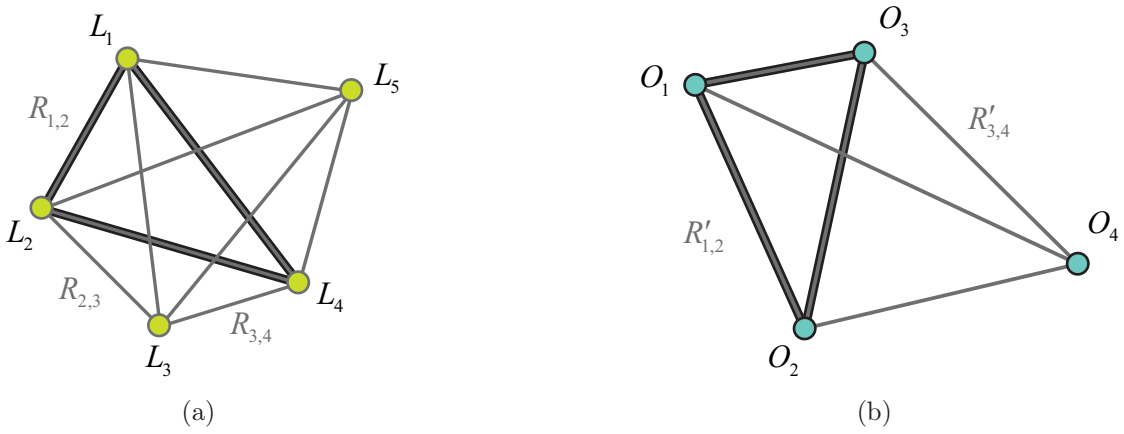


Figura 5.4: Asociación de datos por el método del máximo subgrafo común (MCS).

Entre las ventajas del MCS están su independencia de la posición absoluta de cada submapa —ya que las relaciones se establecen entre parejas de objetos, dentro de cada modelo local—, y que permite gran flexibilidad a la hora de definir el tipo y características de estas relaciones. El algoritmo básico de localización fue presentado en [11], mientras que en [9] se mejoró introduciendo la información disponible sobre la posición relativa entre ambos conjuntos de objetos en forma de restricciones adicionales.

5.3.2. El grafo de correspondencias

Basado en el MCS encontramos el algoritmo de asociación de datos CCDA (*combined constraint data association*). Este método explota la información disponible en una nueva estructura de datos que combina la información sobre las relaciones establecidas dentro de cada conjunto de objetos, y las compatibilidades entre ellas: el grafo de correspondencias (CG) [16].

La construcción de este nuevo grafo se realiza del siguiente modo (véase la figura 5.5). Sus nodos están formados por la asociación entre dos objetos, uno de cada conjunto considerado, de tal modo que sus restricciones unarias o absolutas (por ejemplo, su tipo) no sean incompatibles. Los bordes del CG indican compatibilidad conjunta entre los nodos que conectan, y se construyen teniendo en cuenta las restricciones relativas entre los objetos que pertenecen al mismo conjunto.

Siguiendo con nuestro ejemplo, se puede observar en la figura 5.5 el CG correspondiente a los grafos de relaciones construidos sobre los mapas M_L y M_O. Suponiendo que los objetos $L_1, \ldots, L_4, O_1, \ldots, O_3$ son de tipo 1 (por ejemplo, segmentos), y que los objetos L_5 y O_4 son de tipo 2 (por ejemplo, círculos), en el ejemplo se muestran los nodos del grafo de correspondencias en rojo. Como puede apreciarse, no tiene sentido incluir los nodos correspondientes a emparejamientos como el $\{L_1, O_4\}$, puesto que estos dos objetos son de diferente tipo.

La figura 5.5 muestra también los bordes para nuestro grafo de correspondencias. Por ejemplo, la conexión entre los nodos $\{L_2, O_1\}$ y $\{L_4, O_3\}$ indica que la relación

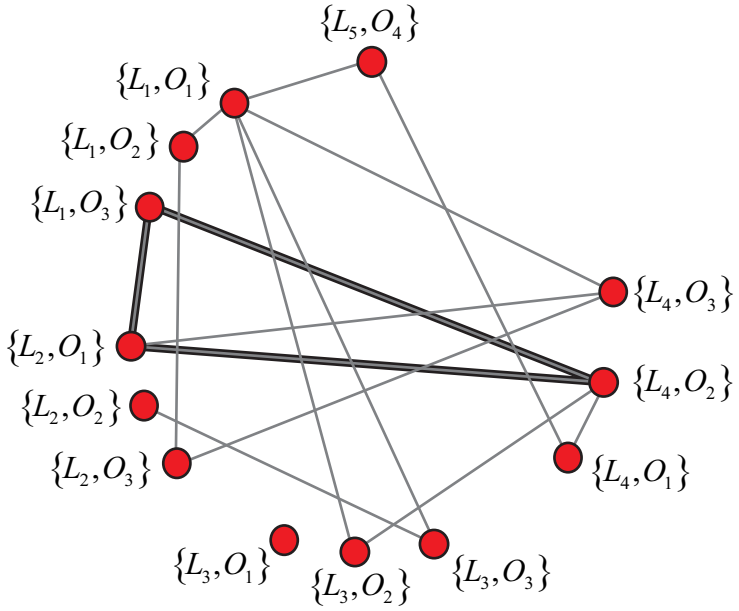


Figura 5.5: Asociación de datos mediante un grafo de correspondencias.

$R_{2,4}$ entre los objetos L_2 y L_4 del mapa M_L, y la relación $R'_{1,3}$ entre los objetos O_1 y O_3 del mapa M_O son compatibles entre sí. Como puede observarse, el CG no es un grafo completo, pudiendo incluso existir nodos (*i.e.* emparejamientos entre objetos de ambos mapas) que no estén conectados con ningún otro, como sucede con el $\{L_3, O_1\}$.

En estas condiciones, cualquier subgrafo completo (o clique) dentro de esta estructura indica la compatibilidad conjunta de todas las asociaciones involucradas. Si buscamos el mayor subgrafo completo (máximo clique) hallaremos el mayor conjunto de asociaciones conjuntamente compatibles. Esta será por tanto la solución óptima al problema de asociación de datos.

Este algoritmo de asociación de datos es conceptualmente equivalente al GCBB (Geometric Constraints Branch and Bound) [132], pero con algunas diferencias. La fundamental es que no se busca sólo empleando la mejor hipótesis disponible, sino simultáneamente para todas las hipótesis posibles que satisfacen todas las restricciones.

En esta tesis, el procedimiento para obtener una correspondencia máxima entre los elementos de ambos conjuntos se formula como la obtención de un máximo clique [87] dentro del grafo de correspondencias construido utilizando objetos extraídos de cada submapa. La utilización directa de nuestra herramienta de modelado —los splines— para establecer estas relaciones no es buena idea; su flexibilidad y adaptabilidad geométrica es aquí un arma de doble filo, no permitiendo establecer métricas fácilmente parametrizables entre dos curvas cualesquiera. Sin embargo, siempre es posible analizar su estructura, y extraer la información geométrica necesaria, como veremos.

5.3.3. El problema del máximo clique

Dado un grafo no dirigido $G = V, E$, donde V es un conjunto de vértices y E es el conjunto de bordes, se dice que dos vértices son *adyacentes* cuando están conectados por un borde. Se dice que el grafo es *completo* cuando dos vértices cualesquiera son adyacentes y, cuando esto sucede, también se dice que el grafo es un *clique*. Un problema típico es la obtención del subconjunto máximo de vértices de V tal que sus elementos forman un clique. En este caso hablamos del "problema del máximo clique" (MCP).

La figura 5.6 ilustra el concepto de clique con un grafo de ejemplo formado por 4 elementos. El estudio de los cliques tiene importantes aplicaciones prácticas en campos como la visión por computador o el procesamiento de señales. En particular, resulta de gran interés en problemas de optimización combinacional, en aplicaciones como el análisis de mercados [164].

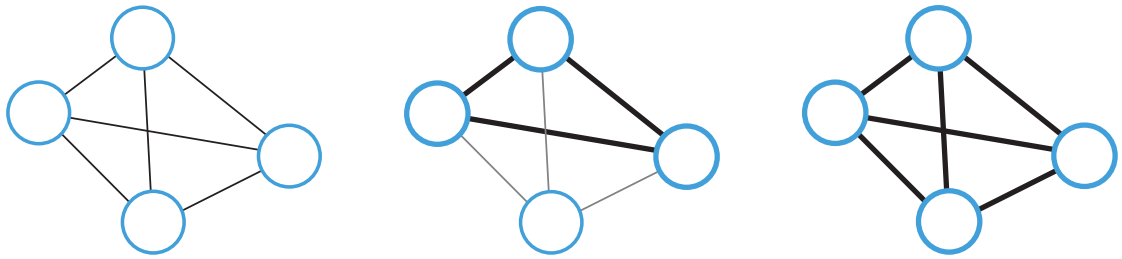


Figura 5.6: Concepto de clique. (a) Grafo original. (b) Clique de cardinalidad 3. (c) Clique de cardinalidad 4 que, en este caso, es máxima y coincide con el grafo de partida.

Ya hemos visto en el apartado anterior que la búsqueda de una correspondencia entre objetos contenidos en dos conjuntos diferentes (submapas) puede ser formulada como la búsqueda de un máximo clique dentro de un grafo de correspondencias, en el cual:

- Cada **vértice** está formado por un **par de objetos**, cada uno de los cuales corresponde a uno de los dos conjuntos considerados.
- Cada **borde** indica la **compatibilidad** entre las dos relaciones entre los objetos pertenecientes al mismo mapa.

Más formalmente, sean $|L| = m$, $|O| = n$ las cardinalidades de ambos conjuntos de objetos con elementos $L = \{L_1, \ldots, L_m\}$, y $O = \{O_1, \ldots, O_n\}$. Se genera un grafo de correspondencias $CG = \{V, E\}$ con vértices $V = \{V_1, V_2, \ldots\}$, cuyos elementos son todas las combinaciones posibles que contienen un elemento del mismo tipo extraído de cada conjunto.

Los vértices de este grafo se conectan entre sí utilizando una distancia generalizada D, independiente de la posición global de cada uno de los elementos que

constituyen cada par de objetos, y que considera sólo posiciones y orientaciones relativas. Esta función se aplica a cada par de vértices en V, obteniéndose un valor verdadero cuando la distancia entre el par de observaciones (u objetos pertenecientes a M_O) es similar a la distancia entre el par de objetos contenidos en M_L. Específicamente, si tenemos dos vértices $V_i = \{L_p, O_q\}, V_j = \{L_r, O_s\} \in V, i \neq j$, se define la función $D : \{V, V\} \rightarrow \{0, 1\}$ tal que:

$$D(V_i, V_j) = \begin{cases} 1 \text{ si } d(L_p, L_r) \approx d(O_q, O_s) \\ 0 \text{ sn cualquier otro caso} \end{cases} \tag{5.4}$$

Así, cada elemento de la matriz de adyacencias M de G codifica la potencial asociación correcta de cada par de elementos perteneciente a cada uno de los submapas bajo consideración (ver figura 5.7). El máximo clique encontrado en el grafo así generado proporciona una solución óptima a la búsqueda de correspondencia.

	$\{L_1,O_1\}$	$\{L_1,O_2\}$	$\{L_1,O_3\}$	$\{L_2,O_1\}$	$\{L_2,O_2\}$	...	$\{L_5,O_4\}$
$\{L_1,O_1\}$	0	1	0	0	0	...	1
$\{L_1,O_2\}$	1	0	0	0	0	...	0
$\{L_1,O_3\}$	0	0	1	0	0	...	0
$\{L_2,O_1\}$	0	0	1	0	0	...	0
$\{L_2,O_2\}$	0	0	0	0	0	...	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$\{L_5,O_4\}$	1	0	0	0	0	...	0

Figura 5.7: Matriz de adyacencias del grafo de correspondencias.

Usando la formulación codificada a nivel de bit para la solución de este problema, explotando al máximo la arquitectura de los computadores actuales, se pueden obtener resultados destacados para la solución del mencionado problema, mejorando soluciones clásicas como del GCBB [132], y abriendo la puerta a implementaciones en tiempo real.

5.3.4. Solución del MCP utilizando *bitboards*

Siguiendo el trabajo previo realizado en *bitboards*, se puede adaptar el algoritmo de propósito general desarrollado en [166] al escenario específico de un grafo generado en el problema de localización global [167]. Este mismo razonamiento es adaptado en esta tesis para resolver el problema de obtener correspondencias entre elementos contenidos en dos submapas del entorno. La solución propuesta está codificada a nivel de bit, y encuentra una solución exacta al MCP explotando el paralelismo entre la arquitectura de un computador y la lógica que aparece cuando se comparan dos conjuntos de objetos.

Los métodos típicos que investigan las asociaciones existentes dentro de un árbol o grafo, y buscan optimizar las soluciones desde el punto de vista del tiempo de cómputo, siempre se implementan en lenguajes de programación que representan la información binaria de una posible asociación —un 0 si no es posible, o un 1 cuando sí lo es— en una variable entera. Al manejar estos datos dentro del procesador se utiliza una palabra completa —32 o 64 bits, generalmente— para almacenar esta información, puesto que los bits individuales no se pueden manipular independientemente. Cuando esta operación se realiza millones de veces, resulta evidente su escasa eficiencia.

En [166] se explotaba este paralelismo lógico existente entre la información contenida en un grafo y la información binaria que maneja un computador, codificando la información en cadenas de bits. Una cadena de bits es una estructura de datos que almacena bits individuales de manera compacta y cuya utilización resulta eficiente para explotar al máximo las capacidades del hardware. Un ejemplo de su aplicación lo encontramos en las colas de prioridad de los sistemas operativos (*e.g.* el kernel de Linux), donde el bit en la posición k tiene valor 1 sólo cuando el proceso k-ésimo se encuentra en la cola.

Estas estructuras también han sido ampliamente utilizadas en el dominio de los juegos de mesa, como el ajedrez [99], donde el término *bitboard* fue acuñado para referirse a ellas. Utilizando cadenas de bits, la complejidad del espacio de búsqueda se reduce en un factor equivalente al tamaño en bits de la palabra del procesador. A pesar de que este hecho no justifica por sí sólo su utilización en problemas de búsqueda en grafos, sí lo hace la reducción de tiempo de cómputo que supone la posibilidad de realizar operaciones a nivel de bit.

Observando la matriz de adyacencias representada en la figura 5.7, se pueden sacar algunas conclusiones. Las inmediatas son que se trata de una matriz simétrica, cuyos elementos diagonales son nulos (se descartan las compatibilidades triviales como consecuencia de comparar cada elemento de un conjunto consigo mismo). Además, cada fila indica las potenciales correspondencias de un elemento del CG con todos los demás vértices. El mismo razonamiento se puede realizar con cualquiera de las columnas, debido a la simetría de esta matriz y a que representa las adyacencias en un grafo no dirigido.

En [167] se pueden encontrar detalles sobre la solución del MCP utilizando una codificación en cadenas de bits. Básicamente, hay que tener en consideración los siguientes elementos:

- **Sobre la codificación**. Los *bitboards* mapean en cada bit individual un elemento de la matriz de adyacencias. Esto implica que el límite superior en la reducción de tiempo de cómputo que consigue esta especial codificación depende del tamaño de la palabra del procesador W_{CPU} [166].

- **Sobre el algoritmo de búsqueda**. Los principales paradigmas para resolver el MCP son *backtracking* y *branch and bound*. Algunos algoritmos que emplean este último paradigma se pueden encontrar en [141,142], siendo la formulación básica la presentada en [40], cuyo pseudocódigo se presenta en la figura 5.8.

- **Sobre la complejidad del algoritmo**. El MCP es un problema NP-Duro [79]. Para un grafo con n vértices ($|V| = n$), la búsqueda bruta de un clique de tamaño k tiene un orden de complejidad $O\left(\binom{n}{k} k^2\right)$ que puede aliviarse algo utilizando el paradigma *branch and bound*. La mejora es especialmente importante en el caso del problema de localización, o asociación de objetos entre mapas que nos ocupa. Esto se debe a la naturaleza generalmente dispersa de las matrices de adyacencias obtenidas.

Inicialización: U =Vertex set of G , size=0 function **clique**(U, size) Paso 1: Leaf node ($\lvert U\rvert = 0$) 1. if (*max_size* > *size*) record *max_size* **2. return** Paso2: Fail upper bound (*size* + $\lvert U\rvert$ < *max_size*) **1. return** Paso 3: Expand node from U 1. $v_i := N_{sel}(U)$ 2. $U := U \setminus \{v_i\}$ 3. **clique**$(U \cap N_{adj}(v_i), size+1)$	$N_{adj}(v_i)$ devuelve el conjunto de vértices adyacentes a v_i *max_size*: variable global que almacena el mayor clique encontrado hasta el momento $N_{sel}(U)$ es cualquier estrategia de selección de vértices que escoja uno del conjunto U

Figura 5.8: Forma básica del algoritmo MCP.

La tabla 5.1 muestra los tiempos requeridos por la solución codificada en bit del MCP (BE-MCP) para diferentes números de objetos contenidos en dos mapas diferentes.

Tiempo (s)	*n*							
m	5		8		11		14	
250	0,00	0,02	0,01	0,02	0,03	0,03	0,03	0,05
		0,00		0,00		0,02		0,00
500	0,02	0,03	0,05	0,06	0,10	0,11	0,17	0,19
		0,02		0,05		0,09		0,16
750	0,04	0,06	0,13	0,16	0,28	0,31	0,48	0,52
		0,03		0,11		0,22		0,44
1000	0,09	0,09	0,42	0,52	0,99	1,06	1,79	1,94
		0,06		0,27		0,89		1,58

Cuadro 5.1: Tiempo de procesamiento del algoritmo BE-MCP. En la tabla m y n son el número de objetos contenidos en cada uno de los mapas, y se muestran los valores medio, mínimo y máximo tras 10 experimentos con la misma configuración.

Finalmente, como se comprueba experimentalmente en [167], este algoritmo presenta una razonable inmunidad ante observaciones espurias contenidas en alguno de

los dos conjuntos considerados. Esto es especialmente interesante en el problema de localización global, donde no está garantizado que todas las observaciones adquiridas por el robot, estén de hecho contenidas en el mapa disponible.

5.4. Elementos Usados en la Asociación

Ya sabemos cómo construir mapas utilizando splines para modelar las geometrías de los elementos presentes en el entorno del robot. Pero, ¿qué elementos usar para realizar la asociación entre diferentes mapas? La utilización directa de los puntos de control no parece una buena idea, ya que la misma forma física puede ser representada utilizando diferentes combinaciones de vector de nodos y polígono de control. La presencia de oclusiones, o el simple hecho de que los objetos son modelados utilizando diferentes conjuntos de datos, hacen que la simple utilización de los puntos de control en el proceso de asociación de datos no sea la primera opción a considerar.

No obstante, la disponibilidad de una representación paramétrica del entorno se torna una vez más una ventaja más que un inconveniente. A continuación veremos cómo el análisis geométrico de las formas contenidas en el mapa puede proporcionar gran cantidad de información, permitiendo la extracción de características más simples y fácilmente parametrizables para ser utilizadas en el proceso de asociación de datos. También resultará interesante el hecho de que el número de estas primitivas geométrica es, en general, menor que el número de puntos de control que definen la curva original.

En entornos de interior complejos, no importa cuán curvos sean los elementos arquitectónicos presentes, siempre es posible aproximar su apariencia mediante conjuntos de puntos muestreados, o la simple concatenación de elementos más simples tales como líneas o arcos de circunferencia:

- **Puntos**: Se puede obtener una burda representación del entorno simplemente muestreando cada curva del mapa con un espaciado prefijado sobre el parámetro independiente t. Esta representación, a pesar de ser muy simple, implica un crecimiento innecesario en el número de elementos involucrados en el proceso de asociación. En cualquier caso, los puntos pueden ser entidades de gran utilidad en áreas donde la complejidad del entorno hace complicada la extracción de características más generales. También pueden ser utilizados para representar la localización de elementos puntuales, tales como esquinas, no consideradas en el trabajo presentado en esta tesis.

- **Puntos con curvatura asociada**. Se trata de simples puntos que tienen asociada la curvatura de la pared modelada en esa localización concreta. La consideración de esta característica obviamente enriquece la información disponible durante el proceso de asociación. Por ejemplo, un punto situado en una pared cóncava del entorno nunca podrá ser asociado con otro que se encuentra situado en una superficie convexa o plana. Este hecho podría evitar eventuales ambigüedades en el proceso de asociación.

- **Segmentos**: El análisis de la curvatura puede revelar la presencia de simples objetos planos. En los experimentos realizados, se definió un umbral de $0{,}01m^{-1}$ para la detección de segmentos.

- **Arcos circulares**. Aquellos puntos de un elementos del mapa, cuya curvatura supere el umbral definido para la consideración de segmento, puede ser agrupados por intervalos en arcos de circunferencia.

En [167] se emplearon simples puntos en la resolución del problema de localización global de un robot móvil. En esta tesis, se emplearán fundamentalmente segmentos y arcos extraídos del mapa para realizar el proceso de asociación de datos entre los diferentes mapas que modelan globalmente el entorno.

5.4.1. Extracción de características simples del mapa

Las características presentadas en los párrafos anteriores se pueden obtener sencillamente a partir del análisis de la curvatura de cada una de las curvas spline contenidas en el mapa. La figura 5.9 muestra un ejemplo de un mapa de muestra dibujado a mano. El mapa contiene un elemento curvilíneo de grandes dimensiones y dos elementos rectos (segmentos). La figura también muestra los segmentos y arcos que es posible extraer una vez analizada la geometría de los elementos representados.

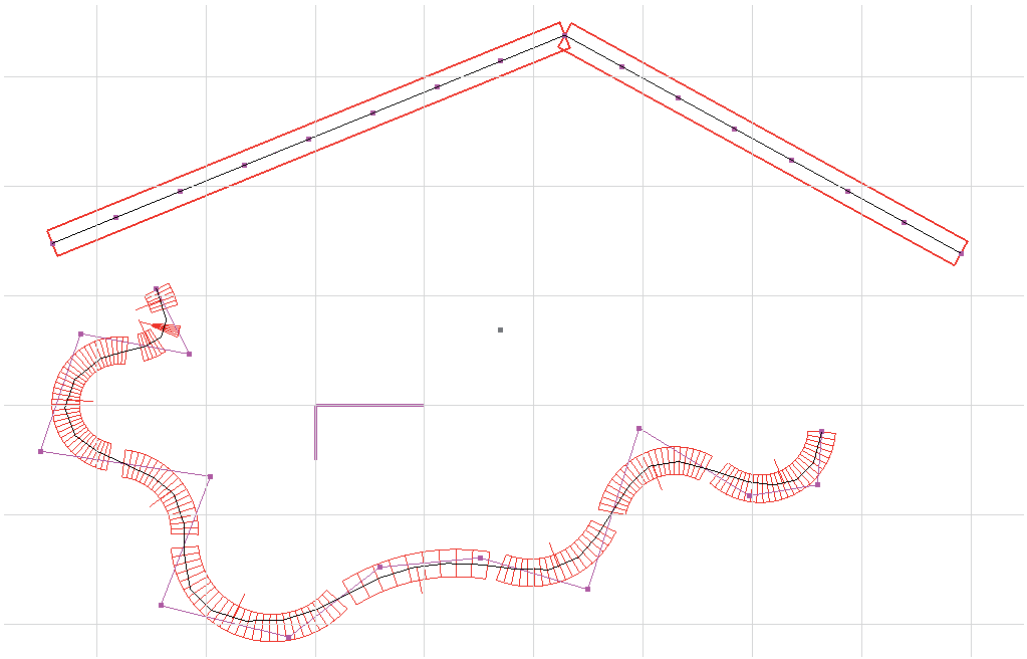


Figura 5.9: Ejemplo que muestra un mapa compuesto por tres splines definidos por un total de 31 puntos de control. El análisis del mapa revela la presencia de 10 características simples: 2 segmentos y 8 arcos de círculo.

El cálculo de la curvatura de un spline en una localización dada se reduce a un simple problema de derivación sobre la curva paramétrica, utilizando la conocida

fórmula dada por la ecuación (5.5). Así, para una curva spline expresada paramétricamente como $\mathbf{s}(t) = [s_x(t), s_y(t)]$, la curvatura para una posición del parámetro t viene dada por:

$$C(t) = \frac{s'_x s''_y - s'_y s''_x}{\left(s'^2_x + s'^2_y\right)^{3/2}} \tag{5.5}$$

Esta extracción de características simples se realiza para cada uno de los submapas construidos, inmediatamente después de que se ha alcanzado el tamaño máximo predefinido, utilizando un paso prefijado constante para muestrear la curvatura a lo largo de toda su extensión. Para la estimación precisa de los puntos extremos que definen cada uno de los elementos extraídos se realiza una búsqueda binaria.

5.4.2. Relaciones invariantes

La construcción de la matriz de adyacencias requiere, como sabemos, de algún tipo de regla que defina si dos relaciones son compatibles o no (ecuación (5.4)). El procedimiento básico para obtener estas relaciones se basa en la comparación de las principales métricas que definen cada tipo de relación establecida entre dos elementos del mismo mapa, y la comprobación de que sus diferencias se encuentran dentro de ciertos límites o tolerancias preestablecidas. Se absorben así posibles errores en el modelo, e inexactitudes en la localización de los elementos —tanto las producidas por la inexactitud de los mapas, como las que se derivan de la posterior extracción de características simples—.

La figura 5.10 muestra algunas de las características que es posible extraer de mapas modelados mediante curvas spline, así como las métricas que es posible establecer entre ellas. Estas métricas son las utilizadas para comparar dos relaciones en dos mapas diferentes, y construir de este modo el grafo de correspondencias y la matriz de adyacencias.

Finalmente, la figura 5.11 muestra un ejemplo con datos reales del proceso de asociación de datos descrito en este capítulo. En este caso se ha realizado la asociación entre un conjunto de objetos detectados en una única observación por los sensores del robot, y los contenidos en un mapa local del entorno construido en algún momento anterior.

