UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE CIENCIAS FÍSICAS

Departamento de Arquitectura de Computadores y Automática

INTEGRACIÓN Y FUSIÓN MULTISENSORIAL EN ROBOTS MÓVILES AUTÓNOMOS

MEMORIA PARA OPTAR AL GRADO DE DOCTOR

PRESENTADA POR

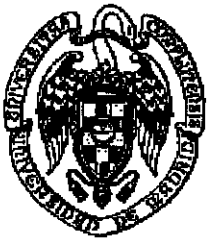
José Antonio López Orozco

Bajo la dirección del doctor

Jesús Manuel de la Cruz García

Madrid, 2003

ISBN: **978-84-669-1585-4**



UNIVERSIDAD COMPLUTENSE DE MADRID
Departamento de Arquitectura de Computadores y Automática
Madrid - ESPAÑA

Integración y fusión multisensorial en Robots móviles autónomos

Memoria para optar al grado de doctor

presentada por

José Antonio López Orozco

Madrid 1998

λ 250 52950

Índice

Introducción

Planteamiento general del problema

La teoría y aplicación de sistemas multisensor en diversas aplicaciones y la gran mejora observada al utilizarlos en problemas tales como el procesamiento de imágenes, aplicaciones militares, reconocimiento de entorno, ayuda al diagnóstico médico, etc., ha provocado un creciente interés en multitud de investigadores. La gran variedad de aplicaciones en las que se puede utilizar y la mejora que produce su utilización ha captado el interés de diversas áreas de conocimiento (estadística, inteligencia artificial, robótica, etc.) y ha propiciado el desarrollo de nuevas técnicas (redes bayesianas, lógica difusa, redes neuronales artificiales, etc.). Aplicaciones clásicas y nuevas conducen a distintas aproximaciones en el tratamiento de la información obtenida y en el tratamiento de los distintos aspectos de la fusión de datos, como son: técnicas de inteligencia artificial y uso de bases de conocimiento, técnicas de procesamiento de señal y teoría de la estimación, técnicas estadísticas o probabilísticas (teoría bayesiana, Dempster-Shafer, ...), técnicas heurísticas, lógica fuzzy, redes neuronales, etc.

La amplia literatura e interés mostrado sobre integración y fusión multisensorial desde mediados de 1980 muestra su gran desarrollo, creciente interés e importancia en múltiples aplicaciones. A su vez surgen nuevos problemas a resolver y nuevas áreas de aplicación. Ejemplos de este desarrollo son los

numerosos de artículos sobre integración y fusión multisensorial publicados en revistas como *International Journal Robotics Research, IEEE System, Man, and Cybernetics, Trans. on Robotics and Automation, Patterns Recognition and Machine Intelligence*, etc.; los numerosos encuentros o monografías relacionados con el tema como *FUSION'89, Proceedings of the IEEE* de enero de 1997, *Proceedings of International Society for Optical Engineering (SPIE)* sobre Sensor Fusión, Remote Sensing, ...; y libros sobre fusión de datos y aplicaciones como [Durrant-Whyte 88], [Hager 90], [Waltz 90], [Hall 92], [Abidi 92], [Manyika 94], [Varshney 96], etc.

De estos trabajos se infiere la necesidad de mejorar la calidad de la información externa utilizada por cualquier aplicación. En lugar de utilizar un único sensor, aunque sea un sensor de alta precisión, es preferible el uso de varios sensores (aunque sean de menor precisión) que sean complementarios o redundantes. Esto permite disponer de diversas medidas que aseguran la fiabilidad de los datos, evitan errores, mejoran el tiempo de respuesta y ayudan a una mejor interpretación de lo que está siendo observado. Éste será el cometido del sistema de fusión e integración multisensorial.

El sistema más intuitivo que realiza integración y fusión multisensorial es el ser humano, que hace uso de sus cinco sentidos para interpretar el entorno que le rodea. Siguiendo con este ejemplo, un sistema de integración y fusión multisensorial se ocupa de generar el estímulo sensorial que represente a un conjunto de medidas de diversos sensores. Posteriormente este estímulo es analizado para encontrar qué estado del mundo se corresponde con un estímulo semejante. No se debe olvidar que la fusión o integración de medidas produce una medida mejorada del entorno observado, es decir, una medida más clara o precisa que cada uno de sus componentes, pero que su cometido no es interpretar dicha información.

Tengamos en cuenta que la percepción de cualquier agente inteligente, y en concreto de un robot móvil, está muy relacionada con el tipo de fuente de información disponible, es decir, con los sensores utilizados. Esto implica que cualquier aplicación que necesite diversos sensores deba utilizar alguna técnica de fusión o integración sensorial, pero su elección está muy condicionada por los sensores empleados. Además, éstos pueden ser muy diversos y la forma de tratar su información muy variable por lo que el sistema de fusión implementado (método

de fusión de las medidas disponibles) suele ser particular de cada problema concreto y no puede ser adaptable a otro tipo de problemas.

Esto provoca que los distintos algoritmos desarrollados sólo puedan ser utilizados en un conjunto muy restringido de aplicaciones y en muchos casos incluso sólo en el problema tratado. Por tanto, sería conveniente desarrollar un sistema de integración y fusión multisensorial que pueda ser utilizable por distintas aplicaciones y distintos sensores.

Para realizarlo se debe procurar que el sistema de integración abstraiga al sensor físico en la medida de lo posible, puesto que la dependencia con el tipo de sensor evita la generalidad. Sin embargo, la abstracción del sensor requiere un procesamiento de la información antes de ser introducido en el sistema de fusión y por tanto, como siempre que se generaliza, a una pérdida de la calidad de la información obtenida. Sin embargo, esta pérdida es suplida con la mejora que significa la fusión multisensorial.

Además de una abstracción de los sensores utilizados, si se desea que el sistema sea versátil, debe ser restringido a un subconjunto concreto de aplicaciones puesto que será imposible realizar un sistema de fusión general para cualquier tipo de aplicación. El tipo de problema escogido son los robots móviles autónomos terrestres porque en ellos la integración y fusión sensorial representa un papel crucial en el correcto funcionamiento del robot. Además, los robots móviles autónomos utilizan una gran cantidad de sensores que representan la mayoría de los sensores disponibles, lo que permite utilizar unas técnicas que podrían ser extensibles a otros problemas y aplicaciones.

Por último, en muchas aplicaciones no existe una única solución sino que depende del modelo del mundo utilizado y de la interpretación que se desee realizar de éste, por lo tanto su diseño debe ser claro y sencillo de forma que permita una fácil adaptación, no sólo de los sensores disponibles, sino que también permita modificar el modelo del entorno o cualquier otro algoritmo que sea necesario. Además, las restricciones de tiempo intrínsecas a los robots móviles implican la utilización de métodos de tiempo real y la utilización de algoritmos rápidos o que tengan la posibilidad de paralelizarse de forma sencilla.

Todo este conjunto de restricciones y necesidades llevará a la elección de una serie de herramientas y algoritmos determinados que conducirán al diseño e implementación del sistema de integración y fusión multisensorial de esta Tesis.

Objetivos de la Tesis

El diseño de un sistema de fusión de datos procedentes de distintos sensores, sistema de integración multisensorial, tiene muy en cuenta hacia qué tipo de aplicación se dirige. Incluso dentro de una misma aplicación las opciones escogidas en su implementación están condicionadas por el tipo y número de los sensores disponibles. En muchas ocasiones esta especificidad conduce a que el aprovechamiento o adaptación de una arquitectura ya desarrollada, bien para utilizarla en otras aplicaciones bien para probar nuevos algoritmos, sea más costosa que el desarrollo de una nueva. Por este motivo, en esta Tesis, no se intenta encontrar el método ideal para realizar la integración multisensorial, sino que se busca un método de integración y fusión de medidas que sea sencillo de utilizar, paralelizable y que permita probar y desarrollar distintos algoritmos sin necesidad de grandes modificaciones. Dentro de las múltiples aplicaciones en las que es posible realizar integración multisensorial nos centraremos en los robot móviles terrestres autónomos.

Durante el desarrollo de la presente Tesis se ha construido un sistema de control y sensorización para robots móviles autónomos terrestres que permite utilizar distintos sensores, o sistemas sensoriales, conectándolos al mismo como si fuese una caja negra. De este modo se pone a disposición del usuario un método de integración mutisensorial para realizar pruebas sobre la utilización conjunta de distintos grupos de sensores (complementarios o/y redundantes) que evita la necesidad de crear una estructura que soporte todo el sistema sensorial. Es decir, se permite que se puedan cambiar los sensores disponibles de forma sencilla y se proporciona una estructura de control clara, sencilla y abierta que facilita la modificación de las tareas existentes y la construcción o adición de capas que le doten de nuevas funcionalidades. El estudio, por tanto, se ha enfocado de la siguiente manera:

1.- *Construcción de un sistema de integración y fusión multisensorial.* Se diseña un sistema de sensorización que sea independiente del tipo y número de sensores que utilice. Para ello se realiza,

- El *estudio de las técnicas más usuales de fusión* de datos, buscando aquellas técnicas que permiten su utilización paralela y asíncrona.

- La preparación de los métodos de fusión utilizados para que sean fácilmente *adaptables y reconfigurables por el usuario*, es decir, que se pueda añadir o suprimir un sensor sin modificar por completo el sistema diseñado.
- El *análisis y clasificación de los distintos tipos de sensores* utilizados en los robot móviles autónomos terrestres, para comprobar que las técnicas escogidas permiten la correcta fusión de sus datos.
- El *diseño de un sensor lógico* que englobe las características de los sensores de un robot móvil y abstraiga al sensor físico del sistema de fusión multisensorial.
- El diseño de un método de fusión de medidas que sea *versátil, distribuido, asíncrono, robusto frente a fallos y fácilmente paralelizable*.

2.- *Implementación de una arquitectura de control* del robot que permita mostrar el uso de este sistema sensorial. Esta arquitectura, permite probar distintas configuraciones sensoriales y diversos algoritmos sin necesidad de modificar el diseño realizado. En este caso se realiza,

- El estudio de las distintas tareas necesarias en todo robot móvil y de su organización. Las tareas que componen nuestro sistema pueden trabajar en paralelo y se comunican por paso de mensajes. Se realiza un *diseño utilizando metodología orientada a objetos*, especificando claramente su funcionamiento interno y el protocolo entre tareas.
- El *diseño de una arquitectura abierta* con un diseño claro y versátil permite que cualquier diseñador pueda hacer uso de algoritmos propios sobre la misma, sin el coste que implicaría el desarrollo de una arquitectura completa.
- La *implementación en tiempo real* de la arquitectura desarrollada para satisfacer, dentro de lo posible, las restricciones de tiempo típicas de los robots móviles.

Organización de la Memoria

La Memoria se ha dividido en dos partes: la integración multisensorial y la arquitectura de control de robots móviles terrestres.

La integración multisensorial se recoge principalmente en los capítulos 1 a 4, donde se describen los sensores y sistemas sensoriales de los robots móviles y cómo integrar las medidas que pueden ofrecernos de modo que sea independiente del tipo de sensores utilizados. Los métodos de integración escogidos pretenden que el sistema final sea asíncrono, paralelizable y fácil de adaptar a cualquier configuración sensorial.

La implementación del sistema de fusión definido, la arquitectura de control diseñada y ejemplos de su utilización, se tratan en los capítulos 5 y 6. En ellos se muestran las tareas necesarias en el control de un robot móvil, las arquitecturas existentes y su relación con el sistema sensorial. Se propone una arquitectura de control que permita utilizar el sistema sensorial diseñado y que sea fácilmente adaptable a las necesidades de cada usuario. Esta arquitectura de control será diseñada con restricciones de tiempo real y utilizando la metodología orientada a objetos.

Con esta división, la organización lógica de los capítulos es la siguiente:

En el capítulo 1 se describe la diferencia entre fusión e integración multisensorial, la clasificación de los procesos de fusión, sus arquitecturas y los algoritmos utilizados. Además se muestran distintos ejemplos de sistemas sensoriales existentes en robots móviles autónomos destacando la diversidad de sensores disponibles y las diferentes configuraciones que pueden darse.

A continuación, en el capítulo 2, se describen distintos sensores disponibles para robot móviles, la relación entre ellos y qué tipo de información pueden proporcionar a un robot móvil terrestre. Se realiza una clasificación de los sensores y métodos de integración en dos grandes grupos: de estado y de entorno. En estos dos grupos se incluyen todos los sensores que se utilizan en un robot móvil y a partir de esta clasificación se definirá un tipo de sensor lógico que sea compatible con las especificaciones requeridas para nuestro sistema. Para su descripción se utiliza la metodología orientada a objetos como forma más clara de representación de una organización compleja.

Para realizar la fusión de datos de estado, en concreto de posición, se revisan los métodos más utilizados, destacando la forma de tratar la incertidumbre de los

sensores, su funcionamiento distribuido, paralelización, asíncronismo y su posibilidad de adaptación dinámica a los distintos tipos de sensores. De ellos se concluye que un método adecuado para realizar la fusión buscada es mediante un filtro de Kalman extendido, distribuido y asíncrono. Todo esto se realiza en el Capítulo 3, donde además se desarrollan las ecuaciones del filtro para su correcto funcionamiento y se muestra con un ejemplo de eficacia, destacando la robustez ante fallos de los sensores.

La fusión de medidas de entorno es diferente a la fusión de medidas de estado y requiere otro tipo de algoritmos. Son necesarias técnicas probabilísticas o técnicas próximas a la inteligencia artificial. En el Capítulo 4 se describen algunas de las técnicas existentes haciendo hincapié en la forma de tratar la incertidumbre de los sensores, y, al igual que en el caso anterior, su funcionamiento distribuido, paralelizable, asíncrono y su posibilidad de adaptación dinámica a distintos tipos de sensores. Se escogen las redes bayesianas como técnica de fusión multisensorial más adecuada para conseguir las características buscadas y se desarrolla un método asíncrono, dinámico y paralelizable.

En este momento se dispondrá de un sistema sensorial paralelizable, distribuido, asíncrono y versátil, pero para mostrar su funcionamiento es necesario dotarle de una arquitectura de control que permita su utilización de forma sencilla y directa, ya sea como una caja negra o como una plataforma para sustituir o probar nuevos algoritmos. Así, en el capítulo 5 se desarrolla una arquitectura de control adecuada para la utilización del sistema sensorial propuesto y se describe cómo se acopla con nuestro sistema de integración multisensorial. Por último se desarrolla un sistema de tiempo real de forma clara y sencilla a partir de la arquitectura y sistema sensorial propuestos. Para ello se escoge una metodología orientada a objetos en tiempo real.

Finalmente, en el capítulo 6, se realiza la implementación del sistema desarrollado. Primero se realizan distintas simulaciones para mostrar la adecuación del diseño a todas las necesidades propuestas, destacando la posibilidad de variar el tipo o número de sensores utilizados, la utilización de distintos robots para diversas aplicaciones o el cambio de algoritmos para probar nuevas técnicas. A continuación, se aplica el sistema de fusión y la arquitectura de control propuestos sobre un robot real. En este caso se muestra la adaptación del sistema a los requerimientos sensoriales de un robot real para destacar su versatilidad y facilidad de uso, así cono su respuesta en tiempo real.

Capítulo 1

Aspectos generales

En este capítulo se muestran los aspectos más destacables de la fusión e integración multisensorial y los problemas que conlleva. Se señalan muchas de las técnicas de fusión existentes, normalmente dependientes de los sensores involucrados, y se muestra que no existe un método ideal para realizar fusión de datos. Todo ello nos ayudará a comprender la amplia variedad de aplicaciones existentes, diseñadas explícitamente para una gran variedad de situaciones distintas. Las aplicaciones más importantes en fusión de datos son las militares, que van desde sistemas empotrados basados en microprocesadores de tiempo real en armamento militar hasta sofisticados sistemas expertos de reconocimiento de tropas y armamento enemigo [Waltz 86; Chellappa 97; Antony 97]. Sin embargo, cada vez adquiere más importancia la utilización de sistemas de fusión multisensorial en aplicaciones civiles, entre las que se pueden destacar sistemas de control remoto, sistemas de diagnóstico médico y robótica [Kam 97]. A pesar de la dificultad en la elección de un sistema o método concreto, en este capítulo se señalan una serie de orientaciones para la elección de la arquitectura, los algoritmos y las técnicas más convenientes en cada caso.

En todo momento se exponen las técnicas usuales o significativas en aplicaciones en general, y se destaca cual es la adaptación o su utilidad en el caso concreto de robots móviles autónomos terrestres.

1.1. Aspectos generales de la integración y fusión multisensorial

El mejor ejemplo de integración multisensorial lo tenemos muy próximo, los seres humanos somos capaces de integrar de forma natural la información de los sentidos (vista, oído, gusto, tacto y olfato). La integración multisensorial es imprescindible para nuestra supervivencia y pueden mostrarse multitud de ejemplos de integración multisensorial realizada por el ser humano. Sin embargo debe tenerse en cuenta que la integración multisensorial no siempre es beneficiosa, sino que depende de la integridad y fiabilidad de los sensores utilizados e incluso su eficacia varía dependiendo de las situaciones en las que se aplique. Siguiendo con el ejemplo humano, un caso de integración multisensorial que nos lleva a una observación equivocada es un ventrílocuo. Cuando observamos a un ventrílocuo y a su muñeco, la información visual (movimiento de los labios del muñeco) y la información auditiva (la voz del ventrílocuo) son integradas dando la sensación de que es el muñeco el que habla. Otro ejemplo de integración errónea que involucra otros sentidos es la integración del olfato y el gusto. Todos hemos experimentado, en alguna ocasión, que si en el momento de saborear una sustancia olemos algo repulsivo nos provocará arcadas. Una prueba sencilla es ofrecer un vaso de agua (incolora, inodora e insípida) cuyo exterior se ha impregnado con gasolina (muy volátil y de olor desagradable) pero sin que esté próximo al borde ni contacte en ningún momento con el agua. El pobre conejillo de indias jurará que el agua sabe a gasolina.

A pesar de estos ejemplos las ventajas de la integración multisensorial son muchas, lo que lleva a utilizarla siempre que nos sea posible, en nuestros sistemas inteligentes. Las ventajas son [Luo 89]:

(1) *Redundancia*, ayuda a mejorar la fiabilidad del sistema debido a que reduce la incertidumbre de las medidas individuales al observarlas con distintos sensores de distintas precisiones.

(2) *Complementariedad*, permite observar ciertas características del entorno que son imposibles de observar con sensores individuales.

(3) *Oportunidad*, permite captar la información con el sensor que primero esté disponible aumentando además la velocidad del sistema sensorial.

(4) *Coste de la información*, normalmente la integración de distintos sensores permite que el coste del sistema sea menor que si se utilizase un único sensor que tuviese las mismas prestaciones.

Fusión frente a integración

Debe distinguirse entre integración y fusión de datos de múltiples sensores. *Integración multisensorial* se refiere al uso de la información que proporcionan distintos dispositivos sensoriales para ayudar a la consecución de una tarea del sistema. *Fusión multisensorial*, por otra parte, se refiere a cualquier estado en el proceso de integración donde hay una combinación (o fusión) de diferentes fuentes de información sensorial en un cierto formato común. Esta descripción de fusión multisensorial podría aplicarse a un único dispositivo que ofreciese información sensorial a lo largo del tiempo. De estas dos definiciones puede deducirse que muchos sistemas que realizan integración multisensorial utilizan la información de los distintos sensores para guiar la operación de otros o para cooperar en la consecución de una tarea común pero sin que nunca lleguen a fusionar la información de sus sensores. Por ejemplo, [Schneider 96] utiliza sensores de ultrasonidos para la percepción de entorno (esquiva de obstáculos) y visión para la búsqueda de balizas; o [Lallement 98] que utiliza radar laser para localizar balizas y obtener su posición con precisión y visión para la identificación de dicha baliza.

En este trabajo de investigación se busca un sistema multisensorial para el guiado de vehículos autónomos terrestres que permita la integración y fusión de la información de diferentes sensores de la forma más conveniente posible. Esto llevará a que en algunas ocasiones se utilicen dichos términos indistintamente, aunque nos centraremos principalmente en el problema de la fusión multisensorial.

Fusión multisensorial

Como se señala en [Hall 92], la fusión multisensorial es un proceso. Este proceso es sencillo en cuanto a su concepto se refiere, pero enormemente complejo en su implementación. La esencia de la fusión es el manejo de la incertidumbre asociada a los datos que se fusionan, pero teniendo en cuenta que no es el único factor a considerar en el desarrollo de sistemas de fusión de datos. Por un lado, la aplicación específica determina el fenómeno observado y los tipos

de sensores utilizados. Por otro lado, cada aplicación implica el tipo de razonamiento que se debe realizar sobre lo medido. Este tipo de razonamiento nos conduce a la búsqueda y aplicación de las técnicas necesarias. Por ejemplo, los sistemas de ayuda en la estimación de la posición, velocidad y atributos de un objeto generalmente necesitarán técnicas numéricas como son los algoritmos de asociación y los algoritmos de estimación, ya sea para la estima de la propia posición [Fung 83] o para el seguimiento de otros objetos [Berg 83]. Otras aplicaciones pueden requerir razonamientos a más alto nivel y precisar técnicas basadas en razonamiento o sistemas expertos, como las encargadas de inferir el estado de la batalla en aplicaciones militares como VIDS, que utiliza un sistema experto basado en reglas, o IPS, que utiliza guiones para representar el conocimiento (estas y otras aplicaciones de fusión de datos militares pueden encontrarse en la revisión realizada en [Hall 92]); y otras aplicaciones no militares como MYCIN, sistema médico basado en factores de certeza [Shortliffe 75], o PROSPECTOR, sistema experto bayesiano para prospección geológica [Duda 79].

Otros factores que influyen en el desarrollo del sistema de fusión son las decisiones a tomar en tiempo real, el número de inferencias que se realizan por unidad de tiempo, la velocidad de llegada de los datos, el número y tipo de sensores, etc. Estas restricciones, como ocurre en el control de un móvil en tiempo real, nos inducen a utilizar ciertas técnicas frente a otras que necesitan mayor tiempo de procesamiento. Una evaluación sistemática del funcionamiento en tiempo real de sistemas de integración multisensorial en robótica puede encontrarse en [Zheng 89].

Finalmente, la capacidad de los sensores, las restricciones en la comunicación y el sistema de procesamiento o fusión también afectan a la elección de la arquitectura escogida. *En nuestro caso se persigue que los datos transmitidos entre agentes sea el mínimo indispensable.*

Todos los factores enumerados anteriormente, como diversidad de aplicaciones, necesidades de tiempo real, sensibilidad de los sensores, etc., implican un amplio abanico de estructuras de control para manejar esa disparidad de fuentes de información; cada una aplica una técnica distinta de integración de los datos sensados. Las arquitecturas que se utilizan poseen muchos puntos en común, pero en la mayoría de los casos no son reutilizadas para otras aplicaciones, fundamentalmente debido a su fuerte dependencia de los sensores empleados.

Por tanto es conveniente revisar qué tipos de procesos de fusión podemos distinguir, qué clase de información se puede fusionar, cómo puede ser tratada y qué arquitectura de fusión y control puede utilizarse. Así, en la sección 1.1.1 se describen dos clasificaciones de los procesos de fusión: a nivel del tratamiento y del uso que se hace de la implementación e inferencias obtenidas; y a nivel del tipo de información fusionada o recibida. En la sección 1.1.2 se describen las arquitecturas que permiten tratar y fusionar los datos procedentes de los distintos sensores y los distintos algoritmos utilizados para realizar fusión de datos.

A continuación, en la sección 1.2 se expone una pequeña muestra de la gran variedad de aplicaciones existentes, aplicaciones industriales, militares, espaciales, etc. que utilizan integración y/o fusión multisensorial. De esta forma se pone de manifiesto la importancia del uso de distintos sensores en cualquier sistema que deba interactuar con (o desenvolverse en) su entorno. Estas aplicaciones muestran la cantidad de soluciones propuestas dependiendo de las necesidades de cada caso.

Por último, la sección 1.3 enumera distintos robot móviles terrestres y cómo realizan la fusión e integración multisensorial, que es el tema central de esta Memoria. En la relación de vehículos no se ha buscado una completa enumeración de todos los existentes, sino una muestra de la evolución que se ha seguido, los sistemas utilizados y las distintas configuraciones sensoriales elegidas.

1.1.1. Clasificación de los procesos de fusión

La multitud de métodos de fusión y la variedad de aplicaciones existentes dan lugar a una gran riqueza de sistemas de fusión. Éstos se orientan según la aplicación, los datos medidos o el tipo de tratamiento de los datos. Así, para comprender mejor qué es la fusión de datos y cómo puede ser aplicada en cada caso es conveniente revisar qué tipos de procesos podemos distinguir y qué tipo de información se puede fusionar.

Existen multitud de clasificaciones de los procesos de fusión dependiendo del aspecto que se considere, en nuestro caso destacamos dos clasificaciones diferentes de los procesos de fusión, véase figura 1-1: (1) a nivel del tratamiento y uso que se hace de la implementación e inferencias obtenidas; y (2) a nivel del tipo de información fusionada o recibida.

Estas dos clasificaciones nos mostrarán aspectos importantes que serán tratados en capítulos posteriores.

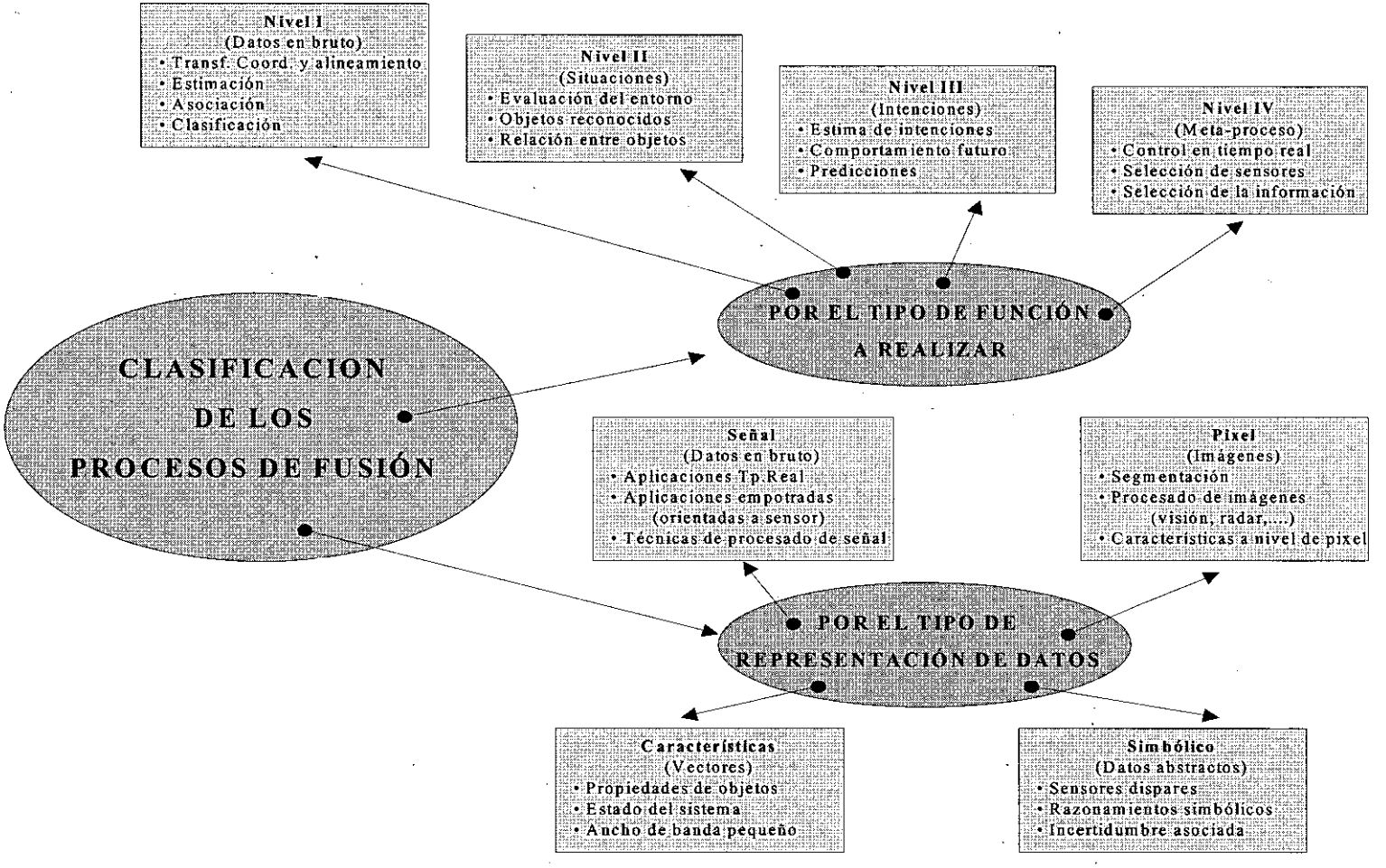


Figura 1-1 . Clasificación de los procesos de fusión de datos

1.1.1.1 A nivel funcional

La división de los procesos de fusión desde el punto de vista de su funcionalidad fue realizada por primera vez por la comisión Data Fusion de la Joint Directors Laboratories (JDL). Esta comisión se constituyó en 1986 e integraba a los principales grupos de trabajo en fusión de datos. Su finalidad era unificar la terminología existente referente a fusión y proporcionar un marco de trabajo común [White 91]. Como resultado de este trabajo se obtuvieron dos modelos diferentes, dependiendo desde qué punto de vista se estudiase: el primer modelo [White 88] identificaba tres niveles de fusión según los productos procesados. El segundo modelo [Kessler 92] hacía referencia a la arquitectura necesaria para realizar la fusión de datos.