5.5. Conclusiones

En este capítulo se ha presentado una solución para abordar uno de los problemas de la solución EKF al problema del SLAM, que hace uso de la representación paramétrica de los objetos contenidos en el mapa. Para ello, se ha optado por acotar la carga computacional del algoritmo limitando el tamaño de los mapas construidos en el número de objetos (puntos de control) que contienen. De esta manera, se

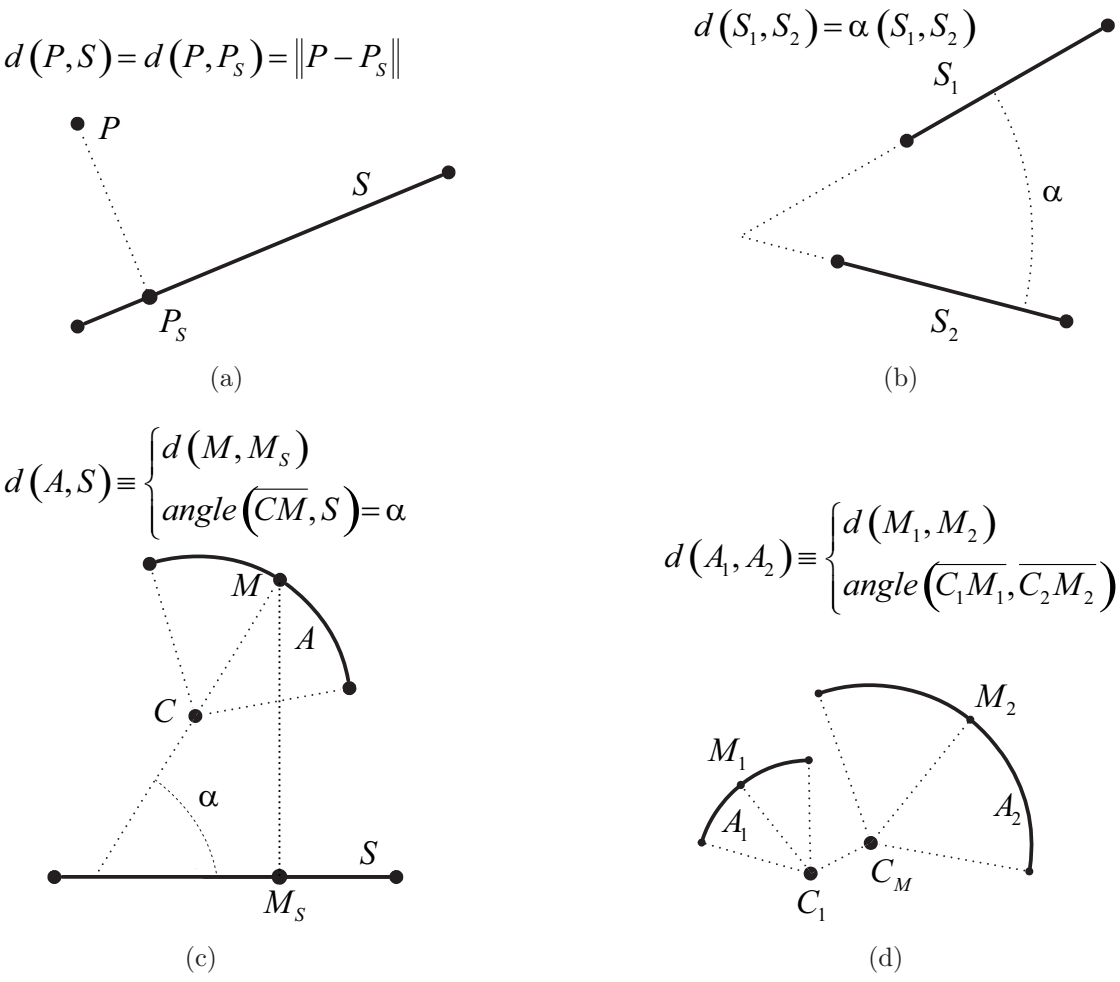
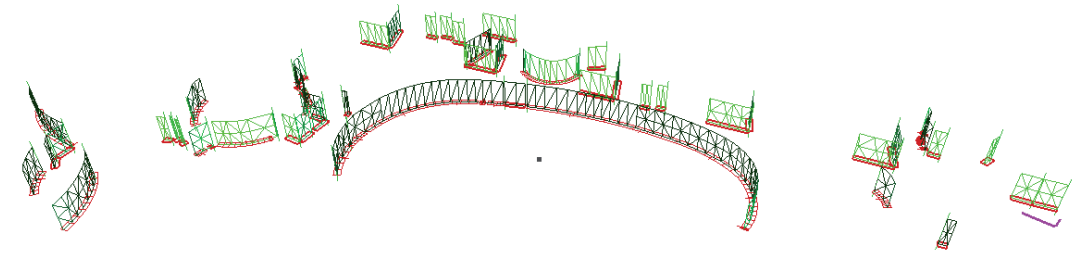
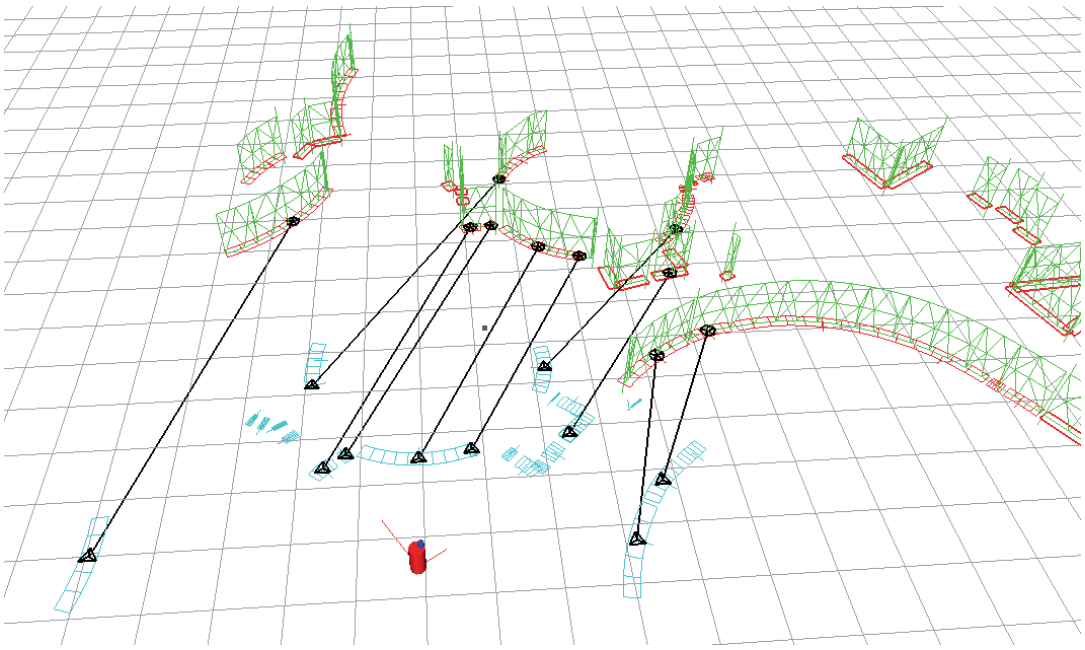


Figura 5.10: Relaciones establecidas entre las características extraídas de un mapa. Relación entre un punto y un segmento (a), entre dos segmentos (b), entre un arco y un segmento (c) y entre dos arcos (d).



(a)



(b)

Figura 5.11: Ejemplo de asociación de datos. Asociación de datos entre un conjunto de 26 observaciones (representadas en azul) y un conjunto de 68 objetos contenidos en otra representación en el mapa (mostrados en rojo) que se muestra completo en la parte superior. La mayoría de las observaciones son correctamente emparejadas con sus correspondientes características presentes en el mapa en tan solo 0,06 segundos.

construye una cadena de representaciones locales, cuyos sistemas se organizan en un vector de estado que se estima mediante un nuevo filtro extendido de Kalman.

Si bien esta solución basada en la descomposición del problema global en unidades más pequeñas no es contribución original de esta tesis, si lo es la manera de razonar sobre los mapas y extraer información geométrica de los mismos. Así, una vez disponible un modelo paramétrico del entorno, en el que cualquier geometría se ve representada mediante relaciones funcionales de un parámetro independiente, es posible analizar la curvatura de los diferentes elementos y extraer información más simple como puntos, segmentos, o arcos de circunferencia.

Las relaciones geométricas que se establecen entre estos elementos contenidos en cada submapa, permiten buscar correspondencias entre las diferentes representaciones locales que permiten actualizar o corregir la estrucutra global de la constelación de submapas. Para ellos, se ha utilizado un algoritmo especial que codifica la información que proporciona el grafo de correspondencias, y obtiene soluciones muy eficientes para el problema de obtener en él un máximo clique. La solución a este problema (MCP) proporciona una solución óptima al problema de encontrar correspondencias entre dos conjuntos de objetos.

Esta solución, además, no depende del conocimiento previo de la orientación relativa entre ambas representaciones, y es robusta ante la presencia de errores en la posición de los objetos, o de observaciones incorrectas.

Capítulo 6

Resultados Experimentales

6.1. Introducción

Con el objetivo de comprobar y validar los procedimientos y algoritmos propuestos en esta tesis, se han llevado a cabo numerosos experimentos tanto con datos reales como simulados. Los experimentos con datos simulados por computador han permitido comprobar las propiedades de precisión y consistencia de los algoritmos. Los experimentos con datos reales permiten verificar la aplicabilidad de estas técnicas, su utilidad y, en algún caso, compararlas con procedimientos existentes.

Respecto al análisis comparativo con técnicas existentes de modelado, cabe realizar tres comentarios de gran importancia:

- **La técnica de modelado presentada en esta tesis es única en su clase** y difícil de comparar con algoritmos existentes. Esto se debe a que los métodos actuales de SLAM geométrico no son capaces de manejar adecuadamente estructuras curvilíneas. Cuando los entornos contienen geometrías de este tipo, los métodos actuales tratan de asimilarlas a entidades más simples cuando esto es posible. Por ejemplo, estimando el centro (y tal vez el radio) de características aproximadamente circulares, como troncos de árboles [89, 90, 212].

- Cuando se desea aproximar el contorno real de elementos curvílíneos presentes en el entorno, **ningún método geométrico actual tiene validez**, debiéndose recurrir a técnicas de *scan matching* [121] o basadas en trayectorias [134] y almacenando la información bruta suministrada por el sensor, muy diferentes a la solución EKF utilizada en esta tesis.

- Desde el punto de vista del **rendimiento computacional** del algoritmo, así como del de su **consistencia**, tanto las ventajas como los inconvenientes, así como las soluciones que palían sus limitaciones, son **equivalentes a las de cualquier otra solución SLAM-EKF**. Lo que aquí se plantea es una nueva técnica de amplía considerablemente el dominio de aplicabilidad de soluciones existentes.

En este capítulo se presentan en primer lugar algunos experimentos que ayudan a comprender las técnicas de aproximación de datos adquiridos mediante un sensor láser (sección 6.2), mostrando la importancia del orden y espaciado del vector de nodos a la hora de construir las curvas. Se continúa con resultados que muestran las propiedades del algoritmo desde el punto de vista de su precisión y consistencia (secciones 6.3 y 6.4), para terminar con los resultados experimentales que muestran mapas construidos con datos adquiridos en entornos reales (secciones 6.5 y 6.5.1).

El capítulo se cierra con algunas conclusiones recogidas en la sección 6.6.

6.2. Aproximación de Datos Puntuales

Esta tesis ataca el problema del SLAM tratando de explotar la máxima cantidad de información posible suministrada por un sensor de gran popularidad en robótica móvil como es el telémetro láser. Estos sensores proporcionan, por cada medición, un número de medidas de distancia variable (generalmente entre 181 y 361 medidas, dependiendo de la resolución angular empleada) recorriendo un determinado rango angular. Se ha visto en el capítulo 3 cómo aproximar estas medidas puntuales mediante curvas B-spline.

El elemento de partida para conseguir este objetivo es el vector de nodos, que se emplea para generar el conjunto de funciones básicas cuya combinación lineal da lugar a la curva de aproximación. Los puntos de control se obtienen mediante un simple proceso de minimización, resolviendo un sistema de ecuaciones lineales en el sentido de mínimos cuadrados del error cometido.

En esta sección se pretende estudiar brevemente la influencia de dos parámetros fundamentales sobre la calidad de la aproximación obtenida: el **espaciado internodal** y el **orden** de la curva empleada.

Para ellos se ha simulado un robot estático situado en el origen de coordenadas frente a una pared curvilínea definida por la siguiente ecuación en coordenadas polares:

$$z = 5 + 0{,}5 \cdot sin\,(5\varphi) \tag{6.1}$$

siendo φ el ángulo polar en radianes y z la distancia en metros de un punto de la curva al origen.

El sensor láser simulado tiene un rango de medida de 180° de manera que es posible cubrir el semiplano superior $y > 0$ y un rango lineal de 8 metros. Para cada medida a lo largo de este rango angular, tomada con una resolución de 1°, se ha agregado ruido Gaussiano generado con una distribución de media 0 y desviación típica $\sigma_L = 5$ mm. En todos los experimentos se ha calculado el error cuadrático medio (MSE) cometido en la aproximación tras 50 simulaciones Monte Carlo, utilizando diferentes configuraciones del vector de nodos o del orden de la curva empleada.

Para el cálculo del error cuadrático medio se ha tomado como residuo la dife-

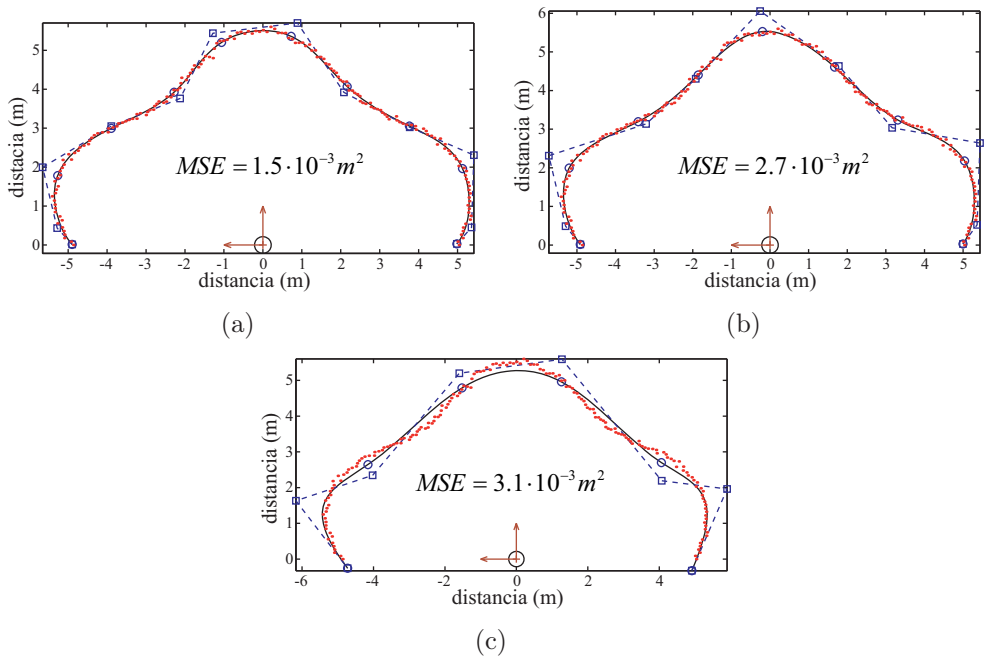


Figura 6.1: Influencia del espaciado internodal en el ajuste de datos. la longitud total de la curva es 20,42 m. En cada caso el B-spline cúbico de aproximación está formado por (a) 9 tramos ($\rho_a = 2{,}27$ m), (b) 8 tramos ($\rho_b = 2{,}55$ m), y (c) 5 tramos ($\rho_c = 4{,}1$ m).

rencia entre la distancia real de un punto de la curva al origen, y la distancia del correspondiente punto del spline de aproximación al origen.

6.2.1. Influencia del espaciado internodal

La figura 6.1 muestra los resultados de aproximar la geometría descrita por la ecuación (6.1) utilizando diferentes espaciados internodales en cada caso. La longitud total de la curva que se pretende aproximar es de 20,42 metros. la figura 6.1(a) muestra el resultado de aproximarla mediante un spline formado por 9 tramos polinomiales, la figura 6.1(b) cuando se emplean 8 tramos, y la figura 6.1(c) cuando sólo se utilizan 5 tramos. En todos los casos el orden de la curva es $\kappa = 4$ (spline cúbico).

Como puede apreciarse, el resultado es perfectamente lógico; a medida que aumenta el número de tramos polinomiales que componen la curva, para un mismo orden, el resultado mejora notablemente. Por este motivo es necesario alcanzar un compromiso entre el número de tramos polinomiales a utilizar, y el número de puntos de control que serán necesarios para definir la curva.

Aumentar el número de tramos polinomiales supone reducir el espaciado internodal para el vector de nodos generado. Dicho con otras palabras: aumentar el número de nodos. Puesto que para un B-spline la diferencia entre el número de nodos y el de puntos de control permanece constante, e igual al orden κ de la curva, se deduce que a más tramos polinomiales (mayor precisión de la aproximación), mayor número de

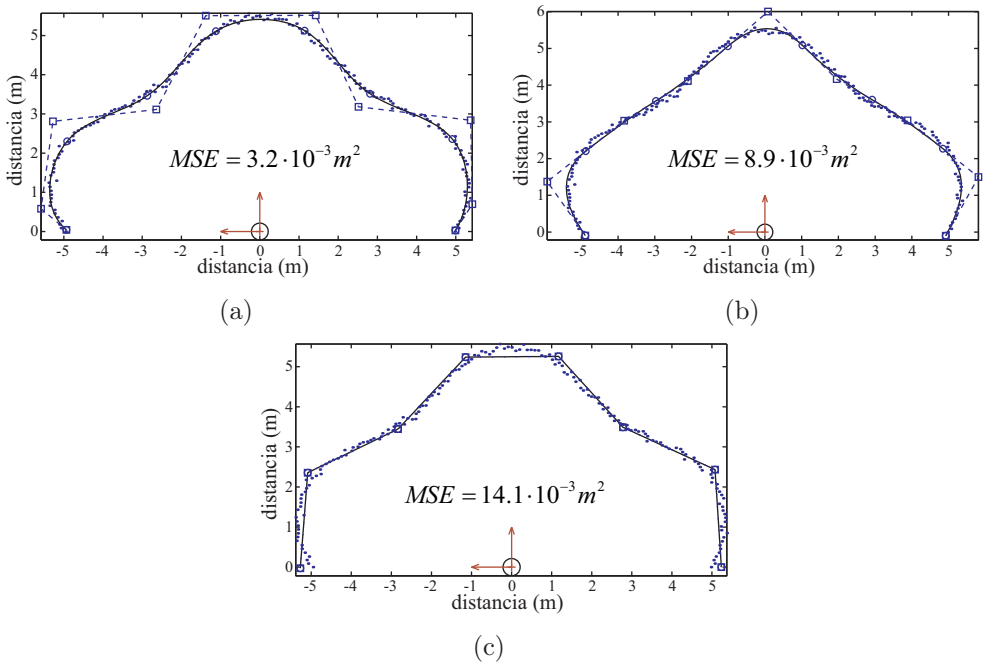


Figura 6.2: Importancia del orden del B-spline de aproximación. Aproximación de datos ruidosos mediante (a) un spline cúbico ($\kappa = 4$), (b) un spline cuadrático ($\kappa = 3$) y (c) un spline lineal ($\kappa = 2$).

puntos de control. Por tanto, mayor crecimiento del tamaño del mapa (en número de elementos), y mayor crecimiento del coste computacional del algoritmo.

Para entornos de interior habituales, un espaciado internodal de 1 o 2 metros resulta más que suficiente.

6.2.2. Influencia del orden de la curva

Otros de los elementos que se pueden determinar a la hora de aproximar una geometría es el orden de la curva empleada. Los splines cúbicos se han convertido en un estándar en el mundo de los gráficos por computador, por su adecuado compromiso entre coste computacional, versatilidad, y capacidad de representar cualquier tipo de geometría.

La figura 6.2 muestra la importancia del orden de la curva de aproximación elegida a la hora de aproximar una geometría arbitraría. Se aprecia que los resultados son mejores a medida que el orden de los diferentes tramos polinomiales crece. Esto es debido a que se dispone de más grados de libertad para aproximar adecuadamente la geometría.

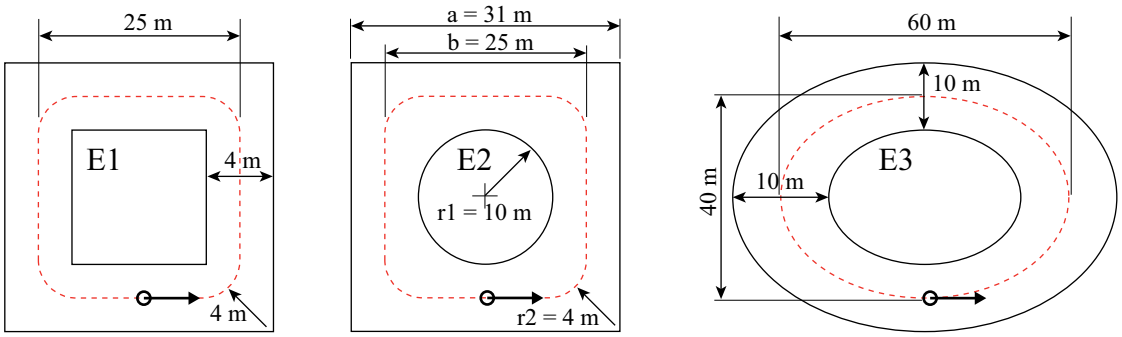
Sin embargo, debe resaltarse el hecho de que cuando los elementos presentes en el entorno son simples segmentos lineales, un B-spline de aproximación lineal es más que suficiente. Esto es importante por el hecho de que el mínimo número de puntos de control necesarios para definir una B-spline coincide con el orden de la curva. Así, un spline cúbico precisa al menos de 4 puntos de control, mientras que el spline

lineal más simple se puede definir con sólo 2; los puntos inicial y final de un único segmento lineal. Por tanto, a pesar de que los splines cúbicos pueden representar adecuadamente geometrías rectilíneas, su uso es poco eficiente desde el punto de vista del rendimiento computacional, puesto que incrementa de manera innecesaria el número de elementos estrictamente necesarios para modelar el entorno.

6.3. Precisión de los Algoritmos

Con el objetivo de evaluar la precisión y eficacia de los algoritmos, se han realizados experimentos con datos simulados utilizando tres entornos sintéticos diferentes. Estos entornos se pueden observar en la figura 6.3. La figura 6.3(a) muestra un entorno cíclico formado únicamente por paredes planas, al cual nos referiremos como **pasillo cuadrangular**.

Este primer entorno no representa ningún reto para algoritmos de SLAM geométrico tradicionales basados en el uso de segmentos como entidad descriptiva del entorno. Por supuesto, cuando se utilizan splines como herramienta de modelado la existencia de segmentos tampoco es un inconveniente; antes al contrario, un segmento es un tipo particular de curva. El modelado mediante splines no hace sino extender las capacidades de representación a situaciones más generales. En particular, un segmento lineal puede ser representado utilizando un spline lineal (orden $\kappa = 2$), pero también utilizando splines de orden superior, como por ejemplo los cúbicos.



(a) Pasillo cuadrangular (b) Pasillo mixto (c) Pasillo anular

Figura 6.3: Entornos sintéticos utilizados en la simulaciones. (a) Pasillo cuadrangular, formado únicamente por tramos de pared rectos. (b) Entorno formado por una mexcla de estructuras rectas y curvas. (c) Pasillo elíptico, formado exclusivamente por elementos curvos.

La figura 6.3(b) muestra un entorno algo más complejo, al que nos referiremos como **entorno mixto**. En este caso la parte interna del pasillo es modelada mediante un elemento de sección circular (un cilindro en el espacio tridimensional). La presencia de geometrías curvas impediría una descripción completa del entorno utilizando simples segmentos como herramienta de modelado. Se podría considerar la existencia de un círculo en el entorno, e intentar estimar la posición de su centro y radio dentro del marco de trabajo del filtro extendido de Kalman. Sin embargo,

hay que recalcar el hecho de que, incluso en este caso, el algoritmo de modelado debería conocer a priori la existencia de una geometría precisamente circular en el entorno. La técnica de modelado que propone esta tesis es inmune a esta dependencia geométrica. Al no presuponer la existencia de una geometría concreta, intenta interpretar los datos suministrados por el sensor tal cual le llegan. La única característica que el algoritmo se encarga de buscar, de manera activa, son las esquinas, como ya sabemos.

Finalmente, la figura 6.3(c) muestra un entorno verdaderamente complicado para cualquier algoritmo de localización y modelados simultáneos existente. Aquí la ausencia de elementos rectilíneos es total, y el entorno cíclico ha sido modelado empleando únicamente dos elipses. A este entorno lo denominamos **pasillo anular**.

En los tres casos, la línea roja de trazos muestra la trayectoria recorrida por un robot móvil simulado en el seno del entorno virtual. Dicho robot está equipado con un láser capaz de tomar medidas en un rango angular de 180° hacia delante y una distancia máxima de 8 metros. Las medidas son simuladas midiendo la distancia del robot a los obstáculos del entornos con una precisión de 1°, lo cual supone un total de 181 medidas que es capaz de adquirir el sensor. A estas medidas se agrega ruido gaussiano generado artificialmente con desviación típica $\sigma_L = 0{,}05m$.