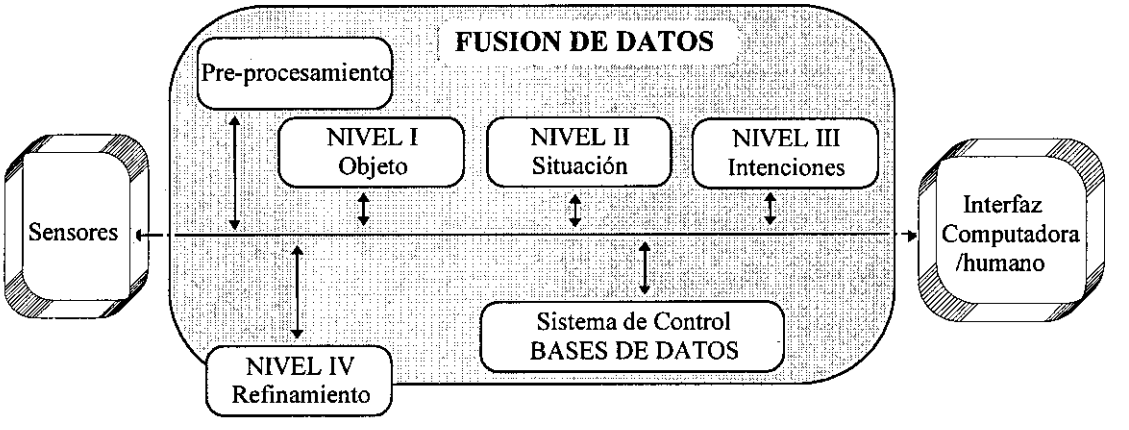


Figura 1-2. Modelo del proceso de fusión orientado a niveles.

El primer modelo, que distingue tres niveles según el procesamiento realizado sobre los datos, es ampliado por Hall [Hall, 97] con un cuarto nivel. Este modelo, como puede verse en la figura 1-2, está orientado a la funcionalidad de los procesos y además de los cuatro niveles conviene destacar: un módulo de entrada, que representa los sensores del sistema; y otro módulo de salida, el interfaz computadora/humano encargado de comunicar los resultados del proceso (alarmas, resultados, ...) y de recibir órdenes, informes de operadores humanos, inferencias y probabilidades, etc. Los niveles funcionales del sistema son,

Nivel I:

Es el nivel encargado de aceptar los datos de los sensores y fusionar dichos datos para obtener una estima de la localización e identidad de los objetos o

entidades que afectan al sistema. Por ejemplo, en aplicaciones militares representaría una base de datos del estado de la batalla: localización, identificación y características de los objetos observados (tanques, barcos, radares, etc.). En nuestro caso, un robot móvil, representaría el mapa o una base de datos del entorno que rodea al móvil: localización de los obstáculos, identificación y localización de objetos como puertas, balizas naturales, etc. Este nivel realiza cuatro funciones principales:

(1) Transformación de los datos del sensor en coordenadas y unidades consistentes.

(2) Extensión de las estimas temporales de la posición o atributos de un objeto.

(3) Aplicación de técnicas de estimación estadísticas.

(4) Clasificación o estimación de la identidad de un objeto.

En general, para cualquier aplicación, este nivel se ocupa de la identificación y localización de los fenómenos de interés.

Nivel II:

Actúa sobre los resultados del Nivel I para realizar la evaluación de la situación, desarrollando una descripción de la relación entre objetos y sucesos en el propio entorno donde se encuentran. Las inferencias obtenidas se basan en análisis espaciales, temporales y posicionales del Nivel I. Para poder interpretar correctamente los datos, estos análisis se realizan teniendo en cuenta el contexto en el que se han recibido, de igual forma que lo haría un experto humano.

Nivel III:

Este nivel se encuentra por encima de los anteriores y se ocupa de realizar un análisis del conjunto de la situación y de realizar inferencias y predicciones sobre el sistema y los objetos observados. En sistemas militares se ocupa de estudiar la peligrosidad de la situación, estimando la vulnerabilidad de amigos y enemigos, las intenciones del enemigo, etc. Este nivel es bastante complejo puesto que no sólo trata con los datos que recibe de los niveles inferiores, sino que intenta predecir futuros comportamientos, para lo cual necesita disponer de un cierto conocimiento, normalmente adquirido mediante aprendizaje y contenido en grandes bases de datos, como política de entorno, situación actual, doctrina del enemigo, etc. En este nivel son útiles las técnicas correspondientes a la teoría de juegos. En aplicaciones no militares este nivel se ocupa de identificar potenciales

fallos mecánicos de un sistema o, en nuestro caso, detectar intenciones de un objeto en movimiento o realizar predicciones sobre el camino más seguro en función de hipótesis sobre las zonas que tendrán menos congestión en el futuro.

Nivel IV:

Este nivel es considerado como un meta-proceso. Supervisa la actuación de los demás procesos y tiene las siguientes funciones:

- Supervisar el proceso de fusión de datos para proporcionar información sobre control en tiempo real y funcionamiento a largo plazo.
- Determinar qué información es necesaria para mejorar el resultado de la fusión en los múltiples niveles (identidades, posiciones, inferencias, ...).
- Determinar las necesidades de cada sensor para obtener la máxima información (tipo de sensor, base de datos, etc.).
- Localizar y dirigir a los sensores adecuados para realizar cada misión específica.

Se debe entender que este modelo es genérico y que la clasificación en los Niveles 1-4 es artificial y fue desarrollada fundamentalmente para obtener un léxico y una terminología consistente sobre fusión de datos. *El desarrollo de esta tesis se basa en encontrar una arquitectura y métodos de fusión aplicables al nivel I, que corresponde al nivel básico de un robot móvil autónomo terrestre.* El diseño será lo más versátil posible para que permita, en posibles ampliaciones del sistema, la adición de los niveles posteriores como capas superiores sobre la arquitectura propuesta.

1.1.1.2 A nivel de representación

Cuando se realiza fusión, los datos pueden provenir de distintos sensores en un mismo instante de tiempo o de un único sensor a lo largo de un cierto periodo. La fusión realizada con estos datos puede clasificarse [Luo 92], según el tipo de información recibida, en cuatro niveles: de señal, de pixel, de características o simbólico.

Cada uno de estos niveles también se caracterizará por cómo es modelada la información y por los métodos de fusión que utiliza. La mayoría de los sensores proporcionan datos que pueden fusionarse en uno o más de estos niveles. La clave

está en encontrar un método que permita realizar de forma sencilla la fusión de estos tipos de datos.

Nivel de señal

Se utiliza en aplicaciones en tiempo real y puede verse como un superconjunto de las técnicas de procesamiento de señales, que combinan las señales de un grupo de sensores para proporcionar una señal del mismo tipo pero más precisa o de mejor calidad. Este tipo se señales deben estar sincronizadas o en su defecto se deberán estimar sus valores en algunos instantes determinados de tiempo.

Los métodos de fusión a nivel de señal requieren una serie de suposiciones. Las más comunes son la adición de un término de ruido gaussiano de media cero y la independencia estadística entre los errores en las medidas de cada sensor. Una introducción a los problemas de cualquier método de fusión a nivel de señal basado en estas suposiciones puede encontrarse en [Richardson 88], dondè se demuestra que, en la mayoría de los casos, en los métodos basados en estimación óptima la inclusión de información redundante mejora la estima final. En este nivel se pueden destacar distintos métodos de optimización, filtros de Kalman, etc.

Nivel de pixel

La fusión a este nivel permite mejorar técnicas de procesado de imágenes como segmentación o extracción de características, debido al aumento de la información asociada a cada pixel al combinar multiples imágenes. La fusión puede tener lugar en distintos niveles de representación: fusión de señales en bruto de multiples sensores antes de su correspondencia con un pixel específico; la fusión de los pixeles correspondientes en cada una de las imágenes obtenidas por los sensores para formar una imagen fusionada; , y, el uso de los pixeles correspondientes en las distintas imágenes para realizar la segmentación o la extracción de características.

La fusión a nivel de pixel es útil en términos de funcionamiento global del sistema puesto que utiliza funciones que tienen unos requerimientos de calculo muy fuertes que normalmente impide su uso en tiempo real. Para facilitar dicho cálculo se suelen utilizar imágenes de un único sensor o de sensores que sean lo más semejantes posibles (ej. visión estereoscópica). Los métodos utilizados para fusión deben poseer facilidades para procesamiento paralelo (o al menos en grupos

de pixeles locales) y además el método escogido debe ser fácilmente adaptable para procesar una amplia variedad de imágenes de diferentes sensores.

Nivel de características

La fusión a nivel de características se utiliza para obtener una mayor certeza de que la característica extraída de la información del sensor corresponde a un aspecto importante del entorno, o para crear nuevas características asociadas que serán usadas por el sistema.

Características típicas extraídas de una imagen y fusionadas entre sí son esquinas y regiones de similar intensidad o profundidad; aspectos importantes son su forma geométrica, posición, orientación y tiempo. De esta forma características que están presentes repetidamente se fusionan y por ello su precisión aumenta, y por el contrario aquellas que no son confirmadas nuevamente se consideran erróneas y son eliminadas. El ancho de banda de comunicación que se requiere a este nivel es mucho menor que el de los niveles anteriores.

Nivel simbólico

Este nivel permite la fusión de información de alto nivel de abstracción de diferentes sensores. Se utiliza principalmente cuando se fusionan sensores muy distintos o que se refieren a regiones diferentes del entorno. Para ello se utilizan métodos de razonamiento simbólico que hacen uso de información a priori procedente de un modelo propio del mundo o de fuentes externas al sistema. Las características fusionadas normalmente tienen una medida de incertidumbre para representar tanto el grado de concordancia como cualquier incertidumbre inherente en la información proporcionada por los sensores. Esta incertidumbre se utiliza para indicar el peso relativo que un dato particular tiene en el proceso de fusión. En inferencia estadística, la incertidumbre de los símbolos se representa por probabilidades y su fusión se realiza como probabilidades condicionales, que aumentarán o disminuirán mediante el proceso de fusión. El tiempo, en este nivel, no es tan importante como en los niveles anteriores puesto que los símbolos fusionados no suelen variar en función de éste.

Por supuesto, la frontera entre estos niveles no está tan clara en sistemas de fusión reales. En esta memoria *se desarrollará un método de fusión a nivel de señal/pixel para las medidas directas de los sensores y un método de fusión a nivel de características/simbólico para aquellos sensores que realizan ciertos preprocesamientos e inferencias propias.*

1.1.2 Arquitecturas de fusión

La elección correcta de la arquitectura de fusión es fundamental para el desarrollo de cualquier sistema. La cuestión fundamental es dónde combinar los datos procedentes de dos o más sensores. Esta elección afectará a la calidad del producto fusionado, a la naturaleza de los algoritmos, a las técnicas utilizadas, a la complejidad del procesamiento lógico y al ancho de banda de las comunicaciones entre el sensor y el sistema de fusión. La elección de la arquitectura no es arbitraria, dependerá tanto de la naturaleza de los sensores como de las inferencias necesarias.

Un modelo que muestra la arquitectura de fusión de los tres niveles anteriores puede verse en la figura 1-3 [Waltz 90]. La principal característica que podemos destacar, y que también se busca en nuestra arquitectura, es su naturaleza puramente distribuida. También se puede destacar: (1) la distinción en la fusión de datos del nivel I entre sensores similares y distintos; (2) la importancia que da a la utilización de bases de datos en todos los niveles; y (3), que los procesos son multinivel, con realimentaciones entre niveles y pueden estar distribuidos geográficamente.

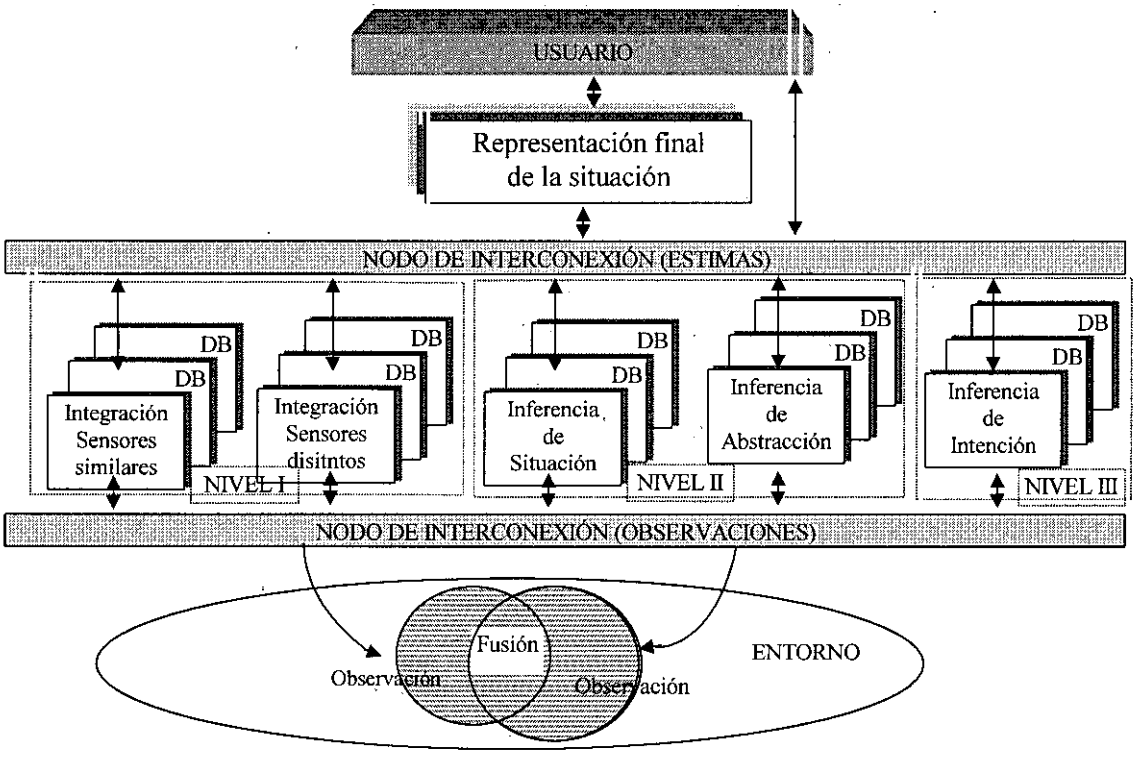


Figura 1-3. Modelo del proceso de fusión orientado a su arquitectura.

Centrándonos en el Nivel I existen tres tipos de arquitecturas [Hall 97] que se diferencian según la información fusionada y las técnicas de fusión aplicadas. Dichos tipos son: centralizada, autónoma e híbrida.

1.1.2.1. Arquitectura de fusión centralizada

En este modelo se fusionan datos en bruto provenientes de múltiples sensores para realizar una estima de la posición, velocidad, atributos o identidad de un objeto concreto, véase la figura 1-4. En este caso sensores similares observan la misma entidad. Se entiende por sensores similares a sensores del mismo tipo, como por ejemplo [Jain 95] donde se fusionan datos de diferentes cámaras que observan la misma escena; o también a sensores cuyos datos en bruto pueden ser directamente fusionados, por ejemplo [Shetty 96] donde la información de una imagen de un radar 3-D y de una imagen visual se fusionan directamente para la detección de maniobras en un sistema de seguimiento.

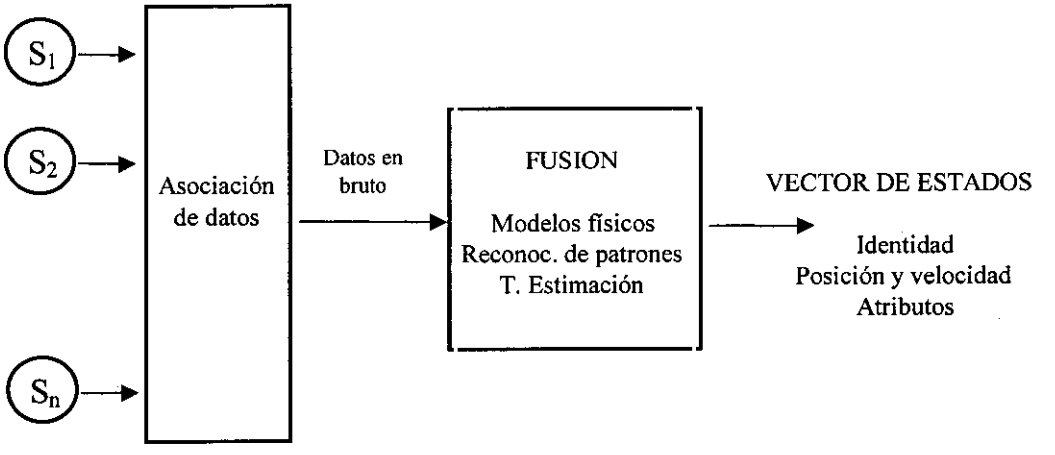


Figura 1-4. Fusión centralizada

Este tipo de arquitectura tiene la ventaja de que existe una mínima perdida de información, puesto que la fusión se realiza directamente a partir de los datos observados por los sensores. Pero también tiene serios inconvenientes, el principal es que sólo es aplicable a sensores similares, lo cual es una seria restricción en un sistema general. Incluso aunque los sensores sean idénticos, en muchos casos se presentan dificultades para su fusión cuando se encuentran separados geográficamente. Además, en esta aproximación es necesaria una gran cantidad de comunicación, fuertes necesidades de cálculo y complejos modelos físicos.

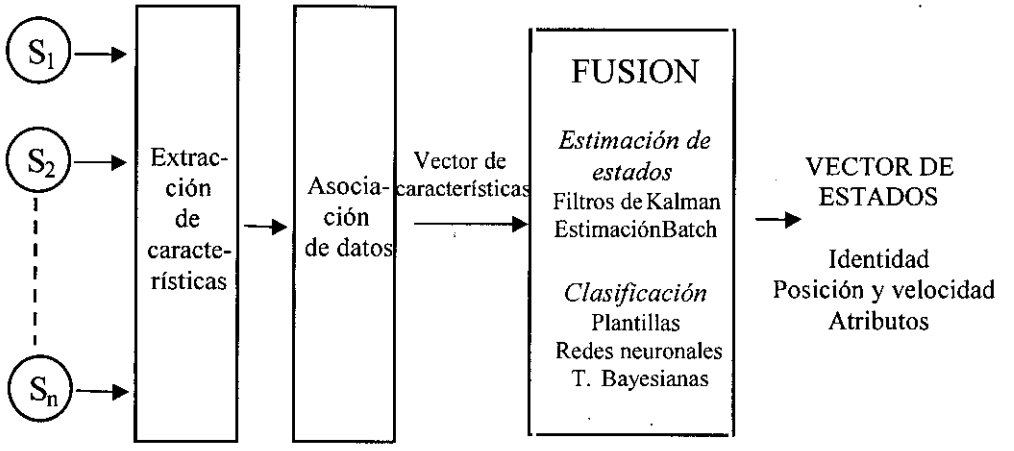


Figura 1-5. Fusión centralizada con vector de características

Una alternativa, a esta arquitectura, es permitir que exista un preprocesamiento en cada sensor, de forma que proporcione un vector de características, figura 1-5. Éstas son transmitidas a un sistema de fusión central, donde se realiza la asociación y la consiguiente fusión. Esta aproximación implica una pérdida de información procedente de los sensores, puesto que los datos que observa cada sensor sólo vienen representados por un vector de características. Por ejemplo [Jain 95] donde se fusionan medidas de diferentes cámaras que observan la misma escena, y muestra que el error disminuye a medida que aumenta el número de cámaras (o el número de *frames*).

El uso del vector de características simplifica el proceso de asociación de datos, reduce las necesidades de comunicación entre los sensores y el proceso de fusión y, lo que más nos interesa, permite que sensores que no son similares puedan fusionar sus datos. Esto nos muestra la necesidad de que los sensores sean capaces de realizar un preprocesamiento inicial de los datos antes de ser fusionados.

1.1.2.2. Arquitectura de fusión autónoma

En esta arquitectura, véase la figura 1-6, los datos de los sensores vienen representados por un vector de atributos o de características. Cada sensor realiza un procesamiento sobre los datos para determinar un vector de estado o determinar la identidad de un objeto. De este modo, la salida de cada sensor es una decisión (una declaración de identidad o la estima de un vector de estado). Estas decisiones son la entrada a un proceso de fusión, el cual realiza la asociación y la fusión de identidad o la estima del estado global. Claro está que esta

arquitectura representa mayor pérdida de información que la anterior, pero permite que datos procedentes de sensores completamente distintos sean fácilmente comparables. Es necesario tener en cuenta, como dificultad añadida en este tipo de sistemas, que ahora los datos no son condicionalmente independientes.

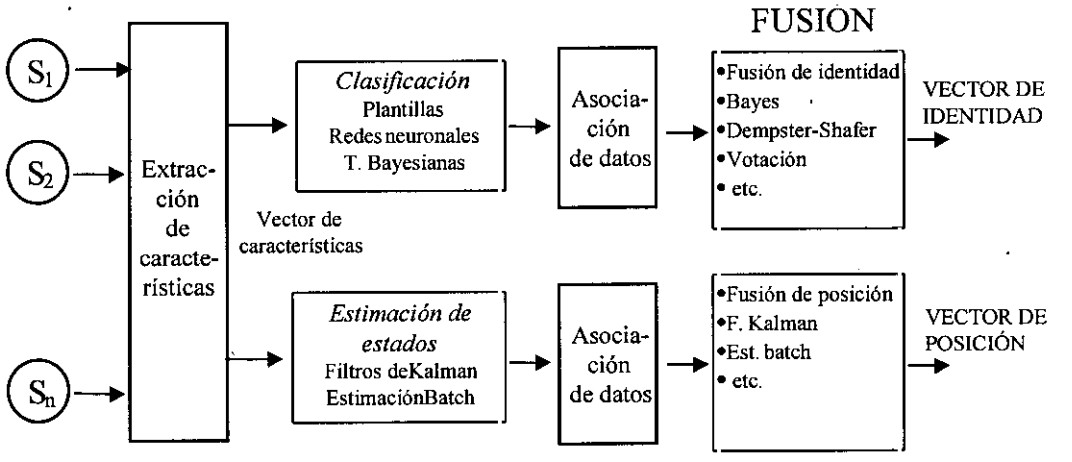


Figura 1-6. Fusión autónoma

1.1.2.3. Arquitectura de fusión híbrida

Combina la arquitectura centralizada y autónoma. En esta arquitectura, todo tipo de datos (datos en bruto, vectores de características y datos de decisión) son entradas para el proceso de fusión. En este caso, al englobar ambos métodos, se necesita una gran capacidad de cálculo, mucho mayor que en los casos anteriores. Esto es, cálculo y procesamiento de datos en bruto y cálculo y razonamientos sobre estimas y vectores de características. En este sistema las necesidades de comunicación entre los sensores y el sistema de fusión se incrementa enormemente. Sin embargo, al combinar tanto datos en bruto como vectores de características se aprovecha de las ventajas de ambos obteniendo una fusión de datos óptima.

Un sistema general que permite la utilización de cualquier tipo de sensor de que disponga el usuario, lleva a diseñar una arquitectura híbrida, pero debido a que los requerimientos en tiempo real son muy fuertes será conveniente reducir en lo posible los datos en bruto, por tanto *en nuestro caso se tenderá a aproximarse más a una arquitectura de fusión autónoma.*

1.1.3. Algoritmos de fusión de datos

La selección de algoritmos para la fusión de datos procedentes de sensores similares o distintos se complica enormemente cuando se pretenden combinar datos erróneos o incompletos pertenecientes a un entorno complejo en tiempo real. No existe ningún "manual" de análisis de métodos de fusión, aunque sí multitud de técnicas que pueden aplicarse. Una revisión detallada de los métodos matemáticos utilizados en fusión de datos puede encontrarse en [Hall 92].

La selección de las técnicas de fusión de datos depende de la implementación específica y del sistema donde es usada. Esto implica tener una amplia perspectiva del sistema completo y la consideración simultánea de distintos puntos de vista: requerimientos del sistema, restricciones del usuario, algoritmo de diseño, métodos numéricos y estadísticos, interfaz hombre/máquina, integridad del sistema, ... Pero siempre se deben tener en cuenta los siguientes puntos:

1.-*Máxima eficacia:* los algoritmos escogidos deben ser capaces de realizar correctas inferencias en presencia de datos erróneos o con incertidumbre. Estos algoritmos deben poder tratar la realidad correctamente aunque la información a priori sobre las funciones de densidad de probabilidad o la información estadística de los datos no estén disponibles.

2.-*Restricciones operativas:* la selección del algoritmo debe considerar restricciones operativas tanto para el procesado automático de datos (necesidades de tiempo real, entorno de trabajo,...) como en las herramientas para ayudar al análisis de la fusión (p. e. un interfaz adecuado con el usuario).

3.-*Recursos eficientes:* los algoritmos seleccionados necesitan minimizar el uso de los recursos de la computadora, como por ejemplo el tiempo de CPU y la entrada/salida.

4.-*Flexibilidad operacional:* en la evaluación de los algoritmos es interesante tener en cuenta su flexibilidad. Es decir, permitir introducir modificaciones en el sistema tanto por parte del usuario como por parte del diseñador. Por ejemplo para adaptarlo a la recepción de otro tipo de datos, como en el caso de un sistema con inferencias de orden superior, donde debería preverse que podría

ser necesario cambiar los datos a priori (por ejemplo en las plantillas).

5.-*Crecimiento funcional:* El diseño del flujo de datos y la elección de los algoritmos deben permitir el crecimiento de su funcionalidad a medida que el sistema lo necesite. Por ejemplo, se diseña un sistema que fusiona datos para reconocer unos objetos determinados y en un futuro se puede desear ampliar su capacidad de reconocimiento e inferencia sobre ellos.

Algoritmos y técnicas	Alineación datos			Correlación datos/objetos			Estimación posición/velocidad				Estimación identidad			
	E	T	U	S	R	A	P	AE	I	O	M	C	D	I
Transf. De coordenadas	X	X	X	X			X							
Modelos sensores							X							
Modelos físicos							X	X		X	X		X	
Medidas de asociación				X	X	X					X	X		
Ecuaciones de movimiento						X						X	X	
Métodos de optimización								X					X	
Filtros de Kalman							X	X						
Error de covarianza									X					X
Inferencia bayesiana						X							X	X
Dempster-Shafer						X							X	X
Votación						X							X	
Reconocimiento patrones					X	X						X	X	
Plantillas				X	X	X				X			X	
Sistemas expertos									X	X			X	X
Conjunto fuzzy									X				X	X

Tabla 1.1. Técnicas de fusión de datos y su aplicación (extraída de [Hall 92])

E: Espaciales T: Temporales U: Unidades S: Proyección R: Correlación
A: Asignación C: Comparación P: Predecir observación AE: Actualizar estados
I: Incertidumbres O: Manejo de objetos M: Medidas de identidad D: Declaración identidad

Estos cinco puntos deben ser tenidos siempre en cuenta a la hora de seleccionar algoritmos de fusión de datos, pero más aun en el desarrollo de nuestro sistema, puesto que *se busca que permita el uso de distintas fuentes de información (sensores), que sea aplicable a diversas situaciones y que su funcionalidad sea ampliable por el usuario.*

Debido a la amplia variedad de aplicaciones de fusión y de algoritmos aplicables, es imposible proporcionar unas directrices sobre qué técnicas son

aplicables en la fusión específica de unos datos. Aun así, en la tabla 1.1 (extraída de [Hall 92]) se describen distintas técnicas y se muestra su aplicación.

Todos estos algoritmos y técnicas de fusión de datos pueden utilizarse como algoritmos de fusión multisensorial y pueden clasificarse dentro de una de estas dos grandes teorías [Sorenson 70]: *teoría de la detección y teoría de la estimación*. A pesar de que estos algoritmos están sujetos a un gran desarrollo matemático, cuando son aplicados a ejemplos concretos constituyen una solución particular al problema general de la fusión multisensorial.

- **Decisión o teoría de la detección.** En este caso las medidas son comparadas con hipótesis alternativas para decidir cual de ellas las describe "mejor"; por tanto nos indicará la situación de nuestro sistema. El método de Bayes proporciona una forma de actualizar el grado de creencia que se tiene (expresado como una probabilidad) sobre una hipótesis basándose en el conocimiento a priori y en las observaciones recientes.