En todos los casos el robot realiza un doble recorrido de la trayectoria, comenzando en el punto marcado por el círculo marcado sobre esta y deteniéndose en el mismo punto tras haber completado dos vueltas al entorno. Las medidas odométricas simuladas también son contaminadas con ruido gaussiano cuyas desviaciones típicas son las siguientes:

- Para el experimento del pasillo cuadrangular:

$$\begin{aligned}\sigma_x &= 0{,}001 + 0{,}1\Delta x_r m \\ \sigma_y &= 0{,}001 + 0{,}1\Delta y_r m \\ \sigma_\phi &= 0{,}02 + 0{,}1\Delta \phi_r m\end{aligned}$$

- Para el experimento del pasillo mixto:

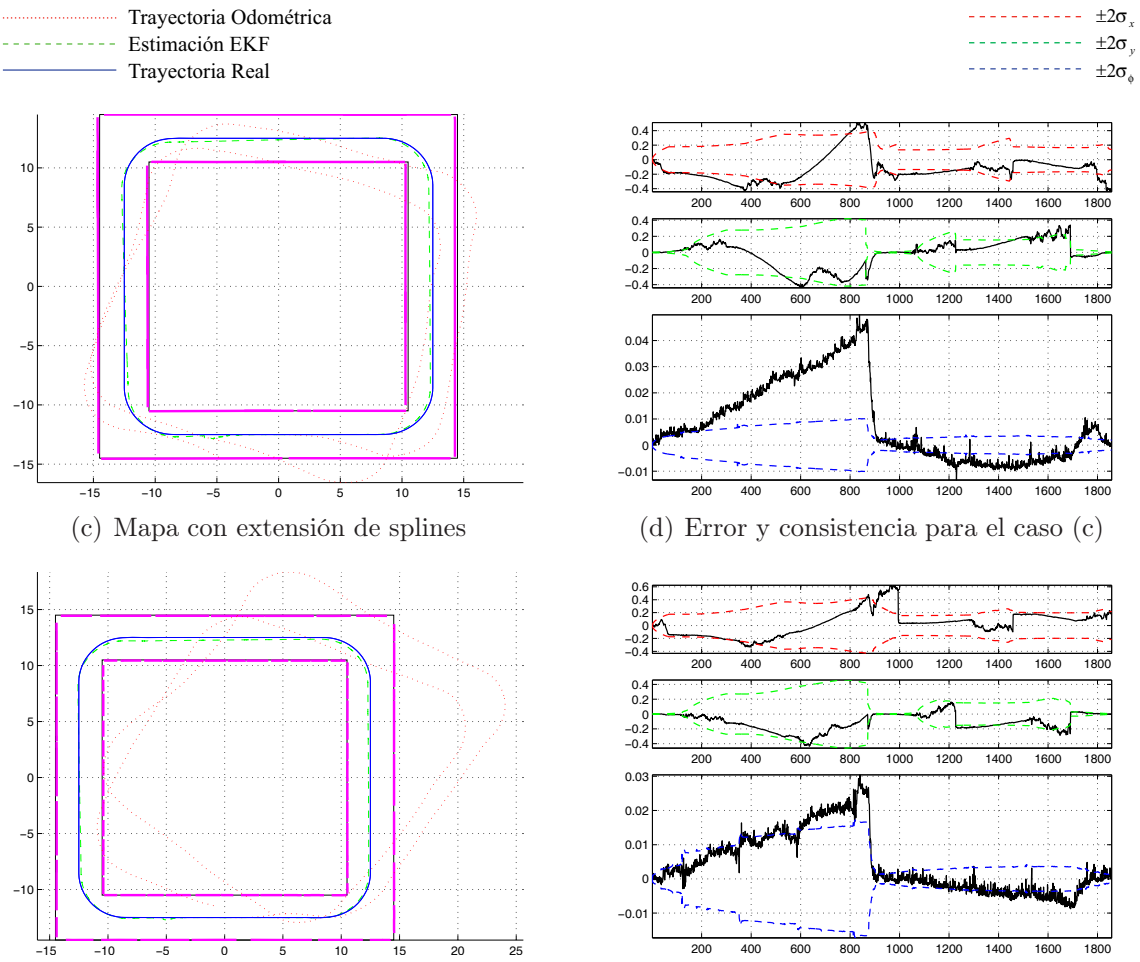
$$\begin{aligned}\sigma_x &= 0{,}001 + 0{,}1\Delta x_r m \\ \sigma_y &= 0{,}001 + 0{,}1\Delta y_r m \\ \sigma_\phi &= 0{,}02 + 0{,}1\Delta \phi_r m\end{aligned}$$

- Para el experimento del pasillo anular:

$$\begin{aligned}\sigma_x &= 0{,}001 + 0{,}05\Delta x_r m \\ \sigma_y &= 0{,}001 + 0{,}05\Delta y_r m \\ \sigma_\phi &= 0{,}02 + 0{,}1\Delta \phi_r m\end{aligned}$$

En las figuras 6.4, 6.5 y 6.6 se muestran los resultados de los experimentos de SLAM para el caso en que los splines son progresivamente extendidos a medida que

se descubren nuevos tramos de los objetos contenidos en el mapa (mitad superior de las figuras) tal y como se explicó en la sección 4.6.2 (página 114), y para el caso en que esta extensión no sea realizada (mitad inferior de las figuras). En esta última situación, las observaciones asociadas con objetos contenidos en el mapa son utilizadas para actualizar la estimación, mientras que se espera a obtener observaciones no emparejadas con ningún objeto del mapa para incluirlas en el mapa como una nueva entidad.



(c) Mapa con extensión de splines

(d) Error y consistencia para el caso (c)

(e) Mapa sin extensión de splines

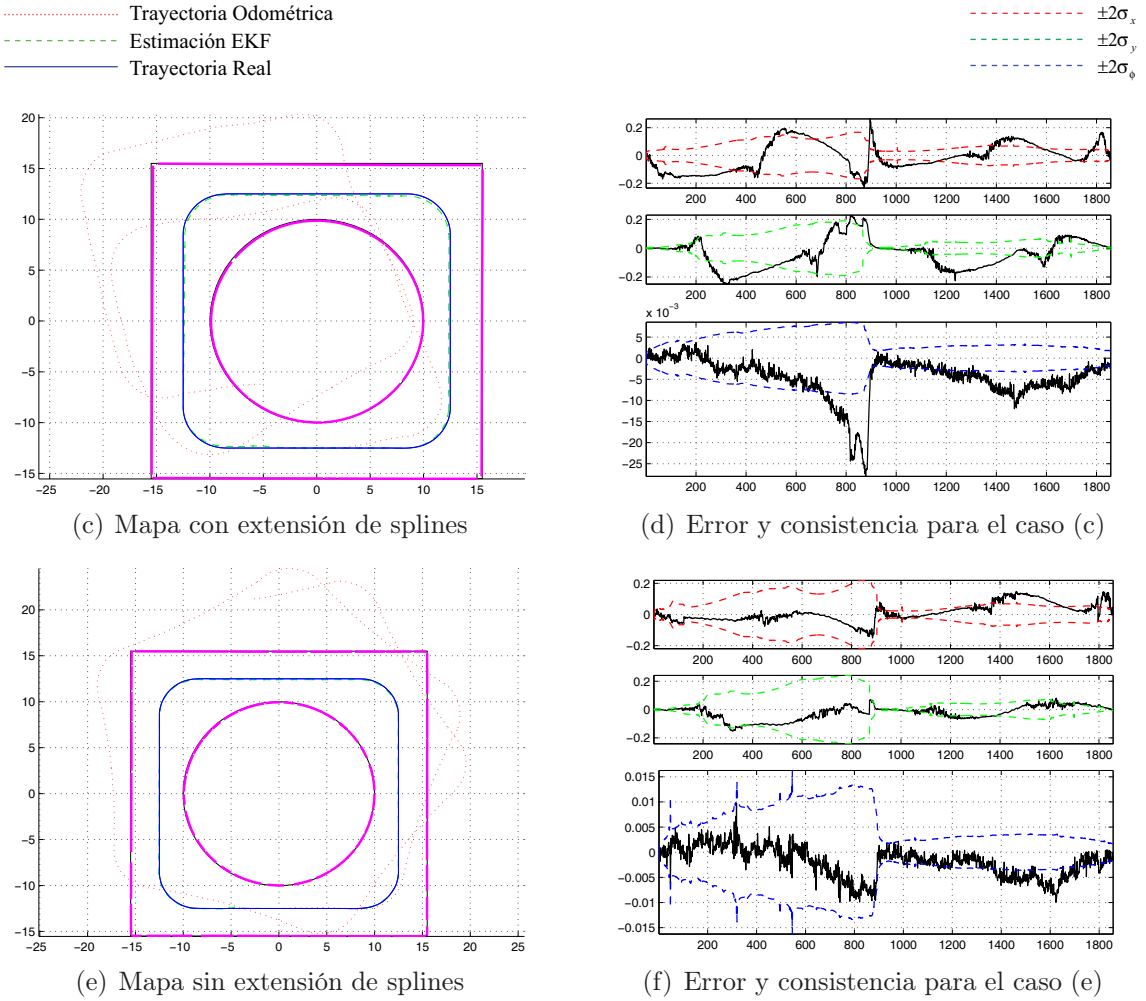
(f) Error y consistencia para el caso (e)

Figura 6.4: Experimentos de precisión y consistencia para el pasillo cuadrangular de la figura 6.3(a).

Las tres figuras muestran en su mitad izquierda los mapas obtenidos en cada experimento, junto con la trayectoria real, la trayectoria odométrica, y la obtenida por el algoritmo de SLAM. La configuración real del entorno sintético se muestra en trazo oscuro, mientras que el mapa obtenido se ha dibujado en color magenta. Se observa la buena calidad de los resultados obtenidos, que muestran mapas que representan adecuadamente las diferentes estructuras reales de los entornos. En el caso de las figuras 6.4 y 6.5 las líneas magentas se superponen casi perfectamente con su verdadera ubicación, mientras que los resultados correspondientes al pasillo anular (Fig. 6.6) son, como podía esperarse, algo peores.

Sin embargo, la calidad de los mapas correspondientes al pasillo anular resulta especialmente llamativa si se tienen en cuenta los siguientes puntos:

- Ambos mapas representan adecuadamente la configuración real del entorno, al menos desde un punto de vista topológico, a pesar de que exista un error apreciable en la posición de los diferentes elementos.
- Ningún algoritmo de SLAM geométrico existente es capaz de modelar adecuadamente entornos de estas características.
- La calidad de los resultados es especialmente buena, si se tiene en cuenta la pésima calidad de la odometría disponible. Esto viene a indicar que la integración de las medidas proporcionadas por el láser, realmente cumple su función permitiendo la adecuada corrección de la trayectoria estimada para la máquina.



(c) Mapa con extensión de splines

(d) Error y consistencia para el caso (c)

(e) Mapa sin extensión de splines

(f) Error y consistencia para el caso (e)

Figura 6.5: Experimentos de precisión y consistencia para el pasillo mixto de la figura 6.3(b).

Por otra parte, cuando existe verdadera certeza acerca de la trayectoria real de la máquina a lo largo del experimento —como sucede en las simulaciones—, es

posible realizar algunas comprobaciones que permitan tener una idea de la calidad de los algoritmos desde el punto de vista de su consistencia. Por este motivo se ha incluido en la mitad derecha de cada figura la representación de los diferentes errores odométricos (en la coordenada x, ϵ_x, en la coordenada y, ϵ_y, y en la orientación ϕ_r, ϵ_ϕ) frente al doble de la desviación típica estimada para cada uno de ellos, que es extraída de la matriz de covarianzas del sistema en cada instante, $\mathbf{P}_{k|k}$.

El filtro extendido de Kalman asume que todas las variables estocásticas involucradas en el proceso de estimación, siguen una distribución gaussiana centrada en el auténtico valor de la variable. Si esto es cierto, el error cometido en la estimación debería permanecer la mayoría del tiempo dentro de los límites definidos por 4 veces la desviación típica estimada para esa variable. En particular, para una distribución normal, el 95 % de los errores cometidos deberían estar comprendidos dentro del intervalo definido por más menos la desviación típica de cada una de las variables:

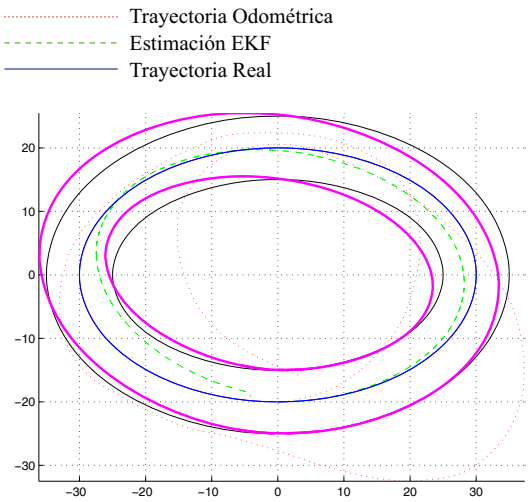
$$-2\sigma_x \leq \epsilon_x \leq 2\sigma_x \tag{6.2}$$
$$-2\sigma_y \leq \epsilon_y \leq 2\sigma_y \tag{6.3}$$
$$-2\sigma_\phi \leq \epsilon_\phi \leq 2\sigma_\phi \tag{6.4}$$

Lamentablemente, una de las limitaciones del filtro de Kalman es la inevitable aparición de inconsistencia, que resulta evidente en ocasiones tras ser procesadas unas cuantas muestras. Esto se debe fundamentalmente a las linealizaciones introducidas en las ecuaciones de los modelos empleados, que hacen que los ruidos presentes en el proceso (a pesar de que estos puedan ser verdaderamente Gaussianos en origen) no se vean propagados en la realidad como ruido Gaussiano.

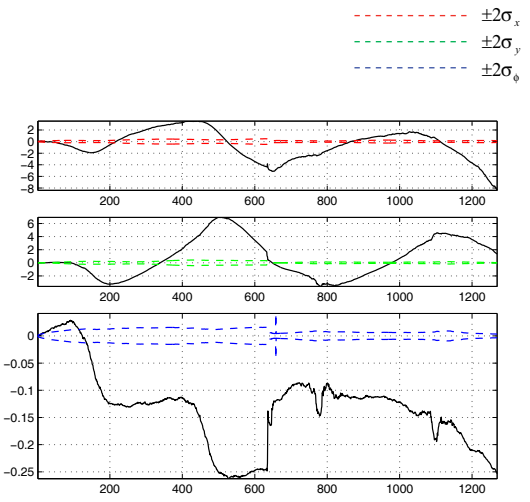
En las figuras se pone de manifiesto que el algoritmo propuesto comparte las mismas limitaciones de otras soluciones del tipo SLAM-EKF. Más tarde o más temprano, la inconsistencia del algoritmo resulta evidente, y ello puede llevar a resultados impredecibles a la hora de cerrar bucles, que pueden perjudicar seriamente la estimación final obtenida.

Finalmente, cabe resaltar el hecho de que la extensión de los splines parece tener un efecto negativo sobre la consistencia del algoritmo. Así, para un mismo experimento, se observa que la inconsistencia tiende a manifestarse antes cuando las extensiones son realizadas, que cuando estas no se aplican. Esto es algo perfectamente lógico, puesto que la manera de integrar la nueva información sobre un elemento existente en el mapa, depende de la parametrización escogida para los nuevos datos que, a su vez, es obtenida utilizando la mejor estimación disponible para la posición del robot. Por lo tanto, al realizar las extensiones se introducen simplificaciones y aproximaciones adicionales que, como no podía ser de otro modo, repercuten en las propiedades del algoritmo.

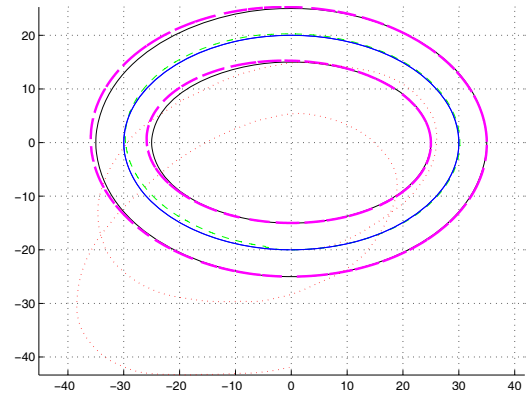
Sobre este aspecto incidiremos de nuevo en la sección siguiente.



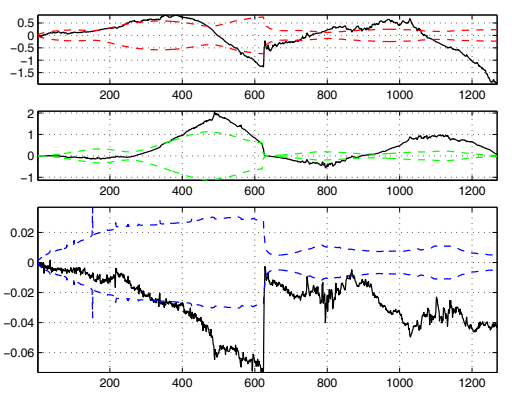
(c) Mapa con extensión de splines



(d) Error y consistencia para el caso (e)



(e) Mapa sin extensión de splines



(f) Error y consistencia para el caso (e)

Figura 6.6: Experimentos de precisión y consistencia para el pasillo anular de la figura 6.3(c).

6.4. Consistencia de los Algoritmos

Es bien conocido el hecho de que una de las mayores limitaciones de las soluciones SLAM-EKF es la inconsistencia del algoritmo debida a los errores de linealización [44], que pueden originar un fallo irrecuperable de la estimación cuando la auténtica incertidumbre de la orientación del robot supera un límite [12].

El origen y las causas de esta inconsistencia cuando los objetos del mapa contienen información angular también han sido estudiados con profundo detalle en [158].

A continuación analizamos con algo más de detalle las propiedades del algoritmo desde el punto de vista de su consistencia.

6.4.1. Un síntoma de inconsistencia: excesiva ganancia de informacion

Como Bailey *et al.* señalan en [12], un síntoma de la inconsistencia es la excesiva ganancia de información, que origina que cualquier implementación de SLAM basada en EKF se vuelva optimista tras un cierto período de tiempo. Por optimista se entiende aquí que la auténtica covarianza del sistema es mayor que la estimada por el filtro.

Este hecho se ponía de manifiesto en los resultados experimentales del apartado anterior, donde se observaba claramente que la distribución del error cometido en la estimación presentaba una desviación típica que sobrepasaba con creces la extraída de la matriz de covarianzas estimada tras un cierto intervalo de tiempo, variable en cada caso.

Para analizar el efecto de la ganancia de información en el problema del SLAM, se ha repetido el experimento descrito en la sección 6.2 utilizando splines cúbicos para la aproximación de los contornos, y ejecutando el algoritmo de SLAM completo para un robot estático en el origen de coordenadas.

Si el robot no se mueve, el sentido común nos dice que la incertidumbre asociada a la localización del robot nunca debería disminuir, sin importar el número de observaciones que el robot realice de su entorno.

Sin embargo, el resultado experimental que se recoge en la figura 6.7 nos indica todo lo contrario. Se han utilizado diferentes valores para la desviación típica inicial de la orientación de la máquina ($\sigma_\phi(0|0) = 0{,}01 \quad rad$, $\sigma_\phi(0|0) = 0{,}05 \quad rad$, y $\sigma_\phi(0|0) = 0{,}2 \quad rad$,), y se han promediado los resultados de 50 simulaciones Monte Carlo para cada uno de estos valores.

En cada caso, la desviación típica estimada para la orientación del robot muestra un inmediato descenso. Además, cuanto mayor es la incertidumbre inicial en la orientación, más rápida y abrupta es la caída, y menor es el valor final de la estimación obtenida.

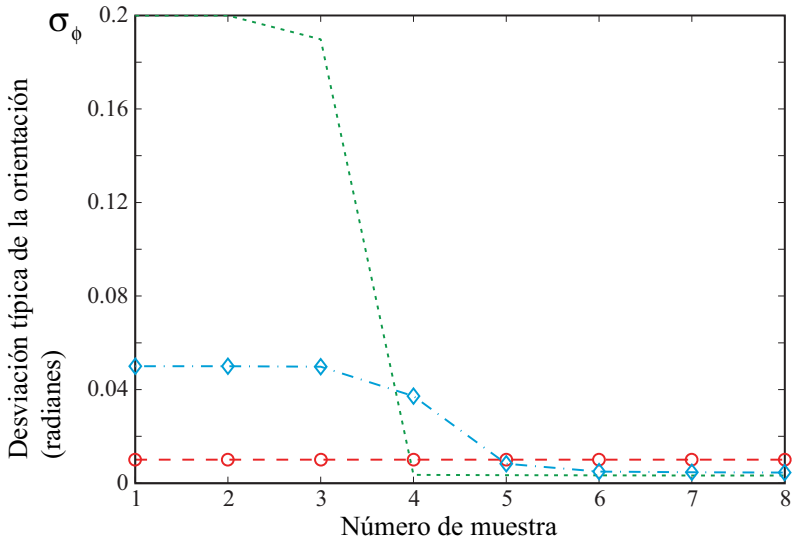


Figura 6.7: ganancia de información en la orientación para un vehículo estacionario. La desviación típica estimada σ_ϕ cae abruptamente tras pocas actualizaciones del filtro. Los resultados muestran valores medios de 50 simulaciones Monte Carlo para tres valores iniciales diferentes.

6.4.2. Experimentos Monte Carlo de consistencia

Los resultados experimentales de la sección 6.3 sugieren que la extensión de los splines tiene algún tipo de impacto sobre la consistencia del algoritmo. Aquí se analiza en detalle este efecto utilizando el entorno artificial mostrado en la figura 6.8. Un robot simulado completa en este caso un único bucle. Los ruidos odométricos simulados siguen distribuciones normales con desviaciones típicas $\sigma_x = 5 + 0{,}1\Delta x$ mm, $\sigma_y = 5 + 0{,}1\Delta y$ mm y $\sigma_\phi = 0{,}02 + 0{,}1\Delta\phi$ radianes. Además se simulan medidas adquiridas por un sensor láser montado sobre la plataforma, con un alcance de 8 metros y ruido normal de media 0 y desviación típica $\sigma_L = 5$ mm.

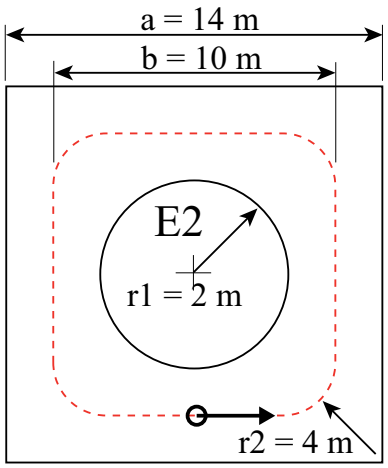


Figura 6.8: Entorno sintético utilizado en los experimentos de consistencia.

Cuando el verdadero estado del vehículo $\mathbf{x}_r\,(k)$ es conocido en cada instante, se puede utilizar el error cuadrático de estimación normalizado (NEES) para caracte-

rizar el algoritmo desde el punto de vista de su consistencia. Dicho error se define como [12]:

$$\epsilon(k) = (\mathbf{x}_r(k) - \hat{\mathbf{x}}_r(k|k))^T \mathbf{P}_r^{-1}(k|k)(\mathbf{x}_r(k) - \hat{\mathbf{x}}_r(k|k)) \tag{6.5}$$

El NEES medio tras la ejecución de N simulaciones Monte Carlo, bajo la hipótesis de que el filtro es consistente y aproximadamente lineal, y que los ruidos son gaussianos, sigue una distribución χ^2 con tantos grados de libertad como la dimensión del espacio, $dim(\mathbf{x}_r)$ (en nuestro caso 3). Por lo tanto el valor promedio de $\epsilon(k)$ se aproxima a la dimensión del estado a medida que N tiende a infinito:

$$E[\epsilon(k)] = dim(\mathbf{x}_r) = 3 \tag{6.6}$$

Así, podemos validar las hipótesis anteriores comprobando que este es precisamente el caso. La figura 6.9 muestra los resultados correspondientes a 50 simulaciones Monte Carlo para 3 configuraciones diferentes. En todos los casos se muestra para la gráfica del NEES la franja en la que deberían permanecer el 95 % de las muestras, limitada por el intevarlo [2,36, 3,72].

En la primera fila se muestran los resultados en el caso en que los splines son progresivamente extendidos. La segunda fila muestra los resultados para el mismo experimento, pero en este caso la matriz de ruido del sensor ha sido multiplicada por un factor de 3 al ser utilizada en el filtro. Finalmente, la tercera fila muestra los resultados para el caso en que las curvas no son extendidas, y se utiliza la auténtica matriz de covarianzas que modela el ruido del sensor. También se ha incluido el error cuadrático medio, promediado para las 50 ejecuciones, que da una idea de la buena precisión obtenida por el algoritmo:

$$MSE = (\mathbf{x}_r(k) - \hat{\mathbf{x}}_r(k|k))^T (\mathbf{x}_r(k) - \hat{\mathbf{x}}_r(k|k)) \tag{6.7}$$

En todos los casos el filtro se vuelve optimista apenas han sido procesadas las primeras 50 muestras. Además, el hecho de incrementar la covarianza del sensor empleada en las ecuaciones del filtro, no parece mejorar el resultado de manera importante.

La aparición de inconsistencia resulta mucho más abrupta y evidente cuando los splines son extendidos, como cabía esperar. A pesar de este peor comportamiento en el caso de realizarse la extensión de los objetos, se defiende su empleo por el siguiente motivo: puesto que la inconsistencia de la estimación es una característica inherente e inevitable de cualquier implementación SLAM-EKF, se prefiere realizar la extensión de los objetos puesto que así se ralentiza o atenúa el crecimiento en el número de puntos de control contenidos en el mapa.