  La teoría de la decisión, y en concreto la teoría bayesiana, es aplicable directamente al procesamiento de datos de múltiples sensores para realizar decisiones binarias (ej: en el problema de la detección, observar si una señal está presente o ausente) o decisiones m-arias (ej: en el problema de clasificación, identificar en cual de las m clases de hipótesis encajan mejor las medidas). Pero pueden utilizarse otras técnicas como la teoría de Dempster-Shafer, máxima verosimilitud, factores de certeza, etc.

- **Teoría de la estimación**. Es una extensión al proceso de decisión por el cual la estimación de un parámetro se hace utilizando múltiples medidas observadas de variables que pueden estar directamente relacionadas con el parámetro en cuestión. Como ejemplos de la teoría de la estimación se pueden enumerar, los mínimos cuadrados de Gauss (1795), el método de máxima verosimilitud de Fisher (1912), la estimación estadística de Kolmogorov y Weiner (de la década de los cuarenta) sobre series de medidas en tiempo continuo y discreto o el estimador de estados de Kalman (1960), entre otros.

  En todos los casos se supone:

- Que se puede desarrollar un modelo lineal que relacione las medidas con el parámetro en cuestión,
- que existe un conocimiento estadístico de los errores de las medidas y, en algunos casos, de su comportamiento,
- que se conoce la fuente de las medidas a priori.

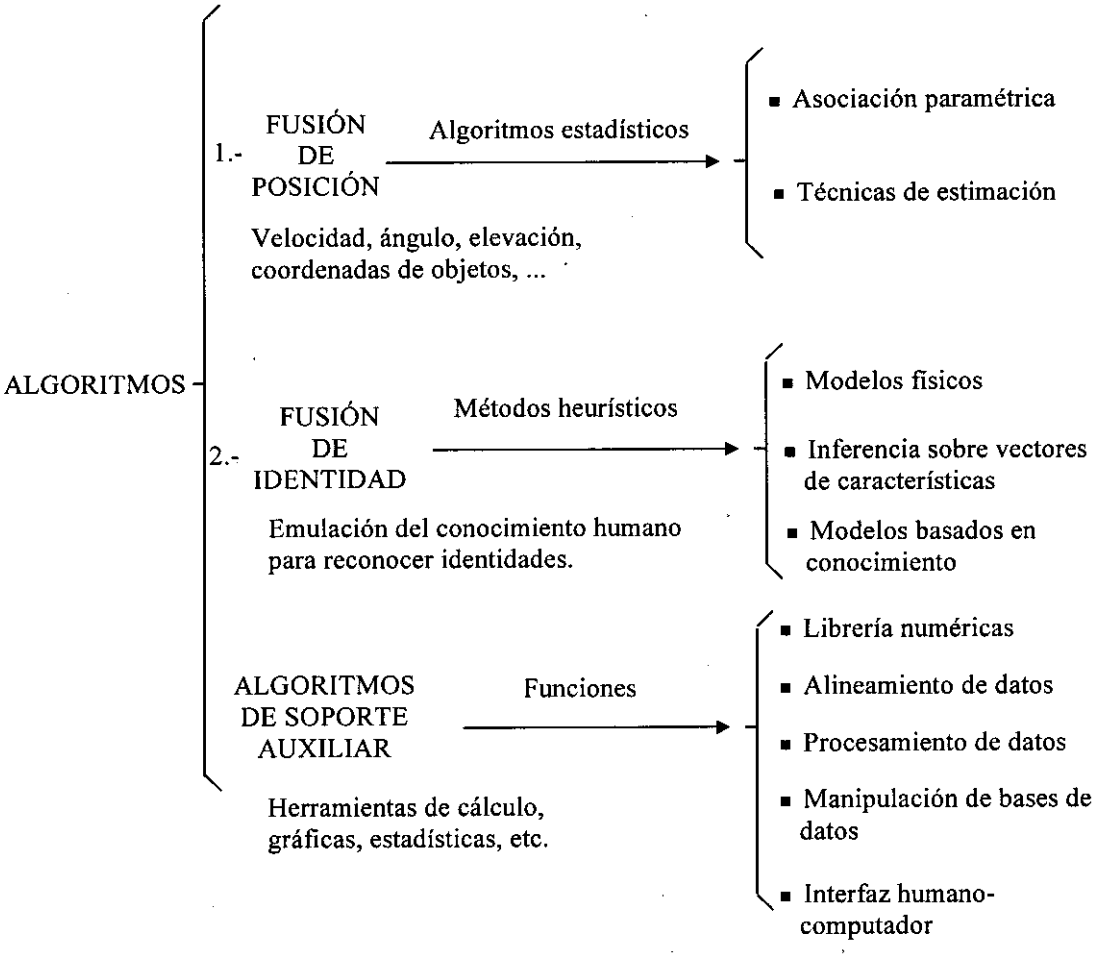


Figura 1-7. Categorías en la fusión multisensorial.

Estas dos teorías nos lleva a clasificar los algoritmos de fusión de datos en dos grandes categorías: fusión de datos de posición y fusión de datos identidad. En la figura 1-7 se ha incluido una subcategoría correspondiente a todas aquellas herramientas auxiliares imprescindibles en la utilización de los algoritmos de fusión como por ejemplo las librerías numéricas o las herramientas para manejo de bases

de datos. *Esta clasificación nos determinará la definición de nuestro modelo de sensor lógico* que se expondrá en el capítulo 2.

1..1.3.1. Algoritmos de fusión de datos de posición

Los algoritmos de fusión de posición, o estimación de la posición, son conocidos desde antaño y como se ha indicado anteriormente abarcan desde los mínimos cuadrados de Gauss de finales del siglo XVIII hasta el filtro de Kalman.

Estas técnicas de fusión de datos utilizan modelos físicos y suposiciones estadísticas sobre los ruidos de las observaciones para representar los datos observados en un vector de estado. El algoritmo de fusión proporciona una estima del vector de estado que mejor se ajusta (en cierto sentido matemático) a los datos observados. Para conseguir esta estima, estos algoritmos realizan correcciones o refinamientos sucesivos del vector de estado por medio de la diferencia entre los datos actuales (observados) y las observaciones predichas (basadas en el modelo, a partir del estado y la observación actuales). En múltiples sensores, puede utilizarse un modelo físico distinto para cada sensor.

En estos sistemas, los datos medidos por los sensores primero deben ser alineados (transformación de los datos a las unidades y al sistema de referencia adecuados), y a continuación se realizan otras dos operaciones sobre ellos:

- *Asociación paramétrica.* Se encarga de asociar unas observaciones con otras, de forma que indique si esa observación es correcta, con qué otros datos observados debe ser fusionada o qué elemento ya reconocido anteriormente le corresponde. Para ello es necesario definir una medida de asociación que cuantifique la proximidad entre observaciones, aunque por supuesto no tiene por qué ser métrica. Las medidas de asociación suelen incluir coeficientes de correlación, medidas de distancia, coeficientes de asociación o medidas probabilísticas de semejanza [Hall, 92; cap. 3].
- *Estimación del vector de estados.* Una vez que la asociación de datos ha sido llevada a cabo (los datos se han agrupado según pertenecen a una entidad u objeto), se puede proceder a la estimación. Por estimación se entiende la búsqueda del valor del vector de estado (posición, velocidad, parámetro o coeficientes del modelo, ...) que mejor se ajuste, en un cierto sentido matemático, a los datos observados. En la práctica la asociación y la estimación

están relacionadas para conseguir un mejor funcionamiento del sistema. Ejemplos de problemas de estimación pueden ser:

- Uso de datos de posición para determinar la localización de un objeto estacionario (posicionamiento o localización).
- Combinación de datos de posición de múltiples sensores para determinar la posición y velocidad de un objeto en función del tiempo (seguimiento).
- Estima de las características de una entidad (figura, forma, etc.) a partir de los datos observados (identificación).
- Estima de los parámetros de un modelo, que representan o describen los datos observados.

Una vez tomado un cierto vector de estado y todas las relaciones de éste con las medidas y el sistema, es necesario elegir un cierto criterio de optimización con respecto al cual estimar el mejor ajuste con los datos. Los criterios de optimización más utilizados están basados en mínimos cuadrados, funciones de máxima verosimilitud o formulaciones bayesianas. Para encontrar el vector de estados que minimiza la función escogida se siguen dos estrategias:

- Modo batch, se obtienen todos los datos simultáneamente o se espera hasta que se hallan obtenido todos los datos necesarios y entonces se realiza la estimación del estado del sistema. Un ejemplo de estas técnicas es el estimador de mínimos cuadrados [Lewis 86; Bar-Shalom 93].
- Modo secuencial, la modificación en la estima del vector de estado se realiza con cada nueva observación. Un ejemplo de este tipo de técnicas son los mínimos cuadrados recursivos, el filtro alfa-beta [Kalata, 84] o el filtro de Kalman [Gelb 74; Anderson 79].

El uso de una u otra estrategia dependerá de la aplicación escogida, de las necesidades de tiempo y del número de datos de que se disponga en cada instante.

1.1.3.2. Algoritmos de fusión de identidad

En este caso el problema consiste en combinar la declaración de identidad de datos de múltiples sensores para estimar con mayor certeza la identidad de un objeto. La idea es obtener a partir de la combinación de declaraciones de los

distintos sensores una estima más precisa y clara que las de los sensores individuales.

De los datos obtenidos por los sensores, infrarrojos, espectro visible, imágenes, etc. se extrae un vector de características que represente los datos observados. Ejemplos de transformaciones para extraer características son algoritmos de procesamiento de imágenes, reconocimiento de patrones (por ejemplo esquinas o patrones artificiales fijos), etc., aunque pueden existir sensores que proporcionen directamente declaraciones de identidad.

Al igual que en la fusión de posición, es necesario hacer un preprocesamiento de los datos y realizar una asociación para clasificar las declaraciones de identidad en grupos que representan observaciones del mismo objeto.

La fusión de identidad puede tener lugar a tres niveles:

1. A nivel de los datos en bruto (previo a la extracción de características).
2. A nivel del vector de características (previo a la declaración de identidad).
3. A nivel de decisión (después de que cada sensor ha hecho una declaración independiente de identidad).

La elección de cuándo se realiza la fusión depende de los tipos de datos disponibles y del tipo de preprocesamiento realizado por los sensores.

En la práctica, la fusión de identidad se realiza simultáneamente a la de posición, pero existen dificultades que impiden realizarlo de la misma forma:

- Los modelos físicos generales para declaraciones de identidad no existen o son muy difíciles de desarrollar. Cuando se encuentra un modelo, este debe desarrollarse para cada objeto observado. La gran carga computacional que se necesita para cada modelo y los cientos de potenciales objetos que podemos encontrarnos implican la imposibilidad de aplicarlos.
- La fusión de identidad es jerárquica; en el nivel más bajo de inferencia los objetos se identifican como emisores individuales (balizas, radio, ...) o como entidades físicas (esquina, pared, ...). En niveles altos se infieren relaciones entre distintos componentes o deducción de características más intangibles como por ejemplo la

intención de ataque de un enemigo o la dirección más probable que seguirá un móvil.

Así, aunque el uso de modelos para estimar los datos observados por los sensores y deducir su identidad (mediante la comparación entre las observaciones predichas por el modelo y los datos actuales mediante técnicas de estimación clásica) es perfectamente factible, pueden ser muy complejos y la estima de la identidad muy costosa. Esto lleva a que se planteen otras formas de fusión de características como son,

1.- *Técnicas de inferencia basadas en características*: Estas técnicas no utilizan modelos físicos, sino una correspondencia directa entre los datos referentes a la identidad de los objetos y a la declaración de identidad. Se dividen en dos grandes categorías:

   a.- Técnicas paramétricas o estadísticas, que requieren suposiciones a priori sobre las propiedades estadísticas (por ejemplo, la función de distribución) de los datos referentes a la identidad. Son técnicas como la inferencia clásica [Duda 79], inferencia mediante redes bayesianas [Pearl 86], inferencia con Dempster-Shafer [Dempster 67; Shafer 76], etc.

   b.- Técnicas no paramétricas, que no requieren información a priori. Métodos de este tipo son las plantillas, redes neuronales artificiales [Minsky 69; Hopfield 82], métodos de entropía y de votación, etc.

2- *Modelos basados en conocimiento*: Estos métodos intentan imitar el proceso de inferencia del análisis humano en el reconocimiento de la identidad de un objeto. En este grupo se encuentran técnicas como plantillas lógicas, sistemas (expertos) basados en conocimiento y la teoria fuzzy [Zadeh 75]. De un modo u otro, estos métodos se basan en una percepción de cómo los humanos procesan la información para llegar a unas conclusiones sobre la identidad de los objetos.

1.1.3.3. Algoritmos de soporte auxiliar

Por último indicar que en los niveles descritos anteriormente se necesitan gran número de funciones como librerías de métodos numéricos, algoritmos de alineamiento de datos, preprocesamiento de datos, manipulación de bases de

datos, rutinas de presentación de datos y resultados, etc. En un sistema de fusión de datos estas funciones pueden superar el 80 % del software utilizado [Waltz 90].

Resaltar que la función auxiliar más compleja e imprescindible en todo proceso de fusión de datos es la encargada de acceder a las bases de datos del sistema. El control de una base de datos incluye funciones de acceso y manipulación de las bases de datos, protección de datos, compresión, consultas relacionales, etc. Este control es particularmente complejo debido a la gran variedad de datos manejados (p. e. imágenes, vectores o datos individuales) y a las altas velocidades necesarias tanto para acceso como para archivado de datos procedentes de un cierto sensor. Aunque éstas y otras funciones auxiliares son imprescindibles en cualquier sistema y su utilidad es indiscutible no nos detendremos más en ellas.

1.2. Aplicaciones que realizan fusión e integración multisensorial

Existen multitud de aplicaciones donde cada día se utilizan un mayor número de sistemas robóticos que trabajan de forma autónoma o semi-autónoma (con ayuda de un operador humano). Pero cada vez hay más aplicaciones (industria, agricultura, exploraciones submarinas o espaciales, etc.) donde se requieren sistemas completamente autónomos. Esto implica dotar a estos robots de cierta inteligencia y capacidad para desenvolverse en entornos dinámicos y complejos. Es decir, deben dotarse de capacidades esenciales como percepción, conocimiento, razonamiento, aprendizaje y acción. En [Azouaoui 98] puede encontrarse una revisión de distintos sistemas robóticos autónomos y cómo se hace hincapié en la inteligencia y reactividad de estos sistemas.

El funcionamiento de los robots autónomos puede entenderse como agentes inteligentes [Russell 95], que perciben su entorno mediante sensores y responden o actúan sobre ese entorno mediante actuadores. La conexión entre sensorización y actuación puede ser un sencillo procesamiento de señal o puede involucrar toma de decisiones complejas, interpretación de metas u otros aspectos de razonamiento complejos. Pero sea cual sea el tipo de razonamiento empleado, el sistema siempre emprenderá aquellas acciones que favorezcan la obtención del máximo rendimiento, beneficio o consecución de un objetivo basándose en las evidencias aportadas por la percepción realizada y por su conocimiento incorporado.

Por lo tanto, un mayor desarrollo de los sistemas sensoriales de los robots utilizados en las distintas aplicaciones (dotándolos de múltiples sensores redundantes o complementarios) llevará a un incremento en la autonomía, seguridad, calidad y versatilidad de sus operaciones. Esto explica el mayor esfuerzo realizado actualmente en la investigación dirigida al desarrollo de mejores sistemas de control y percepción. A continuación se enumeran distintas aplicaciones donde los robot empleados utilizan múltiples sensores.

1.2.1 Industriales

En muchas aplicaciones industriales [Kafrissen 84; Groover 89] es necesario dotar al robot de múltiples sensores para aumentar su capacidad sensorial, dotándole de mayores prestaciones, para conseguir un funcionamiento más seguro, preciso y versátil. Los sensores más ampliamente utilizados en robótica industrial

son sensores táctiles, de fuerza, de proximidad y alcance, de contacto, visión artificial y sensores de medidas de ciertas variables de interés (temperatura, presión, etc.). Ejemplo de aplicaciones industriales son:

- *Manipulación de materiales*: Aquí podemos incluir los robot autónomos móviles utilizados para carga, descarga y transporte de materiales, piezas y herramientas, tanto en entorno cerrados como robots elevadores en fábricas [Espinosa 98] o transporte en entornos abiertos, como por ejemplo vehículos pesados en un puerto como [Durrant-Whyte 96]. Otros robots distintos son los brazos manipuladores o células robóticas encargados de manejar piezas en cintas transportadoras [Groover 89; cap. 11] y robots con manos y dedos semejantes a las humanas (una revisión de algoritmos sobre manos y dedos robóticos puede encontrarse en [Shimoga 96]).

- *Fabricación de componentes*: donde los robot se utilizan para fabricar piezas o elementos cuyo coste de fabricación si lo realizasen operarios humanos sería mucho mayor. Por ejemplo, se pueden encontrar robots realizando trabajos de soldadura [Hedenborn 94] o en el pulido de piezas.

- *Inspección*: aunque depende de la naturaleza de las piezas inspeccionadas, cualquier técnica de fusión e integración multisensorial puede ser aplicable. Esta puede ser:

  - explícita, es decir, como una operación separada durante o después del proceso de manufactura para comprobar la integridad de las piezas fabricadas y realizar un control de calidad [Hedenborn 95], por ejemplo la inspección de la calidad de las baldosas producidas [Fernández 98] o un sistema multisensor de inspección de soldaduras [Lacey 93].

  - implícita, cuando verifica la integridad de las piezas mientras se manipulan. Los sensores más utilizados son visión, por ejemplo [Conners 97] para inspección de la calidad de los tableros producidos en una industria maderera, sensores táctiles [Shimoga 96] y ultrasonidos.

- *Ensamblaje*: esta es una de las aplicaciones más complejas de la robótica industrial, puesto que además de las típicas necesidades de

reconocimiento de entorno, el ensamblaje requiere unos métodos de planificación especiales al existir ciertas restricciones en los grados de libertad debido a la disposición de las piezas a ensamblar y a que en muchas ocasiones requiere la colaboración de varios robots. Un ejemplo de fusión sensorial para ensamblaje puede encontrarse en [Suarez 95].

1.2.2. Militares

La integración y fusión multisensorial ha tenido siempre un gran interés y desarrollo en aplicaciones militares, como demuestra el gran número de aplicaciones existentes (véase [Waltz 92] o [Hall 92] para distintos ejemplos) y de proyectos financiados con fondos militares como los proyectos DARPA o los financiados por la NASA. Entre las aplicaciones militares existentes se pueden destacar,

- *Estado de la batalla*: [Campos 98] utilizan normalmente sistemas expertos y realizan complejos razonamientos sobre los datos obtenidos para indicar la posición del enemigo, la peligrosidad de la situación, intenciones de tropas enemigas, etc. Un ejemplo de análisis de imágenes para reconocimiento de objetos e interpretación de escenas puede encontrarse en [Nandahkumar 97].

- *Detección y seguimiento*: Los sistemas de defensa aérea y más generalmente los sistemas de control y orden C^2 (*Command and Control*) necesitan una clara representación de la situación aérea para poder tomar decisiones con seguridad. Para ello es necesario desarrollar sistemas de detección y seguimiento de múltiples aeronaves que sean seguros y fiables. Existen multitud de técnicas probabilísticas y no probabilíticas que pueden ser aplicables, véase [Bar-Shalom 90] para una descripción más amplia. Estos sistemas utilizan múltiples sensores para mejorar su resolución como son, radar, infarrojos, visión, etc.

1.2.3. Espaciales

En este grupo incluimos las aplicaciones referentes a detección y seguimiento de cuerpos en el espacio (astros) o el aire (aviones), sistemas que navegan por el globo terrestre utilizando sistemas inerciales, localización mediante satélites, GPS (Global Positioning System) u otros sensores, y también los sistemas

desarrollados para la exploración de otros planetas. La mayoría de estos sistemas podrían incluirse en mayor o menor medida en la sección anterior, pero se describen a parte para poner de manifiesto su utilización no militar.

- *Detección y seguimiento*: estos sistemas de fusión de datos suelen ser semejantes a las aplicaciones militares pero dedicadas a aplicaciones civiles como por ejemplo la detección y seguimiento de aviones para el control del tráfico aéreo en aeropuertos [Yeddanapudi 97].

- *Navegación inercial*: los sistemas de navegación inercial o INS (Inertial Navigation Systems) aprovechan las propiedades inerciales de los sensores montados sobre el vehículo (giróscopos y acelerómetros) para obtener de forma autónoma, mediante la integración de las medidas de fuerza y velocidad angular, la posición y velocidad del móvil [Britting 71]. La ventaja de este sistema es que es autónomo y no necesita fuentes externas, pero debe integrarse junto con otros sensores de posición debido al crecimiento de sus errores provocados por las integraciones realizadas. Existen multitud de sistemas que utilizan INS junto con otros sensores como por ejemplo [McMillan 88] donde se integra el INS con Transit, odómetros, brújula magnética y altímetro; o [Pham 92] que usa INS y GPS conjuntamente.

- *Exploración planetaria*: a primeros de 1970 se comenzaron a construir prototipos de vehículos autónomos para exploración planetaria equipados con distintos sensores. El Mars ROVER [Thompson 77] es el prototipo del PathFinder [Dubov 95], el primer vehículo que, en 1997 y por primera vez en la historia, realizó tareas de recogida de muestras y exploración de otro planeta: Marte. Para poder trabajar sobre terrenos agrestes en lugar de diseñar robots con ruedas (como el Pathfinder), se han desarrollado robots con patas como el AMBLER [Bares 89; Krotkov 96], de seis patas.

1.2.4. Otras

Existen muchas más aplicaciones a parte de las enunciadas anteriormente que aplican integración y/o fusión multisensorial. Entre otras podemos destacar,

- *Conducción y guiado automático*: engloba a los sistemas de ayuda a disminuidos físicos como son sillas de ruedas autónomas [Bell 94; Yanco 98; Pires 98] o perros lazarillos [Borestein 98] basados normalmente en sensores de ultrasonidos o sonar. También se incluyen aquí multitud de

sistemas de conducción de vehículos en carreteras, que se vienen desarrollado desde los primeros tiempos de la robótica móvil [Thorpe 88]. Principalmente se basan en sistemas de visión [Salgian 98]. Actualmente se están realizando muchos esfuerzos en la automatización de vehículos en autopistas [Bishop 97; Thorpe 97] como demuestra la aparición de proyectos de investigación como el PATH de California o el proyecto europeo PROMETHEUS.

- *Exploración e inspección*: se han desarrollado distintos sistemas para exploración terrestre o acceso a zonas que es peligroso o complicado para un ser humano como por ejemplo para inspección de altos hornos, para limpieza de edificios [Zong 98], quillas de buques y otros lugares inaccesibles [Balaguer 98], etc.

- *Exploración submarina*: se han desarrollado muchos sistemas robóticos submarinos, puesto que a parte de los peligros que conlleva la exploración submarina, es mucho más factible, por ejemplo, dejar a un robot durante semanas en el fondo del océano tomando muestras, tarea que a un ser humano le resultaría prácticamente imposible. La demanda de robots submarinos avanzados está creciendo enormemente en la actualidad, dando lugar a un gran desarrollo de vehículos totalmente autónomos y especializados. Una amplia revisión de sistemas autónomos submarinos puede encontrarse en [Valavanis 97].

- *Sensorización remota*: un caso especial es cuando disponemos de sensores distribuidos geográficamente, como en el caso de los de tipo pluviómetro para estudio de precipitaciones, para predicción del tiempo, medidas oceánicas, estudios hidrográficos, etc. [Daniel 97] .

- *Aplicaciones médicas*: se han desarrollado multitud de sistemas para ayuda en aplicaciones médicas. Podemos destacar dos vertientes: la ayuda al diagnóstico mediante sistemas expertos como MYCIN [Shortliffe 75] o DIAVAL [Diez 94]; y los sistemas robóticos semiautónomos como ayudantes en cirugía en operaciones de próstata o en operaciones con lamparascópia en cavidades abdominales [Hurteau 94] o en microcirugía [Salcudean 94]. Otros ejemplos de sistemas de fusión de datos para aplicaciones médicas pueden encontrarse en [Assi 98].

1.3. Integración multisensorial en robots móviles

En esta sección no se pretende dar una descripción exhaustiva de todos los tipos de robots móviles existentes, sino una breve revisión histórica de éstos resaltando los más significativos. Con ello se quiere destacar la utilización de múltiples sensores y la variedad de implementaciones, arquitecturas y algoritmos utilizados en fusión para robot móviles autónomos.

Uno de los primeros robot móviles fue SHAKEY, desarrollado por el Stanford Research Institute (SRI) de la Universidad de Stanford desde 1966 hasta 1972 [Nilsson 69]. El principal objetivo de este proyecto fue estudiar procesos para control en tiempo real de robots que interactúen en entornos complejos. Constaba de una cámara de vídeo, un radar óptico y varios sensores de contacto. Operaba en un entorno muy artificial y con un mapa a priori del entorno. Aunque desarrolló nuevas técnicas de resolución de problemas y planificación no disponía de un modelo del mundo real ni se centró en la sensorización, lo que implicaba una utilidad muy reducida.

HILARE [Giralt 84], desarrollado por el Laboratoire d'Automatique et d'Analyse des Systèmes (CNRS) de Toulouse, fue un proyecto iniciado en 1977. Su finalidad era desarrollar un robot experimental para investigar en percepción y planificación de robots. Fue el primer robot móvil en crear un modelo del mundo en un entorno desconocido usando información de múltiples sensores [Harmon 86]. Utilizaba sensores de visión (visón estereoscópica), sonar, sensores de contacto y radar láser para reconocimiento de entorno, y se elige un grupo u otro de sensores dependiendo de las características y distancia del objeto. Para la estima de posición utilizaba tres métodos: posicionamiento absoluto mediante balizas activas, calculo de la trayectoria mediante integración y posicionamiento relativo mediante reconocimiento de balizas naturales. Cada uno de los métodos era complementario y se usaba para reducir el error de los demás. Su representación del entorno era mediante mapas expresados como grafos.

El robot de inspección terrestre (*Ground Surveillance Robot*) [Harmon 87] era un vehículo autónomo diseñado para viajar en terreno natural. Usaba sensores acústicos y de visión para detección de obstáculos, y cámaras y radar láser para reconocimiento de objetos. La integración de los distintos sensores se realizaba por medio de una pizarra, donde a cada elemento del mundo detectado le

correspondía una lista de propiedades en las que se incluía el tiempo y la medida de la precisión y confianza de la estima.

El JPL Mars Rover [Thompson 77; Gat 90] estaba englobado en un proyecto iniciado a primeros de los setenta, del Jet Propulsion Laboratory de la NASA para desarrollar un prototipo de vehículo para exploración planetaria. La suspensión del proyecto en 1978 provocó un retraso en el desarrollo del prototipo que fue retomado a primeros de 1990 para desarrollar un prototipo para una exploración de Marte. El vehículo constaba de cámaras de TV, radar láser y un brazo para tomar muestras.

EL NAVLAB y Terregator [Thorpe 88; Herbert 88; Thorpe 91b] son dos vehículos terrestres autónomos desarrollados por el Instituto de Robótica de la Universidad Carnegie-Mellon (CMU). Su principal finalidad era la investigación en el desarrollo de una arquitectura capaz de soportar procesamiento paralelo y desarrollar nuevas técnicas de percepción, fusión e integración multisensorial y navegación [Torpe 91a]. Utilizaba sensores de visión (cámaras TV de color), radar láser y sonar. El sistema de control constaba de distintos módulos ejecutándose independientemente relacionados entre sí a través de una estructura denominada "pizarra-blanca", que difiere de la pizarra clásica en que cada módulo continua su ejecución mientras se realizan las peticiones de lectura de datos y sincronización. Otro interesante vehículo del CMU es Alvin que desarrolla VITS [Turk 88], un nuevo método para extracción de bordes y segmentación para circulación en carreteras que es robusto ante cambios en condiciones atmosféricas, modificaciones en el sensor, en la navegación, etc. Este vehículo dio lugar a ALVINN (Autonomous Land Vehicle In A Neural Network) [Pormeleau 93; Jochem 93] que utiliza un red neuronal Backk-Propagation para analizar la imagen de la carretera y obtener la dirección que debe mantener el vehículo para no salirse de la carretera.

La serie HERMIES [Weisbin 89] (*Hostile Enviroment Robotic Machine Intelligence Experimental Serves*) son robots móviles desarrollados por la Oak Ridge National Laboratory para investigar plataformas capaces de navegar y manipular objetos autónomamente en entornos complejos con total seguridad, tales como edificios de reactores nucleares. Utiliza un array sonar para detectar obstáculos y múltiples cámaras para buscar objetos conocidos en el entorno (como paneles de control) para su manipulación. El mundo es modelado a priori mediante una estructura en forma de red (o grafo). Si un obstáculo es detectado

bloqueando un enlace entre dos nodos se vuelve al último nodo visitado y se calcula un nuevo camino. Utiliza un sistema de reglas para planificación de alto nivel y toma de decisiones, incluyendo integración y fusión.

Blanche [Nelson 88; Cox 89] es un robot experimental desarrollado por los laboratorios AT&T Bell Labs diseñado para operar autónomamente en entornos estructurados, realizando sobre él pruebas de lenguajes de programación de robots, integración de sensores y detección de errores. Como sistema sensorial para el entorno utiliza un radar láser.

El ALV (*Autonomous Land Vehicle*) [Payton 90; Payton 91] fue desarrollado como parte del Programa de Computación Estratégica de la Agencia de Proyectos de Investigación Avanzada de Defensa (DARPA) del Departamento de Defensa de Estados Unidos. Fue diseñado para probar distintas técnicas existentes e investigar en el estado del arte sobre vehículos autónomos. La finalidad inicial del proyecto era el desarrollo de vehículos capaces de circular por una carretera de forma autónoma. Posteriormente se utilizó también para desarrollar vehículos capaces de navegar campo a través y probar algoritmos de esquiva de obstáculos [Olin 91].

En el departamento de Inteligencia Artificial del MIT (Massachusetts Institute of Technology) y dirigido por Brooks, se han desarrollado diversos robots basados en una arquitectura de control basada en capas y totalmente reactiva [Brooks 86]. Su investigación se centra en el desarrollo de mecanismos de control y sensorización de bajo nivel robustos utilizando una arquitectura de control puramente distribuida.

Otros robots "históricos" que se pueden mencionar, aunque no sea una lista completa (por otra parte inmensa), de robots móviles terrestres son el MARS (o Mobi) de Stanford [Kriegman 87; Kriegman 89], el robot KAMRO de la universidad de Karlsruhe [Rembold 88], etc.

En la actualidad podríamos mencionar muchos otros robots desarrollados en multitud de universidades y centros de investigación. Podemos destacar el numeroso desarrollo e investigación sobre robots de la Universidad de Michigan, dirigido por Borenstein, en el estudio de técnicas de localización de robots [Borestein 95a], esquiva de obstáculos [Borestein 95b] y nuevos robots móviles como el OmniMate [Borestein 97], robot de múltiples grados de libertad capaz de moverse en cualquier dirección y girar al mismo tiempo. Otros centros de investigación en robots móviles son el MIT, ya indicado anteriormente, que continua investigando en nuevas tecnologías como visión artificial y lenguaje

natural y ha desarrollado software y hardware para robots de bajo coste [Ferrell 93] y robots humanoides como Cog [Brooks 98]. El instituto de robótica Georgia Tech, que investiga en comportamiento reactivo y comportamiento natural basado en esquemas de comportamiento (*Schemas*) [Arkin 95; Arkin 97a; Arkin 97b], también en percepción, navegación, etc.[Arkin 91; Arkin 93]. La Universidad de Massachusetts (UMASS) que investiga en temas relacionados con ensamblaje, navegación, modelos biológicos, etc.[Grupen 93; Gonçalves 98]; o el instituto francés INRIA (*Institut National de Recherche en Informatique et en Automatique*) con una amplia variedad de áreas de investigación de las que nos interesan percepción, visión artificial y robots móviles.

También se pueden destacar centros españoles como el departamento de Ingeniería de Sistemas y Automática de la universidad de Málaga donde han construido los robots RAM-1 y RAM-2 [Torres 93; González 98] sobre los que investigan en temas relacionados con robot móviles e integración sensorial. El departamento de Ingeniería de Sistemas, Automática e Informática Industrial de la Universidad Politécnica de Cataluña, en temas relacionados con integración multisensorial para robots manipuladores en trabajos de ensamblaje [Suarez 95]. El Área de Ingeniería de Sistemas y Automática de la universidad Carlos III de Madrid, trabajan con robots móviles en interiores y exteriores investigando en localización y percepción basada en marcas [Armingol 98]. La División de Ingeniería de Sistemas y Automática del departamento de Ingeniería Electrónica e Informática Industrial de la E.T.S.I. Industriales de la Politécnica de Madrid [Escalera 95]. El Centro Politécnico Superior de Ingenieros de la universidad de Zaragoza, que investigan en temas relacionados con integración multisensorial [Tardós 91; Castellanos 98]; y otros muchos departamentos y centros que son imposible enumerar todos aquí. Los temas de investigación de los grupos enumerados son mucho más variados que los señalados, sólo se han resaltado trabajos relacionados con esta Tesis, aunque en ciertos casos la investigación actual haya derivado hacia temas distintos a la percepción, fusión multisensorial o robots móviles.

Por último destacar entidades como DARPA, del departamento de defensa americano, la NASA, junto con el JLP, el DOD (*Department of Defense*) y el DOE (*Department of Energy*) y proyectos europeos como PANORAMA (*Perception and Navigation System for Mobile Applications*) en el que se han desarrollado sistemas de percepción y navegación para múltiples aplicaciones en entornos estructurados o PROMETHEUS para vehículos que puedan circular autónomamente en

carreteras. Y también entidades españolas como la CICYT o el Plan Nacional de I+D del Ministerio de Educación y Ciencia, que han financiado y continúan financiando multitud de proyectos de investigación sobre robots móviles, integración y fusión multisensorial en robots, promoviendo programas específicos en el desarrollo e investigación de nuevas técnicas sobre sistemas sensoriales, percepción, etc.

Capítulo 2

Modelo sensorial

El desarrollo de un sistema de integración y fusión multisensorial implica el conocimiento del tipo de sensores que se van a utilizar y de la aplicación en la que se usan, en nuestro caso robot autónomos terrestres. Para este caso concreto los sensores se clasifican en dos grandes grupos: sensores de estado y sensores de entorno, y se realiza una breve descripción de cada tipo de sensor.

A partir del estudio y clasificación realizados se busca una estructura para el sistema sensorial que permita obtener las características buscadas en esta tesis: un sistema sensorial paralelizable, asíncrono y distribuido que sea independiente del número y tipo de sensores utilizados. Para ello se aprovecha la metodología orientada a objetos para desarrollar un tipo de sensor abstracto adecuado: el sensor lógico. Finalmente se muestra un ejemplo de su funcionamiento para dos sensores de entorno y dos de estado. Mediante un diagrama de paso de mensajes y una traza de mensajes se pone de manifiesto su funcionamiento paralelo, distribuido e independencia del número de sensores utilizados.

2.1. Introducción

Los sensores son la parte más importante de un robot móvil, representan sus cinco sentidos y sin ellos no puede averiguar donde se encuentra, que le rodea, ni por supuesto, moverse con seguridad. Por ello, antes de realizar un sistema de integración versátil, reconfigurable e independiente del tipo y número de sensores utilizados, es imprescindible conocer los diferentes tipos de sensores que pueden utilizarse en un vehículo autónomo. Este conocimiento es necesario para diseñar un sensor lógico y un modelo sensorial general que permita conseguir los objetivos propuestos.

Además de un estudio de los sensores existentes es necesario tener en cuenta varios aspectos relativos tanto a los sensores como a los procesos de fusión, por ejemplo el tratamiento que se hace a las medidas observadas, si se utiliza un único sensor o un sistema con múltiples sensores, cual es la arquitectura de fusión escogida y el tipo de las decisiones que pueden tomarse.

1.- *Tratamiento de las medidas*: las observaciones realizadas por un sensor deben ser procesadas adecuadamente para llegar a deducir de ellas las variables buscadas. El tratamiento aplicado sobre las observaciones puede ser muy diverso, pero las funciones pueden clasificarse en detección y en estimación. Un sensor puede realizar una o ambas funciones a la vez y se definen como,

- Estimación. Es el proceso por el cual se determina el valor de las variables consideradas (en algún sentido óptimo) a partir de la observación realizada. Es decir, es un proceso por el cual se transforma o se realiza una correspondencia entre la observación y una estima del estado estudiado.
- Detección. Es el proceso por el cual, establecida una hipótesis, determina qué observación se aproxima más a ésta (el espacio de observación se divide en regiones distintas de forma que cada una represente una hipótesis). La detección puede ser:
  - *binaria*, cuando se realiza una decisión única sobre una hipótesis de forma que se confirme o se niege, por ejemplo la decisión sobre la presencia o no de un objeto (H_0, objeto detectado; H_1, objeto no presente).
  - *m-aria*, cuando se aplican múltiples hipótesis sobre los datos observados, de forma que cada hipótesis representa un grado de

creencia distinto, por ejemplo el uso de múltiples hipótesis para detectar un objeto, donde representan un cierto grado de creencia en la existencia de un objeto. Otra aplicación de detección m-aria es la clasificación, donde la observación, descrita por medio de vectores de características, se compara con m hipótesis para comprobar en qué caso se aproxima más o menos a cada hipótesis (por ejemplo esquina, pared, rincón, puerta o desconocido).

2.- *Fusión sensorial*: tengamos en cuenta que en sensores simples el proceso de detección y estimación es parte integral del sensor, pero en un sistema multisensor el proceso de detección y/o estimación puede situarse fuera del sensor. En este caso existirá un nodo o procedimiento de fusión externo y cada sensor será un contribuyente individual al resultado final. Otra posibilidad es que cada sensor realice la fusión de los datos. Para ello se debe dotar al sistema sensorial (y a cada sensor individualmente) de herramientas para su funcionamiento distribuido y encontrar cual es la información que es necesaria transmitir a los otros sensores para que realicen la fusión. En este caso también existirá un nodo externo de fusión, pero actuará más como registro del resultado obtenido e interfaz con el robot móvil que como nodo de fusión propiamente dicho.

3.- *Arquitectura del sistema de fusión*: se pueden distinguir, como se indicó en el Capítulo 1, dos estructuras de fusión distintas: centralizada y distribuida. Recordemos que en la arquitectura de fusión centralizada todas las observaciones de los sensores son pasadas a un nodo de fusión central que realiza una decisión global a partir de un espacio de observación global, mientras que en la distribuida cada sensor puede tomar primero una decisión a partir de sus propias observaciones y envía estas decisiones a un nodo de fusión (global) donde se realiza la decisión final. La elección de la arquitectura será determinante para la elección del modelo sensorial.

4.- *Tipo de decisiones realizadas*: las decisiones que puede tomar un nodo de fusión donde se realiza la detección o clasificación pueden ser:

- *Duras* son decisiones tomadas irrevocablemente, es decir, se dispone de distintas hipótesis y cuando se escoge una como válida no se varia de opinión.
- *Blandas* son decisiones en las que no está definida claramente una opción como correcta, sino que cada hipótesis tiene su grado de

certeza. En este caso nuevas medidas pueden provocar un cambio en la hipótesis predominante.

Estos cuatro aspectos influyen directamente en el método de fusión y en la estructura del sistema sensorial escogido. Estos aspectos implican la división del sistema de fusión y de los sensores, como veremos más adelante y en capítulos posteriores, en dos clases: uno especializado en la estimación del estado del robot y con decisiones duras; y otro basado en la teoría de la detección y con decisiones blandas. En la medida de lo posible ambos métodos serán distribuidos y con fusión descentralizada.

Así, en la sección 2.2 se describen los sensores que pueden utilizarse en un robot móvil terrestre autónomo, su aplicación y ventajas e inconvenientes de su utilización. A partir de esta descripción, en la sección 2.3 se obtiene una clasificación de los sensores que pueden utilizarse en nuestro sistema de integración y se realiza un estudio de qué información puede considerarse común a estos sensores y qué funciones son necesarias para su utilización. Estas conclusiones son el punto de partida de la sección 2.4, donde se define a partir de la metodología orientada a objetos un tipo abstracto de sensor, denominado *sensor lógico*. A partir de este sensor lógico se diseña el sistema de fusión general que utiliza estos sensores y se muestra un ejemplo para clarificar su funcionamiento.

2.2. Sensores en un robot móvil

En el mercado existen infinidad de sensores distintos cuyo estudio detallado es una ardua tarea y no es la finalidad de este trabajo. En la literatura existen exhaustivas descripciones de distintos sensores utilizados en un robot móvil, como por ejemplo [Borenstein 95; Borenstein 97] y [Everett 95]. Lo que se busca aquí es realizar una clasificación de los sensores desde el punto de vista de la información que pueden proporcionar al robot y las inferencias que pueden realizarse sobre ellos, para poder realizar una integración de los sensores de la forma más efectiva posible. Información hace referencia tanto a las medidas tomadas como al tipo de decisión que cada sensor puede realizar.

La clasificación realizada, en la que se pueden incluir la mayoría de los sensores utilizados en un robot móvil, nos permitirá extraer la información necesaria para crear un modelo común de sensor. A partir de la cual se realiza una integración más sencilla y directa de los datos que proporcionan los sensores y obtener una información más completa que la obtenida con cada sensor independientemente. Además, la información extraída puede ser realimentada a éstos para que, con las correcciones adecuadas, mejoren sus posteriores observaciones.

En esta sección se muestran distintos tipos de sensores utilizados en la construcción de robots móviles y que pueden incorporarse en nuestro robot. Nuestra clasificación consta de dos tipos principales de sensores:

- *Sensores de estado*, que indican la posición, velocidad, orientación del robot y cualquier otra medida del estado del robot. Estos sensores permiten localizar al robot a partir de una serie de medidas (relativas o absolutas) o proporcionan información sobre algún estado interno de interés, pero en ningún momento proporcionan directamente datos sobre el entorno que le rodea.
- *Sensores de entorno*: principalmente encargados de la localización de obstáculos y de la construcción de mapas. Permiten al robot tener "conciencia" de lo que le rodea, interpretar su entorno, esquivar obstáculos y encontrar el mejor camino para conseguir el objetivo propuesto.

Una clasificación de los sensores puede verse en la figura 2-1, donde se ha mostrado un sensor típico de cada clase como representación de su grupo.

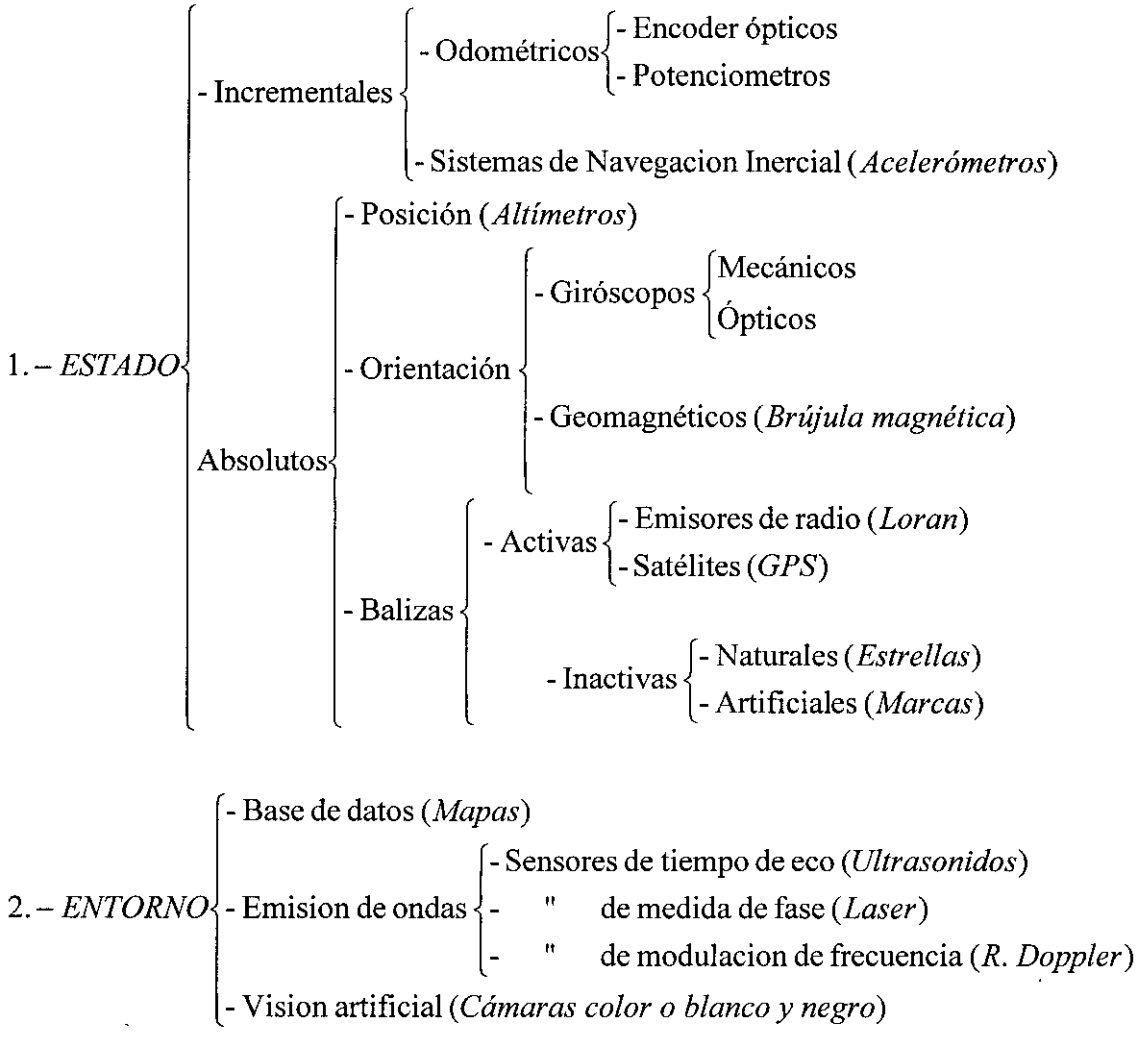


Figura 2-1. Clasificación de los sensores utilizados en un robot móvil.

2.2.1. Sensores de estado

Los sensores de estado del robot, son aquellos que permiten estimar el estado actual del móvil. En este tipo de sensores se pueden incluir cualquier estado del funcionamiento del robot. En nuestro caso nos centraremos exclusivamente en sensores de posición. Por posición entendemos el estado de movimiento del vehículo, es decir, sus coordenadas x, y, z, orientación o cualquier otra variable de interés que represente el estado del movimiento del móvil como por ejemplo su velocidad, balanceo, etc. En nuestro caso se estimarán únicamente x, y, y orientación (al considerar vehículos terrestres no tendremos en cuenta la altura). Por supuesto, no todos los sensores de posición tienen por qué

proporcionar información sobre todas las variables de posición. Por ejemplo, una brújula sólo es capaz de tomar medidas sobre la orientación del móvil.

Estos sensores permiten contestar a una de las preguntas claves en la navegación: ¿Dónde estoy? [Borenstein 95]. La localización del robot dentro de su entorno se realiza a partir de las medidas proporcionadas por esta clase de sensores. Estas medidas pueden clasificarse en medidas de posición *incrementales o relativas*, que necesitan como referencia una posición anterior del robot e integraciones sucesivas para estimar la posición actual, como por ejemplo la odometría ("dead-reckoning") y los sistemas de navegación inercial (INS); y en medidas de posición *absolutas*, que permiten calcular la posición actual del robot con respecto a un cierto sistema de referencia fijo y sin tener en cuenta estados anteriores. Sensores de este tipo son la brújula magnética, las balizas activas (por ejemplo, los emisores de radio o el GPS) o las balizas pasivas, tanto artificiales como naturales. Las medidas absolutas son más precisas que las primeras pero tienen el inconveniente de que dependen de objetos o sistemas externos al robot y no siempre están disponibles. Así, un buen procedimiento es la utilización de ambos tipos de medidas, utilizando las medidas relativas en ausencia de referencias absolutas.

Un esquema de estos sensores puede verse en la figura 2-2, donde los sensores se clasifican en dos grandes grupos:

1.- *Sensores de estima directa*: son aquellos que no necesitan una interpretación del entorno del robot para estimar la posición. Estos sensores obtienen las estimas de posición de forma directa o con un tratamiento sencillo de los datos. Sus decisiones sobre las estimas serán duras, indicando únicamente la medida estimada y su error asociado. Estos pueden clasificarse en autónomos o en dependientes de fuentes externas.

Los autónomos son aquellos que estiman la posición del vehículo sin necesidad de ninguna señal externa. Entre estos cabe destacar los sensores geomagnéticos, odométricos e inerciales, como por ejemplo giróscopos, acelerómetros, brújulas, etc.

Los sensores dependientes de fuentes externas, estiman la posición del vehículo a partir de una señal recibida por cierta fuente o fuentes externas al vehículo. Estas señales externas típicamente son señales electromagnéticas como señales de radio, luminosas, etc. Estos sensores

aunque son más precisos, tienen el inconveniente de que pueden provocar la detección del móvil o incluso la manipulación o destrucción de la señal de referencia (la fuente emisora) por parte de un enemigo, lo que los hace pocos recomendables en aplicaciones militares en caso de guerra. Entre estos cabe destacar emisores de radio (radiobalizas), faros, GPS, etc.

2.- *Sensores basados en percepción de entorno*: necesitan observar e interpretar los datos recibidos del entorno del robot y estiman la posición relacionando el vector de características extraído con otro almacenado en una base de datos. En este caso sus decisiones suelen ser blandas, ofreciendo múltiples hipótesis sobre la interpretación de las medidas observadas. Utilizan balizas naturales o artificiales situadas en el entorno, pero que no emiten ninguna señal, sino que deben ser localizadas por el sensor, lo que implica la interpretación del entorno observado. En este apartado se encuentran emisores de ultrasonidos, visión artificial, sonar, etc.

En la figura 2-2 se puede observar esta clasificación y un ejemplo de sensor de cada tipo. Con ellos tenemos una amplia representación de aquellos que pueden utilizarse para obtener la localización del robot móvil en todo instante. Una descripción más extensa de los distintos sensores que pueden utilizarse en un robot móvil para su localización se puede encontrar en [Everett 95], [Gonzalez 96] o [Borenstein 97].

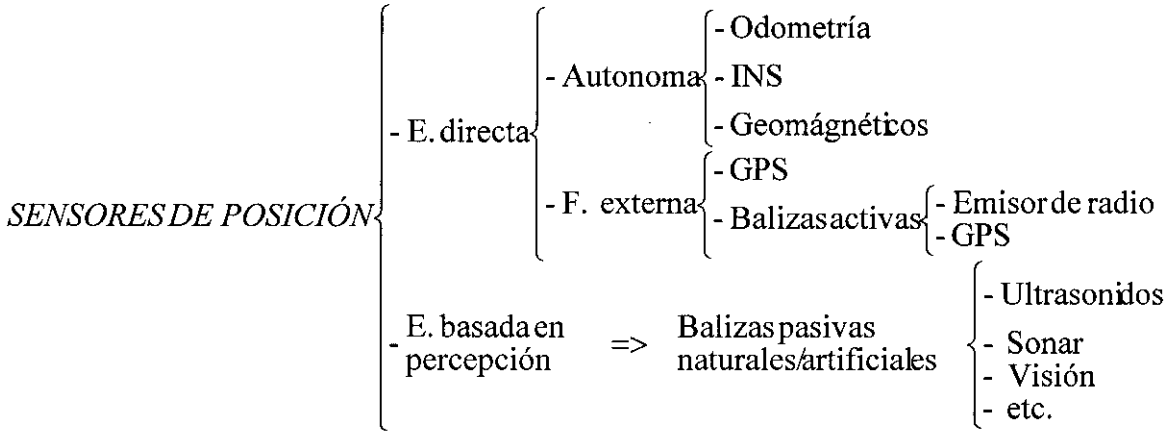


Figura 2-2 . Sensores de posición

Como ya se ha indicado anteriormente, cada uno de los sensores de posición deberá realizar un preprocesamiento de las señales que observan. La estima de posición por medio de balizas pasivas no es tratada en este momento, sino que será estudiada junto con los sensores de entorno, puesto que se utiliza en muchas de las técnicas necesarias para detección de entorno.

Tipo de sensor		Sensor	Tipo de Información	Referencia	Notas
Incremental		Odómetro INS	Posición, Velocidad Orientación	Local- respecto al robot Global – ejes inerciales	Recomendable reset periódico, debido a su error acumulativo.
Absoluto	**De posición**	Altímetro	Posición	Local – ejes geográficos	Aconsejable ayuda de otros sensores.
	De orientación	Giróscopo Brújula Magnética	Orientación	Local – ejes del robot Global – N. Magnético	
	Balizas activas	GPS Emisión de ondas	Posición, Velocidad Orientación	Local – respecto de la posición conocida del emisor	Posibilidad de pérdida de la recepción de la señal.

Tabla 2.1. Características de los sensores de posición

2.2.1.1. Sensores incrementales

Los sensores incrementales, conocidos en la literatura científica como *Dead reckoning* (derivado de la expresión inglesa *"deduced reckoning"*), son aquellos sensores que calculan la posición actual de un móvil a partir de una posición previa y de la integración de su velocidad o aceleración en un cierto intervalo de tiempo.

Entre estos sensores cabe destacar a los odómetros y a los sensores inerciales. Características comunes a este tipo de sensores se encuentran recogidas en la tabla 2.1.

Sensores odométricos

El odómetro es un dispositivo que se incorpora en el propio vehículo y se encarga de medir el desplazamiento o velocidad de éste. Dependiendo del mecanismo que utilicen (propio del método físico aplicado) pueden ser ópticos, magnéticos, inductivos o capacitivos, potenciómetros, etc.

En los robots móviles, el sensor odométrico más popular es el odómetro óptico. Básicamente, su mecanismo es un foco de luz dirigida a un fotodetector que es interrumpido por un disco rotante con un cierto patrón opaco/transparente a dicha luz. La recepción o no del haz luminoso debido a la rotación del disco que está unido al eje de interés, representa una relación directa de la rotación (velocidad o posición) de dicho eje.

Existen básicamente dos tipos de encoders ópticos: el incremental y el absoluto. El encoder incremental mide la velocidad de rotación de un eje y por tanto puede inferir posición relativa. El encoder absoluto mide directamente posición angular y por tanto puede inferir velocidad. Si la inestabilidad en la información de posición no es tenida en cuenta (se corregirá con otros elementos) los encoders incrementales poseen un interfaz más sencillo y proporcionan una resolución equivalente a los absolutos con un coste mucho menor [Everett 95].

Debido a su sencillez y naturaleza digital, estos sensores representan un sensor de bajo coste y buena inmunidad al ruido, lo que explica su amplia utilización en todo tipo de robots móviles y su uso prácticamente obligado. Su mayor problema es que utiliza el desplazamiento relativo de cada rueda para encontrar mediante integración la nueva posición y orientación del vehículo, lo que da lugar a que los pequeños errores aumenten con el tiempo. Otro riesgo importante es la posible aparición de errores debido a deslizamiento o patinados de las ruedas, que pueden ser corregidos con ayuda de otros sensores.

Sensores inerciales

Los sensores inerciales son sensores que se basan en las propiedades inerciales de los cuerpos. Estos son giróscopos y acelerómetros que suelen acoplarse en una plataforma de forma que los acelerómetros miden la aceleración en cada uno de los ejes del vehículo y los giróscopos miden la orientación del vehículo con respecto a unos ejes inerciales. Así, mediante una doble integración de las medidas de los acelerómetros (utilizando la orientación indicada por los giróscopos para su transformación a coordenadas inerciales) se obtiene la posición del móvil sin ninguna ayuda externa. A este procedimiento se le conoce como Sistema de Navegación Inercial, INS (*Inertial Navegation System*) [Britting 71].

La gran desventaja del INS es que, debido a la doble integración de la información obtenida de los sensores, los errores en el cálculo de la posición y velocidad del móvil aumentan enormemente con el tiempo, siendo típicamente de

1 milla náutica por hora. Por ello normalmente sus medidas se integran, mediante un filtro de Kalman, con otras fuentes externas para conseguir altas precisiones [Maybeck 79]. Uno de los sensores más utilizados que mejor se complementa con el INS es el GPS, que suelen integrarse juntos mediante un filtro de Kalman, un ejemplo de simulación de un sistema INS con ayuda de GPS puede encontrarse en [López-Orozco 95b]. Un ejemplo en un robot móvil terrestre en exteriores puede encontrarse en [Barshan 95].

La principal dificultad en su utilización, es la correcta colocación sobre el vehículo para evitar errores, lo que implica diseños muy precisos y diversas pruebas para detección de errores en la ortogonalidad de la disposición de los elementos utilizados, lo que se conoce como alineamiento. Técnicas de alineamiento de sistemas de navegación ligados pueden encontrarse en [Aranda 89; Ruipérez 94].

2.2.1.2. Sensores de orientación

Estos sensores son muy importantes en el posicionamiento de un robot, puesto que ayudan a compensar uno de los mayores problemas de la odometría: su error acumulativo. En caso de que el robot tenga alguna pequeña desviación lateral de su trayectoria, por pequeña que sea, al cabo un cierto tiempo será importante, ya que al basarse en integraciones iteradas el error crecerá muy rápidamente con el tiempo. Por esto cuanto mejor se conozca la orientación mejor podremos dar la posición del robot y, por tanto, corregirla.

Los tipos más comunes son los giróscopos, basados en propiedades inerciales, y las brújulas, basados en las propiedades geomagnéticas. Este tipo de sensores, independientemente de su principio físico o construcción, proporcionan básicamente el mismo tipo de datos, véase la tabla 2.1.

Giróscopos

Pueden ser clasificados en dos amplias categorías dependiendo del principio físico en el que se basan: giróscopos mecánicos y giróscopos ópticos.