Supongamos por ejemplo que queremos modelar un entorno formado por una única pared de 20 metros de longitud. Para hacerlo, generaremos vectores de nodos

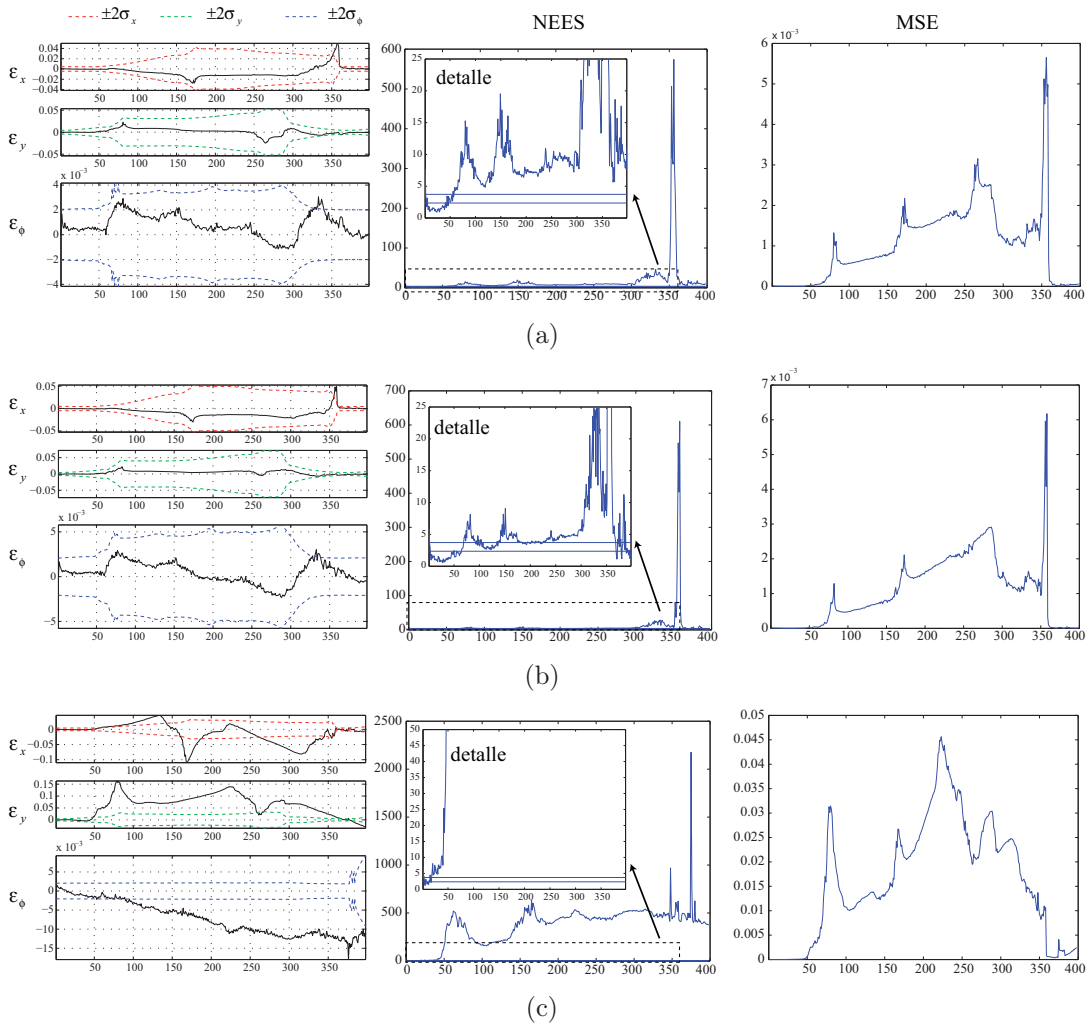


Figura 6.9: Experimentos Monte Carlo de consistencia (50 ejecuciones). La primera columna muestra los errores de localización ϵ_x, ϵ_y y ϵ_ϕ junto al intervalo definido por 4 veces sus respectivas desviaciones típicas: $\pm 2\sigma_x$, $\pm 2\sigma_y$, $\pm 2\sigma_\phi$ (las dimensiones lineales están en metros y las angulares en radianes). La segunda columna muestra el NEES promedio y la tercera el error cuadrático medio, MSE, también promediado para los 50 experimentos. Por filas: (a) No se realizan extensiones de objetos, (b) No se realizan extensiones y la matriz de ruido de los sensores se multiplica por un factor de 3 y (c) se realiza la extensión de objetos.

con un espaciado internodal aproximado de 2 metros. Realizando el proceso de extensión, podríamos describir completamente el entorno utilizando un único spline formado por $20/2 = 10$ tramos polinomiales. Si el spline es cúbico, esto supone que estará definido por 17 nodos y 13 puntos de control.

Supongamos ahora que no realizamos las extensiones y, para situarnos en un escenario desfavorable, que hemos necesitado 10 splines individuales, cada uno de 2 metros, para describir la misma geometría. Incluso en el caso de que cada una de estas curvas estuviera formada por un único tramo polinomial, serían necesarios tantos puntos de control como el orden escogido para representarla. Si en este caso también hemos utilizado splines cúbicos, nos da un total de $10 \cdot 4 = 40$ puntos de control para representar la misma geometría.

Así pues, el hecho de extender los objetos, a pesar de empeorar las propiedades de consistencia del filtro (por otra parte, algo inevitable), permite ralentizar el crecimiento del tamaño del mapa y, por lo tanto, luchar contra el otro gran obstáculo que es el coste computacional de la implementación.

6.5. Experimentos con Datos Reales

Se han realizado diferentes experimentos con datos extraídos en entornos reales para validar los resultados presentados. Primeramente se presentan los resultados obtenidos utilizando un único filtro, para luego presentar resultados utilizando la técnica de submapas descrita en el capítulo 5.

6.5.1. Mapas obtenidos con un único filtro

La figura 6.10 muestra dos mapas del mismo entorno, con predominancia de geometrías planas (paredes típicas, en este caso paneles de separación de expositores en una feria). El mapa de la figura 6.10(a) se ha construido en este caso utilizando B-splines cúbicos como herramienta de modelado, y contiene un total de 81 objetos definidos por 332 puntos de control. En el caso de la figura 6.10(a), se ha utilizado el mismo conjunto de datos, pero procesado en este caso utilizando únicamente B-splines lineales (orden $\kappa = 2$).

Como puede apreciarse, los resultados son prácticamente equivalentes, pero en este segundo caso el mapa está formado por 83 curvas definidas por un total de 166 puntos de control. Es clara por tanto la reducción del número de elementos que conforman el mapa y, por consiguiente, la disminución del coste computacional del algoritmo a la hora de actualizar la estimación.

El mapa de la figura 6.11 reproduce el resultado de la figura 6.10(a), mostrando los puntos de control de los splines cúbicos que conforman el mapa.

La figura 6.12 muestra el mapa de un entorno más grande y complejo, que contiene una mezcla de elementos arquitectónicos planos y curvos, con clara predominancia de estos últimos. Se trata del Museo de las Ciencias “Príncipe Felipe”, cuyo aspecto

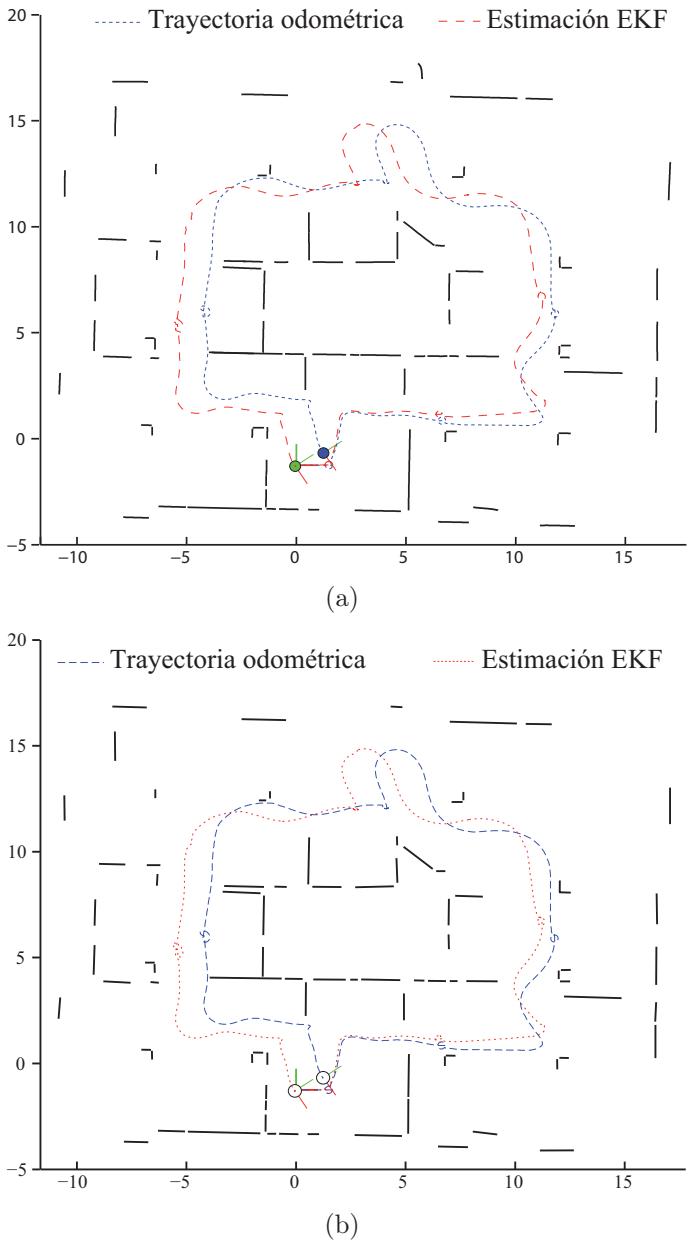


Figura 6.10: Mapa de la feria Indumática celebrada en la Escuela Técnica Superior de Ingenieros Industriales de la UPM en Madrid. (a) Utilizando splines cúbicos y (b) utilizando splines lineales.

Figura 6.11: Mapa de la feria Indumática con splines cúbicos, mostrando los puntos de control.

interior se puede observar en las imágenes de la página 16. En este caso el mapa está formado por un total de 462 puntos de control que definen un total de 96 splines.

Tanto en el experimento de la feria Indumática como en el del Museo de las Ciencias, se empleó un robot B21r dotado de un telémetro láser SICK LMS200 que fue empleado para adquirir datos a una frecuencia de 5 Hz. En todos los casos los mapas son construidos utilizando un espaciado fijo de 2 nodos/m para la generación de los vectores de nodos. En ningún caso se ha utilizado ningún tipo de restricción geométrica.

Una de las características del SLAM es que a la hora de probar los algoritmos no es necesario trabajar directamente con la plataforma móvil. Se pueden guardar los datos adquiridos durante una exploración, para más tarde procesarlos en un computador. Esto permite intercambiar conjuntos de datos con otros investigadores del mismo área, de manera que los resultados puedan ser evaluados comparativamente. Este es el caso de Radish [102], el repositorio del que se ha obtenido el conjunto de datos cuyo mapa se muestra en la figura 6.13(a) [1].

En este caso el entorno ha sido modelado utilizando splines cúbicos, procesando las 5620 primeras muestras del conjunto de datos. Como puede apreciarse, los resultados son equivalentes a los obtenidos mediante un mapa de ocupación de celdillas (figura 6.13(a)), pero en este caso el modelado es mucho más compacto, siendo los elementos del entorno modelados individualmente mediante curvas paramétricas.

6.5.2. Técnica de submapas

A continuación se presentan resultados obtenidos con la técnica de submapas descrita en el capítulo 5. Como ya se comentó entonces, el principal objetivo de esta técnica es limitar el coste computacional de los mapas, descomponiendo el problema global en unidades individuales más sencillas. Llegados a este punto, se vuelve de especial importancia la posibilidad de analizar la estructura geométrica de los diferentes submapas generados, de manera que se puedan establecer relaciones entre ellos que permitan actualizar en consecuencia la estructura global del mapa.

La figura 6.14 muestra el mismo mapa de la figura 6.13, pero en este caso se han generado diferentes submapas limitados en tamaño a 80 puntos de control —lo cual permite su construcción en tiempo real sin problemas—, y se ha empleado la técnica de asociación de datos descrita en el capítulo 5 para establecer correspondencias entre los distintos modelos locales, de manera que su configuración global pueda ser actualizada mediante un filtro extendido de Kalman.

La figura 6.14(a) muestra los diferentes submapas construidos hasta el momento en que se cierra el primer bucle. En el detalle de la misma figura, se muestra las asociación obtenida, que es utilizada para actualizar el modelo tal como puede verse en la figura 6.14(b). Finalmente, en la figura 6.14(c) se muestra la configuración final del mapa tras completar dos bucles alrededor del circuito principal.

En la figura 6.15 se muestran lo resultados para el Museo "Príncipe Felipe" uti-

[1] Gracias a Dieter Fox por proporcionar estos datos.

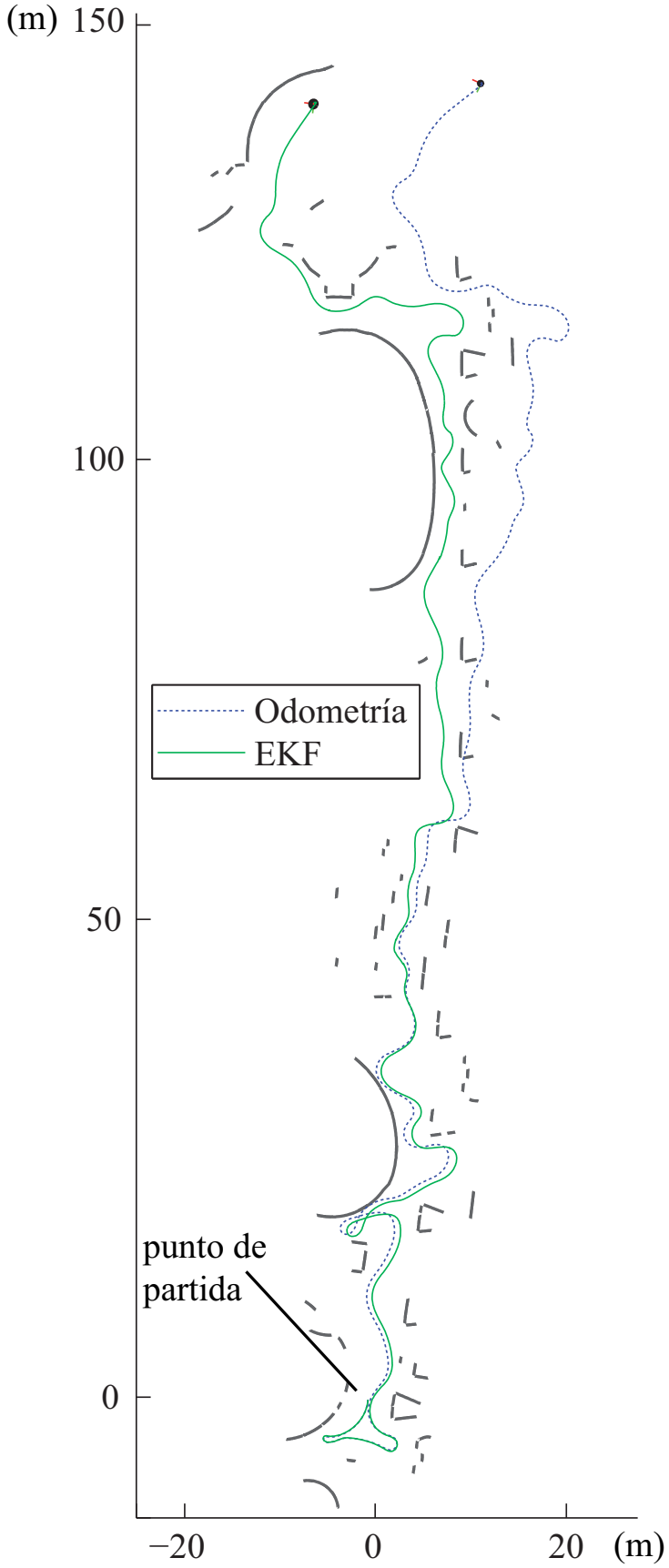
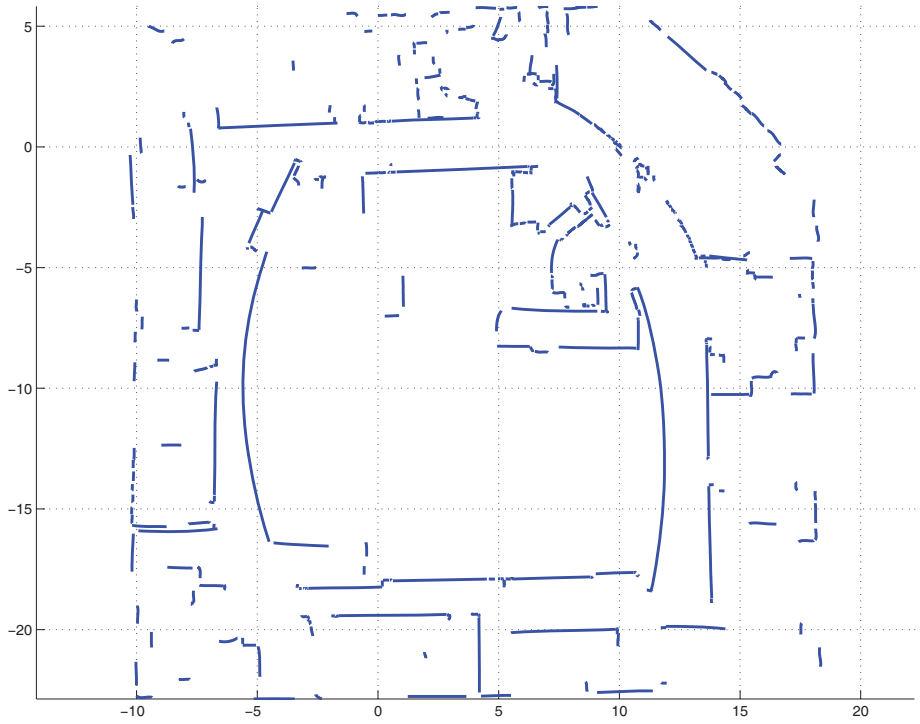


Figura 6.12: Mapa del Museo de las Ciencias «Príncipe Felipe» (Valencia, España)



(a)

(b)

Figura 6.13: Mapa de los laboratorios de Intel en Seattle construido con splines cúbicos (a) y con un mapa de ocupación de celdillas (b).

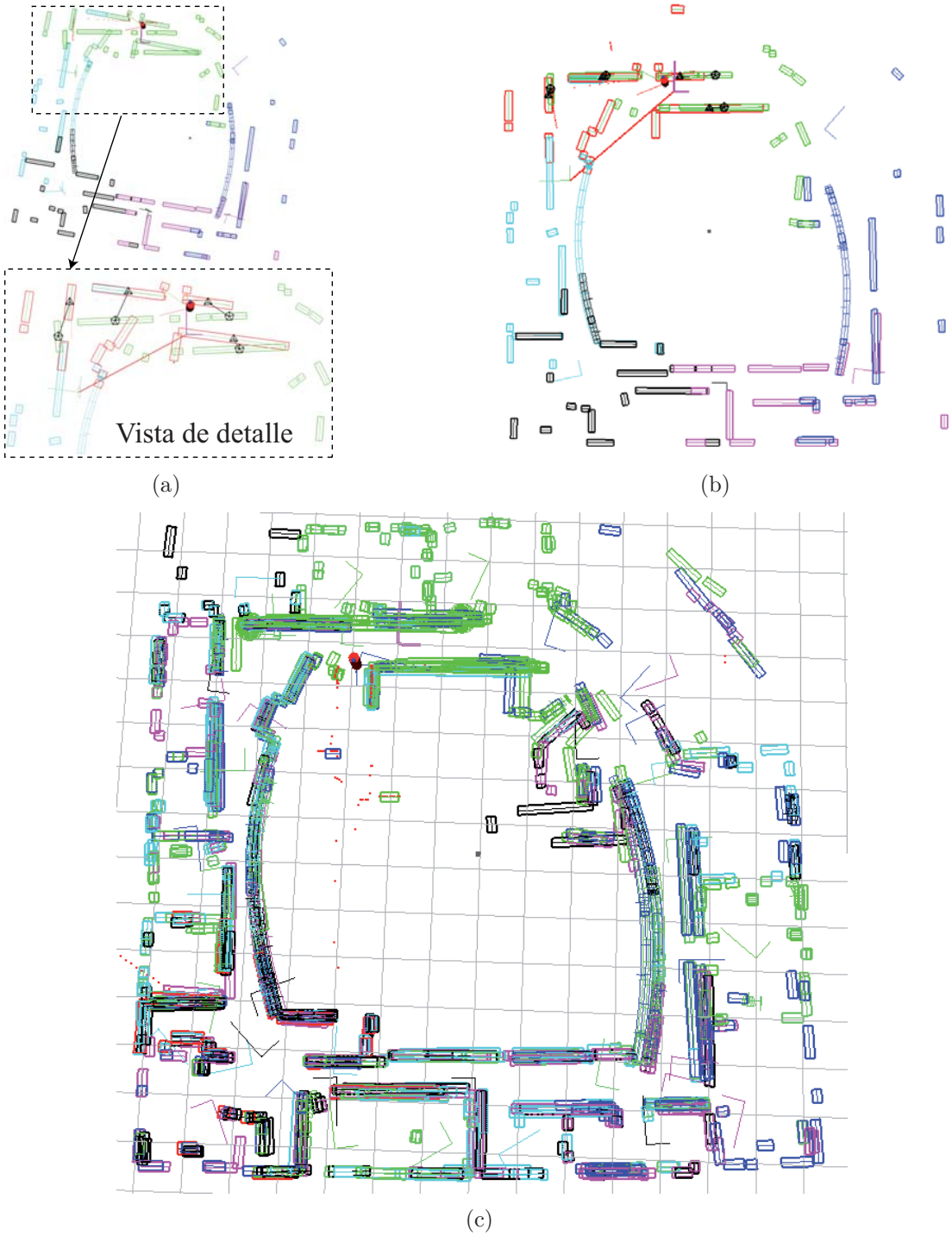


Figura 6.14: Mapa de los Laboratorios de Investigación de Intel en Seattle, con técnica de submapas. (a) Configuración antes de cerrar el primer bucle. (b) Configuración tras cerrar el primer bucle y actualizar las posiciones relativas de los diferentes submapas. (c) Configuración final del modelo.

lizando la técnica de submapas. Como puede observarse, es posible obtener un modelado paramétrico de este entorno de grandes dimensiones (más de 150 metros de un extremo al otro del mapa), en el cual estructuras de gran complejidad han sido modeladas mediante B-splines cúbicos.

Se han resaltado dos zonas mediante círculos rojos. Se indican de este modo puntos en los que la superposición entre los mapas que modelan estas áreas no es todo lo perfecta que cabría esperar. Esto se debe a la falta de estructura común en los mapas involucrados en el proceso de asociación de datos en la zona. Esto se debe a que el robot, al visitar nuevamente esta parte del entorno, no ha vuelto a observar correctamente o en toda su extensión los elementos físicos contenidos en submapas construidos con anterioridad.

Se aprecia así una gran limitación de la técnica de asociación propuesta en el capítulo 5 a la hora de establecer correspondencias entre mapas locales, en comparación con los resultados obtenidos en el problema de localización global. En este último caso, se dispone de un mapa completo del entorno, y un conjunto de observaciones que permiten localizar al robot en el mismo. Es, por tanto, sencillo que el proceso de asociación de datos encuentre una correspondencia correcta entre la observación adquirida y los objetos que se encuentra en el mapa completo del entorno.

En el caso que nos ocupa, es importante que los diferentes submapas tengan un tamaño suficiente, y que un número suficiente de estructuras se encuentren repetidas en submapas próximos para que la asociación de datos dé sus frutos. En caso de no existir suficiente estructura común, la asociación no será posible, o incluso será incorrecta (sobre todo en el caso de entornos muy simétricos). Es preciso por tanto disponer de algún tipo de estrategia más inteligente que contemple estas limitaciones a la hora de adquirir datos de un nuevo entorno (en todos los mapas presentados en esta sección, los datos han sido adquiridos mediante guiado manual de la plataforma móvil).

6.6. Conclusiones

En este capítulo se han mostrado y validado experimentalmente las propiedades de los algoritmos presentados en esta tesis.

Los experimentos con datos simulados han permitido valorar las propiedades del algoritmo de estimación, tanto desde el punto de vista de su convergencia como desde el punto de vista de su consistencia. Se ha mostrado que la inconsistencia es una característica inevitable e intrínseca de cualquier solución EKF al problema del SLAM y, por tanto, la solución presentada en esta tesis no se diferencia en este sentido de otras similares. Sin embargo, se han mostrado mapas cuya predominancia de elementos curvos los hace imposibles de modelar empleando técnicas geométricas existentes hasta el momento. Además, la calidad de los mapas obtenidos es excepcional, y su representación basada en la utilización de curvas paramétricas los hace idóneos desde el punto de vista de su compacidad, ofreciendo nuevas posibilidades para extraer información geométrica de los mismos, y razonar sobre su estructura.

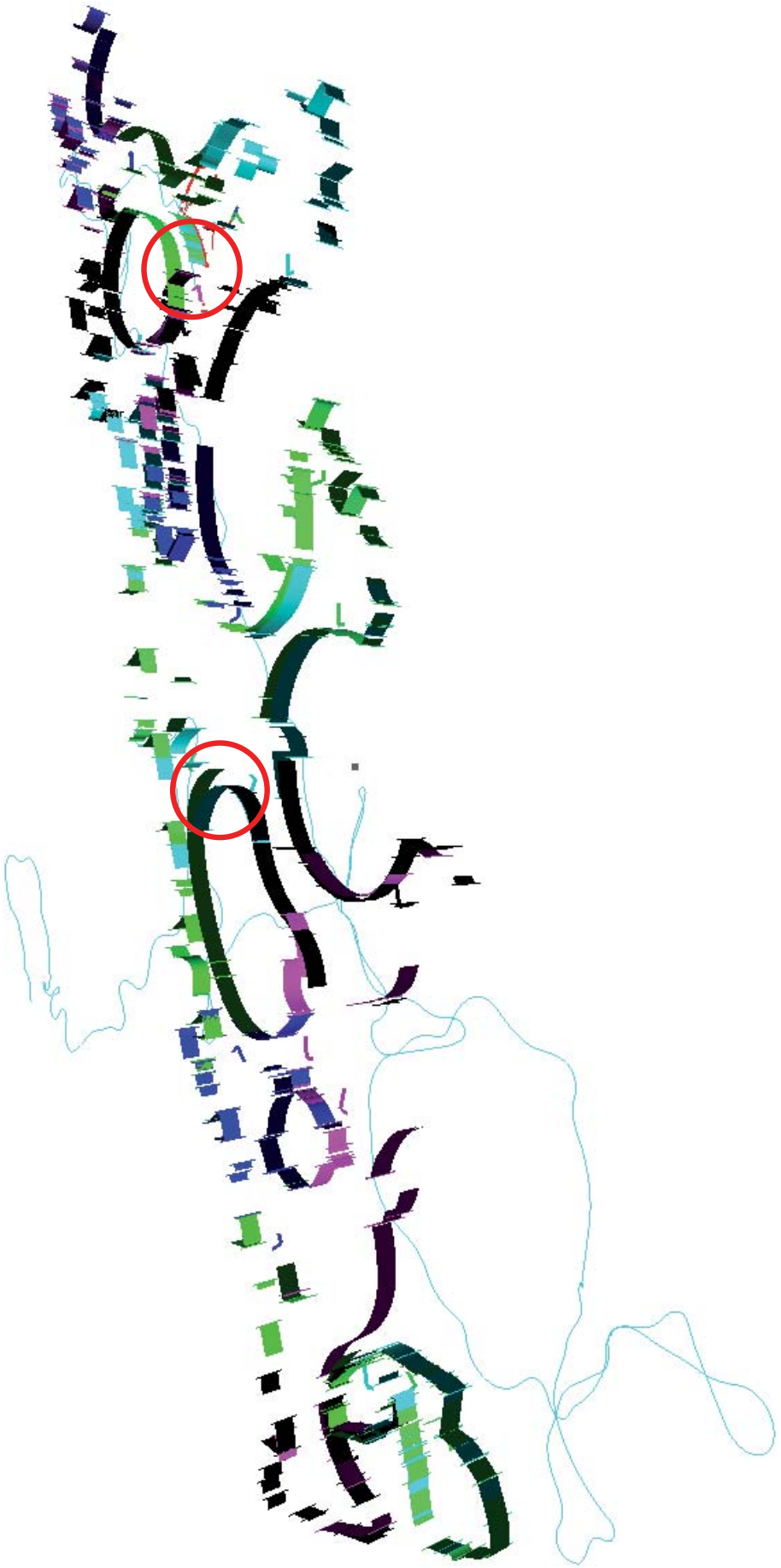


Figura 6.15: Mapa del Museo de las Ciencias "Príncipe Felipe" construido con técnica de submapas.

Los experimentos con datos reales también ofrecen resultados importantes. Ahora, no sólo es posible modelar geométricamente entornos que antes sólo podían ser descritos empleando otras técnicas menos afines al modo de razonamiento espacial humano. Además los mapas obtenidos son extremadamente compactos, y la pureza de sus líneas los hace idóneos para establecer relaciones entre los diferentes elementos del entorno. Se han mostrado resultados utilizando tanto splines lineales como cúbicos, siendo estos últimos los preferidos en el caso general por su mayor versatilidad y capacidad para representar geometrías curvilíneas. Sin embargo, cuando sean rectos los elementos geométricos del entornos, los splines lineales son más que suficientes, siendo los mapas generados similares a los obtenidos mediante un modelado basado en segmentos.

Las propiedades de los nuevos mapas construidos son aprovechadas en la técnica de modelado basada en la descomposición en submapas de tamaño acotado. Es posible así establecer relaciones de correspondencia entre elementos contenidos en distintos mapas, de manera que esta información pueda ser utilizada para actualizar la estructura global de la representación. Se aborda de esta manera el problema del coste computacional del algoritmo, abriéndose las puertas a su implementación en tiempo real. Cabe destacar la coherencia topológica de los mapas obtenidos. Para utilizarlos en tareas de navegación, el robot simplemente tendría que alternar entre los diferentes submapas a medida que se desplaza por el entornos, siendo esta representación igualmente válida para tareas de planificación global.

Capítulo 7

Conclusiones

En este capítulo final de la tesis se resumen las principales conclusiones de la misma, destacando su contribución original y aportaciones al actual estado de desarrollo de las soluciones al problema del SLAM.

7.1. Principales Aportaciones

Con esta tesis se ha presentado una nueva metodología para la localización y el modelado simultáneos en entornos arbitrariamente complejos. Por primera vez, se ha mostrado cómo la potencia matemática y versatilidad geométrica que proporcionan las curvas B-spline, encajan de manera natural en el marco de trabajo del SLAM-EKF. Se consigue así la representación de estructuras físicas complejas de manera paramétrica.

A pesar de la aparente dificultad que esta simbiosis pudiera entrañar, se han presentado algoritmos matemáticos cuya programación resulta sencilla en un computador. A decir verdad, uno de los objetivos no escritos desde el comienzo de esta tesis, ha sido el de intentar mantener los desarrollos, tanto desde el punto de vista conceptual como desde el punto de vista algorítmico, tan sencillos como fuera posible. Este objetivo se ha conseguido, sintetizando ecuaciones cuya forma matricial no sólo facilita su interpretación, sino que también permite su cómoda programación y eficiencia computacional.

Más concretamente, el modelo de observación propuesto —inspirado en las técnicas de trazado de rayos de los gráficos por computador— constituye un artificio matemático-geométrico que, no solo proporciona un mecanismo efectivo para predecir las medidas de un sensor láser a partir del mapa disponible, sino que también simplifica y facilita la obtención de los Jacobianos involucrados en la estimación del estado mediante un filtro extendido de Kalman.

Parece, pues, evidente que cuando otras descripciones geométricas del entorno resultan insuficientes —o imposibles—, cualquier algoritmo de SLAM podría beneficiarse de la versatilidad de esta nueva representación paramétrica.

Si hubiera que resumir el contenido de esta tesis en una sola frase, se podría decir lo siguiente:

> Se ha presentado una metodología que permite extender las actuales fronteras del SLAM geométrico, facilitando el modelado eficiente de entornos poco estructurados, sin tener que depender en la existencia de una primitiva concreta a ser detectada por los sensores del robot.

Así pues, si hasta ahora los algoritmos existentes de SLAM geométrico veían limitada su aplicación a entornos con la suficiente estructura como para poder extraer geometrías sencillas como puntos o segmentos, ahora esto ya no es necesario.

Además, hay que destacar que el método propuesto tiende a descartar la menor cantidad de información posible, integrando datos que con otras metodologías de modelado geométrico se perderían. La información estocástica asociada con las medidas individuales proporcionadas por los sensores del robot queda adecuadamente encapsulada en los puntos de control de las curvas que describen el entorno, todo ello gracias a la aplicación de algoritmos matriciales que transfieren esta información tanto al inicializar un nuevo objeto en el mapa, como al extenderlo progresivamente. La información espacial almacenada de esta forma en la matriz de covarianzas del sistema, permite actualizar el mapa de manera eficiente cuando se cierran bucles durante la exploración.

Finalmente, se ha mostrado cómo la representación analítica de los objetos contenidos en el mapa, constituye una gran ventaja respecto a otras representaciones capaces de modelar estructuras complejas. Nos referimos aquí a los mapas de ocupación de celdillas, o a la técnicas basadas en trayectorias que mantienen el conjunto de datos completo adquirido por el sensor. Si bien es cierto que con este modelado se puede seguir perdiendo parte de información, en la mayoría de las ocasiones se corresponde con estructuras difícilmente modelables (como vegetación, o superficies con una extremada rugosidad o variabilidad geométrica, poco frecuentes), cuya inclusión en el mapa aporta poco a efectos de su utilización posterior. En cualquier caso, hay que hacer notar que esta manera de modelar el entorno no es incompatible con otros procedimientos existentes, resultando especialmente apropiada para su combinación de ellas, adoptando estrategias híbridas existentes en la bibliografía (suponiendo la información más densa modelada con otras técnicas, al mapa construido mediante SLAM-EKF).

Así pues, si bien sigue siendo improbable poder explotar el total de la información adquirida por el sensor, sí es cierto que el método propuesto supera con creces a cualquier algoritmo geométrico existente. Y presenta la ventaja adicional frente a otros mapas densos, de permitir el razonamiento e interpretación geométrica sobre los objetos modelados. Esto puede ser empleado, como se ha visto en esta tesis, para mejorar sustancialmente el proceso de asociación de datos; no sólo en tareas de localización global del robot, sino también a la hora de aplicar técnicas de submapas, que descomponen el problema global en fragmentos más simples.

A modo de resumen general, se comentan brevemente los resultados de cada capítulo, señalando las contribuciones que en ellos se presentan:

- En el capítulo 1 se ha introducido al lector en el dominio de conocimiento de la robótica móvil en general, y en el de la localización y modelado simultáneos en particular. Primero, y desde una perspectiva histórica, se ha mostrado la eterna inquietud del hombre por construir seres artificiales que, aplicando la técnica disponible en cada momento, le sirvieran como mero entretenimiento, como ayuda en su labor, o como simple objeto de regalo o agasajo. La búsqueda del movimiento como imitación de la vida, y el continuo afán de superación en los objetivos tecnológicos planteados, nos ha llevado, ya entrados en el sigo XXI, a disponer de avanzadas plataformas móviles cuya presencia en nuestras vidas es cada vez más frecuente y menos sorprendente.

  Sin embargo, muchos retos e interrogantes permanecen aún abiertos. Uno de ellos —tal vez el más crucial, si pretendemos disponer algún día de robots verdaderamente autónomos, e independientes de nuestra intervención en las labores que les son encomendadas—, es el de obtener mapas del entorno que sirvan a estas máquinas para planificar y ejecutar sus tareas. Cuando se pretende que sean las propias máquinas las que construyan este modelo, el problema se complica de manera extraordinaria; nuestros robots son sofisticados, pero distan mucho de ser perfectos, al igual que sus creadores.

  A pesar de que se han realizado importantes avances en la resolución de este problema de modelado, queda aún mucho por hacer. El problema se complica cuando los entornos no son lo suficientemente estructurados como para que algunas de las soluciones más exitosas puedan ser aplicadas sin obstáculos. Así pues, se impone la necesidad de obtener métodos de modelado y descripciones del entorno cada vez más generales —y, tal vez sea esto lo más importante— que permitan el posterior razonamiento y la interpretación de los mapas construidos; tanto por un mente humana, como por una artificial. Por este motivo se ha querido generalizar las representaciones existentes basadas en modelar el mundo mediante expresiones analíticas; apelando a la unidad última de información espacial del entorno: el objeto, sin importar su forma.

- En el capítulo 2 se han comentado algunas de las soluciones más populares y exitosas al complejo problema del SLAM. Es precisamente esta complejidad la que ha encumbrado a las soluciones que emplean técnicas probabilísticas, que atacan el problema sin tratar de escapar de los principales obstáculos que se plantean: el ruido de los sistemas sensoriales, y la inexactitud de los modelos empleados.

  Como no podía ser de otra manera, cualquier solución al problema tiene que recurrir a determinadas simplificaciones o suposiciones, que hagan la solución, si bien aproximada, al menos practicable. A pesar de que es imposible resumir en un capítulo todos los métodos recogidos en la bibliografía acumulada a lo largo de más de 20 años de investigación en este campo, se ha querido poner de manifiesto la relevancia de algunos de ellos. Por ejemplo, la de aquellos basados en la utilización de un filtro extendido de Kalman, que mantienen información espacial que relaciona a todos los objetos incluidos en el mapa entre sí, y permite actualizar sus posiciones a medida que se va adquiriendo más información.

Quizá lo fundamental de este capítulo sea la definición de tres fronteras principales en el estado actual de las soluciones: (i) La que tiene que ver con el coste computacional de los algoritmos, (ii) la relacionada con las limitaciones de modelado a la hora de representar adecuadamente el entorno, y (iii) la que tiene que ver con la adecuada asociación entre lo que detecta el robot y lo que existe en el mapa, o entre la información contenida en dos mapas diferentes.

Es en estos tres puntos en los que se ha querido incidir en esta tesis. Fundamentalmente en lo que se refiere a la mejora de las técnicas de modelado geométrico. Pero también adoptando soluciones existentes al problema del coste computacional —mostrando que el método propuesto no es ajeno, ni incompatible con estos beneficios—, y mostrando las ventajas de la representación paramétrica de los objetos en el proceso de asociación de datos.

- Puesto que esta tesis plantea como herramienta de modelado la utilización de curvas spline, *i.e.* curvas polinomiales por tramos, no podía faltar un capítulo dedicado a su presentación. Así, el capítulo 3 recoge algunos aspectos fundamentales de esta técnica que, desde su nacimiento para resolver los problemas de modelado planteados por las industrias naval y aeronáutica, ha encontrado aplicación —¡y de qué modo!— en la arquitectura moderna.

  Dado que los splines son empleados de hecho en el diseño de muchas de las actuales edificaciones, no parece descabellado intentar su uso en el procedimiento inverso; *i.e.*, pasar de la realidad física, de nuevo, a su descripción matemática. Dicho y hecho: en este capítulo se presenta la formulación matemática de las curvas B-spline, junto con algunas de sus propiedades más importantes. Dichas propiedades no sólo sirven para mostrar su versatilidad y facilidad de manejo (aunque su formulación recursiva resulte poco atractiva en un principio) y ayudan a comprender y asimilar su potencial aplicación a nuestro problema, sino que se utilizarán en desarrollos presentados en capítulos ulteriores.

  Los B-splines —que no son otra cosa que curvas spline, expresadas como combinación lineal de un cierto tipo de funciones básicas— proporcionan una descripción compacta de geometrías complejas y, lo que es más importante, muy intuitiva. Así, su definición depende únicamente de la generación de un vector de nodos (que determina el número y modo en que los tramos polinomiales se unirán para dar lugar a la curva) y un conjunto de puntos de control, cuyo polígono interpolador da una idea de la apariencia general de la curva. Una vez definido el vector de nodos, el ajuste de los puntos de control resulta un mecanismo cómodo e intuitivo para refinar el modelo del entorno. Más aún teniendo en cuenta la facilidad con que las incertidumbres asociadas a las medidas brutas del láser pueden ser encapsuladas en estos puntos, mediante la simple formulación matricial del método elegido para su aproximación.

  A pesar de no ser esta la única forma de aproximar conjuntos de datos ruidosos mediante curvas spline, ha sido la elegida por dos motivos: primero, por ser la más simple, y segundo, por proporcionar un método formulable matricialmente capaz de ajustar eficazmente los datos en un único paso. Esto último facilita la propagación de la información estocástica desde la información adquirida por el sensor a su modelado en el mapa.

- El capítulo 4 recoge las principales contribuciones de esta tesis, que se enumeran a continuación. Todas ellas son aportaciones originales, pensadas por y para el objetivo final de facilitar la inclusión de las curvas spline como herramienta de modelado en el marco de trabajo del SLAM-EKF.

    1. Se ha mostrado un mecanismo simple, pero efectivo, para segmentar las medidas brutas adquiridas por un telémetro láser, agrupándolas en conjuntos pertenecientes a un mismo objeto del entorno.
    2. Cada conjunto de medidas adquiridas es aproximado mediante una función B-spline, de manera que su geometría queda definida en el mapa, y representada por los puntos de control almacenados en el vector de estado del sistema. El simple mecanismo de aproximación también facilita la inclusión en la matriz de covarianzas de la información estocástica que relaciona espacialmente todos los elementos del mapa entre sí y con el propio robot.
    3. El mecanismo de asociación propuesto, no solo permite establecer una correspondencia robusta entre las observaciones realizadas por el robot y los objetos contenidos en el mapa, sino que también permite establecer una correspondencia paramétrica entre cada par de elementos asociados que será de utilidad (i) a la hora de actualizar la estimación (permitiendo la aplicación del modelo de observación desarrollado de manera eficiente, al proporcionar excelentes valores iniciales para la iteración de Newton-Raphson), (i) para la eventual extensión de un objeto del mapa.
    4. Se ha propuesto un modelo de observación eficiente y eficaz, que elude las limitaciones que plantea la no observabilidad de los puntos de control, trasladando el problema a la predicción de las medidas individuales que se espera adquirir con el sensor.
    5. El mencionado modelo de observación también permite la deducción natural de los Jacobianos involucrados en el proceso de estimación. Se destaca la simplicidad de las fórmulas obtenidas, que hacen olvidar los temores planteados por la formulación recursiva de las funciones básicas B-spline.
    6. Se muestra cómo todos los resultados anteriores encajan de manera simple y natural en el marco de trabajo del SLAM-EKF, permitiendo la estimación recursiva e incremental del modelo del entorno a medida que es explorado por un robot móvil.
    7. Se proporcionan algoritmos para elongar los objetos contenidos en el mapa. Esto es posible gracias a la adaptación de un algoritmo recursivo que permite la transformación del vector de nodos asociado a un spline, dando cabida a nuevos tramos representativos de nuevas zonas descubiertas. Finalmente, se obtienen de nuevo expresiones matriciales que facilitar la integración de la nueva información estocástica adquirida por el sensor, ampliando así el estado de la manera más consistente posible.

- Visto que una de las mayores limitaciones de la solución EKF al problema del SLAM es su coste computacional —cuadrático con el número de objetos contenidos en el mapa—, el capítulo 5 muestra una manera de paliar este

problema por la simple descomposición en modelos de tamaño limitado, cuya estructura global se actualiza mediante un nuevo filtro extendido de Kalman.

Si bien no es esta una solución original de esta tesis, sí lo es la manera de extraer información del modelo, y establecer relaciones entre los diferentes submapas que, a modo de observaciones virtuales, permiten corregir sus posiciones relativas y obtener una representación global más acorde con la realidad. Más concretamente, el problema se aborda buscando un máximo clique dentro del grafo de correspondencias, aprovechando una implementación de muy reciente aparición basada en la codificación de esta información en cadenas bits. De esta manera se aprovecha el paralelismo entre la estrucutra de datos generada y la manera de operar del hardware de un computador, consiguiendo resultados únicos en su clase desde el punto de vista del tiempo de cómputo necesario.

- Finalmente, el capítulo 6 presenta los resultados que evalúan, demuestran y validan los algoritmos presentados en capítulos anteriores.

  Se comienza con un breve estudio experimental de la influencia del espaciado internodal y el orden de la curva spline a la hora de aproximar datos contaminados por un cierto nivel de ruido. Desde el punto de vista del espaciado internodal, no hay una regla universal que permita establecer su valor. Para entornos de interior habituales un valor de entre 1,0 y 2,0 metros suele ser razonable, sirviendo el último en la mayoría de las ocasiones. En cualquier caso, serán las condiciones geométricas del entorno las que determinen el valor a emplear, en función de la complejidad geométrica del mismo. Desde el punto de vista del orden, los splines cúbicos son un estándar de hecho para representar cualquier tipo de geometría, si bien los splines lineales son suficientes para representar entornos con paredes planas.

  Se prosigue con experimentos con datos simulados que permiten verificar la precisión y consistencia de los algoritmos. Si bien la precisión conseguida es excepcional —más aún teniendo en cuenta que no es posible modelar algunos de los entornos empleados con técnicas geométricas actuales—, desde el punto de vista de la consistencias surgen las mismas limitaciones de cualquier otra implementación EKF. Estas limitaciones son ineludibles y de muy temprana aparición, y no parecen verse atenuadas inflando artificialmente la covarianza del sensor empleada en el filtro.

  Los mapas con datos reales permiten evaluar la aplicabilidad práctica de los métodos propuestos. Cabe resaltar el hecho de que el set de datos adquirido en el Museo “Príncipe Felipe” entraña grandes dificultades para su procesamiento. No sólo por el hecho de la presencia de gran número de objetos dinámicos (personas) en el entorno —como es habitual que suceda en un museo—, sino por la escasa estructuración de este y la clara predominancia de estructuras curvilíneas. Estas condiciones provocan que ningún otro método de modelado existente hasta el momento haya sido capaz de procesar adecuadamente estos datos.

7.2. Futuros Desarrollos

Las nuevas metodologías propuestas en esta tesis abren la puerta a un sin fin de futuras aplicaciones y desarrollos que, desde nuestro punto de vista, resultan muy prometedores. Algunos tienen que ver con la posible mejora en los resultados aquí presentados; se trataría en este caso de aplicar y combinar resultados existentes en la bibliografía, relacionados tanto con la teoría de las curvas spline, como con las soluciones al problema del SLAM o el procesamiento y análisis de datos espaciales. Otros constituyen extensiones de las soluciones propuestas en esta tesis.

A continuación se enumeran algunos de ellos.

- Se plantea la mejora del mecanismo de segmentación de datos propuesto. Seguramente, métodos de *clustering*[1] más sofisticados permitirían determinar los objetos individuales presentes en una observación del sensor con mayor robustez [76, 107, 183, 193], descartando medidas espurias (conocidas en la bibliografía como *outliers*[2]), elementos a tener muy en cuenta especialmente al dar el salto al modelado tridimensional (ver Fig. 7.1).

- Exploración de técnicas alternativas de ajustes de datos. Por una parte, planteando la utilización de formulaciones alternativas de las curvas polinomiales por tramos, como los NURBS [77], los X-splines [30], los Beta-splines [17, 108], las curvas de resolución variable [165]... Por otra, explorando otros métodos de ajuste que, si bien son en general métodos iterativos, presentan otras ventajas desde el punto de vista de la calidad del ajuste, la selección automática del vector de nodos, o la capacidad de representar con una única curva puntos singulares como las esquinas [21, 70, 83, 120, 143, 149, 211].

- Mejoras en método de asociación de datos propuesto en el capítulo 4. En primer lugar aplicando la solución propuesta en el capítulo 5 con el objetivo en este caso de relacionar objetos contenidos en diferentes submapas. Pero también empleando técnicas de identificación de formas que se apoyan directamente en la formulación geométrica proporcionada por las curvas spline [19, 194, 205] y otras basadas en el análisis estadístico o geométrico de las formas [22, 200].

  En [200], por ejemplo, se utiliza el concepto de métrica reflejada definido en [93] para obtener una métrica normalizada invariante ante transformaciones afines, que es utilizada para comparar formas entre sí, y encontrar similitudes.

- Como ya se ha comentado, los métodos propuestos no son incompatibles ni excluyentes de otras técnicas de localización y modelado simultáneos existen-

[1] El *clustering* consiste en la clasificación de objetos en diferentes grupos o, más precisamente, en la partición de un set de datos en subconjuntos (*clusters*), de manera que los datos contenidos en cada uno de ellos compartan algún rasgo en común (generalmente su proximidad, de acuerdo con alguna métrica previamente definida). Puede ser considerado como el problema más importante de aprendizaje no supervisado.

[2] En estadística, un *outlier* es un elemento de un conjunto de datos que es significativamente diferente a los otros datos de la colección, o que parece implicar un patrón que es inconsistente con el grueso de la evidencia de datos. Dicho con otras palabras, se trata de una observación que está numéricamente alejada del resto de los datos.

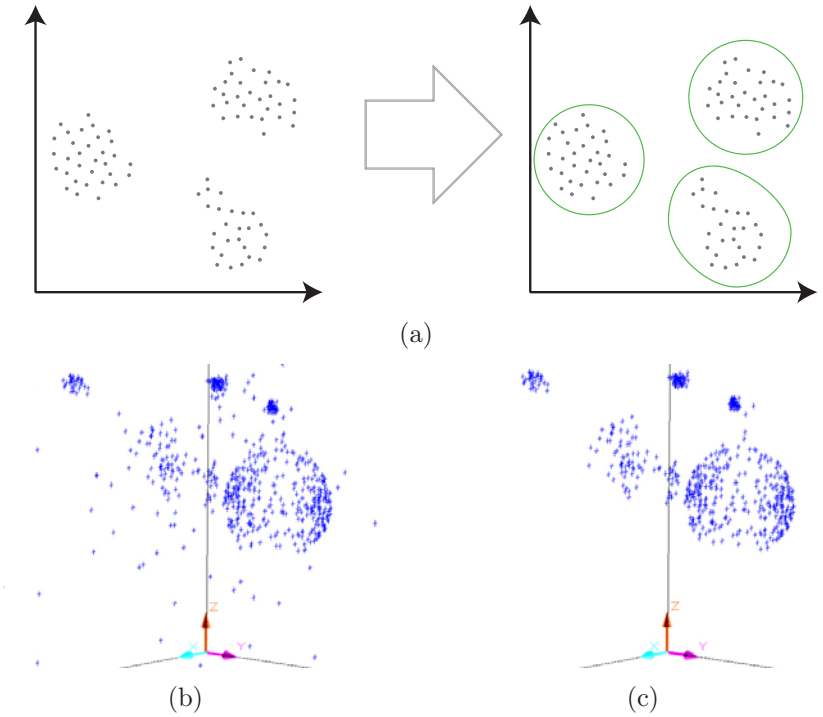


Figura 7.1: Concepto de *clustering* (a). Eliminación de *outliers* en un conjunto de datos tridimensional [183]: (b) datos originales y (c) datos procesados.

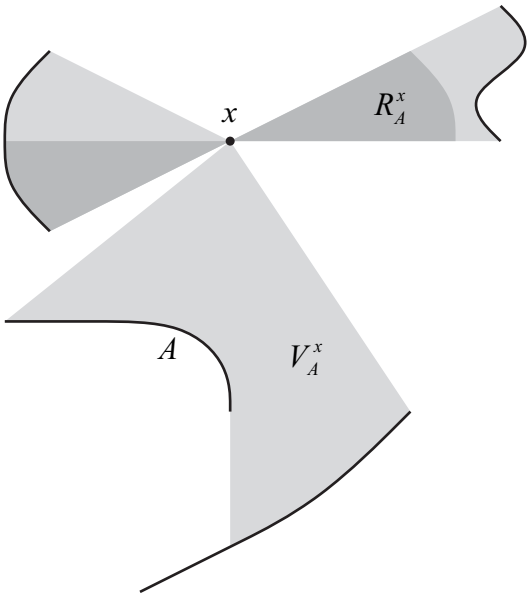
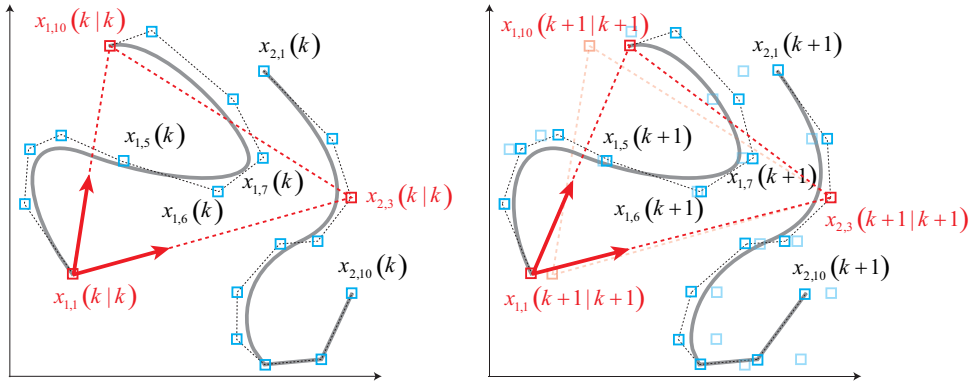


Figura 7.2: Estrella de visibilidad (V_A^x, sombreada en gris claro) y estrella de reflexión (R_A^x, en gris oscuro) para el punto $x \in \Re^2$ dada la configuración de curvas A [200].

tes. En particular, se consideran especialmente interesantes las técnicas híbridas [138] que aprovechan la potencia y eficacia de la solución EKF para cerrar bucles de manera consistentes, superponiendo representaciones más densas como, por ejemplo, un mapa de ocupación de celdillas.

Además estas soluciones reducen el coste computacional del algoritmo. Permiten seleccionar un conjunto de puntos de control cuyas posiciones son las únicas que se actualizan mediante un filtro extendido de Kalman. Estos elementos definen un conjunto de regiones triangulares locales, que proporcionan sistemas de referencia para el resto de objetos contenidos en el mapa. Esta solución se ilustra gráficamente en la figura 7.3 adaptada al caso del modelado mediante curvas B-spline. Los puntos de control representados en azul ven actualizada su posición como consecuencia de la corrección del mallado representado en rojo mediante un EKF (se ha representado un único triángulo, por claridad).



(a) (b)

Figura 7.3: Mapas métricos híbridos adaptados al modelado mediante B-splines. (a) Configuración en el instante k. (b) Configuración en el instante $k+1$, donde se ha mantenido en un color más tenue la configuración en el instante temporal anterior.