Giróscopos mecánicos: Son sensores basados en las propiedades inerciales del spin de un cuerpo que gira sobre sí mismo muy rápidamente, lo que da lugar a una dirección claramente diferenciada y cualquier fuerza aplicada dará lugar a una precesión del giróscopo alrededor de su eje.

Giróscopos ópticos: Estos giróscopos tiene ninguna o pocas partes móviles, de modo que son insensibles a efectos gravitatorios y, por tanto, eliminan la

necesidad de cárdanes, que es uno de los inconvenientes de los giróscopos mecánicos. El giróscopo óptico se basa en patrones de interferencias formados al colisionar dos rayos laser, lo que permite estimar velocidad y orientación del vehículo.

Un amplio tratamiento matemático y una completa descripción de giróscopos mecánicos y ópticos utilizados en navegación inercial, puede encontrarse en [Stieler 82].

Sensores geomagnéticos

El sensor más conocido de este tipo es la brújula magnética, que se encarga de marcar la dirección del flujo del campo magnético terrestre.

Este sensor presenta una serie de problemas: la no estabilidad del campo magnético, debido a fluctuaciones en el tiempo y en el espacio; la diferencia de localización de los polos magnético y geográfico (el magnético está inclinado unos 11°) que debe ser corregida; y la dificultad de su uso en interiores, ya que el campo magnético se ve afectado por líneas eléctricas o por estructuras metálicas.

Estos inconvenientes aconsejan que su uso sea complementado con medidas de otros sensores, eliminando o reduciendo así las imprecisiones cometidas en el cálculo de la orientación. Como proporcionan medidas absolutas de orientación, únicamente necesita realizar la traslación de ángulos al sistema de referencia del robot.

2.2.1.3. Balizas activas

En este apartado se enumeran sensores utilizados para el seguimiento o la recepción de señales de fuentes externas (balizas). Las balizas se clasifican en activas e inactivas.

Se entiende por balizas activas aquellas que emiten alguna clase de señal, ya sea luz (espectro visible o no), acústica, etc. Podemos distinguir aquí entre *balizas activas naturales*, signos o señales utilizados como orientación y localización pero no expresamente diseñadas para ello, como por ejemplo las estrellas, luces de una casa, etc.; y las *balizas activas artificiales*, expresamente construidas por el hombre para utilizarlas como señales de localización, por ejemplo transmisores de ultrasonidos o de radio, GPS, fuentes de luz como faros, etc.

La estima de la posición a partir de ellas no se basa en la interpretación del entorno, sino en el principio de triangulación, bien a partir de medidas de ángulos,

de distancias o de combinación de ambas. El número mínimo de balizas requeridas para estimar la posición dependerá del tipo de sistema empleado, pero en la práctica, para minimizar los errores cometidos en la estimación y las pérdidas de señal, se suelen utilizar un número elevado de balizas simultáneamente.

Las características comunes de estos sensores pueden observarse en la tabla 2.1 y entre otros podemos destacar:

GPS

GPS (*Global Positioning System*) o NAVSTAR (*NAVigation Satellite Timing And Ranging*) [Parkinson 83, Bruin 87, Ellowitz 92] es un sistema de localización por satélites que fue desarrollado en 1973 por el Departamento de Defensa de los Estados Unidos. Como sucesor del sistema de posicionamiento TRANSIT [Danchik 90], se diseñó para mejorar algunas de las deficiencias de éste, de forma que fuese capaz de cubrir no sólo las necesidades de navegación y tiempo de la época, sino también las necesidades de futuros compromisos militares. Posteriormente, en 1983, el sistema fue modificado para que pudiera ser utilizado por cualquier usuario. Así surgieron dos servicios: el PPS (*Precise Positioning Service*), de muy alta precisión y uso restringido a aplicaciones militares de Estados Unidos y otras autorizadas; y el SPS (*Standard Positioning Service*) cuya señal, al estar degradada debido a la introducción deliberada de un error en el reloj y en los mensajes transmitidos por los satélites (SA, Selective Availability), sólo permite precisiones de aproximadamente 100 metros [Leick 90], aunque pueden ser mejorados hasta precisiones del orden de centímetros con el uso de GPS diferenciales [Snyder 92].

Este sistema también proporciona medidas absolutas a partir de la señal recibida por, al menos, tres satélites. El preprocesamiento en este caso consiste en la elección de los satélites que proporcionen una mayor fiabilidad. Para calcular la posición del usuario, se mide la distancia aparente (pseudorango) a tres satélites dispersos geométricamente. Por pseudorango se entiende la solución calculada por el receptor, que tendrá un error debido a la diferencia entre el tiempo real del sistema GPS y el tiempo supuesto por el equipo del usuario. Un modelo para cálculo del pseudorango puede encontrarse en [Friess 88]. Una muestra del funcionamiento del sistema GPS como una toolbox de Matlab puede encontrarse en [López-Orozco 95a].

Como podemos ver este tipo de balizas está orientado a la localización de vehículos en entornos externos y vehículos que recorren grandes distancias.

Otras balizas

Otros emisores de señales diseñados por el hombre para servir como referencia al vehículo dependen del sistema utilizado para su detección. Así pueden ser balizas emisoras de ultrasonidos, infrarrojos, señales luminosas, etc.

La fiabilidad y precisión de este tipo de sensores depende del tipo de señal utilizada (infrarrojos, láser, luz visible, etc.) del medio en el que se encuentra el móvil (más o menos ruidoso a la señal), de la distancia de las balizas al móvil y del número de balizas utilizadas en la triangulación en cada instante. Aunque es uno de los métodos de localización absoluta más precisa, también tiene ciertos inconvenientes como el situar las balizas en lugares determinados de forma que se puedan observar simultáneamente el mayor número posible en todo momento, o evitar zonas con ruido o contaminación acústica, electromagnética, etc.

Estos inconvenientes implican que estas técnicas no sean utilizadas en entornos que varíen constantemente o en entornos no estructurados.

2.2.2. Sensores de entorno

Estos sensores proporcionan datos relativos sobre el entorno del robot. Estos datos pueden ser utilizados de dos formas distintas: para la localización del móvil o para la detección de obstáculos o la construcción de mapas.

En el caso de utilizarlos para localización del móvil, la información proporcionada por el sensor es contrastada con una base de datos o un mapa del "mundo" incorporado en el robot para, por medio una simple comparación (*matching*), estimar la posición del robot en el entorno. Estos sensores se denominan balizas inactivas y balizas activas naturales.

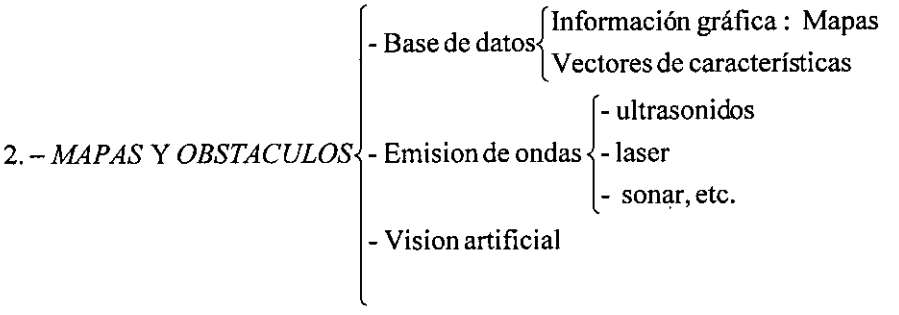


Figura 2-3. Sensores de entorno.

Otra utilización de estos sensores es para la construcción de un mapa (en caso de que no existiese ninguno almacenado en el robot) o para completar uno existente (indicando obstáculos que no aparecían en el plano inicial), y a partir de él planificar un conjunto de acciones que le lleven a la consecución de su objetivo final o trazar un camino seguro que le lleve al destino deseado. En este caso, los sensores suelen utilizarse principalmente para evitar cualquier obstáculo que pueda aparecer en el camino del robot.

	Sensor	Tipo de Información	Referencia	Medidas	Notas
Base de datos	Mapas Fichas etc.	- Mapas de bits - Vector de características	Global – ejes del laboratorio	Absolutas	Tratamiento de los datos almacenados
Emisor de ondas	Ultrasonidos Radar-laser Sonar, etc.	- Distancia a objetos	Local – ejes del robot	Absolutas	Precisión lineal con la distancia. Numerosas fuentes externas de error.
Visión	1 cámara 2 cámaras	- Distancia a objetos - Vector de características	Local – ejes del robot	Absolutas	Grandes necesidades de cálculo y tiempo de proceso

Tabla 2-2. Características de los sensores de localización y construcción de mapas

Podemos distinguir, como puede verse en la figura 2-3 y en la tabla 2-2, varias clases de sensores que se describen con más detalle a continuación.

2.2.2.1. Base de datos

Cualquier información del entorno que esté almacenada o disponible para el robot se tratará por extensión como un sensor. Este sensor estará dotado de un

sistema de recuperación de la información almacenada y de un método para el tratamiento de ésta que puede ser desde una simple transformación de coordenadas a un complejo análisis inteligente de la base de datos en cuestión [Fayyad 96].

Las bases de datos son utilizadas principalmente como dos fuentes de conocimiento:

- Conocimiento a priori del entorno del robot. En este caso se conecta al sistema de fusión de entorno para actualizar el conocimiento sobre la información percibida.
- Referencia en sistemas de localización. En este caso se utiliza como referencia para los sensores de balizas naturales o artificiales. Estas bases pueden conectarse directamente al sistema sensorial de las balizas como conocimiento de la referencia o al sistema de fusión de entorno y que el sensor de baliza pida a éste la información que necesita.

2.2.2.2 Emisores de ondas

La mayoría de los sensores utilizados para construir un mapa o detectar obstáculos miden distancias. Podemos distinguir tres formas diferentes de medir distancias según el tipo de onda emitida y su tratamiento:

1.- Sensores de tiempo de vuelo. Sensores que obtienen la distancia a un objeto midiendo el tiempo que una onda emitida tarda en viajar al objeto y volver reflejada al receptor. Son los más utilizados.

2.- Sensores de desfase de onda. Sensores que miden el desfase entre una onda continua emitida y la posteriormente recibida.

3.- Sensores de modulación de frecuencia. Sensores basados en el radar de frecuencia modulada (FM). Calculan la distancia del objeto directamente a partir de la diferencia de frecuencias entre la onda emitida y la reflejada.

Estos sensores suelen utilizar sonidos, ultrasonidos, laser o fuentes ópticas. Un parámetro importante que diferencia estos métodos es la velocidad de los pulsos de energía emitidos (para el sonido aproximadamente 0.3 m/ms y para la luz 0.3 m/ns) que da lugar a la resolución y alcance propios de cada sensor.

A diferencia de los pulsos de corta duración emitidos por los sensores de tiempo de vuelo, los sensores de medida de desfase y de modulación de frecuencia emiten ondas continuas y permiten la obtención de la dirección y velocidad de los objetos observados.

Una de las ventajas de estos sistemas se debe a la propia naturaleza de las señales utilizadas: la distancia absoluta a un objeto está directamente disponible sin necesidad de análisis complicados y sin necesidad de tener en cuenta propiedades ni orientaciones de la superficie destino. Además estos sensores mantienen una precisión en la distancia de modo lineal siempre que pueda obtenerse el eco mientras que en la triangulación la precisión disminuye a medida que la distancia al destino aumenta.

A pesar de sus ventajas, existen varias fuentes potenciales de error:

- La velocidad de propagación. En sistemas basados en ondas acústicas, como el sonar o ultrasonidos, la velocidad del sonido se ve claramente afectada por cambios en la temperatura y, en menor medida, por la humedad, por lo que estos errores deben ser tenidos en cuenta y corregidos.
- Incertidumbres en la detección de las ondas. Se debe a variaciones en rango de la energía de la señal recibida debido a variaciones de reflectividad en el superficie del destino. Así, en el caso de detección por umbral fijo podría provocar que objetos muy reflectantes pareciesen más cercanos.
- Consideraciones temporales. Debido a la diferencia de velocidad entre la luz y el sonido, al comparar ambos sistemas se puede ver que los primeros requieren una circuitería muy compleja debido a los requerimiento de respuestas muy rápidas, por lo que son claramente más caros aunque tienen una mayor precisión. Además, debido a limitaciones físicas estos sensores presentan zonas muertas que deben ser tenidas en cuenta.
- Interacción con la superficie. Cuando las ondas golpean un objeto, cualquier eco detectado representará una porción muy pequeña de la onda original. La energía restante se refleja o puede absorberse por el objeto dependiendo de las características de la superficie y del ángulo de incidencia del rayo. Así, en lugar de no recibir ninguna señal se recibirán distintos rayos, especialmente en el uso de ultrasonidos. Además si la fuente de transmisión se encuentra en un ángulo o excede de un valor crítico, la

energía reflejada puede quedar fuera del umbral del receptor. Por último, las señales difundidas en todas direcciones pueden reflejarse en más de un objeto, devolviendo al receptor varias señales falsas o ruidos. Para compensar estos errores las medidas se realizan repetidamente y se promedian, llegando a niveles de ruido aceptables pero a expensas de un cierto tiempo adicional.

Todos estos errores deben ser compensados en el preprocesamiento del sensor correspondiente. En caso de que no puedan ser compensados se deberá tener en cuenta en la incertidumbre de la medida obtenida.

2.2.2.3 Visión artificial

La visión artificial tiene como finalidad la reproducción artificial del sentido de la vista [Maravall 93] (para ello engloba distintos subsistemas informáticos). Los sistemas de visión suelen utilizar una cámara o dos. El uso de una única cámara está restringido a menudo a la búsqueda de balizas (naturales o artificiales) y al reconocimiento de patrones para la localización del móvil. La aplicación de dos cámaras lleva a la visión estereoscópica que permite calcular la distancia de los objetos observados y por tanto la construcción de mapas.

Las operaciones básicas en un sensor de visión son: la adquisición de imágenes (digitalizar una señal continua de vídeo); el preprocesamiento (conversión de los niveles de gris, transformaciones geométricas, transformación del histograma y filtrados espacial y frecuencial); y al análisis automático de imágenes (segmentación, extracción de características y clasificación).

Preprocesamiento:

Es el tratamiento de la señal de vídeo para resaltar los objetos buscados o para destacar formas o características ocultas. Las operaciones más comunes son,

- Conversión de los niveles de gris, que se basa en una transformación de los niveles de gris de la imagen para facilitar su tratamiento.
- Transformaciones geométricas, como la corrección de la perspectiva y la reconstrucción tridimensional de los objetos de una escena.
- Transformación del histograma, manipulación del histograma de la imagen. Permite aumentar/disminuir el contraste de la imagen permitiendo, en ciertas ocasiones, distinguir mejor diversos objetos.

- Filtrado espacial y frecuencial, se utiliza para la extracción de bordes, aunque suelen se una operación terriblemente costosa en tiempo y memoria del computador.

Análisis automático de imágenes:

Una vez realizado el prepocesamiento de la imagen se debe proceder a su análisis. Corresponde a las etapas de análisis automático de una escena digital, y son: segmentación de la escena, extracción de las características de los objetos y reconocimiento de éstos.

- Segmentación: Se define como el proceso dirigido a dividir la imagen digital en zonas disjuntas con significado propio. Este significado dependerá de la imagen, así como de los objetos que se localicen para su interpretación. Existen dos tipos de segmentaciones: de regiones o zonas y de objetos. La operación es la misma, lo que las distingue es lo que se desea segmentar.

  Las técnicas de segmentación, independientemente de a qué estén orientadas, se pueden dividir en tres grupos:

  (1) Aplicación de umbrales de nivel de gris; basado en el empleo del histograma. Consiste en distinguir objetos distintos dentro de una escena a partir de sus diferentes niveles de gris.

  (2) Agrupación por rasgos comunes; utilizada comúnmente en el reconocimiento de formas.

  (3) Extracción de bordes; la mayor cantidad de información de un objeto se encuentra en sus bordes, de modo que la mayoría de los objetos pueden reconocerse a partir de su forma.

- Extracción de características propias: Cada objeto tiene una serie de parámetros característicos como perímetro, grado de circularidad, momentos de orden n, etc. Estos rasgos permiten distinguir al objeto de otros de la misma escena.

- Reconocimiento de objetos: Corresponde con la clasificación de los objetos detectados. Existen multitud de técnicas para reconocimiento de objetos desde clásicas, basadas en discriminantes como medidas de distancias, contrastes de hipótesis, etc.; o heurísticas como redes neuronales, lógica borrosa, etc.

2.3. Información sensorial en un robot móvil

A partir de la revisión de los distintos tipos de sensores que pueden utilizarse en un robot móvil y que pueden emplearse en nuestro sistema sensorial, se infieren dos tipos de informaciones claramente diferenciadas que es necesario fusionar: *información de estado e información de entorno*. Ambas informaciones serán tratadas por módulos de fusión distintos. Uno de estos módulos se encargará de realizar la integración de medidas numéricas correspondientes al estado del móvil (estima de la mejor posición, velocidad, orientación, etc.); y el otro módulo de fusión debe ocuparse de la integración de las estimas obtenidas a partir de las medidas del entorno.

A partir de los datos del entorno también podría derivarse información sobre la posición del móvil (comparando lo percibido con una base de datos de balizas o con un mapa del entorno). Esto implica dejar abierta la posibilidad de interacción entre ambos, aunque este tipo de funcionamiento sería por petición desde el sistema de fusión de estado (encargado de la estima de posición) al sistema de fusión de entorno. Veamos, antes de realizar el sistema de fusión qué tipo de datos vamos a integrar y qué información adicional es necesaria.

Posición	Velocidad	Orientación	Ocupación de celdas	Tipo de objetos
• INS • Odómetro • GPS • Mapa/Balizas pasivas y visión. • Balizas activas	• Acelerómetros (INS) • 1-2 visión[1] • Odómetro • GPS • Ultrasonidos	• Torreta de ultrasonidos • Giróscopos (INS) • Brújula	• 2-visión • Torreta de ultra-sonidos • Ultrasonidos • Láser • Mapa	• 1-2 visión • Mapa

Tabla 2-3. Datos ofrecidos por distintos sensores.
[1]Por 1-2 visión se entiende la visión con una única cámara (1 visión) y la visión con dos cámaras o visión estereoscópica (2 visión), cuyas posibilidades son muy diferentes.

2.3.1. Datos proporcionados por los sensores

Entre la gran variedad de datos que pueden ofrecer los sensores, los más representativos pueden encontrarse en la tabla 2-3, nos ocuparemos

principalmente de medidas de posición y de características del entorno. Los datos mostrados en la tabla 2-3 corresponden a las dos categorías indicadas en la sección anterior. El sistema de fusión diseñado a continuación tiene como finalidad integrar información de cada uno de estos tipos de sensores.

La información que es necesaria extraer de cada uno de los sensores disponibles y que puede integrarse en una estima común en nuestro sistema de fusión puede agruparse en dos grandes grupos, véase en la figura 2-4:

a) Datos de LOCALIZACIÓN. Indican la posición, velocidad y orientación del móvil. Junto con la información de la posición del móvil estimada por cada sensor es necesario indicar el error cometido en la estimación debido a la precisión de la medida, al modelo del sistema, a los ruidos del entorno, etc.

b) Datos de ENTORNO. Indican la ocupación del espacio que rodea al móvil. La ocupación se da en función del enrejado o divisiones que se han hecho del espacio visible. Esta ocupación se indica junto con un grado de certeza (probabilidad de ocupado). Además, según el tipo de sensor utilizado, puede ofrecer información sobre el tipo del objeto que observa (siempre que se conozca) mediante un código. Como complemento al tipo de objeto se indica la distancia a la que se encuentra y el error cometido en esa estimación.

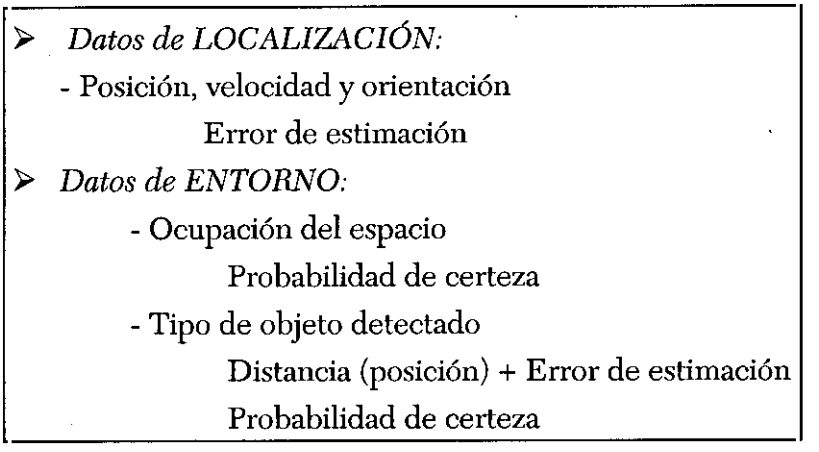


Figura 2-4 . Información de los sensores que se integra.

2.3.2. Funciones para utilización de los sensores

Además de definir el tipo y estructura de los datos que va a manejar el sensor del robot, es necesario definir una serie de funciones para utilizar dichos sensores.

Funciones de mantenimiento:

Son funciones más propias de la configuración inicial del sistema que del momento de la navegación. Aun así, estas funciones pueden utilizarse, y algunas se utilizan, en la navegación propiamente dicha. No manipulan los datos ofrecidos por los sensores, sino su disposición, existencia, sincronización, etc.

- Creación/adicción de un sensor del sistema.

Corresponde a la modificación software de un archivo de definición de sensores, es decir, dar de alta a un sensor. Esto se realiza en la definición del sistema sensorial y tiene vigencia hasta que se modifique nuevamente. Cuando se habla de adición nos referimos a que el tipo de sensor ya existe y únicamente es necesario indicar la existencia de otro sensor del mismo tipo. Pero si ese sensor es la primera vez que se utilizará en el sistema, estamos hablando de creación y en este caso, además de lo anterior, se debe introducir toda la información necesaria para su uso, como por ejemplo el modelo (físico y de probabilidades) del sensor, el tipo, etc.

- Eliminación/anulación de un sensor del sistema.

La anulación y la eliminación de un sensor tienen el mismo efecto: la no utilización de los datos generados por ese sensor (si es que existen). La anulación equivale a la adición, es la orden dada al sistema de ignorar los datos producidos por ese sensor (esta anulación puede ser temporal y dinámica, al igual que la adición). La eliminación se realiza en la inicialización y es la no utilización de un sensor durante todo el proceso de navegación. En este caso equivale a la eliminación física del sensor, sea real o no esta eliminación.

- Sincronismo del sensor (Tiempo).

Corresponde a la petición de tiempo por parte de cada sensor. Con esta función se puede calcular el tiempo transcurrido, con referencia a un reloj "universal", es decir, común a todos los procesadores que utilizan los sensores. Con esto nos aseguramos la utilización de una misma referencia temporal sea cual sea el

lugar donde se usa cada sensor. Recordemos que los sensores son tratados como sistemas distribuidos donde es importante que todos tengan una misma referencia de tiempo.

Funciones de petición de información:

Son las funciones que tratan propiamente con la información obtenida por el sensor. Ya sea con los datos medidos directamente o con la estima del estado o de su entorno en la configuración escogida. Estas funciones son menos generales que las primeras y normalmente son propias de cada sensor utilizado.

- Solicitud/Modificación del estado del sensor (funcionamiento, tiempo, ...)

Es la petición, al propio sensor, de su estado. En este caso se pide que indique todos aquellos datos necesarios para conocer su funcionamiento, como son su "tiempo" en el momento de la petición, nivel de funcionamiento, etc. O también se le pueden dar órdenes para modificar su funcionamiento, orientación, etc.

- Solicitud de Información de estado

El sensor al recibir esta petición, proporciona a su solicitante la estima de las variables de estado en consideración y que es capaz de calcular y su covarianza de error. Un caso típico en robots móviles es ofrecer la estima de la posición actual del robot junto con el instante de tiempo de la obtención de dicha posición. Esta posición irá además acompañada por el error que se cree cometido en esa estima. Si el sensor puede proporcionar datos de velocidad o de orientación, los ofrecerá al igual que los de posición, siendo el tiempo común para todos.

- Solicitud de Entorno

El sensor al recibir esta petición, proporciona datos sobre la ocupación de las celdas que han cambiado significativamente desde su última actualización; también proporciona la distancia y tipo de objetos reconocidos recientemente o que han modificado su tipo. Todos estos datos deben ir acompañados de su correspondiente tiempo y probabilidad de certeza.

2.4. Sensor general propuesto

Para construir un sistema de fusión versátil, capaz de fusionar medidas de sensores tan dispares es necesario encontrar un método que permita al sistema de fusión la abstracción de los sensores utilizados. De esta forma se permite que el sistema sea portable, que pueda sustituir o añadir más sensores de forma sencilla y que sea adaptable a distintos tipos de sensores. Por tanto, la idea básica es definir un tipo, o en su defecto varios tipos abstractos, de sensor.

2.4.1. Sensor lógico

La primera propuesta de abstracción y generalización del funcionamiento de un sensor fue dada por Henderson y Shilcrat [Henderson 84a]. A esta abstracción se la denomina *sensores lógicos*. Un sensor lógico contiene tanto una descripción del sensor como los procedimientos para controlarlo, y representa una definición abstracta de un sensor para proporcionar un marco de trabajo uniforme sobre el que realizar la integración multisensorial. Esto es, separa los detalles del sensor físico del sistema de fusión de manera similar a lo que ocurre con un tipo abstracto de datos que separa los detalles algorítmicos de la aplicación que los utiliza.

Un ejemplo que muestra la utilización de sensores lógicos y su utilización en un sistema de integración multisensorial puede encontrarse en [Henderson 84b], donde se define el MKS (*Multisensor Kernel System*). Este sistema de integración multisensorial utiliza vectores de características obtenidos por dos sensores de bajo nivel (cámaras para visión estereoscópica y un escaner laser) en una estructura tridimensional denominada "grafo de proximidad espacial". El sistema es capaz de integrar y analizar datos de diferentes sensores, aunque sólo utiliza sensores de reconocimiento de entorno y no realiza un tratamiento de los errores en la detección de datos. Otro ejemplo de sistema robótico que utiliza sensores lógicos es el brazo manipulador de [Zeng 89]. En él se desarrolla un método sistemático para realizar integración multisensorial y se define una estructura jerárquica por capas para el control de movimiento del robot y todos los sensores lógicos y procesos de control son tratados como módulos objeto con paso de mensajes entre ellos.

Una extensión de los sensores lógicos son los *comportamientos lógicos* [Henderson 90]. Estos sirven como marco uniforme para la integración multisensorial, incrementando la portabilidad de los sistemas de control del robot. Para ello se definen sensores y actuadores lógicos de forma que las tareas

elementales del robot se describen como comportamientos lógicos que se componen de sensores y actuadores lógicos.

2.4.2. Desarrollo orientado a objetos del sensor lógico

Una de las características de un sistema orientado a objetos es que tiende a expresarse en términos del problema tratado y representa una maquina virtual que organiza nuestra abstracción de las entidades clave del problema. Utilizando esta metodología, cambiar los detalles del hardware del sistema sólo afecta a nuestra abstracción de las capas más bajas del sistema.

Nuestro sensor lógico, al igual que el sistema de fusión que se va a diseñar, se basa en el paradigma orientado a objetos. Los primeros ejemplos de utilización del paradigma orientado a objetos aplicado a los sensores lógicos pueden encontrarse en [Henderson 87; Allen 87]. Otra aproximación a la utilización de objetos como sensores lógicos se encuentra también en [Zeng 89]. Un ejemplo actual de una definición orientada a objetos de sensores y actuadores (el LSA, *Logical Sensor Actuator*) puede encontrarse en [Budenske 97].

El diseño orientado a objetos de nuestro sistema utiliza la notación de Booch [Booch 94]. Esta metodología de diseño presenta una notación unificada que engloba y, en la medida de lo posible, elimina las diferencias estéticas con otras notaciones como la de Jacobson o la OMT (*Object Modeling Technique*) de Rumbaugh [Rumbaugh 91]. En los últimos años, como consecuencia de la necesidad de unificación de las diversas notaciones existentes (Booch, Rumbaugh y Jacobson) ha surgido la metodología UML (*Unified Modeling Language*) [Douglass 98], enfocada a dar unas directivas generales al análisis y al diseño orientado a objetos. Sin embargo, se ha preferido utilizar la notación de Booch ya que, como una notación clásica, se encuentra muy extendida y el análisis y diseño orientado a objetos con ella será mucho más familiar a cualquier diseñador.

A partir de los conocimientos adquiridos en las secciones precedentes y utilizando los beneficios que ofrece la metodología orientada a objetos, es posible diseñar un sistema sensorial y un sistema de fusión de forma general y versátil. Los beneficios buscados de la definición de nuestro sensor lógico mediante el paradigma orientado a objetos son:

- Abstraer los detalles de funcionamiento y los detalles del sensor físico del sistema de control de alto nivel.

- Permitir una reconfiguración más fácil del sistema sensorial.
- Proporcionar una mayor tolerancia a los fallos del sensor.
- Capacitar la adición y supresión de sensores de forma sencilla.
- Inducir al desarrollo incremental o por capas del sistema sensorial.

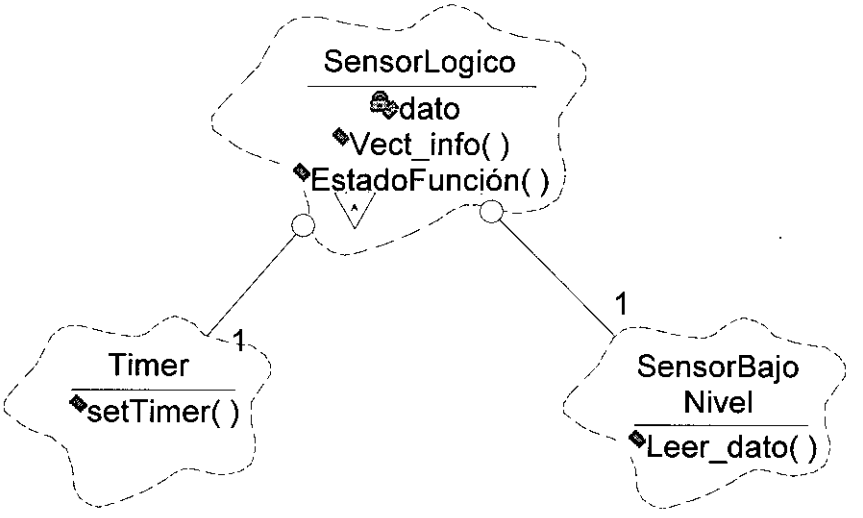


Figura 2-5. Clase abstracta *SensorLógico*.

El sensor lógico se representa como una clase abstracta, figura 2-5, cuyas funciones se encargan del mantenimiento del sistema sensorial, de la adquisición de información y de la comunicación con el sistema de fusión. Por tanto, la clase abstracta *SensorLógico* proporciona:

- Interfaz con el sensor utilizado. Se encarga de recibir de (o pedir, según su funcionamiento) la clase *SensorBajoNivel* la información correspondiente a los datos observados por cada sensor físico (función *leer_dato()*).
- Interfaz con el sistema de fusión. Transmite los datos de cada sensor al sistema de fusión en el formato necesario para su correcta interpretación y fusión (función *Vect_info()*). También puede recibir información procedente del sistema de fusión dirigida al sensor.
- Interprete de órdenes. Se encarga de traducir las órdenes o peticiones de información del sistema de fusión a los sensores lógicos correspondientes. Estas pueden ser órdenes de

mantenimiento como la creación, adición, eliminación o anulación de un sensor; o las encargadas de configurar su funcionamiento, como son el periodo de muestreo requerido, centrar la atención en zonas determinadas, buscar información concreta, etc. Estas órdenes pueden traducirse en un funcionamiento distinto del sensor de bajo nivel (función *EstadoFunción()*).

- Sincronismo. Utiliza un temporizador para sincronismo, adquisición del tiempo actual, pedir información a los sensores periódicamente, etc. Para ello usa los servicios de una clase Timer (*setTimer()*).

Como se puede deducir de la revisión de los distintos tipos de sensores utilizados en un robot móvil mostrada en secciones anteriores, podemos distinguir entre dos tipos de informaciones claramente diferenciadas, como son la de localización y la de entorno. Esto nos lleva a definir dos clases principales de sensor a partir de la clase abstracta sensor lógico: la clase sensor de entorno (*SenEntorno*) y la clase sensor de estado (*SenEstado*), mostradas en la figura 2-6.

- Sensor de entorno: Hereda las propiedades y funciones de un sensor lógico, pero además usa los servicios de un sensor de bajo nivel que se encarga de utilizar el sensor físico, procesa la información de entorno recibida y ofrece su estima al sistema de fusión de entorno. La información enviada al sistema de fusión consiste en un vector de características con su probabilidad de estima (la función *VectEntorno()*, redefinición de la función virtual *Vect_info()*). Este vector es obtenido a partir de la correspondencia realizada entre los datos obtenidos por la clase *SensorBajoNivel* y la base de características que posee.
- Sensor de estado: Se encarga de medir las variables correspondientes al estado del robot (posición, velocidad, orientación, etc.). Al igual que el sensor de entorno, hereda las funciones del sensor lógico y utiliza los servicios de un sensor de bajo nivel. Pero el sensor de bajo nivel se centra en medir variables cuyo preprocesamiento es mucho menor que en el sensor de entorno. Este sensor no necesita una base de conocimiento, pero sí requiere el modelo del sistema y el modelo del entorno. Con él puede predecir o estimar el estado del móvil (*VectEstado()*) a partir de las medidas tomadas.

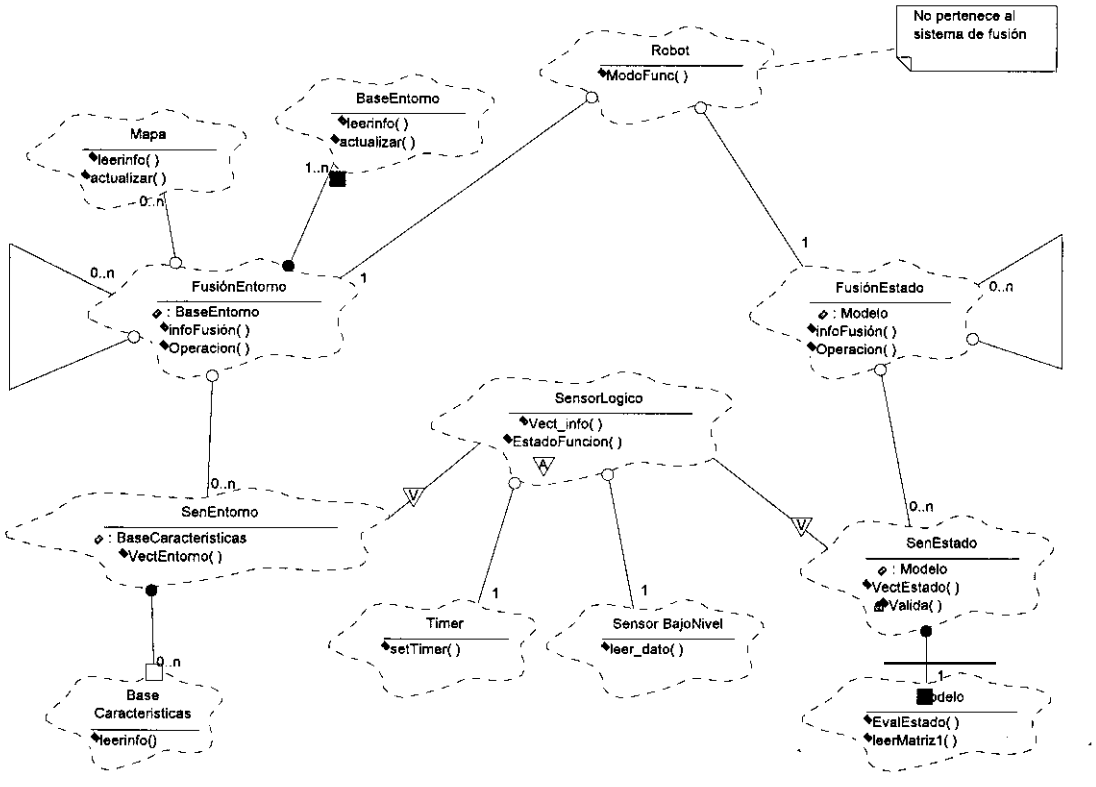


Figura 2-6. Diagrama de clases.

Como puede verse en la figura 2-6, la información de cada una de las clases de sensor será tratada por un módulo de fusión distinto (podría utilizarse una única clase de fusión, pero es más natural la división del sistema de fusión en dos módulos diferenciados). Así, el robot dispone de dos mecanismos de fusión de datos dependiendo de si desea información sobre el estado del robot o sobre el entorno donde se encuentra y permite la actualización de la información de cada tipo con velocidades distintas. Los algoritmos de fusión utilizados en cada uno de los métodos serán estudiados en capítulos posteriores, pero brevemente son:

1.- Sistema de fusión de medidas de estado (clase *FusiónEstado*): se encarga de realizar la integración de las medidas numéricas correspondientes al estado del móvil (estima de mejor posición, velocidad, etc.). Las variables de interés se agruparán en un vector de estado y la integración de los vectores de estado estimados por cada uno de los sensores se integrarán en una estima mejor mediante un filtro de Kalman, como se verá en el Capítulo 3.

2.- Sistema de fusión de medidas de entorno (*clase FusiónEntorno*): se ocupa de la integración de las medidas procedente de los sensores que perciben del entorno que rodea al robot. Los objetos detectados o reconocidos por cada uno de los sensores se integran en una sola estima mediante una red de Bayes. El procedimiento para ello se estudia en detalle en el Capítulo 4.

2.4.3. Caso de utilización (escenario)

A continuación se muestra un sencillo ejemplo con dos sensores de entorno y dos de estado. Este ejemplo muestra la interacción entre el sistema de fusión y el sistema sensorial, haciendo hincapié en los sensores lógicos definidos.

Un diagrama de los objetos que componen el sistema de ejemplo puede verse en la figura 2-7. En este diagrama, por simplicidad, sólo se muestran los objetos necesarios para entender el mecanismo de fusión, es decir, aquellos que influyen directamente en el funcionamiento global del sistema. Como se puede ver se han ocultado todos los objetos que se encuentran contenidos en otros, éstos son: la clase *Modelo*, la clase *BaseCaracterísticas* y la clase *baseEntorno*. Se sobreentiende que se están utilizado normalmente, pero no se indica su uso explícitamente para que el funcionamiento global pueda seguirse sin dificultad. También se han indicado únicamente el inicio de los mensajes que establecen un "canal de comunicación" o "petición de una tarea" entre clases, pero no se muestra ni la información devuelta ni los mensajes entre objetos para cada una de las medidas tomadas por los sensores (al ser repetidos), sino que se supone que se realizan las operaciones indicadas para cada lectura de los sensores.

El funcionamiento general de los sensores y del sistema de fusión es el siguiente:

- El robot indica al sistema de fusión que necesita información de entorno y de posición (estado). Puede indicar que sea periódica o por petición.
- El sistema de fusión tiene dos objetos que, cuando adquieren nueva información o es requerida por el robot según su funcionamiento, envían el dato fusionado al robot (*fEstPos* y *fEntMapa*). El sistema de fusión lo componen como mínimo, estos dos objetos y otro más por cada sensor lógico utilizado.

- Cada nodo de fusión pide datos a su correspondiente sensor lógico y cuando los reciben, los tratan para obtener una estima mejor. El método de fusión es diferente según sean datos de estado o de entorno.
- Si es un objeto de fusión de estado, primero se realiza una estima local (que envía a los demás nodos de fusión) y a continuación mejora esta estima con las estimas recibidas procedentes de otros nodos de fusión. Una vez tiene la estima mejorada la envía al objeto de fusión superior para que se la ofrezca, si procede, al robot.
- En los objetos de fusión de entorno, la predicción local se realiza a partir de la información recibida del sensor lógico y de la base de entorno (similar a una memoria o base de conocimiento) que posee. Una vez realizada la estima final, la comunica a los nodos de fusión para que la utilicen para mejorar su propio conocimiento del entorno. El objeto superior posee una base de entorno donde almacena todo el conocimiento obtenido a partir de las estimas de todos los nodos de fusión de entorno. Este conocimiento es enviado al robot cuando lo solicita.

Como complemento al diagrama de clases puede observarse su correspondiente diagrama de traza de mensajes, figura 2-8, donde pueden verse los mensajes enviados entre cada objeto y su funcionamiento paralelo en el sistema.

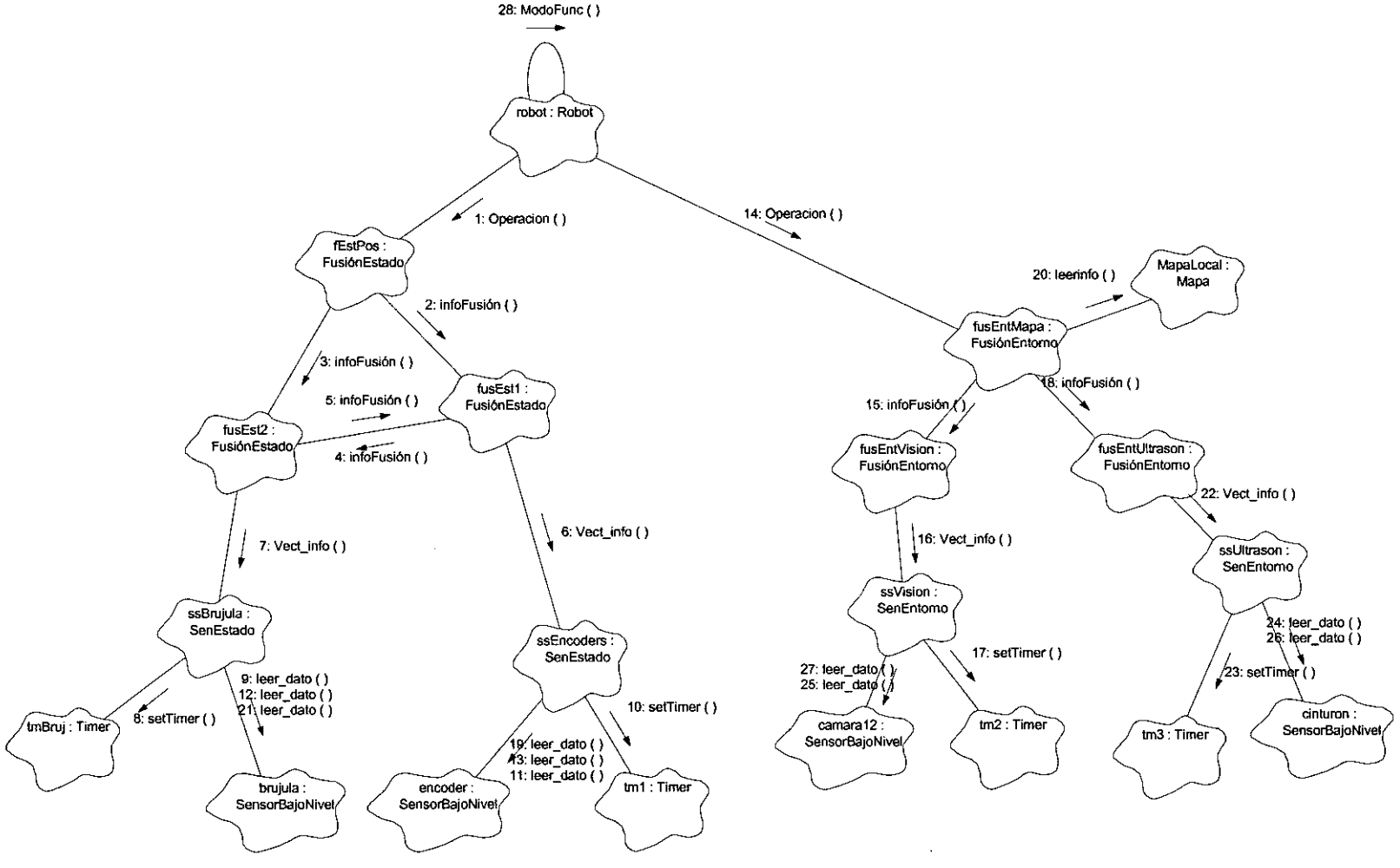


Figura 2-7 . Diagrama de mensajes de los objetos del ejemplo propuesto.

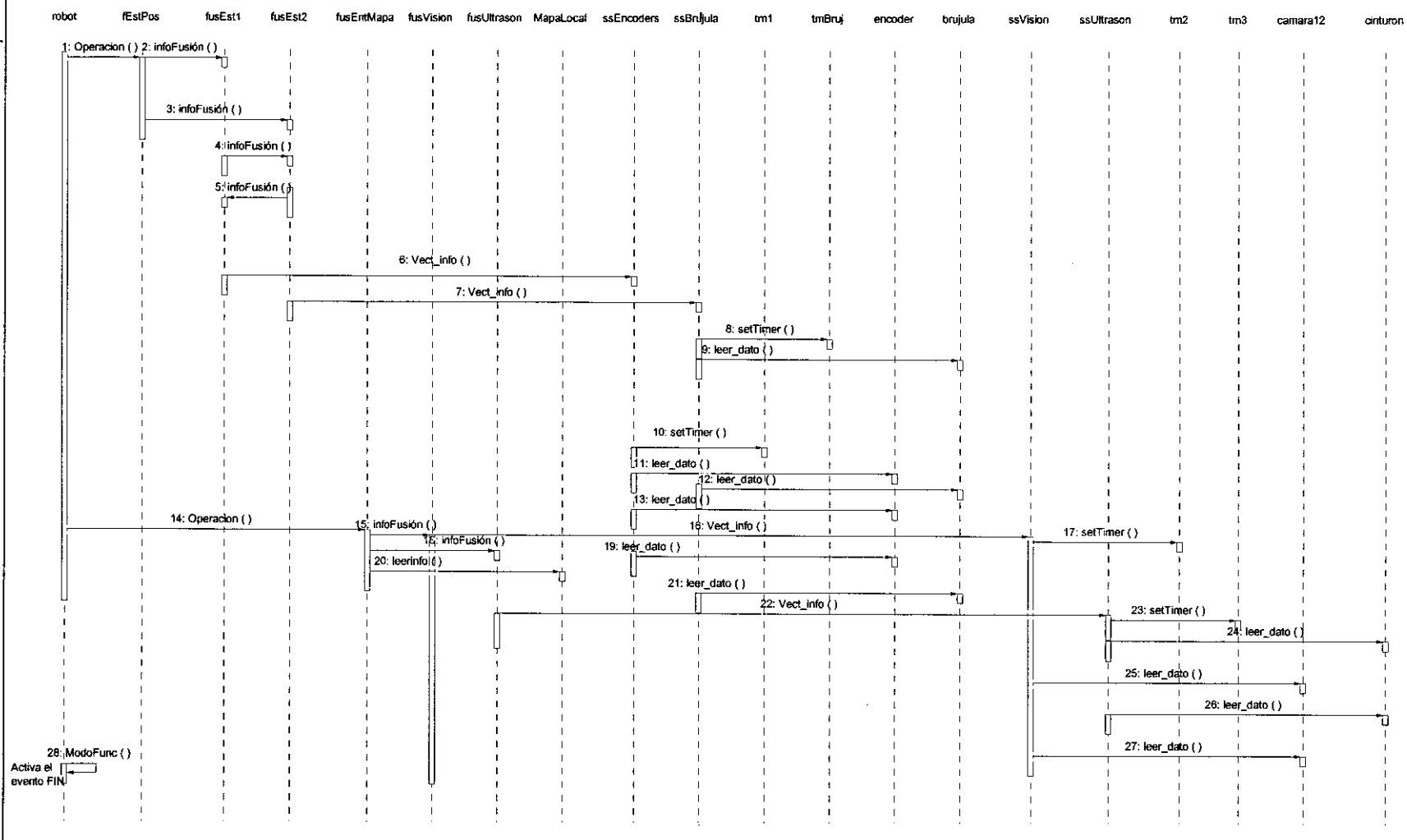


Figura 2-8. Diagrama de traza de mensajes del ejemplo propuesto.

2.5. Resultados del capítulo

En este capítulo se han mostrado los sensores utilizados comúnmente en un robot móvil terrestre. Se ha hecho hincapié en el tipo de información que pueden ofrecer, clasificándola en dos tipos principales: de estado y de entorno. A partir de esta clasificación se ha diseñado, utilizando el paradigma de la metodología orientada a objetos, un sensor lógico que engloba las propiedades comunes de estos sensores. Finalmente, utilizando este sensor lógico, se ha diseñado la estructura de un sistema de fusión que permite fusionar la información procedente de los distintos sensores del robot.

La estructura del sistema diseñado permite que sea independiente del número de sensores utilizados y que éstos puedan trabajar en paralelo y en una arquitectura distribuida. Los algoritmos de fusión necesarios para conseguir las especificaciones indicadas en este capítulo serán desarrollados en los capítulos siguientes.

Capítulo 3

Fusión multisensorial de medidas de posición

En este Capítulo se desarrolla un método de fusión distribuido, asíncrono y robusto de vectores de estado. En concreto se estudia la fusión de medidas de posición procedentes de uno o varios sensores.

Inicialmente se presenta una breve descripción de los algoritmos de fusión existentes que nos conduce a la descripción del algoritmo utilizado: el filtro de Kalman. Se utiliza un filtro extendido distribuido que se basa en el filtro de Kalman distribuido diseñado por Rao, pero al que se le han realizado dos importantes modificaciones: (1) mejora del método de validación externa para adaptarlo a cualquier sensor y (2) adaptación del filtro para su funcionamiento asíncrono.

Además se ha hecho hincapié en su funcionamiento distribuido y robusto, mostrando que el sistema puede funcionar y evolucionar correctamente aunque se produzca un fallo en alguno de los sensores disponibles. Para ello se supone un robot móvil con tres sensores típicos (odómetro, brújula magnética y captadores de balizas artificiales). Con este ejemplo, representativo de cualquier robot, se realizan distintas simulaciones para poner de manifiesto el funcionamiento y la eficacia de los algoritmos desarrollados frente a diversas situaciones. Por ejemplo ante el fallo en la adquisición de medidas o errores en la comunicación entre sensores.

3.1 Introducción

En este Capítulo se presenta un algoritmo o sistema distribuido y asíncrono para la fusión de las medidas de posición procedentes de distintos sensores de un robot móvil. Como ya se ha indicado en el Capítulo 2, existen multitud de sensores que proporcionan información referente a la localización del robot, véase por ejemplo [Everett 95], [Gonzalez 96] o [Borenstein 97]. Estos sensores pueden situarse sobre el vehículo o distribuirse geográficamente, pudiéndose cambiar por otros sensores de posición con una sencilla modificación y tratarse como un sensor lógico descrito en el capítulo anterior sin prejuicio para el sistema implementado. Pero además de la flexibilidad en la elección de los sensores se persigue que este método de fusión sea robusto ante fallos o ante la eliminación de un sensor. Por ello, un funcionamiento distribuido y asíncrono del sistema de fusión es el más indicado porque permite que el sistema actúe correctamente ante distintas anomalías y con independencia del número y tipo de los sensores utilizados.

El sistema de fusión de medidas de posición que se presenta se basa en la teoría de la estimación y utiliza el filtro de Kalman distribuido diseñado por Rao, pero con unas modificaciones motivadas básicamente por dos causas:

1.- Generalizar su aplicación a cualquier sensor o tipo de sensor; para ello se utilizará un filtro de Kalman extendido y se modifica la validación externa para adaptarla a cualquier modelo de sensor deseado.

2.- Convertirlo en un sistema de fusión distribuido totalmente asíncrono; para ello se adaptarán las ecuaciones desarrolladas de forma que se tenga en cuenta el retardo en la transmisión de la información entre sensores.

La organización de este capítulo es la siguiente. En la sección 3.2 se revisan los algoritmos más utilizados para realizar fusión de medidas de posición. Se destaca la asociación de datos como el preprocesamiento imprescindible para garantizar una correcta fusión y se estudia la fusión de medidas de posición como un caso particular de estimación de vectores de estado. En la sección 3.3 se muestra el método escogido para realizar la fusión (un filtro de Kalman) y cómo se adapta para su funcionamiento distribuido, desarrollando una nueva forma de cálculo para validar datos procedentes de otros sensores de una forma general. En la sección 3.4 se modifica el algoritmo de Kalman distribuido para que funcione de

forma asíncrona. Finalmente, en la sección 3.5 se realizan distintas simulaciones del sistema para mostrar la efectividad de los algoritmos utilizados. En ellas se compara nuestro filtro con el filtro de Kalman centralizado mostrando su funcionamiento frente a un filtro óptimo y destacando la robustez de nuestro sistema frente a fallos y a la supresión de alguno de los sensores.

3.2 Algoritmos de fusión posicional

El cálculo de la posición, cinemática o estado de un móvil a partir de nueva información se conoce como estimación de estado, y es la base de la fusión de múltiples medidas referentes al estado de un objeto. Las medidas a fusionar pueden provenir de un único sensor o de múltiples sensores y ofrecer información, independiente o no, sobre la dinámica de un mismo objeto o sobre distintos objetos dinámicos.

En el caso de múltiples sensores, estos pueden encontrarse incluidos en una misma plataforma o distribuidos geográficamente. Esto conduce a diferentes filosofías en cuanto a recursos computacionales y de comunicación. Un sistema sensorial distribuido inducirá a un procesamiento y fusión local de los datos y a un funcionamiento autónomo de los sensores.

Este capítulo se centra en el caso de múltiples sensores, distribuidos o no geográficamente, observando un único estado, en concreto el estado del robot móvil. No se estudia la problemática de seguir múltiples objetos [Bar-Shalom 90], aunque este método podría ser utilizado para ello. En el seguimiento de objetos el planteamiento es diferente al de este trabajo, centrándose principalmente en el problema de asociación de datos y en objetos con una alta dinámica y cambios bruscos de trayectoria. Esta problemática, de gran interés científico, es utilizada ampliamente en aplicaciones militares como por ejemplo [Libby 96; Shetty 96 ó Yeddanapudi 97].

Los procesos que se encuentran directamente relacionados con la obtención (y fusión) de información de posición o cinemática son: la asociación de datos y la estimación de estados.

3.2.1 Asociación de datos

La asociación de datos se encarga de encontrar la correspondencia entre observaciones de múltiples sensores (o de un único sensor tomando datos a lo largo del tiempo) y los individuos u objetos identificados. Así, la asociación permite encontrar correspondencias:

- entre observaciones,
- entre observaciones y objetos ya detectados, o

- entre distintos objetos.

Las dos últimas correspondencias son propias de sistemas de seguimiento o vigilancia y se centran más en el reconocimiento de objetos. La primera es la que nos interesa en este momento y es aplicable tanto a seguimiento como a estimación de estados.

Los métodos de decisión aplicables en la asociación de datos pueden clasificarse en:

- Métodos heurísticos, basados en una simple regla o en alguna función de clasificación.
- Métodos probabilísticos no bayesisanos, basados en test de hipótesis o en funciones de verosimilitud. En estos casos una vez que se toma la decisión (sea o no correcta) no hay vuelta atrás, este tipo de decisiones se conocen como *duras*.
- Métodos probabilíticos bayesianos, basados en probabilidades (condicionales) a posteriori que son calculadas mediante la fórmula de Bayes a partir de probabilidades a priori. Estas decisiones son *blandas*, puesto que su resultado se utiliza en decisiones posteriores y permite el cambio de opinión.

La medida de asociación determina la proximidad entre dos observaciones, es decir, determina si dos observaciones pueden declararse como relativas a la misma identidad física, esto es, si proceden de una fuente común.

Las medidas de asociación incluyen coeficientes de correlación, medidas de distancia, coeficientes de asociación o medidas de similitud probabilística, y pueden ser utilizadas para asociar medidas referentes a vectores de estado o a vectores de características.

Para la fusión de medidas de posición utilizaremos un método probabilístico no bayesiano, en concreto utilizaremos un test de aceptación. El test o contraste utilizado para comprobar si la medida recibida es aceptada o rechazada será un test Chi-cuadrado. Se elige así por varios motivos:

- Los test de hipótesis son muy apropiados cuando no existen modelos físicos y lo que necesitamos es determinar si las observaciones satisfacen un cierto nivel de confianza.

- La distribución Chi-cuadrado representa aquella distribución resultante de la suma de cuadrados de distribuciones gaussianas. Por ello, si se considera que el ruido de los sensores puede modelarse como un ruido gaussiano de media cero, y se realiza un test sobre la norma de los residuos de las observaciones que deseamos asociar, será apropiado utilizar un test Chi-cuadrado como contraste para la aceptación o rechazo de las medidas.
- El test es muy sencillo de aplicar con unos requerimientos de cálculo muy bajos, lo que resulta muy conveniente para las altas frecuencias de muestreo de los sensores de posición de un robot móvil.

Este será el método utilizado en nuestro sistema para realizar la asociación de datos de las medidas de posición procedentes de los distintos sensores de posición utilizados en nuestro móvil. Puesto que este test se utiliza para comprobar si una medida de posición debe ser utilizada en el proceso de fusión lo denominamos *test de validación* o simplemente *validación*.

3.2.2 Estimación de estado

Mediante la estimación se busca el vector de estado que mejor se ajuste en todo momento a los datos observados. En nuestro caso nos referimos a medidas cuantitativas o relativas al estado del móvil (posición, orientación, velocidad,...). Existen multitud de técnicas de estimación, pero siempre se debe tener en cuenta cuatro aspectos importantes:

1.- *El modelo del sistema*. Se debe diseñar correctamente el modelo del sistema, donde se incluyen las ecuaciones de observación, ecuaciones de movimiento, la dinámica del sistema, las variables del vector de estado, etc.