- Aplicación de técnicas típicas en la manipulación de curvas spline como la elevación y reducción de grado, y la inserción y eliminación de puntos de control [148]. Se pretende así una gestión más inteligente de la utilización de estas curvas, empleando splines lineales cuando se detecte la presencia de segmentos, y restringiendo el uso de splines cúbicos a geometrías más complicadas. Además se podría emplear un espaciado internodal variable (no fijo, como en los experimentos aquí presentados), que se adecuara mejor a la realidad física representada.

- Explotar la ganancia de información que supone la representación paramétrica del entorno, para extraer conocimiento de más alto nivel de abstracción, trasladando el modelado a un escenario más topológico una vez que el nivel de representación métrico queda resuelto. Se pretende por tanto detectar la presencia de puertas, ventanas, corredores, y caracterizar las diferentes habitaciones que encuentre el robot en su exploración (según su geometría, dimensiones, orientación...) estableciendo relaciones espaciales y lógicas entre todos estos elementos.

- Extensión de los resultados aquí presentados al espacio tridimensional. Esta extensión parece relativamente sencilla, al menos desde el punto de vista conceptual y desde el punto de vista matemático en una primera aproximación.

  Recordemos por ejemplo la definición de curva B-spline de orden κ en el espacio $\Re^n$, presentada en la página 59 del capítulo 3, que se repite aquí para mayor comodidad del lector:

$$\mathbf{s}(t) = \sum_{i=0}^{n} \mathbf{x}_i \beta_{i,\kappa}(t) \tag{7.1}$$

  siendo como sabemos $\mathbf{x}_i \in \Re^n, (i = 0 \dots n)$ sus puntos de control y $\Xi = \{\xi_0, \dots, \xi_{n+\kappa}\} \subset \Re$ el vector de nodos empleado para generar el conjunto de funciones básicas.

  Del mismo modo, una superficie B-spline de órdenes κ y λ en el espacio $\Re^n$ viene definida por una expresión biparamétrica (*i.e.* su evaluación depende de dos parámetros independientes, u y v) que tiene el siguiente aspecto [159]:

$$\mathbf{q}(u,v) = \sum_{i=0}^{m} \sum_{j=0}^{n} \mathbf{x}_{i,j} \beta_{i,\kappa}^{u}(u) \beta_{j,\lambda}^{v}(v) \tag{7.2}$$

  En este caso es necesario definir dos vectores de nodos (uno por cada dirección paramétrica, o coordenada generalizada):

$$\Xi^u = \left\{\xi_0^u, \dots, \xi_{n+\kappa}^u\right\} \subset \Re$$

  y

$$\Xi^v = \left\{\xi_0^v, \dots, \xi_{n+\lambda}^v\right\} \subset \Re$$

  así como un conjunto de puntos $\mathbf{x}_{i,j} \in \Re^n, i = 0, \dots, m, j = 0, \dots, n$ que determinan una malla de control. Las funciones $\beta_{i,\kappa}^{u}(u)$ y $\beta_{j,\lambda}^{v}(v)$ no son otra cosa que las funciones básicas B-spline correspondientes a cada una de las direcciones biparamétricas u y v, respectivamente, y cuya formulación recursiva es de nuevo:

$$\beta_{i,1}^{u}(u) = \begin{cases} 1 & \xi_i^u \leq u < \xi_{i+1}^u \\ 0 & \text{en otro caso} \end{cases} \tag{7.3}$$

$$\beta_{i,\kappa}^{u}(u) = \frac{(u - \xi_i^u)}{\xi_{i+\kappa-1}^u - \xi_i^u} \beta_{i,\kappa-1}^{u}(u) + \frac{\left(\xi_{i+\kappa}^u - u\right)}{\xi_{i+\kappa}^u - \xi_{i+1}^u} \beta_{i+1,\kappa-1}^{u}(u) \tag{7.4}$$

$$\beta_{i,1}^{v}(v) = \begin{cases} 1 & \xi_i^v \leq v < \xi_{i+1}^v \\ 0 & \text{en otro caso} \end{cases} \tag{7.5}$$

$$\beta_{i,\lambda}^{v}(v) = \frac{(v - \xi_i^v)}{\xi_{i+\lambda-1}^v - \xi_i^v} \beta_{i,\lambda-1}^{v}(v) + \frac{\left(\xi_{i+\lambda}^v - v\right)}{\xi_{i+\lambda}^v - \xi_{i+1}^v} \beta_{i+1,\lambda-1}^{v}(v) \tag{7.6}$$

  Como puede verse, el orden de una superficie B-spline puede ser diferente para cada una de las direcciones definidas por los parámetros u y v.

Apéndice

A.1. El Método de Newton-Raphson

El método de Newton-Raphson o, simplemente, método de Newton, es una de las metodologías más conocidas para calcular iterativamente los ceros de una función. Dada una función real de variable real $f(t), t \in [a,b] \subset \Re$, con primera derivada continua ($f \in C^2(R)$), el método de Newton-Raphson intenta buscar la raíz de la siguiente ecuación:

$$f(t) = 0 \tag{A-1}$$

Dicho con otras palabras: se trata de buscar el valor $t^* \in [a,b]$ que verifica $f(t^*) = 0$.

El método se apoya en la suposición de que inicialmente se conoce una buena estimación t_0 del valor de t^*. Así, supongamos que queremos encontrar una raíz de la ecuación (A-1) en las proximidades de $t = t_0$. El método simplemente aproxima la función original en un entorno del punto $[t_0, f(t_0)]$ mediante la recta tangente a la curva, y calcula el cero de esa tangente que, en términos generales, será una mejor aproximación de la raíz buscada.

A partir de este momento se puede repetir iterativamente el proceso, tomando en cada ocasión la mejor aproximación disponible de t^*, e interrumpiéndolo cuando se considere que la estimación está lo "suficientemente cerca" de la raíz real (donde ese "suficientemente cerca" dependerá de la aplicación particular considerada). Esta idea intuitiva que plantea el método aparece representada esquemáticamente en la figura 4.

Así, en cada iteración, la estimación de la raíz se ve actualizada mediante la siguiente fórmula:

$$t_{n+1} = t_n - \frac{f(t_n)}{f'(t_n)} \tag{A-2}$$

donde f' denota la derivada de f respecto del parámetro t:

$$f'(t) = \frac{df}{dt}(t) \tag{A-3}$$

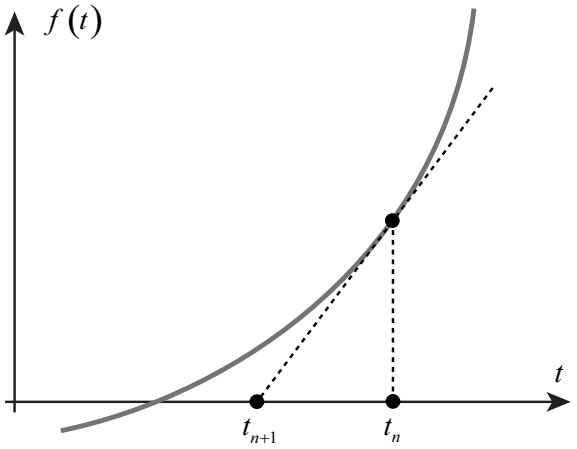


Figura 4: Ejemplificación gráfica de una iteración del método de Newton-Raphson para encontrar una raíz de la ecuación $f(t) = 0$ en las proximidades del valor del parámetro $t = t_n$.

La anterior ecuación (A-2) puede ser fácilmente obtenida haciendo uso del desarrollo en serie de Taylor de la función $f(t)$ en un entorno del punto t_n:

$$f(t + \Delta t) = f(t_n) + f'(t_n)\,\Delta t + f''(t_n)\,\frac{(\Delta t)^2}{2} + \ldots \tag{A-4}$$

y manteniendo únicamente los primeros términos (lo cual es equivalente a aproximar la función por la tangente en el punto t_n):

$$f(t + \Delta t) \approx f(t_n) + f'(t_n)\,\Delta t \tag{A-5}$$

De esta manera, la expresión (A-5) puede ser empleada para calcular el incremento aproximado que hay que aplicar sobre la estimación del cero, para obtener una aproximación más correcta al cero real. Así, haciendo $f(t + \Delta t) = 0$ en la anterior ecuación y despejando Δt podemos obtener:

$$\Delta t_n = -\frac{f(t_n)}{f'(t_n)} \tag{A-6}$$

de manera que podemos calcular la nueva estimación, que presumiblemente será más precisa:

$$t_{n+1} = t_n + \Delta t = t_n - \frac{f(t_n)}{f'(t_n)} \tag{A-7}$$

El método de Newton-Raphson ofrece una rápida convergencia cuando se dispone de una buena estimación inicial de la posición del cero buscado, $t_0 \approx t^*$. Desafortunadamente, su comportamiento puede ser altamente impredecible e inestable en las proximidades de una asíntota horizontal, o de un extremo local de la función.

A.2. El Filtro Extendido de Kalman

A.2.1. Una Herramienta Útil

Para cualquier sistema en evolución que queramos imaginar (ya sea un proceso químico, la evolución del mercado bursátil, un sólido en caída libre, o un robot móvil) el ser humano, intentará buscar modelos que representen adecuadamente determinados aspectos de interés del mismo, estableciendo relaciones entre las variables involucradas.

Sin embargo, cualquier aproximación determinista al modelado e intento de control de un sistema real presentará siempre carencias y limitaciones de base. Esto es debido a tres razones fundamentales [125]:

- **Ningún modelo es perfecto**.
- La evolución de los sistemas viene en general afectada por numerosas **perturbaciones** que, en la mayoría de los casos, son difíciles o imposibles de controlar y modelar.
- Los sensores proporcionan siempre **información parcial e inexacta** acerca de los sistemas observados.

Así pues parece claro que cualquier técnica que permita tener en cuenta las anteriores limitaciones contribuirá a mejorar los resultados de la observación y el control de sistemas reales.

El filtro de Kalman es un filtro recursivo lineal de mínimos cuadrados capaz de estimar el estado de un proceso, teniendo en cuenta el modelado estadístico del ruido que afecta a dicho proceso, y al sistema de medición empleado. Su aplicación a sistemas no lineales se consigue previa linealización de las ecuaciones, dando lugar al filtro extendido de Kalman (*"Extended Kalman Filter" – EKF*). En los siguientes apartados se realizará un breve repaso de esta teoría y de las matemáticas involucradas. Para una descripción más detallada del filtro y de sus orígenes se puede consultar [125, 182, 206].

A.2.2. El Filtro de Kalman Discreto

El filtro de Kalman intenta estimar el estado $x \in \Re^n$ de un sistema gobernado por la siguiente ecuación lineal estocástica en diferencias:

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \tag{A-8}$$

teniendo como información un vector de medidas $z \in \Re^m$ cuyo valor depende del estado del sistema:

$$z_k = Hx_k + v_k \tag{A-9}$$

En las anteriores ecuaciones A-8 y A-9, w_k y v_k son *variables aleatorias* que representan el ruido inherente al proceso y a la medida, respectivamente. Se asume que se trata de ruido blanco, independiente uno de otro y con una distribución de probabilidad normal:

$$p\left(w\right) \sim N\left(0, Q\right) \tag{A-10}$$
$$p\left(v\right) \sim N\left(0, R\right) \tag{A-11}$$

A pesar de que en la práctica las matrices de varianzas y covarianzas del proceso y de la medida (Q y R, respectivamente) pueden variar en cada paso temporal, se asume aquí que son constantes.

El filtro realiza la estimación del estado en dos pasos: primeramente se realiza una predicción del estado (utilizando el modelo disponible del proceso), para luego corregir dicha predicción en función de las medidas (ruidosas) obtenidas, y utilizando el modelado de las incertidumbres, dado por las matrices Q y R. Así, las ecuaciones del filtro de Kalman se pueden agrupar en dos etapas: *etapa de actualización temporal* o de *predicción*, y etapa de *actualización de la medida* o de *corrección*.

1. **Etapa de predicción**: Se encargan de realizar una predicción del estado y de la covarianza del error de estimación en el instante temporal k a partir de la información disponible en el instante $k-1$ (o de las suposiciones iniciales, si se trata de la primera muestra). Primero se realiza la estimación *a priori* de el estado x_k^-:

   $$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1} \tag{A-12}$$

   y la estimación *a priori* de la covarianza del error de estimación, P_k^-:

   $$P_k^- = AP_{k-1}A^T + Q \tag{A-13}$$

   En la ecuación A-12 intervienen las matrices A y B de la ecuación A-8 y la última estimación disponible del estado, x_{k-1}, mientras que en la ecuación A-13 entra en juego la matriz de varianzas y covarianzas del proceso, Q (ecuación A-10), y la matriz A que relaciona linealmente el estado en el instante $k-1$ con el estado en el instante posterior.

2. **Etapa de corrección**: Aquí se utiliza la realimentación que proporcionan los sensores, teniendo en cuenta la incertidumbre de la medida que éstos facilitan. Perimero se calcula la *ganancia de Kalman*, K_k:

$$K_k = P_k^- H^T \left(H P_k^- H^T + R \right)^{-1} \tag{A-14}$$

Esta ganancia es utilizada para corregir la estimación a priori del estado, $\hat{x}_k^-$, utilizando las medidas sensoriales:

$$\hat{x}_k = \hat{x}_k^- + K_k \left(z_k - H\hat{x}_k^- \right) \tag{A-15}$$

En la anterior ecuación, al término $\left(z_k - H\hat{x}_k^- \right)$ se le denomina *innovación de la medida*, o *residuo*. Finalmente, se puede calcular la covarianza del error de estimación cometido mediante la siguiente ecuación:

$$P_k = \left(I - K_k H \right) P_k^- \tag{A-16}$$

Precisamente lo que hace el filtro de Kalman es escoger la ganancia K_k de tal manera que se minimize estadísticamente la covarianza del error a posteriori, P_k. Para un análisis y explicación más detallados de la justificación de las anteriores ecuaciones A-12 a A-16 se puede consultar [125], [36].

A.2.3. Ejemplo: Estimación de una Constante

A continuación se muestra un sencillo ejemplo de aplicación del filtro de Kalman discreto. Supongamos que estamos estudiando un proceso o sistema en el cual deseamos investigar el valor de una magnitud C que sabemos constante. Así pues, el modelo de nuestro sistema tendría la siguiente apariencia, si adaptamos la ecuación A-8:

$$x_k = x_{k-1} + w_k$$

Por otra parte, disponemos de un sensor rudimentario para investigar el valor de la anterior constante. Esto nos da la siguiente ecuación de medida:

$$z_k = x_k + v_k$$

Las ecuaciones del algoritmo del filtro de Kalman discreto quedan como sigue:

1. Etapa de predicción:

$$\begin{aligned} \hat{x}_k^- &= \hat{x}_{k-1} \\ P_k^- &= P_{k-1} + Q \end{aligned} \tag{A-17}$$

2. Etapa de corrección:

$$K_k = P_k^- \left(P_k^- + R\right)^{-1}$$
$$\hat{x}_k = \hat{x}_k^- + K_k \left(z_k - \hat{x}_k^-\right)$$
$$P_k = \left(1 - K_k\right) P_k^-$$

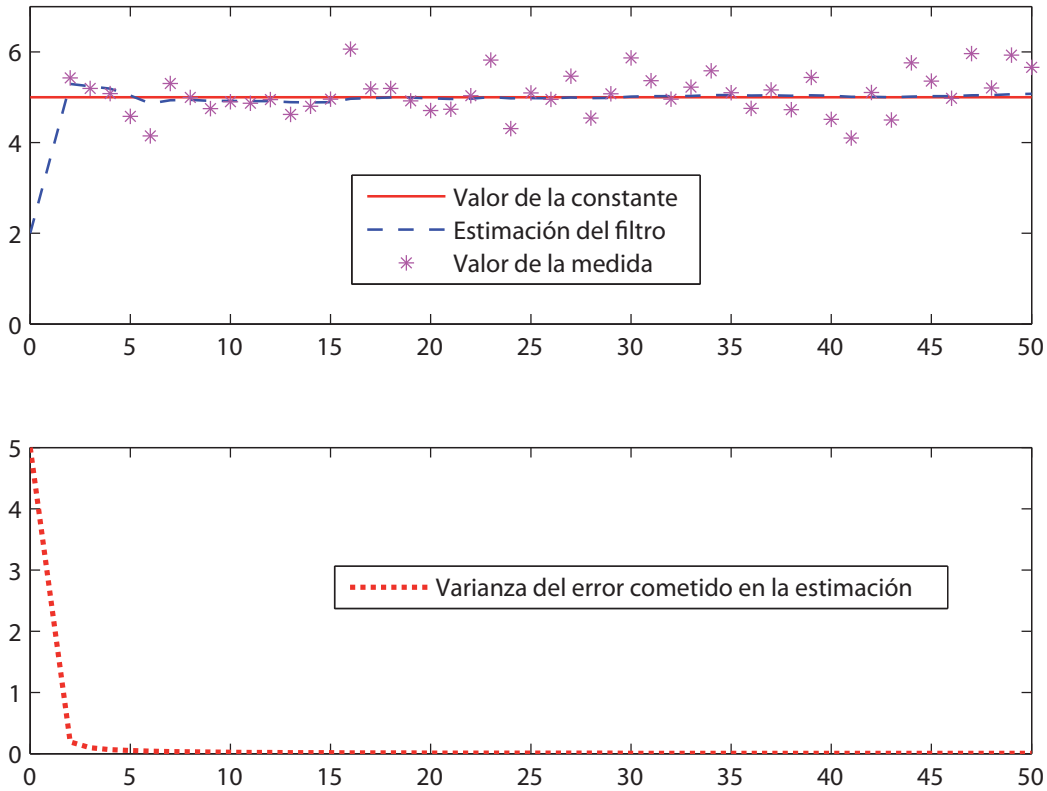


Figura 5: Estimación de la constante C = 5.0 mediante un filtro de Kalman discreto (I). En la imagen superior se observa el valor real de la constante (en línea continua) junto con la estimación suministrada por el filtro a lo largo de 50 muestras (en línea discontinua) y los valores adquiridos por el sensor en cada instante (marcados por asteriscos). El sensor está afectado en este caso por un ruido blanco de media 0 y varianza 0,2 En la parte inferior de la figura se observa cómo desciende rápidamente el valor de la varianza de la estimación, desde la suposición inicial ($P_0^- = 5$) hasta un valor muy cercano a cero. Nótese la calidad de la estimación, a pesar de la inexactitud del sensor empleado en las mediciones.

En la figura 5 aparecen representados los resultados obtenidos al aplicar el anterior filtro en la estimación del valor de una constante $C = 5$, con un ruido en el proceso $p\left(w\right) \sim N\left(0, 3 \cdot 10^{-6}\right)$, utilizando un sensor afectado por un ruido blanco, $p\left(v\right) \sim N\left(0, 0{,}2\right)$. En este caso se ha utilizado como suposición inicial del valor de la constante $x_0^- = 2{,}0$ con una varianza de la estimación $P_0^- = 5$. Este valor inicial de la varianza (covarianza, para el caso multivariable) del error cometido en la estimación también ha de ser suministrado inicialmente al filtro, y ha de ser razonable y acorde con nuestra suposición de medida inicial, y la idea que tengamos del valor real de la magnitud medida. En este caso se ha tomado un valor grande para

esta varianza, asumiendo que consideramos que nuestra suposición inicial está muy alejada del valor real. En la figura 6 se muestran los resultados de repetir el experimento en idénticas condiciones, pero utilizando una valor inicial para la varianza del error cometido en la estimación mucho menor ($P_0^- = 0{,}05$). Se puede observar cómo la convergencia del filtro en este caso es mucho más lenta, al haber supuesto que nuestra suposición inicial era mucho más acertada de lo que a posteriori se revela.

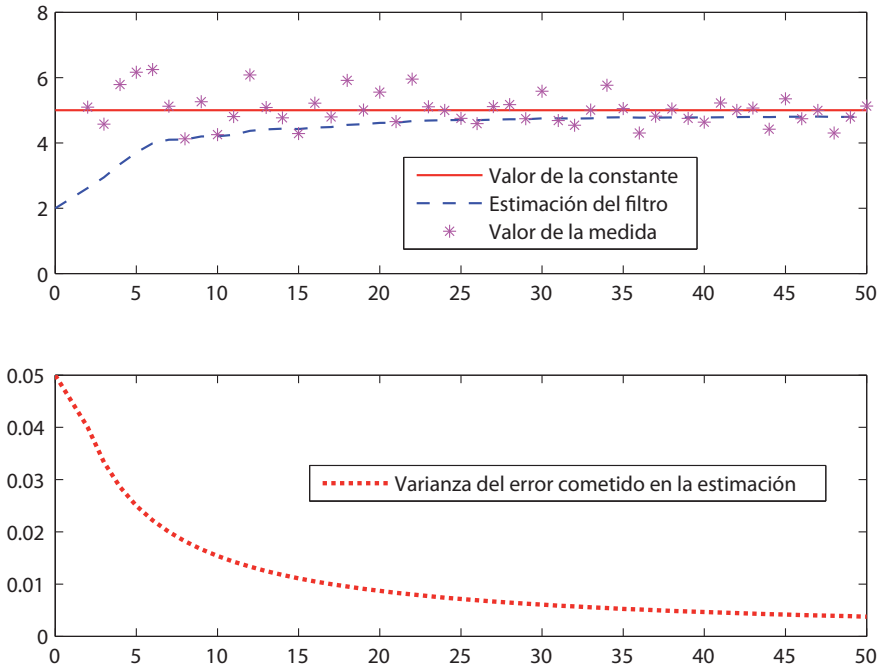


Figura 6: Estimación de la constante C = 5.0 mediante un filtro de Kalman discreto (II). Se han utilizado los mismos parámetros que en el caso representado en la figura 5, pero en este caso el valor inicial de la varianza del error de estimación es $P_0^- = 0{,}05$. El filtro funciona, pero su convergencia es mucho más lenta

A.2.4. El Filtro Extendido de Kalman

Las anteriores ecuaciones son válidas, como se ha dicho, cuando el sistema viene descrito por una ecuación lineal estocástica en diferencias. Pero si la ecuación que describe el proceso y/o la que relaciona la medida con el estado es no lineal, es preciso modificar las anteriores ecuaciones; el filtro extendido de Kalman consigue estimar el estado de este tipo de sistemas linealizando alrededor de las mejores estimaciones del estado y de la covarianza en cada instante.

Así pues se trata de estimar el estado $x \in \Re^n$ de un sistema que viene ahora gobernado por la siguiente ecuación no lineal estocástica en diferencias:

$$x_k = f\left(x_{k-1}, u_{k-1}, w_{k-1}\right) \tag{A-18}$$

disponiendo de un vector de medidas $z \in \Re^m$, cuya relación con el estado en cada instante es también no lineal:

$$z_k = h\left(x_k, v_k\right) \tag{A-19}$$

En este caso se asumen las mismas hipótesis para los ruidos del proceso y de la medida que en el caso no lineal. Bajo estas condiciones, linealizando las anteriores ecuaciones en las inmediaciones de la estimación del estado, y siguiendo unos pasos similares a los aplicaciones en el algoritmo del filtro de Kalman discreto, se puede llegar a los siguientes pasos para el filtro extendido de Kalman:

1. **Etapa de predicción**: Aquí se realiza, como en el caso lineal, la predicción del estado en el instante k, $\hat{x}_k^-$, y de la covarianza del error de estimación, P_k^-:

$$\hat{x}_k^- = f\left(\hat{x}_{k-1}, u_{k-1}, 0\right) \tag{A-20}$$
$$P_k^- = A_k P_{k-1} A_k^T + W_k Q W_k^T \tag{A-21}$$

2. **Etapa de corrección**: Aquí se realiza la corrección de las anteriores estimaciones utilizando la realimentación sensorial, habida cuenta de la incertidumbre asociada. Primero se calcula la ganancia de Kalman del filtro extendido:

$$K_k = P_k^- H_k^T \left(H_k P_k^- H_k^T + V_k R V_k^T\right)^{-1} \tag{A-22}$$

que se utiliza para corregir la estimación del estado

$$\hat{x}_k = \hat{x}_k^- + K_k\left(z_k - h\left(\hat{x}_k^-, 0\right)\right) \tag{A-23}$$

y para corregir la covarianza de la estimación, que es precisamente lo que este filtro intenta minimizar:

$$P_k = \left(I - K_k H_k\right) P_k^- \tag{A-24}$$

En las anteriores ecuaciones hay cuatro matrices que aparecen como consecuencia de las simplificaciones realizadas en forma de linealizaciones: A_k es la matriz jacobiana de f respecto de x, evaluada en cada instante en las inmediaciones de la mejor estimación disponible del estado, y sin tener en cuenta el ruido:

$$A_k = \frac{\partial f}{\partial x}\left(\hat{x}_{k-1}, u_{k-1}, 0\right) \tag{A-25}$$

W_k es la matriz jacobiana de f respecto de w (ruido del proceso), evaluada en el mismo punto que A_k:

$$W_k = \frac{\partial f}{\partial w}\left(\hat{x}_{k-1}, u_{k-1}, 0\right) \tag{A-26}$$

H_k es la matriz jacobiana de h respecto de x, evaluada en este caso en las inmediaciones de la estimación a priori del estado, obtenida de la ecuación A-20, y suponiendo nulo el ruido de medida:

$$H_k = \frac{\partial h}{\partial x}\left(\hat{x}_k^-, 0\right) \tag{A-27}$$

y V_k es la matriz jacobiana de h respecto de v (ruido que afecta a las medidas), evaluada en el mismo punto que H_k:

$$V_k = \frac{\partial h}{\partial v}\left(\hat{x}_k^-, 0\right) \tag{A-28}$$

Mediante las anteriores ecuaciones, el filtro extendido de Kalman es capaz de incorporar toda la información obtenida de las medidas (sin importar su precisión) y, junto con el conocimiento que se tiene del sistema y los dispositivos de medida, la descripción estadística de las incertidumbres existentes y cualquier conocimiento que se tenga sobre el estado inicial, estimar de manera efectiva el estado del sistema minimizando estadísticamente el error cometido.

A.2.5. Ejemplo: Estimación de la Trayectoria de un Robot

En esta sección se muestra un ejemplo de las ventajas de la utilización del filtro extendido de Kalman a la hora de localizar un robot móvil. En este caso se parte de un entorno en el que hay dispuestas cuatro balizas de posiciones conocidas, recogidas en el cuadro 1.