2.- *El criterio de optimización*. Se debe especificar cual es el mejor ajuste de los datos al modelo y cómo se mide tal ajuste. Entre los criterios existentes cabe destacar los mínimos cuadrados, los mínimos cuadrados ponderados, el criterio de máxima verosimilitud o restricciones bayesianas.

3.- *Método de optimización.* Una vez definido el criterio de optimización, se debe decidir cómo encontrar el vector de estado desconocido que mejor se ajusta a los datos. Métodos de optimización son los directos, ya sean no derivativos (simulated annealing, algoritmos genéticos, ...) o derivativos (gradiente, quasi-Newton, ...); y los métodos indirectos como por ejemplo el método de Newton-Raphson (que es también derivativo).

4.- *Método de procesamiento.* También debe decidirse cómo serán procesados los datos, es decir, se debe escoger entre procesamiento batch (después de obtener todos los datos) o procesamiento secuencial (cada observación se procesa según es recibida, actualizando el vector de estados).

A continuación se describe qué técnica se ha escogido en cada caso.

3.2.2.1 El modelo del sistema

La elección de qué estimar depende de la definición de qué parámetros son necesarios para determinar el estado futuro y de qué parámetros de nuestro sistema pueden ser estimados a partir de los datos observados (observabilidad).

Para un sistema no lineal discreto, normalmente las ecuaciones de movimiento de un robot móvil son no lineales, la predicción del vector de estado y de la observación vendrá dada por,

$$\hat{x}(k) = f(\hat{x}(k-1), u(k), k) + G(k) \cdot w(k) \tag{3.1}$$

$$\hat{z}(k) = h(\hat{x}(k), k) + v(k) \tag{3.2}$$

La ecuación (3.1) representa el modelo de transición de estado del robot, donde $f(\_)$ es el modelo no lineal del sistema, $G(k)$ el modelo del ruido de los estados y $w(k)$ la entrada de ruido.

La ecuación (3.2) representa el modelo de observación de la medida, $z(k)$, en función del estado $x(k)$. $h(\_)$ es el modelo no lineal de observación del sensor y $v(k)$ el ruido correspondiente a dicha observación.

El modelo considerado para el carro será un modelo sencillo, correspondiente a un sistema de dos ruedas independientes. Este modelo puede ser sustituido directamente por otros modelos más complicados siempre que se mantengan los estados utilizados: posición en el plano (x,y) y orientación (θ). En

caso de utilizar otros estados la sustitución del modelo requerirá una manipulación más compleja.

3.2.2.2 El criterio de optimización

El criterio de optimización trata de encontrar el valor del vector de estado $x(k)$ que mejor se ajuste a los datos observados. Sea u_i el residuo o diferencia entre el dato observado por el sensor i y la predicción de los datos que medirá dicho sensor:

$$u_i = [y_i(k) - z_i(k)] \tag{3.3}$$

Para hacer que u_i sea mínimo se utilizan distintos criterios de optimización [Gill 81; Fletcher 87] que se basan en encontrar el vector de estado, $x(k)$, que minimiza (o maximiza) una determinada función. Así, según la función a optimizar tendremos mínimos cuadrados, mínimos cuadrados ponderados (o con pesos), funciones objetivo probabilisticas (criterios bayesianos) y funciones de máxima verosimilitud.

Es conveniente destacar que el criterio de optimización escogido dependerá del conocimiento a priori de los datos observados. En nuestro caso utilizaremos para el tipo y ruido de los datos, las siguientes condiciones (ampliamente utilizadas en la literatura):

a) El error de medida (los residuos encontrados) tienen una distribución Gausiana de media cero,

$$E[w_i(k)] = E[v_i(k)] = 0 \tag{3.4}$$

b) Los errores y ruidos utilizados en el modelo son independientes en distintos instantes de muestreo,

$$\begin{aligned} E[w_i(k) \cdot w^T{}_i(j)] &= Q_i(k) \cdot \delta_{k,j} \\ E[v_i(k) \cdot v^T{}_i(j)] &= R_i(k) \cdot \delta_{k,j} \end{aligned} \tag{3.5}$$

c) El ruido de estado y el de observación son independientes entre si,

$$E[w_i(k) \cdot v_i^T(j)] = 0 \tag{3.6}$$

Bajo estas condiciones (y si el peso tomado en los mínimos cuadrados ponderados es la inversa de la matriz de covarianza de x) todos los criterios de optimización indicados anteriormente son equivalentes.

El método de optimización utilizado por nosotros está basado en el de mínimos cuadrados ponderados (el filtro de Kalman).

3.2.2.3 Método de optimización

Pueden utilizarse distintas técnicas para encontrar el valor del vector de estado que hace óptimo alguno de los criterios anteriores. Existen multitud de algoritmos, pero pueden clasificarse en dos grandes grupos: métodos directos y métodos indirectos [Hall 92].

Los métodos directos tratan el criterio de optimización sin modificarlo para encontrar un valor de x que haga mínimo (o máximo) el criterio utilizado. Estos métodos se basan fundamentalmente en técnicas "hill-climbing".

Los métodos indirectos tratan de resolver simultáneamente las m ecuaciones no lineales. Los métodos indirectos necesitan que el criterio de optimización sea explícitamente diferenciable con respecto a x, lo que en muchas ocasiones no es factible.

Implementaciones de algunos de estos algoritmos que permiten probar muchas de estas técnicas pueden encontrarse en [Press 94].

3.2.2.4 Método de procesamiento

Los datos tomados pueden procesarse de dos formas: por lotes o secuencialmente. El procesamiento por lotes o procesamiento batch, parte de la suposición de que todos los datos están disponibles en el momento de realizar la estimación con alguno de los métodos vistos anteriormente. El ejemplo más clásico de estimación por lotes es en problemas de mejor ajuste de curvas en el caso de la modelización de sistemas o en la estimación de parámetros [Goodwin 77; Astrom 80].

Por el contrario, el procesamiento secuencial se utiliza cuando no es posible disponer de todos los datos en el instante de realización de la estimación. En este caso, la estima del vector de estado, $x(k)$, se actualiza con cada llegada de una nueva observación. En la estima secuencial se tienen en cuenta las $n+1$ observaciones, pero no de forma conjunta como en el proceso por lotes, sino que se tienen en cuenta por medio de la influencia de $x(k)$, que ya tuvo en cuenta las n observaciones anteriores, y la nueva observación $y(k+1)$.

Los principales inconvenientes de la estimación secuencial son sus problemas en la asociación de datos, la divergencia (en la cual el estimador ignora nuevos

datos) y la inicialización. A pesar de ello, los estimadores secuenciales son muy utilizados en seguimiento de objetos y estimación de la posición.

El ejemplo más clásico de estimación secuencial es el filtro de Kalman [Gelb 74; Anderson 79; Grewal 93]. Con ciertas modificaciones con respecto a su funcionamiento clásico, es el algoritmo utilizado en esta tesis para desarrollar el sistema de fusión de medidas de estado y será descrito a continuación.

3.3 Fusión de medidas de posición mediante un filtro de Kalman

Una vez que cada uno de los distintos sensores disponibles obtiene la información referente a la posición del móvil, es necesario integrar estas medidas en una única posición. Para ello utilizamos un filtro de Kalman. Tenemos dos posibilidades de fusión: podemos proporcionar los datos a un sistema central que calcule la mejor estima posible y después devolver a cada uno de los sensores esta estima (filtro de Kalman centralizado) o podemos hacer que cada uno calcule la estima global a partir de los datos calculados por todos sin necesidad del procesador central (filtro de Kalman distribuido).

En concreto nos centraremos en una arquitectura totalmente descentralizada para la fusión de datos procedentes de múltiples sensores. El punto de inicio de nuestro sistema es un filtro de Kalman centralizado extendido, ampliamente difundido en la literatura [Gelb 74; Anderson 79; Grewal 93], a partir del cual se obtienen las ecuaciones necesarias para obtener un filtro de Kalman distribuido [Hasemipour 88; Durrant-Whyte 90; Rao 93].

Con el filtro de Kalman distribuido conseguiremos reducir la cantidad de información transmitida en el sistema, permitiremos que cada uno calcule una estima global independientemente y evitaremos ese nodo de fusión global que podría fallar y hacer que todo el sistema fallase.

Además la comunicación será asíncrona para permitir que si un sensor falla el sistema pueda seguir evolucionando y los demás no se vean afectados por el error de éste. De esta forma conseguimos los dos objetivos que consideramos primordiales: que el sistema de fusión sea robusto frente a fallos de los sensores; y que el sistema sensorial sea independiente del número de sensores utilizados.

Pero antes de describir este filtro, revisamos brevemente las ecuaciones de un filtro de Kalman centralizado, como introducción a la nomenclatura utilizada.

3.3.1 Filtro de Kalman extendido centralizado

El filtro de Kalman es una de las herramientas más utilizadas en la literatura para fusión sensorial en robots móviles. El filtro de Kalman clásico utiliza un modelo lineal del sistema, pero en aquellos casos en los que el sistema no puede ser representado de forma adecuada con un modelo lineal, como típicamente

ocurre en el modelo de movimiento de un robot (que es nuestro caso), se emplean filtros de Kalman modificados. La modificación más utilizada es una linealización de primer orden de las ecuaciones no lineales del modelo, que se conoce como filtro de Kalman extendido (EKF).

En un sentido general, el filtro de Kalman puede considerarse como un estimador de estados, es decir, calcula la estima o valor esperado del estado $\hat{x}(k)$ y su error cuadrático medio en el instante k a partir:

a) del estado en el instante *k-1*, calculado a partir de todas las observaciones tomadas hasta el instante *k-1*,

b) de la nueva observación en el instante k, y,

c) de la predicción del estado $\bar{x}(k)$, correspondiente al modelo del sistema considerado.

Lo que se persigue es encontrar la forma de calcular, para todo instante k≥j, la estima o valor esperado del estado $\hat{x}(k)$ en el instante k a partir de todas las observaciones tomadas hasta el instante j, y además calcular el error cuadrático medio de esta estima. Esto es,

$$\hat{x}(k \mid j) \equiv E[x(k) \mid z(1), z(2), \ldots, z(j)] \tag{3.7}$$

$$P(k \mid j) \equiv E[(x(k) - \hat{x}(k \mid j)) \cdot (x(k) - \hat{x}(k \mid j))^T \mid z(1), z(2), \ldots, z(j)] \tag{3.8}$$

Por tanto, el filtro de Kalman encuentra una descripción matemática de las expresiones anteriores en función del modelo del sistema y del modelo de las observaciones. Es decir, encuentra una solución recursiva a la estima del estado *x(k)*, llamada $\hat{x}(k \mid k)$, en términos de la estima en un instante anterior, $\hat{x}(k-1 \mid k-1)$, y de la nueva observación tomada, *z(k)*.

3.3.1.1 Modelo del móvil

Se utilizará para el modelo de movimiento del robot de nuestro sistema un sencillo modelo no lineal en el que el vector de estado utilizado representa su posición y orientación,

$$\vec{x}(t_i) = (x(t_i), y(t_i), \vartheta(t_i)) \tag{3.9}$$

El modelo de estado del móvil puede ser distinto para cada sensor y depende de la dinámica del robot, el tipo de ruedas motrices utilizadas, la señal de control necesaria, etc. Por ejemplo pueden utilizarse modelos como los utilizados por [Nelson 88] o [Kam 97] donde la señal de control es $u = [V, \gamma]$ (V, es la velocidad

lineal del móvil y γ el giro demandado. Otros modelos como el utilizado en [Hongo 87; Feng 93 o Manyika 94] utilizan las velocidades de cada una de las ruedas independientemente $u=[v^{izq}, v^{der}]$. Este es el modelo continuo utilizado:

$$\begin{aligned}\dot{x}(t) &= \frac{v^{izq}+v^{der}}{2}\cdot\cos\vartheta(t)\\ \dot{y}(t) &= \frac{v^{izq}+v^{der}}{2}\cdot\operatorname{sen}\vartheta(t)\\ \dot{\vartheta}(t) &= \frac{v^{izq}-v^{der}}{b}\end{aligned} \tag{3.10}$$

donde v^{izq} y v^{der} son las señales de control al móvil y corresponden con las velocidades demandadas para la rueda izquierda y la rueda derecha. Este modelo, que será tratado con más detalle en la sección 3.5.1, viene dado por las ecuaciones discretas:

$$x(k+1)=f(k,x(k),u(k)) \tag{3.11}$$

$$f(k,x(k),u(k))=\begin{bmatrix} x(k)+T(k)\cdot\cos\vartheta(k)\\ x(k)+T(k)\cdot\cos\vartheta(k)\\ \vartheta(k)+\Delta\vartheta(k)\end{bmatrix} \tag{3.12}$$

El modelo de observación dependerá de cada sensor utilizado y podrá ser lineal o no. Más adelante se mostrará el modelo de observación utilizado para los sensores empleados en nuestro robot.

Considerando las ecuaciones no lineales del sistema y de observación de un modo general, como las indicadas en (3.1) y (3.2), las ecuaciones del filtro de Kalman son básicamente las mismas que en un sistema lineal y sólo se modifican ligeramente: se realiza una linealización de primer orden del sistema cuando es necesario, manteniendo básicamente las mismas ecuaciones. Así, la linealización de las ecuaciones (3.1) y (3.2), donde el modelo de transición de estado correspondiente será el presentado en la ecuación (3.12), serán:

$$F_i^1(k)\approx\left.\frac{\partial f(x,u,k-1)}{\partial x}\right|_{x=\hat{x}_i(k|k-1)} \tag{3.13}$$

$$H_i^1(k)\approx\left.\frac{\partial h(x,k-1)}{\partial x}\right|_{x=\hat{x}_i(k|k-1)} \tag{3.14}$$

3.3.1.2 Ecuaciones del filtro

Los pasos en el cálculo de un filtro extendido y centralizado son:

a) Predicción:

Se utiliza la estima del estado conocida para, a partir del modelo del sistema y de medida, predecir cual debería ser el estado y la medida en el instante actual.

$$\hat{x}(k \mid k-1) = f(\hat{x}(k-1 \mid k-1), u(k-1), k) \tag{3.15}$$

$$\hat{z}(k \mid k-1) = h(\hat{x}(k \mid k-1), k) \tag{3.16}$$

También se calcula, a partir de la linelización, la estima del error cuadrático medio cometido en dicha predicción:

$$P(k \mid k-1) = F^{1}(k) \cdot P(k-1 \mid k-1) \cdot F^{1^{T}}(k) + G(k) \cdot Q(k) \cdot G^{T}(k) \tag{3.17}$$

a) Observación:

Una vez realizada la predicción se toman nuevas medidas (observaciones). Cualquier sensor tendrá incertidumbre en sus medidas, lo que viene modelado por $v(k)$ en la ecuación (3.2). A pesar de ello, las nuevas medidas, si son válidas, permitirán realizar una estima del estado mucho más precisa que la realizada con la ecuación (3.15). Estas observaciones son denominadas, $z_i(k)$.

b) Validación:

Además del modelo de error de los sensores pueden ocurrir otros errores o imprecisiones que pueden dar lugar a medidas erróneas. Por ello, es necesario comprobar cada observación tomada para asegurarnos de que es válida, es decir, está dentro de un cierto margen de error. Para realizar la validación utilizaremos,

$$r^{T}(k) \cdot S^{-1}(k) \cdot r(k) \leq \gamma \tag{3.18}$$

$$r(k) = z_i(k) - \hat{z}(k \mid k-1) \tag{3.19}$$

$$S(k) = R(k) + H^{1}(k) P(k \mid k-1) H^{1^{T}}(k) \tag{3.20}$$

donde $r(k)$ es conocido como innovación y $S(k)$ su covarianza.

La ecuación (3.18) es un contraste estadístico χ^2 para contrastar una cierta hipótesis H_0, en nuestro caso nos preguntamos si es cierto que

$z_i(k) = \hat{z}(k \mid k-1)$. Esto normalmente es interpretado como la definición de una cierta región de validación centrada en el estado de predicción, de forma que cualquier observación que caiga dentro de dicha región será aceptada como válida. La potencia del contraste (el tamaño de la región de validación) vendrá dada por γ y escogido a partir de la tabla de χ^2 una vez que se ha escogido el intervalo de confianza, normalmente de forma empírica [Bendat 71; López-Cachero 88].

c) Actualización:

Una vez que se ha comprobado que las medidas son válidas, sino serían descartadas, se realizará el último paso: se actualiza el filtro con la medidas nuevas a fin de obtener una estima del estado actual lo más precisa posible. Estas ecuaciones son,

$$\hat{x}(k \mid k) = \hat{x}(k \mid k-1) + K(k) \cdot [z_i(k) - \hat{z}(k \mid k-1)] \tag{3.21}$$

$$P^{-1}(k \mid k) = P^{-1}(k \mid k-1) + H^{1^T}(k) \cdot R^{-1}(k) \cdot H^1(k) \tag{3.22}$$

$$K(k) = P(k \mid k) \cdot H^{1^T}(k) \cdot R^{-1}(k) \tag{3.23}$$

donde la ecuación (3.23) representa la matriz de ganancia del filtro de Kalman.

3.3.2 Filtro de Kalman distribuido

Para entender el funcionamiento de este sistema supongamos que disponemos de varios sensores, véase la figura 3-1, que pueden estimar el vector de estados del móvil a partir del modelo que poseen y de las observaciones que realizan.

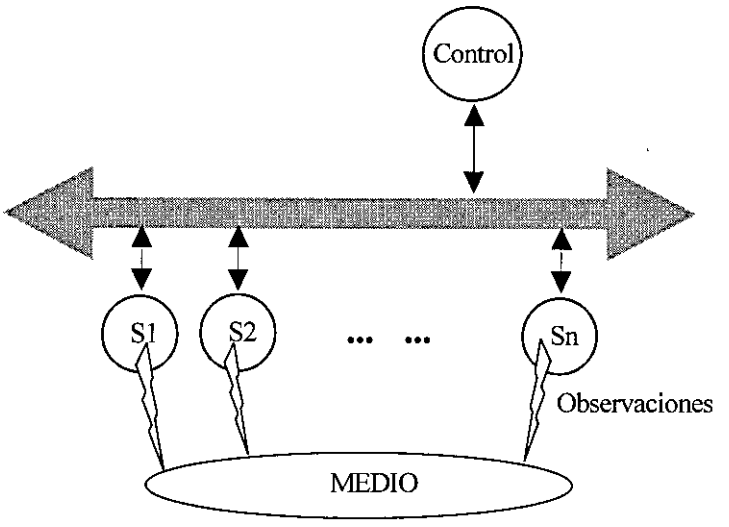


Figura 3-1 . Sensores distribuidos y control

Cada sensor lógico de posición realiza una inicialización, en la que toma el modelo del sistema, de observación y cualquier otro valor que necesite para iniciar su funcionamiento. A partir de aquí el funcionamiento, mostrado en la figura 3-1, sería el siguiente:

1.- El sensor físico realiza una medida de la posición del móvil. Esta medida, después de ser procesada de forma conveniente (preprocesamiento, alineamiento, etc.) es recibida por el sensor lógico de posición correspondiente (véase la figura 3-1).

2.- El sensor lógico de posición, elemento del sistema de fusión distribuido, comprueba que la medida obtenida puede ser utilizada para mejorar la estima que tiene de la posición. En esta validación se utiliza la predicción que cada sensor lógico ha realizado previamente. En la predicción, si su modelo lo necesita, debe incorporar la señal de control que se ha solicitado a los motores y por tanto debe recibir las señales de control que se van enviando a los actuadores de los motores.

3.- Se realiza la estima de la posición del móvil, esta estima es local a cada uno de los sensores.

4.- Cada sensor lógico envía a todos los demás (a todos los que estén a la escucha) datos sobre el estado del robot que ha estimado en ese momento.

5.- Cada sensor lógico recibe la posición que han estimado los demás sensores y a partir de esta información y, siempre que compruebe que esta información es correcta, la utilizará para actualizar y mejorar su propia estima.

6.- Una vez que se confirma qué datos son válidos, estos deben ser fusionados para obtener una estima mejor del estado global.

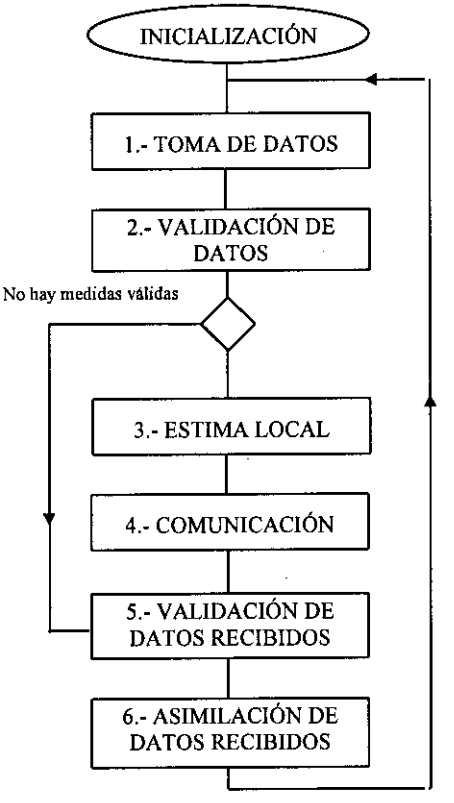


Figura 3-2. Ciclo de fusión.

En todo este proceso debe tenerse en cuenta que los sensores funcionan de forma asíncrona y, por tanto, deben realizarse las correcciones pertinentes para que todos los sensores obtengan la estima correcta. A continuación se muestran las ecuaciones necesarias para realizar cada uno de los pasos indicados.

3.3.2.1 Ecuaciones del filtro y de asimilación

Las ecuaciones del filtro de Kalman distribuido, tienen su base en la descomposición de las ecuaciones centralizadas en m subvectores de diferentes dimensiones m_i correspondientes a las observaciones hechas por cada sensor

individual [Hasemipour 88]. Posteriormente esta descomposición es utilizada para conseguir una arquitectura totalmente descentralizada para fusión multisensorial [Durrant-Whyte 90], pero con comunicación síncrona y sin ninguna validación de los datos filtrados. Finalmente, [Rao 93] modificó las ecuaciones distribuidas anteriores para realizar una comunicación síncrona, donde se realiza una validación de los datos locales y también una validación de los datos recibidos de otros sensores (que denominaremos validación externa). Las ecuaciones para un sistema no lineal discreto, semejantes a las del filtro de Kalman centralizado, son:

a) Ecuaciones de predicción:

$$\hat{x}_i(k|k-1) = f_i(\hat{x}_i(k-1|k-1), u(k), k) \tag{3.24}$$

$$\hat{z}_i(k|k-1) = h_i(\hat{x}_i(k|k-1), k) \tag{3.25}$$

Obsérvese que en este caso las ecuaciones (3.24) y (3.25) difieren de las ecuaciones (3.15) y (3.16) en el subíndice i. Esto indica que estas ecuaciones son locales y son realizadas en el sensor i. Esto ocurre también en las siguientes ecuaciones.

Utilizando las ecuaciones (3.13) y (3.14) para realizar la correspondiente linealización del modelo, obtenemos la matriz de covarianza de la estima que viene dada por,

$$P_i(k|k-1) = F_i^1(k)P_i(k-1|k-1)F_i^{1^T}(k) + G_i(k)Q_i(k)G_i^T(k) \tag{3.26}$$

b) Validación local:

La validación local o interna hace referencia a la comprobación de que el dato observado es coherente con el estado del móvil. Para ello, al igual que en las ecuaciones (3.18)-(3.20) se realizará el contraste de hipótesis de una χ^2 sobre el residuo obtenido de la comparación entre la medida observada y la predicha. Esto es,

$$r_i^T(k) \cdot S_i^{-1}(k) \cdot r_i(k) \leq \gamma \tag{3.27}$$

$$r_i(k) = z^l(k) - \hat{z}_i(k|k-1) \tag{3.28}$$

$$S_i(k) = R_i(k) + H_i(k)P_i(k|k-1)H_i^T(k) \tag{3.29}$$

Podemos ver que la incertidumbre o covarianza del residuo se compone de la incertidumbre en la observación y la incertidumbre transmitida por el estado en la predicción.

c) Ecuaciones de actualización:

$$\tilde{x}_i(k \mid k) = \hat{x}_i(k \mid k-1) + K_i(k) \cdot \left[z_i(k) - \hat{z}_i(k \mid k-1) \right] \tag{3.30}$$

$$P_i^{-1}(k \mid k) = P_i^{-1}(k \mid k-1) + H_i^{1^T}(k) \cdot R_i^{-1}(k) \cdot H_i^1(k) \tag{3.31}$$

$$K_i(k) = P_i(k \mid k) \cdot H_i^{1^T}(k) \cdot R_i^{-1}(k) \tag{3.32}$$

Si el test se supera positivamente se acepta la nueva observación y se corrige la predicción del estado con la nueva medida obteniendo un estima mejor y reduciendo su covarianza. En caso de rechazar la medida ya no se realizará la actualización, ecuaciones (3.30) y (3.31), sino que se utiliza como estima y varianza del estado las prediciones realizadas anteriormente, ecuaciones (3.24) y (3.26).

Finalmente es necesario comunicar la información local obtenida en cada nodo local a los demás sensores. Esta información se envía siempre y cuando se haya validado la medida obtenida por el sensor, puesto que la información relevante que es necesario transmitir es la obtenida debido a las medidas y no las predicciones que realiza ese sensor y que no aporta nada nuevo.

Para determinar qué información debe ser transmitida se descomponen las ecuaciones centralizadas del filtro de Kalman en suma de las distintas contribuciones de los diversos sensores que se poseen y se obtienen las siguientes ecuaciones para la adaptación de la varianza y el estado, una demostración puede encontrarse en [Durrant-Whyte 90]:

$$\hat{x}_i(k \mid k) = P_i(k \mid k) \cdot \left[P_i^{-1}(k \mid k-1)\hat{x}_i(k \mid k-1) + \right.$$
$$\left. + \sum_{j=1}^{m} \left\{ P_j^{-1}(k \mid k)\hat{x}_j(k \mid k) - P_j^{-1}(k \mid k-1)\hat{x}_j(k \mid k-1) \right\} \right] \tag{3.33}$$

$$P_i^{-1}(k|k) = P_i^{-1}(k|k-1) + \sum_{j=1}^{m} \left\{ P_j^{-1}(k|k) - P_j^{-1}(k|k-1) \right\} \tag{3.34}$$

En estas ecuaciones podemos ver que lo único que es necesario transmitir de un nodo a otro es la parte del sumatorio, puesto que corresponde a una información calculada en otro sensor (el sensor j) y por tanto no está disponible en el sensor i. En la ecuación (3.33) el termino del sumatorio correspondiente a cada sensor j se denomina *información del error de estado* y al termino de la ecuación (3.34) *información del error de la varianza*. Tanto uno como otro deben ser transmitidos a todos los otros sensores a fin de que todos puedan calcular la misma estima global.

Estas ecuaciones fueron utilizadas en [Durrant-Whyte 90], pero no tienen en cuenta el problema del asincronismo entre medidas. Para obtener un modelo viable en problemas asíncronos debemos adaptarlas convenientemente. Para ello se parte del modelo de observación y llegamos a las siguientes ecuaciones [Rao 93]:

$$H^{1^T}(k)\cdot R^{-1}(k)\cdot H^1(k)=\sum_{i=1}^{m} H_i^{1^T}(k)\cdot R_i^{-1}(k)\cdot H_i^1(k) \tag{3.35}$$

$$H_i^{1^T}(k)\cdot R_i^{-1}(k)\cdot H_i^1(k)=P_i^{-1}(k\,|\,k)-P_i^{-1}(k\,|\,k-1) \tag{3.36}$$

$$H_i^{1^T}(k)\cdot R_i^{-1}(k)\cdot z(k)=P^{-1}(k\,|\,k)\cdot\hat{x}(k\,|\,k)-P^{-1}(k\,|\,k-1)\cdot\hat{x}(k\,|\,k-1) \tag{3.37}$$

Con estas las ecuaciones y modificando (3.33) y (3.34) obtenemos,

$$\begin{aligned}\hat{x}_i(k\,|\,k)=P_i(k\,|\,k)\cdot\Big[&P_i^{-1}(k\,|\,k-1)\cdot\hat{x}_i(k\,|\,k-1)+\\&+\sum_{j=1}^{m}\left\{H_j^{1^T}(k)\cdot R_j^{-1}(k)\cdot z_j(k)\right\}\Big]\end{aligned} \tag{3.38}$$

$$P_i^{-1}(k\,|\,k)=P_i^{-1}(k\,|\,k-1)+\sum_{j=1}^{m}\left\{H_j^{1^T}(k)\cdot R_j^{-1}(k)\cdot H_j^1(k)\right. \tag{3.39}$$

Estas ecuaciones, equivalentes a (3.33) y (3.34) respectivamente, permitirán que la comunicación asíncrona sea mucho más sencilla de realizar que si se utilizasen las anteriores.

3.3.2.2 Validación de datos externos

La información procedente de otros nodos debe ser validada en el nodo receptor antes de ser utilizada. Esto es necesario porque,

1.- Permite al nodo receptor detectar que un nodo está funcionando incorrectamente y por tanto no se realizará la asimilación de la información que procede de él.