Id. marca	Posición x	Posición y
1	5.2	5.3
2	1.3	2.4
3	2.3	6.3
4	5.2	1.7

Cuadro 1: Posición de las marcas en el terreno

El robot describirá durante el experimento una trayectoria horizontal que pretendemos estimar utilizando un filtro extendido de Kalman. Para determinar su posición, el robot cuenta con sensores odométricos, que proporcionan información sobre el número de vueltas que dan las ruedas, o incremento de distancia recorrida Δd, y la variación en la orientación de las mismas, $\Delta\theta$. Las medidas que facilitan estos sensores vienen afectadas por ruido blanco independiente, con distribución de probabilidad normal conocida (bien porque ha sido proporcionada por el fabricante, bien porque ha sido determinada experimentalmente). A partir de estos datos, y teniendo en cuenta la figura 7 es posible obtener las ecuaciones que permiten estimar la posición y orientación del robot a partir de las medidas de estos sensores. Dicho con otras palabras, el gráfico que aparece en la figura 7 es precisamente el modelo

de nuestro sistema. En la figura se aproxima la trayectoria mediante un arco de circunferencia de radio R_k, recorriendo el robot una distancia Δd que le hace pasar del punto (x_k, y_k) al punto (x_{k+1}, y_{k+1}) variando su orientación en $\Delta\theta$ radianes.

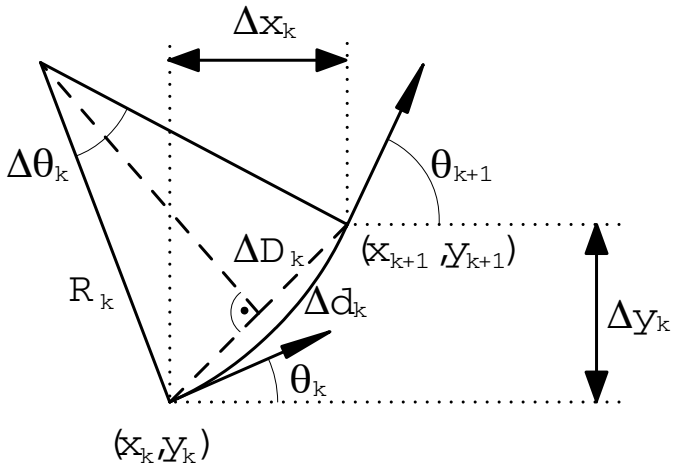


Figura 7: Estimación de posición utilizando la odometría

$$
\begin{aligned}
R_k &= \frac{\Delta d_k}{\Delta\theta_k} \\
\Delta D_k &= 2R_k \sin\left(\frac{\Delta\theta_k}{2}\right) = 2\frac{\Delta d_k}{\Delta\theta_k}\sin\left(\frac{\Delta\theta_k}{2}\right) \\
\Delta x_k &= \Delta D_k \cos\left(\theta_k + \frac{\Delta\theta}{2}\right) \\
\Delta y_k &= \Delta D_k \sin\left(\theta_k + \frac{\Delta\theta}{2}\right)
\end{aligned}
$$

Así pues:

$$x_{k+1} = x_k + 2\frac{\Delta d_k}{\Delta\theta_k}\sin\left(\frac{\Delta\theta_k}{2}\right)\cos\left(\theta_k + \frac{\Delta\theta_k}{2}\right) \tag{A-29}$$

$$y_{k+1} = y_k + 2\frac{\Delta d_k}{\Delta\theta_k}\sin\left(\frac{\Delta\theta_k}{2}\right)\sin\left(\theta_k + \frac{\Delta\theta_k}{2}\right) \tag{A-30}$$

$$\theta_{k+1} = \theta_k + \Delta\theta_k \tag{A-31}$$

Teniendo en cuenta que

$$\left(\frac{\Delta\theta}{2} \longrightarrow 0\right) \Longrightarrow \left(\sin\left(\frac{\Delta\theta}{2}\right) \longrightarrow \frac{\Delta\theta}{2}\right)$$

y que las medidas de la odometría vienen afectadas de ruido, se puede escribir:

$$x_{k+1} \approx x_k + (\Delta d_k + w_{d,k}) \cos\left(\theta_k + \frac{\Delta\theta_k + w_{\theta,k}}{2}\right) \tag{A-32}$$

$$y_{k+1} \approx y_k + (\Delta d_k + w_{d,k}) \sin\left(\theta_k + \frac{\Delta\theta_k + w_{\theta,k}}{2}\right) \tag{A-33}$$

$$\theta_{k+1} = \theta_k + \Delta\theta_k + w_{\theta,k} \tag{A-34}$$

siendo w_θ y w_d, respectivamente, los ruidos asociados a las medidas de rotación angular y desplazamiento lineal, cuya distribución se supone normal de media cero:

$$w_\theta \sim N\left(0, \sigma_\theta^2\right) \tag{A-35}$$

$$w_d \sim N\left(0, \sigma_d^2\right) \tag{A-36}$$

Las anteriores ecuaciones A-32, A-33 A-34 representan el modelo no lineal del sistema, equivalente a la ecuacicón A-18. Nos servirán para realizar las correspondientes predicciones del estado del sistema y a partir de ella se calculan las matrices A_{k+1} y W_{k+1}.

$$A_{k+1} = \begin{bmatrix} 1 & 0 & -\Delta d_k \sin\left(\hat{\theta}_k + \frac{\Delta\theta_k}{2}\right) \\ 0 & 1 & \Delta d_k \cos\left(\hat{\theta}_k + \frac{\Delta\theta_k}{2}\right) \\ 0 & 0 & 1 \end{bmatrix} \tag{A-37}$$

$$W_{k+1} = \begin{bmatrix} \cos\left(\hat{\theta}_k + \frac{\Delta\theta_k}{2}\right) & \frac{-\Delta d_k}{2} \sin\left(\hat{\theta}_k + \frac{\Delta\theta_k}{2}\right) \\ \sin\left(\hat{\theta}_k + \frac{\Delta\theta_k}{2}\right) & \frac{-\Delta d_k}{2} \cos\left(\hat{\theta}_k + \frac{\Delta\theta_k}{2}\right) \\ 0 & 1 \end{bmatrix} \tag{A-38}$$

Por otra parte, el robot dispone de un sensor láser que le permite determinar la posición angular relativa de las balizas del entorno respecto de un sistema de referencia ligado a él. Dicho sensor láser también tiene asociada una incertidumbre a sus mediciones. A partir de la figura 8 es fácil obtener las ecuaciones que relacionan el estado del robot (posición y orientación) con la medida que obtendremos del goniómetro del láser en cada instante, k:

$$\tau_{i,k} = \arctan\left(\frac{y_i - y_k}{x_i - x_k}\right) - \theta_k + v_{i,k} \tag{A-39}$$

siendo (x_i, y_i) las coordenadas cartesianas de la marca i-ésima (conocidas y constantes), (x_k, y_k, θ_k) la posición y orientación del robot en el instante k-ésimo (variables

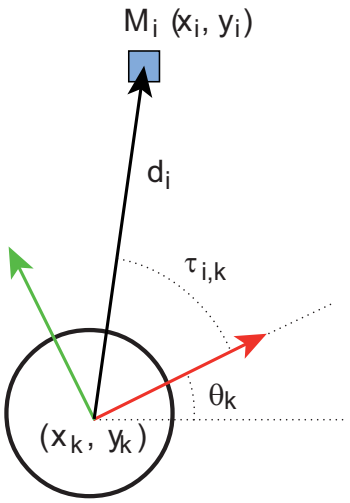


Figura 8: Predicción de la medida del láser

que pretendemos estimar) y $v_{i,k}$ el ruido que afecta a la medida del goniómetro sobre la marca i-ésima en el instante k-ésimo y que suponemos de distribución de probabilidad conocida:

$$v_\tau \sim N\left(0, \sigma_\tau^2\right) \tag{A-40}$$

A partir de la ecuación A-39 es posible realizar predicciones de la medida del láser, que serán utilizadas para calcular los residuos, y también calcular el valor de las matrices H_k y V_k utilizadas en la etapa de corrección del filtro:

$$H_k = \begin{bmatrix} \frac{y_1 - \hat{y}_k^-}{d_{1,k}^2} & -\frac{x_1 - \hat{x}_k^-}{\hat{d}_{1,k}^2} & -1 \\ \vdots & \vdots & \vdots \\ \frac{y_m - \hat{y}_k^-}{d_{m,k}^2} & -\frac{x_m - \hat{x}_k^-}{\hat{d}_{m,k}^2} & -1 \end{bmatrix} \tag{A-41}$$

$$V_k = I_m \tag{A-42}$$

En las anteriores ecuaciones x_j e y_j son las posiciones de la marca j-ésima ($j = 1 \ldots m$), $\hat{x}_k^-$ e $\hat{y}_k^-$ son las estimaciones a priori de la posición del robot en el instante k-ésimo, y $\hat{d}_{j,k}^2$ es la estimación a priori de la distancia del robot a la marca j-ésima en el instante k-ésimo ($j = 1 \ldots m$), que tiene por expresión:

$$\hat{d}_{j,k}^2 = \left(x_i - \hat{x}_k^-\right)^2 + \left(y_i - \hat{y}_k^-\right)^2$$

En las figuras 9 y 10 se pueden observar los resultados de aplicar el filtro en un entorno con cuatro balizas cuyas posiciones se presentan en el cuadro 1. Se ha

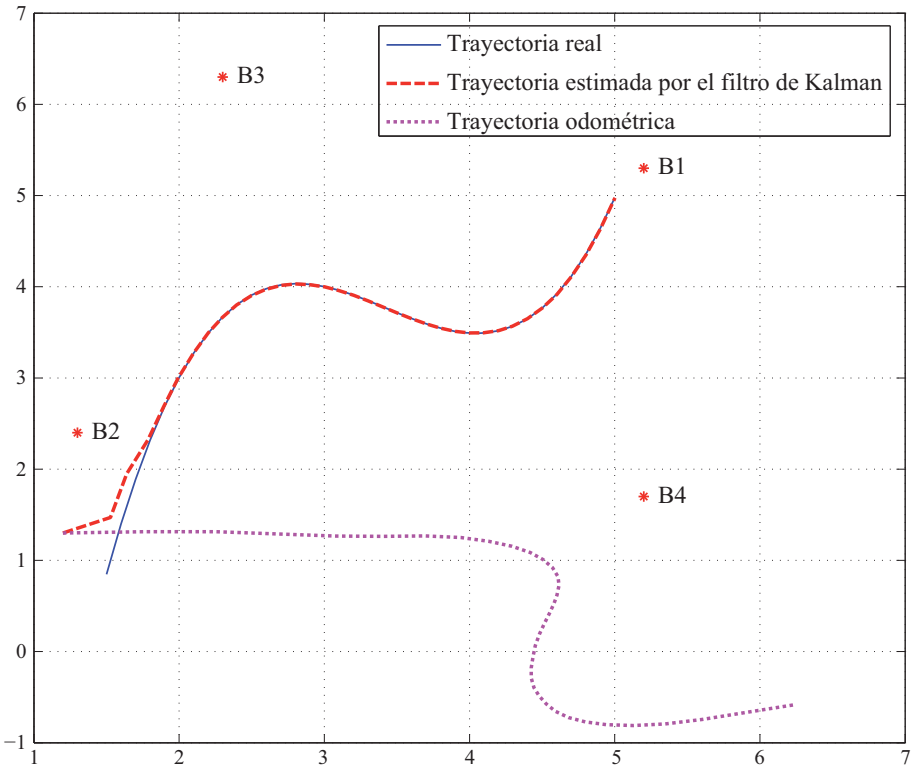


Figura 9: Trayectoria del robot. Estimación Filtro de Kalman con balizas

simulado un ruido en el encoder del *"drive"* con una distribución normal de media 0 y desviación típica de 5 mm. En el caso del *"steer"* se ha añadido ruido de distribución normal, media 0 y desviación típica de 0,06 radianes. Para el goniómetro láser se ha simulado ruido de media 0 y desviación típica de 0,0007 radianes. La estimaciones iniciales han sido de 1.2 metros para la coordenada x del robot, 1.3 metros para la coordenada y, y una orientación de 0 radianes. Se observa cómo la estimación de posición y orientación converge rápidamente con la real en el caso de utilización del filtro de Kalman, mientras que teniendo únicamente en cuenta la información suministrada por los encoders el error crece de manera incontrolable.

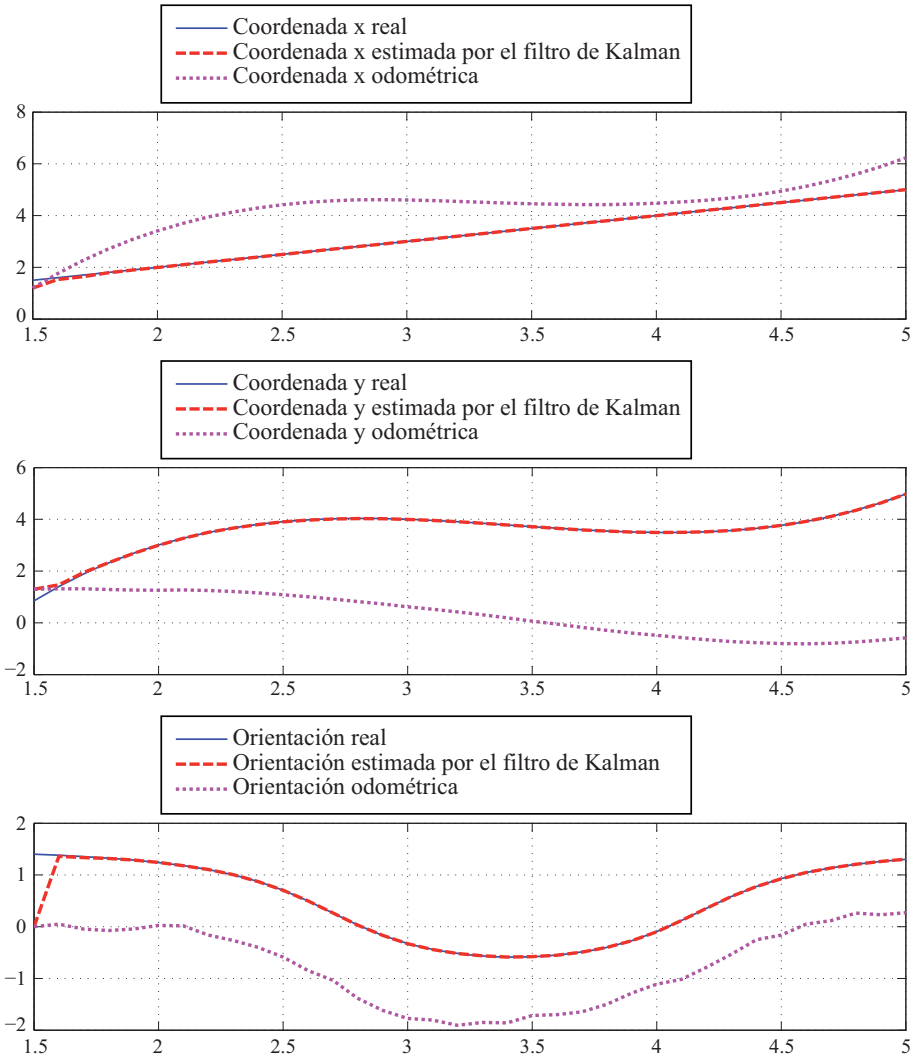


Figura 10: Coordenadas del robot. Estimación Filtro de Kalman con balizas

Bibliografía

[1] O. Abert, M. Geimer, , and S. Müller. Direct and fast ray tracing of NURBS surfaces. In *IEEE Symposium on Interactive Ray Tracing 2006*, pages 161–168, 2006.

[2] J. H. Ahlberg, E. N. Nilson, and J. L. Walsh. *The Theory of Splines and their Applications*. Academic Press, New York, USA, 1967.

[3] J. Alberich. Las flores de Bézier. Elasticidad e inestabilidad en el grafismo digital interactivo. *Artnodes, Revista de Arte, ciencia y tecnología*, 4, 2005.

[4] A. Appel. Modeling in three-dimensions. *IBM Systems Journal*, 7(3&4):310–321, 1968.

[5] A. Appel. On calculating the illusion of reality. In A. J. H. Morrell, editor, *IFIP Congress 1968*, volume 2, pages 945–950, Edinburgh, UK, August 1968.

[6] A. Appel. Some techniques for shading machine renderings of solids. In *AFIPS 1968 Spring Joint Computer Conference Proceedings*, volume 32, pages 37–45, 1968.

[7] A. Atieg and G. A. Watson. A class of methods for fitting a curve or surface to data by minimizing the sum of squares of orthogonal distances. *J. Comput. Appl. Math.*, 158(2):277–296, 2003.

[8] T. Bailey. *Mobile Robot Localisation and Mapping in Extensive Outdoor Environments*. PhD thesis, University of Sydney, Australian Centre of Field Robotics, 2002.

[9] T. Bailey and E. Nebot. Localisation in large-scale environments. *Robotics and Autonomous Systems*, 37(4):261–281, 2001.

[10] T. Bailey, E. Nebot, J. Rosenblatt, and H. Durrant-Whyte. Data association for mobile robot navigation: a graph theoretic approach. In *IEEE Int. Conf. Robotics and Automation*, volume 3, pages 2512–2517, San Francisco, CA, USA, April 2000.

[11] T. Bailey, E. M. Nebot, J. K. Rosenblatt, and H. F. Durrant-Whyte. Data association for mobile robot navigation: A graph theoretic approach. In *in Proc. IEEE Int. Conf. Robotics and Automation*, pages 2512–2517, 2000.

[12] T. Bailey, J. Nieto, J. Guivant, M. Stevens, and E. Nebot. Consistency of the EKF-SLAM algorithm. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.

[13] Y. Bar-Shalom and T. E. Fortmann. *Tracking and data association*. Academic Press Professional, Inc., San Diego, CA, USA, 1988.

[14] R. Barnhill, R. Dube, and F. Little. Properties of Shepard's surfaces. *Rocky Mountain Journal of Mathematics*, 13(2):365–382, 1983.

[15] R. E. Barnhill and R. F. Riesenfeld. *Computer Aided Geometric Design*. Academic Press, 1974.

[16] H. Barrow and R. Burstall. Subgraph isomorphism, matching relational structures and maximal cliques. *Information Processing Letters*, 4(4):83–84, 1976.

[17] B. A. Barsky. *The Beta-Spline: a Local Representation Based on Shape Parameters and Fundamental Geometric Measures*. PhD thesis, Univ. of Utah, december 1981.

[18] R. H. Bartels, J. C. Beatty, and B. A. Barsky. *An Introduction to Splines for Use in Computer Graphics and Geometric Modeling*. Morgan Kaufmann Publishers, Inc., 1987.

[19] R. H. Bartels, J. C. Beatty, K. S. Booth, E. G. Bosch, and P. Jolicoeur. Experimental comparison of splines using the shape-matching paradigm. *ACM Transactions on Graphics (TOG)*, 12(3):179–208, July 1993.

[20] S. Bedi and G. Vickers. Surface lofting and smoothing with skeletal-lines. *Computer Aided Geometric Design*, 6(2):87–96, 1989.

[21] G. Beliakov. Least squares splines with free knots: global optimization approach. *Applied Mathematics and Computation*, 149:783–798, 2004.

[22] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(4):509–522, April 2002.

[23] M. Betke and L. Gurvits. Mobile robot localization using landmarks. *IEEE Transactions on Robotics and Automation*, 13(2):251–263, 1997.

[24] P. Bézier. Définition Numérique des Courbes et Surfaces I. *Automatisme*, XI:625–632, 1966.

[25] P. Bézier. Définition Numérique des Courbes et Surfaces II. *Automatisme*, XII:17–21, 1967.

[26] P. Bézier. Procédé de définition numérique des courbes et surfaces non mathématiques. *Automatisme*, XIII(5):189–196, 1968.

[27] P. Bézier. Mathematical and practical possibilities of UNISURF. In R. Barnhill and R. Riesenfeld, editors, *Computer Aided Geometric Design*, pages 127–152. Academic Press, 1974.

[28] P. Bézier. *Essay de définition numérique des courbes et des surfaces expérimentales*. PhD thesis, University of Paris VI, 1977.

[29] P. Bézier. *The Mathematical Basis of the UNISURF CAD System*. Butterworth-Heinemann, Newton, MA, USA, 1986.

[30] C. Blanc and C. Schlick. X-splines: A spline model designed for the end-user. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 377–386, New York, NY, USA, 1995. ACM Press.

[31] W. Boehm. Cubic B-spline curves and surfaces in computer-aided geometric design. *Computing*, 19:29–34, 1977.

[32] W. Boehm. Inserting new knots into B-spline curves. *Computer-Aided Design*, 12(4):199–201, July 1980.

[33] M. Bolduc, E. Bourque, G. Dudek, N. Roy, and R. Sim. Autonomous exploration: An integrated systems approach. In *In AAAI/IAAI*, pages 779–780, 1997.

[34] M. Bosse, P. M. Newman, J. J. Leonard, M. Soika, W. Feiten, and S. J. Teller. An Atlas framework for scalable mapping. In *ICRA*, pages 1899–1906, 2003.

[35] M. C. Bosse, P. M. Newman, J. J. Leonard, and S. Teller. Simultaneous localization and map building in large-scale cyclic environments using the Atlas framework. *The International Journal of Robotics Research*, 23(12):1113–1139, 2004.

[36] R. G. Brown and P. Y. C. Hwang. *Introduction to Random Signals and Applied Kalman Filtering with Matlab Exercises and Solutions*. Wiley, 3rd edition, November 1996.

[37] J. Buhmann, W. Burgard, A. B. Cremers, D. Fox, T. Hofmann, F. E. Schneider, J. Strikos, and S. Thrun. The mobile robot RHINO. *AI Magazine*, 16(2):31–38, 1995.

[38] C. Calhoun, T. Stahovich, T. Kurtoglu, and L. Kara. Recognizing multistroke symbols. In *AAAI Spring Symposium on Sketch Understanding, AAAI Technical Report SS-02-08*, pages 15–23, 2002.

[39] H. Cardot. Spatially adaptive splines for statistical linear inverse problems. *Journal of Multivariate Analysis*, 81(1):100–119, 2002.

[40] R. Carraghan and P. M. Pardalos. An exact algorithm for the maximum clique problem. *Operations Research Letters*, pages 375–382, 1990.

[41] J. Castellanos, J. Martínez, J. Neira, and J. Tardós. Experiments in multisensor mobile robot localization and map building. In *3rd IFAC Symp. on Intelligent Autonomous Vehicles, IAV'98*, pages 173–178, March 1998.

[42] J. Castellanos, J. Tardós, and G. Schmidt. Building a global map of the environment of a mobile robot: The importance of correlations. In *IEEE Int. Conference on Robotics and Automation, ICRA'97*, pages 1053–1059, Albuquerque, New Mexico, USA, April 1997.

[43] J. A. Castellanos, J. M. M. Montiel, J. Neira, and J. D. Tardós. The SPmap: A probabilistic framework for simultaneous localization and map building. *IEEE Trans. Robot. Autom.*, 15(5):948–952, October 1999.

[44] J. A. Castellanos, J. Neira, and J. D. Tardós. Limits to the consistency of EKF-based SLAM. In *5th IFAC Symposium on Intelligent Autonomous Vehicles, IAV'04*, Lisbon, Portugal, July 2004.

[45] S. Y. Chiem and E. Cervera. Vision-based robot formations with Bézier trajectories. In F. G. et al., editor, *Intelligent Autonomous Systems 8*, pages 191–198. IOS Press, 2004.

[46] E. Cohen, T. Lyche, and R. F. Riesenfeld. Discrete B-splines and subdivision techniques in computer-aided geometric design and computer graphics. *Comput. Gr. Image Process.*, 14:87–111, october 1980.

[47] S. A. Coons. Surfaces for computer aided design. Technical report, Design Division, Mech. Engin. Dept., M.I.T., Cambridge, Massachusetts, 1964.

[48] S. A. Coons. Rational bicubic surface patches. Technical report, MIT, 1968. Project MAC.

[49] S. A. Coons and B. Herzog. Surfaces for computer-aided design. *Journal of Aircraft*, 5(4):402–406, 1967.

[50] H. S. M. Coxeter. *Introduction to Geometry, 2nd Edition*. Wiley, 1989.

[51] A. Davison, Y. G. Cid, and N. Kita. Real-time 3D SLAM with wide-angle vision. In *Proc. IFAC Symposium on Intelligent Autonomous Vehicles, Lisbon*, July 2004.

[52] C. de Boor. On calculating with B-splines. *J. Approx. Theory*, 6:50–62, 1972.

[53] C. de Boor. *A Practical Guide to Splines*. Springer, 1978.

[54] C. de Boor and J. Rice. Least squares cubic spline approximation I – fixed knots. Technical Report CSD TR 20, Dept. of Computer Sciences, Purdue University, 1968.

[55] C. de Boor and J. Rice. Least squares cubic spline approximation II – variable knots. Technical Report CSD TR 21, Dept. of Computer Sciences, Purdue University, 1968.

[56] P. de Casteljau. *Formes à Pôles*, volume 2 of *Mathématiques et CAO*. Hermes, Paris, France, 1985.

[57] L. M. T. de Jesus and G. C. Cawley. Speech coding and synthesis using parametric curves. In *Proc. Eurospeech '97*, pages 597–600, Rhodes, Greece, september 1997.

[58] R. Descartes. *Opera philosophica omnia, in tres tomos distributa, quibus continentur*. Francofurti ad Moenum, 1692.

[59] A. Diosi and L. Kleeman. Laser scan matching in polar coordinates with application to SLAM. In *Proceedings of 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'05)*, Edmonton, Canada, August 2005.

[60] G. Dissanayake, H. Durrant-Whyte, and T. Bailey. A computationally efficient solution to the simultaneous localisation and map building (SLAM) problem. In *IEEE International Conference on Robotics and Automation, 2000*, volume 2, pages 1009–1014, 2000.

[61] G. Dissanayake, P. M. Newman, H. F. Durrant-Whyte, S. Clark, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Trans. Robot. Autom.*, 17(3):229–241, 2001.

[62] A. Doucet, N. De Freitas, and N. Gordon, editors. *Sequential Monte Carlo methods in practice*. Information Science and Statistics. Springer, 2001.

[63] H. L. D. du Monceau. *Éléments de l'Architecture Navale ou Traité Pratique de la Construction des Vaisseaux*. Paris, France, 1752.

[64] H. Durrant-Whyte. Uncertain geometry in robotics. *IEEE Transactions on Robotics and Automation*, 4(1):23–31, 1988.

[65] H. Durrant-Whyte, D. Rye, and E. Nebot. Localisation of automatic guided vehicles. In G. Giralt and G. Hirzinger, editors, *Robotics Research: The 7th International Symposium (ISRR'95)*, pages 613–625. Springer Verlag, 1996.

[66] A. Elfes. Sonar-based real-world mapping and navigation. *IEEE Journal of Robotics and Automation*, RA-3(3):249–265, June 1987.

[67] A. I. Eliazar and R. Parr. DP-SLAM 2.0. In *IEEE International Conference on Robotics and Automation*, pages 1314–1320. IEEE, 2004.

[68] A. I. Eliazar and R. Parr. Hierarchical linear/constant time slam using particle filters for dense maps. In *Advances in Neural Information Processing Systems 18*, pages 339–346, 2006.

[69] J. F. Engelberger. Historical perspective and role in automation. In S. Y. Nof, editor, *Handbook of Industrial Robotics, Second Edition*, chapter 1, pages 1–10. Wiley, March 1999.

[70] I. G. Enting, C. M. Trudinger, and D. M. Etheridge. Propagating data uncertainty through smoothing spline fits. *Tellus Series B - Chemical and Physical Meteorology*, 58(4):305–309, 2006.

[71] C. Estrada, J. Neira, and J. D. Tardós. Hierarchical SLAM: Real-time accurate mapping of large environments. *IEEE Transactions on Robotics*, 21(4):588–596, 2005.

[72] R. Eustice, H. Singh, J. Leonard, M. Walter, and R. Ballard. Visually navigating the RMS Titanic with SLAM information filters. In *Proceedings of Robotics Science and Systems*, pages 57–64, Cambridge, MA, June 2005. MIT Press.

[73] G. Farin. Algorithms for rational Bézier curves. *Computer-Aided Design*, 15(2):73–77, march 1983.

[74] I. D. Faux and M. J. Pratt. *Computational Geometry for Design and Manufacture*. Halsted Press, New York, NY, USA, 1979.

[75] J. Ferguson. Multivariable curve interpolation. *Journal of the ACM*, 11(2):221–228, april 1964.

[76] S. Filin. Surface clustering from airborne laser scanning data. *International archives of photogrammetry remote sensing and spatial information sciences*, 34:119–124, 2002.

[77] J. Fisher, J. Lowther, and C. K. Shene. If you know B-splines well, you also nnow NURBS! In *SIGCSE '04: Proceedings of the 35th SIGCSE technical symposium on Computer science education*, pages 343–347, New York, NY, USA, 2004. ACM Press.

[78] A. Forrest. Interactive interpolation and approximation by Bézier polynomials. *Computer Journal*, 15:71–79, 1972.

[79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Books in the Mathematical Sciences. W. H. Freeman & Co., New York, NY, USA, 1979.

[80] B. Gates. A robot in every home. *Scientific American*, 296(1):58–65, January 2007.

[81] M. Geimer and O. Abert. Interactive ray tracing of trimmed bicubic Bézier surfaces without triangulation. In *The 13-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2005, WSCG 2005*, pages 71–78, Campus Bory, Plzen-Bory, Czech Republic, 2005. University of West Bohemia.

[82] A. S. Glassner, editor. *An introduction to ray tracing*. Academic Press Ltd., London, UK, UK, 1989.

[83] R. Goldenthal and M. Bercovier. Spline curve approximation and design by optimal control over the knots. *Computing*, 72(1–2):53–64, 2004.

[84] W. Gordon and R. F. Riesenfeld. B-spline curves and surfaces. In R. E. Barnhill and R. F. Riesenfeld, editors, *Computer Aided Geometric Design*, pages 95–126. Academic Press, 1974.

[85] W. Gordon and J. Wixom. Shepard's method of "metric interpolation" to bivariate and multivariate interpolation. *Math. of Computation*, 32(141):253–264, 1978.

[86] A. Gray. *Modern Differential Geometry of Curves and Surfaces with Mathematica, Second Edition*. CRC-Press, 1997.

[87] W. E. L. Grimson. *Object recognition by computer: the role of geometric constraints*. MIT Press Cambridge, MA, USA, 1991.

[88] G. Grisetti, C. Stachniss, S. Grzonka, and W. Burgard. A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. In *Proceedings of Robotics: Science and Systems*, Atlanta, GA, USA, June 2007.

[89] J. Guivant and E. Nebot. Optimization of the simultaneous localization and map building algorithm for real time implementation. *IEEE Transaction of Robotic and Automation*, 17(3):242–257, June 2001.

[90] J. Guivant and E. Nebot. Improving computational and memory requirements of simultaneous localization and map building algorithms. In *IEEE International Conference on Robotics and Automation, ICRA'02*, volume 3, pages 2731–2736, Washington, DC, USA, 2002.

[91] J. Guivant, E. Nebot, and H. Durrant-Whyte. Simultaneous localization and map building using natural features in outdoor environments. In *Proc. Intelligent Autonomous Syst. 6*, volume 1, pages 581–588, Venice, Italy, July 2000.

[92] J.-S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, 2000.

[93] M. Hagedoorn and R. Veltkamp. Metric pattern spaces. Technical report, Department of Computer Science, Utrecht University, 1999.

[94] D. Hahnel, W. Burgard, D. Fox, and S. Thrun. An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, volume 1, pages 206–211, Las Vegas, NV, USA, October 2003.

[95] D. Hähnel, W. Burgard, and S. Thrun. Learning compact 3D models of indoor and outdoor environments with a mobile robot. *Robotics and Autonomous Systems*, 44(1):15–27, 2003.

[96] D. Hähnel, S. Thrun, B. Wegbreit, and W. Burgard. Towards lazy data association in SLAM. In *Proc. of the International Symposium on Robotics Research (ISRR)*, 2003.

[97] J. C. Hart. Ray tracing implicit surfaces. In *Siggraph 93 Course Notes: Design, Visualization and Animation of Implicit Surfaces*, pages 1–16, 1993.

[98] E. E. V. Havran. Robust and numerically stable Bézier clipping method for ray tracing NURBS surfaces. In *In Proceedings of 21st Spring Conference on Computer Graphics*, pages 127–135. Press, 2005.

[99] E. A. Heinz. How DarkThought plays chess. *ICCA Journal*, 30(3):166–176, 1997.

[100] R. D. Hersch. Font rasterization: the state of the art. In *Visual and technical aspects of type*, pages 78–109. Cambridge University Press, New York, NY, USA, 1993.

[101] R. Horaud and T. Skordas. Stereo correspondence through feature grouping and maximal cliques. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(11):1168–1180, 1989.

[102] A. Howard and N. Roy. The robotics data set repository (Radish), 2003.

[103] S.-M. Hu, C.-L. Tai, and S.-H. Zhang. An extension algorithm for B-splines by curve unclamping. *Computer-Aided Design*, 34(5):415–419, April 2002.

[104] J. Z. Huang and C. J. Stone. Extended linear modeling with splines. In D. D. I. Dension, M. H. Hansen, C. C. Holmes, B. Malick, and B. Yu, editors, *Nonlinear Estimation and Classification*, pages 213–234. Springer, 2002.

[105] D. G. Huosheng Hu. Landmark-based navigation of industrial mobile robots. *Industrial Robot: An International Journal*, 27(6):458–467, 2000.

[106] J. H. Hwang, R. C. Arkin, and D. S. Kwon. Mobile robots at your fingertip: Bezier curve on-line trajectory generation for supervisory control. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003.

[107] B. Jiang. Extraction of spatial objects from laser-scanning data using a clustering technique. *International archives of photogrammetry remote sensing and spatial information sciences*, 35:219–224, 2004.

[108] B. Joe. Knot insertion for Beta-spline curves and surfaces. *ACM Trans. Graph.*, 9(1):41–65, 1990.

[109] H. Kano, M. Egerstedt, H. Nakata, and C. F. Martin. B-splines and control theory. *Applied Mathematics and Computation*, 145(2–3):263–288, december 2003.

[110] S. Koenig and R. Simmons. Solving robot navigation problems with initial pose uncertainty using real-time heuristic search. In *Proceedings of the International Conference on Artificial Intelligence Planning Systems*, pages 144 – 153, 1998.

[111] J. Kostková and R. Halír. A spline approximation of a large set of points. In *Proc. of the 8th International Conference in Central Europe on Computer Graphics, Visualization and Interactive Digital Media (WSCG2000)*, 2000.

[112] F. N. Krull. The origin of computer graphics within general motors. *IEEE Ann. Hist. Comput.*, 16(3):40–56, 1994.

[113] R. Lakämper, L. J. Latecki, X. Sun, and D. Wolter. Geometric robot mapping. In E. Andres, G. Damiand, and P. Lienhardt, editors, *Discrete Geometry for Computer Imagery (DGCI'05)*, volume 3429 of *Lecture Notes in Computer Science*, pages 11–22, Poitiers, April 2005. Springer.

[114] M. D. Landon and R. W. Johnson. A B-spline-based colocation method to approximate the solutions to the equations of fluid dynamics. In *Proc. 3rd ASME/JSME Joint Fluids Engineering Conference*, San Francisco, CA, US, july 18–23 1999.

[115] J. Leonard and H. Durrant-Whyte. Mobile robot localization by tracking geometric beacons. *IEEE Transactions on Robotics and Automation*, 7(3):376–386, 1991.

[116] J. Leonard and H. Feder. A computationally efficient method for large-scale concurrent mapping and localization. In D. K. J. Hollerbach, editor, *International Symposium on Robotics Research*, 1999.

[117] R. Liming. *Practical Analytical Geometry with Applications to Aircraft*. Macmillan, 1944.

[118] D. Lischinski and J. Gonczarowski. Improved techniques for ray tracing parametric surfaces. *The Visual Computer: International Journal of Computer Graphics*, 6(3):134–152, 1990.

[119] J. S. Liu and R. Chen. Sequential monte carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93:1032–1044, 1998.

[120] Y. Liu, H. Yang, and W. Wang. Reconstructing B-spline curves from point clouds - a tangential flow approach using least squares minimization. In *In International Conference on Shape Modeling and Applications*, pages 4–12, 2005.

[121] F. Lu and E. Milios. Robot pose estimation in unknown environments by matching 2D range scans. *J. Intell. Robotics Syst.*, 18(3):249–275, 1997.

[122] D. H. MacLaren. Formulas for fitting a spline curve through a set of points. Technical report, Boeing, 1958. Appl. Math. Report no. 2.

[123] I. Mahon and S. Williams. Three-dimensional robotic mapping. In *Australasian Conference on Robotics and Automation*, 2003.

[124] W. Martin, E. Cohen, R. Fish, and P. Shirley. Practical ray tracing of trimmed NURBS surfaces. *Journal of Graphics Tools*, 5(1):27–52, 2000.

[125] P. S. Maybeck. *Stochastic models, estimation, and control*, volume 141 of *Mathematics in Science and Engineering*. Academic Press, Inc., 1979.

[126] L. Ming and R. H. Taylor. Spatial motion constraints in medical robot using virtual fixtures generated by anatomy. In *ICRA'04. 2004 IEEE International Conference on Robotics and Automation*, volume 2, pages 1270–1275, New Orleans, USA, 2004.

[127] M. Montemerlo. *FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem with Unknown Data Association*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, July 2003.

[128] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: a factored solution to the simultaneous localization and mapping problem. In *Eighteenth national conference on Artificial intelligence*, pages 593–598, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

[129] L. Montesano, J. Minguez, and L. Montano. Probabilistic scan matching for motion estimation in unstructured environments. In *Proceedings of the Conference on Intelligent Robots and Systems (IROS)*, 2005.

[130] S. Moorehead. *Autonomous Surface Exploration for Mobile Robots*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, August 2001.

[131] H. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *Proc. IEEE Int. Conf. Robot. Autom.*, volume 2, pages 116–121, March 1985.

[132] J. Neira, J. D. Tardós, and J. A. Castellanos. Linear time vehicle relocation in SLAM. In *Proceedings of the 2003 IEEE International Conference on Robotics and Automation, ICRA 2003*, pages 427–433, Taipei, Taiwan, 2003.

[133] J. Neira and J. D. Tardós. Data association in stochastic mapping using the joint compatibility test. *IEEE Trans on Robotics and Automation*, 17:890–897, december 2001.

[134] P. Newman, D. Cole, and K. Ho. Outdoor SLAM using visual appearance and laser ranging. In *International Conference on Robotics and Automation*, 2006.

[135] P. Newman and J. J. Leonard. Consistent, convergent, and constant-time SLAM. In *International Joint Conference on Artificial Intelligence*, Acapulco Mexico, August 2003.

[136] P. M. Newman, J. J. Leonard, and R. J. Rikoski. Towards constant-time SLAM on an autonomous underwater vehicle using synthetic aperture sonar. In *Proceedings of the Eleventh International Symposium on Robotics Research*, Sienna, Italy, October 2003.

[137] J. Nieto, T. Bailey, and E. Nebot. Scan-SLAM: Combining EKF-SLAM and scan correlation. In *Proc. Int. Conf. on Field and Service Robotics*, pages 129–140, 2005.

[138] J. Nieto, J. Guivant, and E. Nebot. The hybrid metric maps (HYMMs): a novel map representation for DenseSLAM. In *IEEE International Conference on Robotics and Automation, ICRA'04*, volume 1, pages 391–396, 2004.

[139] E. Olson, J. Leonard, and S. Teller. Fast iterative optimization of pose graphs with poor initial estimates. In *IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 2262–2269, 2006.

[140] E. Panofsky. *The Codex Huygens and Leonardo da Vinci's art theory*, volume 13 of *Studies of the Warburg Institute*. The Warburg Institute, London, 1940.

[141] P. M. Pardalos and G. P. Rodgers. A branch and bound algorithm for the maximum clique problem. *Comput. Oper. Res.*, 19(5):363–375, 1992.

[142] P. M. Pardalos and J. Xue. The maximum clique problem. *Global Optimization*, 4:301–328, 1994.

[143] H. Park and J.-H. Lee. B-spline curve fitting based on adaptive curve refinement using dominant points. *Comput. Aided Des.*, 39(6):439–451, 2007.

[144] S. Parker, W. Martin, P.-P. J. Sloan, P. Shirley, B. Smits, and C. Hansen. Interactive ray tracing. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Courses*, page 12, New York, NY, USA, 2005. ACM.

[145] L. M. Paz, J. Guivant, J. D. Tardós, and J. Neira. Data association in O(n) for divide and conquer SLAM. In *Proc. Robotics: Science and Systems*, Atlanta, GA, USA, June 2007.

[146] L. M. Paz, P. Piniés, J. Neira, and J. D. Tardós. Global localization in slam in bilinear time. In *IEEE/RSJ Int.Conf.on Intelligent Robots and Systems*, Edmonton, Canada, August 2005.

[147] G. Piccolo, M. Karasalo, D. Kragic, and X. Hu. Contour reconstruction using recursive smoothing splines - experimental validation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2071–2076, San Diego, CA, 2007.

[148] L. Piegl and W. Tiller. *The NURBS Book (2nd ed.)*. Springer-Verlag New York, Inc., New York, USA, 1997.

[149] H. Pottmann, S. Leopoldseder, and M. Hofer. Approximation with active b-spline curves and surfaces. In *PG '02: Proceedings of the 10th Pacific Conference on Computer Graphics and Applications*, page 8, Washington, DC, USA, 2002. IEEE Computer Society.

[150] M. J. D. Powell. The local dependence of least squares cubic splines. *SIAM Journal on Numerical Analysis*, 6(3):398–413, september 1969.

[151] S. Pu. Managing freeform curves and surfaces in a spatial DBMS. Master's thesis, Delft University of Technology, july 2005.

[152] N. V. Puntambekar and A. G. Jablokow. Selection of the number of control points for spline surface approximation. In V. Skala, editor, *WSCG'98 Conference Proceedings*, 1998.

[153] L. Ramshaw. Blossoming: A connect-the-dots approach to splines. Technical report, Digital Systems Research Center, 1987.

[154] R. F. Riesenfeld. *Applications of B-Spline Approximation to Geometric Problems of Computer-Aided Design*. PhD thesis, Syracuse Uiversity, 1972.

[155] D. Rodríguez-Losada. *SLAM Geométrico en Tiempo Real para Robots Móviles en Interiores Basado en EKF*. PhD thesis, Universidad Politécnica de Madrid, 2004.

[156] D. Rodriguez-Losada and F. Matía. Integrating segments and edges in feature-based SLAM. In *Proceedings of IEEE - ICAR 2003. The 11th International Conference on Advanced Robotics*, volume 3, pages 1717–1722, Coimbra, Portugal, June 2003.

[157] D. Rodriguez-Losada, F. Matía, R. Galán, and A. Jiménez. Blacky, an interactive mobile robot at a trade fair. In *IEEE International Conference on Robotics and Automation, 2002 (ICRA '02)*, volume 4, pages 3930–3935, Washington DC., USA, 2002.

[158] D. Rodriguez-Losada, F. Matía, L. Pedraza, A. Jiménez, and R. Galán. Consistency of SLAM-EKF algorithms for indoor environments. *Journal of Intelligent and Robotic Systems*, 50(4):375–397, December 2007.

[159] D. F. Rogers. *An introduction to NURBS: With historical perspective*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2001.

[160] M. Rosheim. Robotic surrogate: work in progress. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 399–403, April 1996.

[161] M. E. Rosheim. In the footsteps of leonardo. *IEEE Robotics & Automation Magazine*, 4(2):12–14, June 1997.

[162] M. E. Rosheim. *Leonardo's Lost Robots*. Springer, 2006.

[163] S. D. Roth. Ray casting for modeling solids. *Computer Graphics and Image Processing*, 18(2):109–144, February 1982.

[164] B. Roy. *Algèbre Moderne et Théorie des Graphes Orientées vers les Sciences Economiques et Sociales*. Dunod, Paris, 1969 (vol. 1) 1970 (vol. 2).

[165] F. Samavati, M. Ali Nur, R. Bartels, and B. Wyvill. Progressive curve representation based on reverse subdivision. In *2003 International Conference on Computational Science and Its Applications, Lecture Notes in Computer Science*, volume 2667, pages 67–78, Montreal, Canada, 2003.

[166] P. San Segundo, R. Galán, F. Matía, D. Rodriguez-Losada, and A. Jiménez. Efficient search using bitboard models. In *ICTAI '06: Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence*, pages 132–138, Washington, DC, USA, 2006. IEEE Computer Society.

[167] P. San Segundo, D. Rodriguez-Losada, R. Galán, F. Matía, and A. Jiménez. Efficient global localization by searching a bit-encoded graph. In *IAV'07*, 2007.

[168] D. Santosh, S. Achar, and C. Jawahar. Autonomous image-based exploration for mobile robot navigation. In *IEEE International Conference on Robotics and Automation*, 2008.

[169] I. J. Schoenberg. Contributions to the problem of approximation of equidistant data by analytic functions, parts a & b. *Quart. Applied Math*, 4:45–99, 112–141, 1946.

[170] D. Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proc. ACM National Conference*, pages 517–524, 1968.

[171] R.-C. C. Shyue-Wu Wang *, Zen-Chung Shih. An improved rendering technique for ray tracing Bézier and B-spline surfaces. *The Journal of Visualization and Computer Animation*, 11(4):209–219, October 2000.

[172] R. Sibson. A brief description of the natural neighbour interpolant. Technical report, Math Department, U. of Bath, 1980.

[173] R. Sibson. A vector identity for the dirichlet tesselation. *Math. Proc. Cambridge Philos. Soc.*, 87:151–155, 1980.

[174] R. Simmons, J. Lopez Fernandez, R. Goodwin, S. Koenig, and J. O'Sullivan. Lessons learned from Xavier. *Robotics and Automation Magazine*, 7(2):33–39, 2000.

[175] J. M. Singh and P. J. Narayanan. Real-time ray-tracing of implicit surfaces on the GPU. Technical Report IIIT/TR/2007/72, International Institute of Information Technology Hyderabad, Deemed University, 2007.

[176] A. W. Sleeswyk. Vitruvius' odometer. *Scientific American*, 245(4):188–200, October 1981.

[177] R. Smith, M. Self, and P. Cheeseman. A stochastic map for uncertain spatial relationships. In *International Symposium of Robotis Research*, pages 467–474, Univ. of California, Santa Clara, California, United States, 1987.

[178] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In J. F. Lemmer and L. N. Kanal, editors, *Uncertainty in artificial intelligence 2*, volume 5, pages 435–461, New York, 1988. Elsevier.

[179] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In I. J. Cox and G. T. Wilfon, editors, *Autonomous Robot Vehicles*, pages 167–193. Springer-Verlag New York, Inc., New York, NY, USA, 1990.

[180] R. Smith, M. Self, and P. Cheeseman. A stochastic map for uncertain spatial relationships. In S. S. Iyengar and A. Elfes, editors, *Autonomous Mobile Robots: Perception, Mapping, and Navigation*, volume 1, pages 323–330. IEEE Computer Society Press, Los Alamitos, CA, 1991.

[181] R. C. Smith and P. Cheeseman. On the representation and estimation of spatial uncertainty. *The International Journal of Robotics Research*, 5(4):56–68, 1986.

[182] H. Sorenson. Least-squares estimation: from Gauss to Kalman. *IEEE Spectrum*, 7:63–68, 1970.

[183] S. Sotoodeh. Hierarchical clustered outlier detection in laser scanner point clouds. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36:383–388, 2007.

[184] C. Stachniss and W. Burgard. Exploring unknown environments with mobile robots using coverage maps. In *Proc. of the International Conference on Artificial Intelligence (IJCAI)*, 2003.

[185] C. Stachniss and W. Burgard. Mapping and exploration with mobile robots using coverage maps. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003.

[186] A. J. Sweeney and R. H. Barrels. Ray tracing free-form B-spline surfaces. *IEEE Comput. Graph. Appl.*, 6(2):41–49, February 1986.

[187] J. D. Tardós, J. Neira, P. M. Newman, and J. J. Leonard. Robust mapping and localization in indoor environments using sonar data. *I. J. Robotic Res.*, 21(4):311–330, 2002.

[188] S. Thrun. An online mapping algorithm for teams of mobile robots. *International Journal of Robotics Research*, 20(5):335–363, May 2001.

[189] S. Thrun. Robotic mapping: a survey. In *Exploring artificial intelligence in the new millennium*, pages 1–35. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002.

[190] S. Thrun, A. Bücken, W. Burgard, D. Fox, T. Fröhlinghaus, D. Hennig, T. Hofmann, M. Krell, and T. Schimdt. Map learning and high-speed navigation in RHINO. In D. Kortenkamp, R. P. Bonasso, and R. Murphy, editors, *AI-based Mobile Robots: Case Studies of Successful Robot Systems*, chapter 1, pages 21–52. MIT Press, Cambridge, MA, 1998.

[191] S. Thrun, W. Burgard, and D. Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping. In *IEEE International Conference on Robotics and Automation*, pages 321–328, 2000.

[192] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press, September 2005.

[193] T. Thuy Vu and M. Tokunaga. Filtering airborne laser scanner data: A wavelet-based clustering method. *Photogrammetric Engineering & Remote Sensing*, 70(11):1267–1274, 2004.

[194] T. Toe and T. V. To. Curve matching by using B-spline curves. In *12th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision '2004 (WSCG '2004)*, 2004.

[195] F. Torres, J. Pomares, P. Gil, S. T. Puente, and R. Aracil. *Robots y Sistemas Sensoriales*. ISBN 84-205-3574-5. Prentice Hall, 2002.

[196] L. N. Trefethen. Barycentric Lagrange interpolation. *SIAM Review*, 46(3):501–517, 2004.

[197] L. N. Trefethen and J. A. C. Weideman. Two results on polynomial interpolation in equally spaced points. *J. Approx. Theory*, 65(3):247–260, 1991.

[198] M. Unser. Splines: A perfect fit for signal processing. In *Tenth European Signal Processing Conference (EUSIPCO'00)*, Tampere, Finland, september 2000. Plenary talk.

[199] M. Veeck and W. Burgard. Learning polyline maps from range scan data acquired with mobile robots. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, volume 2, pages 1065–1070, Sendai, Japan, 2004.

[200] R. C. Veltkamp. Shape matching: Similarity measures and algorithms. In *SMI '01: Proceedings of the International Conference on Shape Modeling & Applications*, page 188, Washington, DC, USA, 2001. IEEE Computer Society.

[201] D. Vernet. Expression mathématique des formes. *Ingenieurs de l'Automobile*, 10:509–520, 1971.

[202] K. J. Versprille. *Computer-Aided Design Applications of the Rational B-spline Approximation Form*. PhD thesis, Syracuse University, feb 1975.

[203] C.-C. Wang, C. Thorpe, and S. Thrun. Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Taipei, Taiwan, September 2003.

[204] S.-W. Wang, Z.-C. Shih, and R.-C. Chang. An efficient and stable ray tracing algorithm for parametric surfaces. *J. Inf. Sci. Eng.*, 18(4):541–561, 2002.

[205] Y. Wang and E. K. Teoh. A novel 2D shape matching algorithm based on B-spline modeling. In *Proceedings of the 2004 International Conference on Image Processing (ICIP 2004)*, volume 1, pages 409–412, Singapore, 2004.

[206] G. Welch and G. Bishop. An introduction to the Kalman filter. Technical Report TR 95-041, University of North Carolina at Chapel Hill, Department of Computer Science, 1995.

[207] T. Whitted. An improved illumination model for shaded display. *Communications of the ACM*, 23(6):343–349, 1980.

[208] S. Williams, G. Dissanayake, and H. Durrant-Whyte. Field deployment of the simultaneous localisation and mapping algorithm. In *15th IFAC World Congress on Automatic Control*, Barcelona, Spain, 2002.

[209] S. Williams, P. Newman, G. Dissanayake, and H. Durrant-Whyte. Autonomous underwater simultaneous localisation and map building. In *IEEE International Conference on Robotics and Automation*, volume 2, pages 1793–1798, San Francisco, USA, April 2000.

[210] G. Wolberg. Image morphing: a survey. *The Visual Computer*, 14(8/9):360–372, 1998.

[211] H. P. Yang, W. W., and J. G. Sun. Control point adjustment for B-spline curve approximation. *Computer-Aided Design*, 36:639–652, 2004.

[212] S. Zhang, L. Xie, M. Adams, and F. Tang. Geometrical feature extraction using 2D range scanner. In *Proc. Fourth Int. Conf. Contr. Automat.*, pages 901–905, Montreal, Canada, June 2003.

[213] Z. Zhang, J. Tomlinson, and C. Martin. Splines and linear control theory. *Acta Applicandae Mathematicae: An International Survey Journal on Applying Mathematics and Mathematical Applications*, 49(1):1–34, october 1997.