2.- Impide que medidas procedentes de otros nodos sean asociadas con medidas que no le corresponden en el nodo receptor (uno de los principales problemas en el seguimiento de múltiples objetos).

3.- Detecta errores en la transmisión de la información, de forma que no utilizará datos que se hayan corrompido durante la transmisión.

La validación en este caso no es tan sencilla como la validación interna, ecuaciones (3.27), debido a que para realizarla sólo se dispone de la información local propia y de la información que recibe de los otros nodos, ecuaciones (3.33) y (3.34) o ecuaciones (3.38) y (3.39).

Supongamos un sensor i que recibe datos procedentes de otro sensor j para su asimilación. La información que recibe de j será el error de estado, $X_j(k)$, y de covarianza, $Y_j(k)$. Por ejemplo,

$$X_j(k) = H_j^{1^T}(k) \cdot R_j^{-1}(k) \cdot z_j(k) \tag{3.40}$$

$$Y_j(k) = H_j^{1^T}(k) \cdot R_j^{-1}(k) \cdot H_j^1(k) \tag{3.41}$$

La validación en este caso no es tan sencilla como la validación interna debido a que para realizarla sólo se dispone de la información local propia y de la información que recibe de los otros nodos. [Rao 93] propone lo siguiente:

> Para la validación externa se realizará un test sobre las medidas observadas por ambos sensores de igual forma que se realizaba en la validación interna:
>
> $$v_{ji}^T(k) \cdot S_{ij}^{-1}(k) \cdot v_{ji}(k) \leq \gamma \tag{3.42}$$
>
> $$v_{ij}(k) = z_j(k) - {}_j\hat{z}_i(k \mid k-1) \tag{3.43}$$
>
> Donde, $v_{ij}(k)$ es el error entre la medida observada por el sensor j, $z_j(k)$, y la predicción de la medida que observaría el sensor j hecha por el sensor i,
>
> $${}_j\hat{z}_i(k \mid k-1) = h_j(\hat{x}_i(k \mid k-1), k) \tag{3.44}$$
>
> Esto se hace así, debido a que cada sensor puede observar medidas referentes a subespacios de estado distintos y, por tanto, no tienen por qué ser comparables. Lo único que puede ser utilizado en el contraste es la parte del estado $x(k)$ que es observado tanto por el sensor j como por el sensor i.
>
> Rao, para realizar dicho contraste, calcula el residuo de la observación proyectado en el espacio de estado. En nuestro caso, mediante la linealización adecuada, se obtiene:
>
> $$v_{ij}(k) = H_j^{1^T}(k)\left[z_j(k) - {}_j\hat{z}_i(k \mid k-1)\right] \tag{3.45}$$

$$S_{ij}(k) = E\{\nu_{ij} \cdot \nu_{ij}^t\} = H_j^{1^T}(k)\left[R_j(k) + H_j^1(k)P_i(k \mid k-1)H_j^{1^T}(k)\right]H_j^1(k) \quad (3.46)$$

$$\nu_{ji}^T(k) \cdot S_{ij}^{-1}(k) \cdot \nu_{ji}(k) \leq \gamma \quad (3.47)$$

Téngase en cuenta que los términos $z_j(k)$ y $R_j(k)$ no son comunicados directamente al nodo i, pero pueden ser obtenidos a partir de la información que recibe el sensor i del sensor j: $\{X_j(k), Y_j(k)$ y $H_j(k)\}$. De esta forma las ecuaciones anteriores quedan,

$$\nu_{ij}(k) = Y_j^-(k) \cdot X_j(k) - H_j^{1^T}(k) \cdot H_j^1(k) \cdot \hat{x}_i(k \mid k-1) \quad (3.48)$$

$$S_{ij}(k) = H_j^{1^T}(k) \cdot \left[\left\{H_j^1(k)Y_j(k)H_j^{1^T}(k)\right\}^{-1} + H_j^1(k)P_i(k \mid k-1)H_j^{1^T}(k)\right] \cdot H_j^1(k) \quad (3.49)$$

donde,

$$Y_j^-(k) = H_j^{1^T}(k) \cdot \left\{H_j^1(k) \cdot Y_j(k) \cdot H_j^{1^T}(k)\right\}^{-1} H_j^1(k) \quad (3.50)$$

3.3.3 Nueva validación de datos externos

Las ecuaciones (3.48) y (3.49) son las ecuaciones utilizadas por Rao [Rao 93] para calcular la validación de los datos externos procedentes de otros nodos sensoriales. A continuación obtendremos nuevas ecuaciones para el cálculo del test que nos permite aceptar los datos recibidos como válidos.

La primera simplificación sobre las ecuaciones propuestas por Rao es la simplificación de la ecuación (3.49), de forma que se reduce el número de operaciones matriciales realizadas. Así, operando sobre (3.41) y (3.50) se obtiene,

$$S_{ij}(k) = H_j^{1^T}(k) \cdot H_j^1(k)\left[Y_j^-(k) \cdot + P_i(k \mid k-1)\right] \cdot H_j^1(k) \cdot H_j^{1^T}(k) \quad (3.51)$$

De esta forma se evita el cálculo de una inversa y se ahorran varios productos de matrices. La ecuación (3.51) obtenida es equivalente a (3.49) y servirá para compararla con las nuevas ecuaciones que obtendremos a continuación para el cálculo de la covarianza del residuo. El motivo de buscar una nueva ecuación para calcular la covarianza de (3.48), necesaria en la validación externa, se debe a que ésta no es válida en un caso general debido a dos razones:

a) La reconstrucción del estado a partir de las medidas observadas por ambos sensores no es óptima utilizando $H^T(k)$.

b) La matriz de covarianza S_{ij} puede ser singular. Esto ocurrirá cuando los sensores posean un modelo de medida en el cual no todos los estados sean observables o en el caso de que $H(k)$ tenga más columnas que filas. Por ejemplo, un sensor que con su medida sólo detecta un estado (por ejemplo x_1) tiene $H=[1\ 0\ 0]$.

Para evitar estos dos inconvenientes se proponen dos modificaciones posibles:

a) *Reconstrucción óptima del estado*:

La reconstrucción óptima del estado se realizará a partir de la pseudoinversa de H, denominada H^-. Esto se hace así porque normalmente H no será cuadrada y no tiene inversa. Se puede calcular de diversas formas [Golub 89] de las que destacamos dos. A partir de la factorización QR, $H_j^-(k) = R^{-1}(k)\cdot Q(k)_1^T$, donde R es una matriz triangular superior no singular y Q corresponde a la base ortonormal para el espacio de las columnas de H. Otra forma de calcularla es utilizando la ecuación correspondiente a la pseudoinversa por la derecha (normalmente se tendrán más estados que observaciones).

$$H_j^-(k) = H_j^{1\,T}(k)\cdot\left[H_j^1(k)\cdot H_j^{1\,T}(k)\right]^{-1} \tag{3.52}$$

Este cálculo de la pseudoinversa de H permite obtener la solución de la ecuación del modelo de medida para que la reconstrucción del residuo del estado, a partir del error entre la observación y la estima, sea optima [Gill 81]. Utilizando la ecuación (3.52) para la realizar la reconstrucción de la observación, ecuación (3.44), se obtiene el residuo siguiente (que se utilizará para realizar el correspondiente test de validación):

$$v_{ij}(k) = H_j^-(k)\cdot\left[z_j(k) - {}_j\hat{z}_i(k\mid k-1)\right] \tag{3.53}$$

Operando sobre (3.53) y teniendo en cuenta que el sensor i sólo dispone explícitamente de $X_i(k)$, $Y_i(k)$ y $H_i(k)$, las ecuaciones (3.40) y (3.41), obtenemos las siguientes ecuaciones para el residuo y su covarianza, semejantes a (3.48) y (3.51).

$$v_{ij}(k) = H_j^-(k)\cdot H_j^1(k)\cdot\left[Y_j^-(k)\cdot X_j(k) - \hat{x}_i(k\mid k-1)\right] \tag{3.54}$$

$$S_{ij}(k) = H_j^-(k)\cdot H_j^1(k)\cdot\left[Y_j^-(k) + \hat{P}_i(k)\right]\cdot H_j^-(k)\cdot H_j^1(k) \tag{3.55}$$

Esto es válido siempre que exista la inversa de $S_{ij}(k)$. Pero cuando H tiene más columnas que filas, como ocurre usualmente, no se puede calcular la inversa de $S_{ij}(k)$.

Esta reconstrucción, aunque nos asegura la reconstrucción optima del estado, tampoco nos es válida, como ocurría en el planteamiento de Rao, puesto que de nuevo no se puede calcular la inversa de $S_{ij}(k)$. Será válida en los casos en los que es válido el método de Rao [Rao 93], pero para un caso general.

b) *Modificación del residuo utilizado*:

Para mejorar el caso anterior y utilizarlo con cualquier modelo se necesita reconstruir el estado del sensor j en el sensor i. Para hacerlo hay que transmitir entre los nodos demasiada información adicional, pero la transmisión de poca información es uno de los objetivos de los sistemas distribuidos. La solución es reconstruir solamente los estados que son visibles por el sensor j en el sensor i, en lugar de estimarlos todos. Para realizar esto, utilizamos para el test de validación el residuo de las observaciones, como se realiza en la validación interna.

$$\nu_{ij}(k) = z_j(k) - {}_j\hat{z}_i(k \mid k-1) \tag{3.56}$$

Operando nuevamente sobre (3.56) y utilizando (3.40) y (3.41) obtendremos las siguientes ecuaciones para el residuo y su covarianza.

$$\nu_{ij}(k) = H_j(k)\left[Y_j^{-i}(k) \cdot X_j(k) - \hat{x}_i(k \mid k-1)\right] \tag{3.57}$$

$$S_{ij}(k) = H_j(k)\left[Y_j^{-i}(k) + \hat{P}_i(k)\right]H_j^T(k) \tag{3.58}$$

Con ellas se comparan la observación del nodo j con lo que habría observado el nodo i si observase el mismo subespacio de estado que el nodo j. Las ecuaciones (3.57) y (3.58) son siempre calculables y $S_{ij}(k)$ siempre tiene inversa.

Las ventajas de este test frente al de Rao, o al caso a), es que es genérico e independiente de la forma de la matriz del modelo de los sensores, H. Además, el número de operaciones que requiere es menor, puesto que la única inversa que hay que realizar corresponde al cálculo de la pseudoinversa de Y que ya se realizaba en [Rao 93]. Recordemos que en todos los casos es necesario comunicar a los otros sensores el modelo de observación H, pero si el modelo de observación del sensor j es invariante en el tiempo, lo cual ocurre

con bastante frecuencia, sólo será necesario enviar esta matriz en la inicialización, con lo que la comunicación entre nodos será únicamente de $X(k)$ e $Y(k)$.

Por lo tanto ésta será la validación entre sensores utilizada en nuestro sistema. En el caso de que no se supere el test, no se considera la información procedente del sensor para el cálculo de la estima del estado.

Por supuesto el filtro obtenido es subóptimo, ya que los sensores no utilizarán los mismos datos que en el caso centralizado. Tampoco todos los sensores dispondrán de la misma información de posición, sino que tendrán ciertas diferencias debido a que no utilizarán los mismos datos en sus estimas. Sin embargo, el vector de estado de todos los sensores serán muy próximos entre sí, y sus errores siempre serán menores que si utilizasen sólo su propia medida.

3.4 Funcionamiento asíncrono

El funcionamiento asíncrono tratado en Rao supone ciertas simplificaciones:

- Que no hay retardo en la comunicación entre nodos.
- Que no hay retraso en el procesamiento de la observación realizada en cada nodo o sensor j.

Esto lleva a suponer que el instante de tiempo t_i en el que el sensor i recibe la información procedente del sensor j, es el instante de tiempo en el que este último obtuvo dicha observación, es decir, $t_i \cong t_j$.

Estas dos suposiciones son difíciles de justificar puesto que no puede garantizarse en un sistema distribuido general que el retraso en la comunicación sea despreciable. Ni siquiera puede utilizarse la simplificación de que el retraso en la comunicación o el tratamiento de los datos observados sea constante. Por ello, es más realista la inserción del tiempo junto con el envío de información a otros sensores. Tengamos en cuenta que en cualquier sistema distribuido siempre debe existir un sistema de sincronización y una misma referencia de tiempo entre los procesos, por lo que la inclusión del instante de medida en los datos enviados no supondrá ningún esfuerzo adicional. Además, el incremento de información tampoco es excesivo, puesto que sólo se trata de una variable simple, y no matrices o vectores como los demás datos.

Con esta suposición y partiendo de las mismas ecuaciones de validación tendremos varios casos, que anteriormente no se contemplaban.

Sean m sensores y cada uno de ellos puede observar una medida sobre el estado del móvil en cualquier instante, es decir, serán sensores asíncronos.

Sean $\{t_1^k, t_2^k, ..., t_i^k, ..., t_m^k\}$ los instantes en los que cada sensor toma una medida y $\{z_1(t_1^k), z_2(t_2^k), ..., z_i(t_i^k), ..., z_m(t_m^k)\}$ las medidas observadas por cada sensor. El superíndice k corresponde al k-ésimo instante de muestreo del estado global, realizado en el instante t^k, por parte del robot. Así, con respecto al instante de medida del sensor i, los instantes de medida de cada sensor serán,

$$\{t_i^k + \eta_{i1}, t_i^k + \eta_{i2}, ..., t_i^k, ..., t_i^k + \eta_{im}\} \tag{3.59}$$

donde η_{ij} representa la diferencia de tiempo entre la k-ésima observación del sensor i y del sensor j.

Además, cada sensor i, en los instantes $t_i^k \in \{t_1^k, t_2^k, ..., t_m^k\}$ enviará los datos:

$$\{t_i^k, X_i(t_i^k), Y_i(t_i^k), H_i(t_i^k)\} \tag{3.60}$$

A continuación tomemos el sensor i para mostrar cómo un sensor concreto realizará la asimilación de la información recibida de los demás sensores. Sea $t_i^{(k-1)} + \eta_{il}$ el instante de la última actualización (medida de un dato o asimilación) del sensor i. Recordemos que siempre que se observa una medida, y es válida, se realiza una predicción y una actualización.

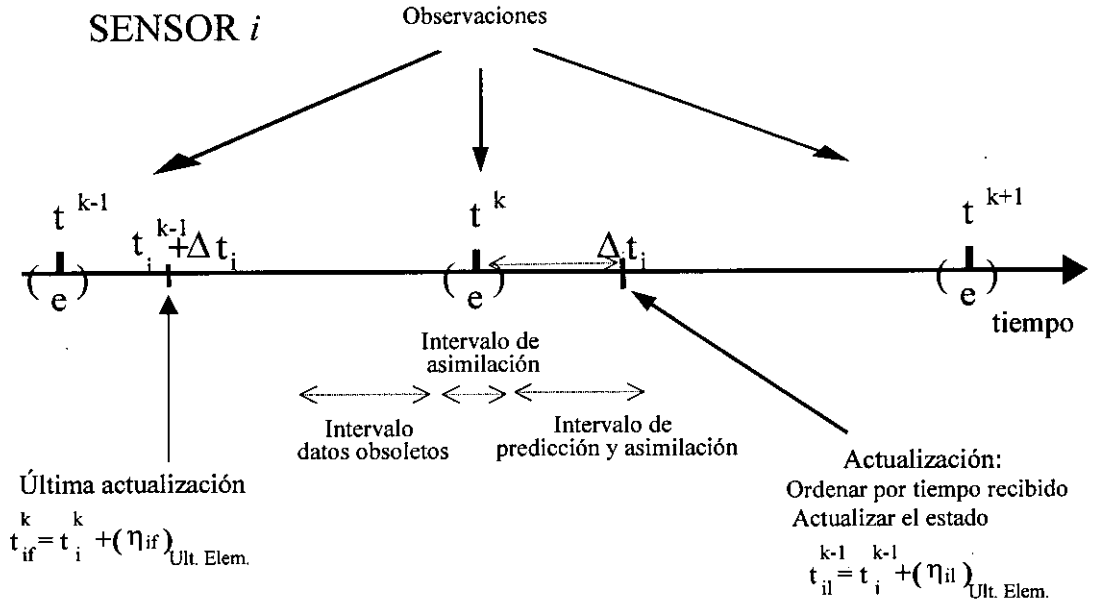


Figura 3-3. Esquema de tiempos de la comunicación entre sensores.

Cualquier sensor recibe datos procedentes de los demás sensores durante un cierto tiempo desde su observación, Δt_i, pasado éste debe asimilar la información recibida junto con su estima. Por ejemplo, en la asimilación en la k-ésima observación del sensor i, véase la figura 3-3, pueden darse tres casos:

1.- *Intervalo de datos obsoletos.*

$$\forall t_j^k \underset{j \neq i}{\in} \{t_1^k, t_2^k, ..., t_m^k\} \ / \ t_j^k < (t_i^k - \varepsilon) \tag{3.61}$$

Si la última actualización es posterior a la información recibida de otro sensor, es despreciada por ser obsoleta. ε representa un intervalo

de tiempo en el que se considera que el retraso es despreciable y pueden asimilarse ambos datos.

2.- *Intervalo de asimilación.*

$$\forall t_j^k \underset{j\neq i}{\in} \{t_1^k, t_2^k, ..., t_m^k\} \ / \ (t_i^k - \varepsilon) \leq t_j^k \leq (t_i^k + \varepsilon) \tag{3.62}$$

En caso de que t_i coincida con t_j o que la diferencia de tiempos entre ambos pueda considerarse despreciable (menor que ε), se realiza la asimilación de los datos con la del sensor que los recibe.

3.- *Intervalo de predicción y asimilación.*

$$\forall t_j^k \underset{j\neq i}{\in} \{t_1^k, t_2^k, ..., t_m^k\} \ / \ (t_i^k + \varepsilon) < t_j^k \tag{3.63}$$

El sensor receptor realiza una predicción del estado del móvil al instante de tiempo recibido y entonces realiza la asimilación de esos datos. En este caso el procedimiento a seguir por el nodo receptor sería:

a.- Ordenar los mensajes recibidos en orden creciente de tiempos.

b.- Predecir hasta el instante del primer mensaje de la lista. Después de la predicción se realiza la asimilación en ese instante de todos los datos que estén en un cierto rango próximos a ese tiempo (si son válidos). Establecer la última actualización del sensor i en ese tiempo de asimilación.

c.- Volver a predecir desde ese último instante al siguiente posterior de la lista y volver a realizar el mismo procedimiento de asimilación. Así se continua hasta el final de los datos recibidos.

Recordemos que ε se elige dependiendo de la aplicación concreta, es decir, de forma que esa variación de tiempo se pueda considerar despreciable y por tanto $t_j^k \cong (t_i^k \pm \varepsilon)$. En muchos casos ε puede ser variable dependiendo de en qué estado se encuentre el móvil.

3.4.1 Ecuaciones de asimilación asíncrona

Por simplicidad utilizaremos dos sensores para mostrar las ecuaciones necesarias en cada uno de los casos mostrados anteriormente. El sensor i recibe

datos en el instante t_i^k y el sensor j observó los datos en el instante t_j^k. Las ecuaciones para esos tres casos serían,

1.- *Intervalo de datos obsoletos*: $t_j^k < (t_i^k - \varepsilon)$

En este caso el sensor i ignora el dato recibido por ser obsoleto.

2.- *Intervalo de asimilación*: $(t_i^k - \varepsilon) \le t_j^k \le (t_i^k + \varepsilon)$

El dato recibido se asimila directamente. $t_j^k \cong t_i^k$.

Sea $t_{il}^{k-1} = t_i^{k-1} + (\eta_{il})_{Ult.Elem.}$ el instante correspondiente a la última actualización u observación "válida" anterior, donde t_i^{k-1} es el instante de la última observación del sensor i y η_{il} es la diferencia de tiempos entre t_i^{k-1} y la observación del sensor l cuyo tiempo ha sido el último aceptado. Sea t_i^k el instante de tiempo de una nueva observación $z_i(t_i^k)$ por parte del sensor i. Si es válida, deberá utilizarse para actualizar su estado antes de asimilar cualquier información procedente de otro sensor.

El estado estimado por el sensor i hasta la recepción del dato procedente del sensor j será:

- Predicción en el sensor i:

$$\hat{x}_i(t_i^k \mid t_{il}^{k-1}) = F_i(\delta t_i) \cdot \hat{x}_i(t_{il}^{k-1} \mid t_{il}^{k-1}) \tag{3.64}$$

$$\hat{z}_i(t_i^k \mid t_{il}^{k-1}) = H_i(\delta t_i) \cdot \hat{x}_i(t_i^k \mid t_{il}^{k-1}) \tag{3.65}$$

$$P_i(t_i^k \mid t_{il}^{k-1}) = F_i(\delta t_i) \cdot P_i(t_{il}^{k-1} \mid t_{il}^{k-1}) \cdot F_i^T(\delta t_i) + G_i(\delta t_i) \cdot Q_i(t_i^k) \cdot G_i^T(\delta t_i) \tag{3.66}$$

donde $\delta t_i = t_i^k - t_{il}^{k-1}$

- Actualización de la medida observada por el sensor i:

$$\hat{x}_i(t_i^k \mid t_i^k) = \hat{x}_i(t_i^k \mid t_{il}^{k-1}) + K_i(t_i^k) \cdot \left[z_i(t_i^k) - H_i(t_i^k) \cdot \hat{x}_i(t_i^k \mid t_{il}^{k-1}) \right] \tag{3.67}$$

$$P_i^{-1}(t_i^k \mid t_i^k) = P_i^{-1}(t_i^k \mid t_{il}^{k-1}) + H_i^T(t_i^k) \cdot R_i^{-1}(t_i^k) \cdot H_i(t_i^k) \tag{3.68}$$

$$K_i(t_i^k) = P_i(t_i^k \mid t_i^k) \cdot H_i^T(t_i^k) \cdot R_i^{-1}(t_i^k) \tag{3.69}$$

- Calcular la información a enviar a los demás (y transmitirla):

$$X_i(t_i^k) = H_i^T(t_i^k) \cdot R_i^{-1}(t_i^k) \cdot z_i(t_i^k) \tag{3.70}$$

$$Y_i(t_i^k) = H_i^T(t_i^k) \cdot R_i^{-1}(t_i^k) \cdot H_i(t_i^k) \tag{3.71}$$

donde, $\hat{x}_i(t_i^k \mid t_i^k)$ es el estado del móvil estimado por el sensor i hasta que se produzca una nueva observación o se reciba información procedente de otros sensores. En caso de recibir información externa, se realizaría la siguiente actualización de la estima local en el sensor i.

$$\hat{x}_i(t_i^k \mid t_i^k) = P_i(t_i^k \mid t_i^k) \cdot \left[P_i^{-1}(t_i^k \mid t_u^{k-1}) \cdot \hat{x}_i(t_i^k \mid t_u^{k-1}) + X_i(t_i^k) + X_j(t_i^k)\right] \quad (3.72)$$

$$P_i^{-1}(t_i^k \mid t_i^k) = P_i^{-1}(t_i^k \mid t_u^{k-1}) + Y_i(t_i^k) + Y_j(t_i^k) \quad (3.73)$$

3.- *Intervalo de predicción y asimilación*: $(t_i^k + \varepsilon) < t_j^k$

La diferencia entre t_i^k y t_j^k es apreciable y por tanto es necesaria una predicción del estado del sensor i hasta el tiempo t_j^k antes de realizar la asimilación de la información recibida.

Predicción:

$$\hat{x}_i(t_j^k \mid t_i^k) = F_i(\delta t_{ij}) \cdot \hat{x}_i(t_i^k \mid t_i^k) \quad (3.74)$$

$$\hat{z}_i(t_j^k \mid t_i^k) = H_i(\delta t_{ij}) \cdot \hat{x}_i(t_j^k \mid t_i^k) \quad (3.75)$$

$$P_i(t_j^k \mid t_i^k) = F_i(\delta t_{ij}) \cdot P_i(t_i^k \mid t_i^k) \cdot F_i^T(\delta t_{ij}) + G_i(\delta t_{ij}) \cdot Q_i(t_j^k) \cdot G_i^T(\delta t_{ij}) \quad (3.76)$$

donde $\delta t_{ij} = \eta_{ij} = t_j^k - t_i^k$

Asimilación:

$$\hat{x}_i(t_j^k \mid t_j^k) = P_i(t_j^k \mid t_j^k) \cdot \left[P_i^{-1}(t_j^k \mid t_i^k) \cdot \hat{x}_i(t_j^k \mid t_i^k) + X_j(t_j^k)\right] \quad (3.77)$$

$$P_i^{-1}(t_j^k \mid t_j^k) = P_i^{-1}(t_j^k \mid t_i^k) + Y_j(t_j^k) \quad (3.78)$$

y se considera como último instante de actualización de la estima $t_{ij}^k = t_j^k$.

Por último, se debe señalar que en un sistema completamente asíncrono, las señales de control al robot y los periodos de medida de cada uno de los sensores son distintos. Además la frecuencia con la que realiza las medidas el sensor físico no tienen porqué coincidir con el del sensor lógico. Todo esto lleva a situaciones más complejas que la mostrada en la figura 3-4.

Por ejemplo, en la figura 3-4, se muestra cómo el periodo de control es mayor que la frecuencia de medida del sensor i, por lo que recibe distintas señales de control dentro de un mismo periodo de observación. En este caso se realizan una serie de predicciones y asimilaciones para cada señal de control u, con el fin de tener en todo instante la estima más precisa.

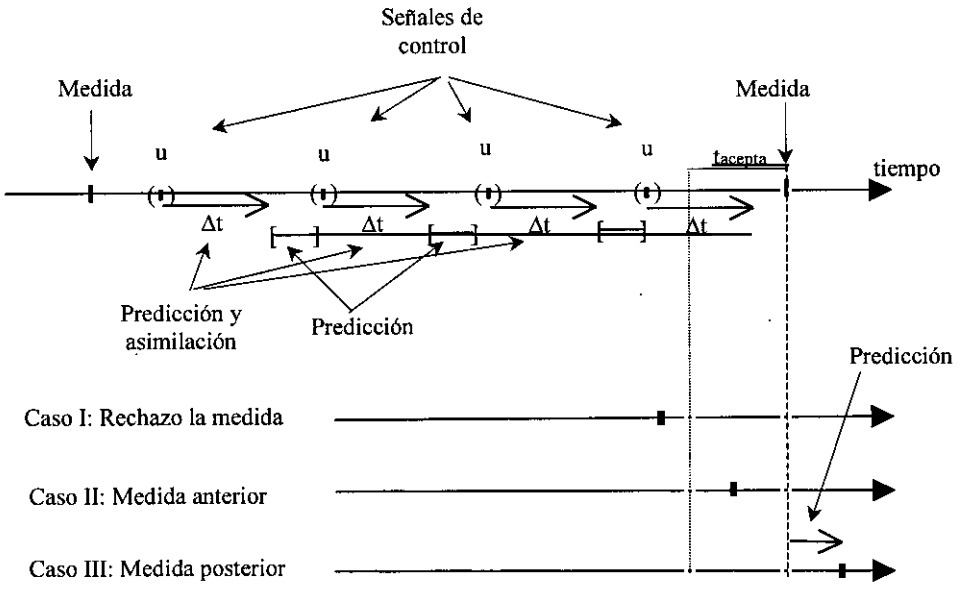


Figura 3-4. Asincronismo entre señales de control, de medida y observaciones.

El asincronismo existente entre la observación del sensor físico y la observación del sensor lógico lleva a que ocurran tres casos diferentes, obsérvese la figura 3-4:

Caso I: rechazo de la medida si el tiempo de la observación correspondiente supera el tiempo que considera el usuario que es el retraso es permisible o despreciable, denominado *tacepta*.

Caso II: si la observación es anterior a la solicitud de la medida, pero se encuentra dentro del intervalo de tiempo desprecible, *tacepta*, se considera que la observación tiene el mismo tiempo que el instante de medida, y se predice hasta ese punto, para posteriormente actuar como ya se ha indicado.

Caso III: si la es realizada en un instante de tiempo posterior a la solicitud de la medida se actúa como ya se ha indicado.

Como podemos ver, de esta forma conseguimos tratar de forma síncrona, desde un control de alto nivel, a un sistema que trabaja con sensores totalmente asíncronos en su bajo nivel. Aunque el control y el tratamiento de alto nivel se realice a partir de observaciones síncronas, éstas han sido obtenida a partir de la mejor estima posible en cada momento por parte de sensores totalmente asíncronos.

3.5 Resultados experimentales

Para ilustrar el funcionamiento asíncrono de nuestro sistema de fusión de medidas de posición estimaremos la posición de un robot móvil que realiza un movimiento en dos dimensiones. El vector de estado utilizado es $\boldsymbol{x}=[x,y,\vartheta]^T$, donde x e y son las coordenadas del móvil en un sistema de referencia fijo determinado y ϑ representa su orientación respecto al eje X. El modelo del carro es similar al utilizado en [Kam 97].

Los sensores utilizados para estimar la posición del móvil son, véase la figura 3-5: unos odómetros (S1), una brújula magnética (S2) y un receptor de radio (S3). En este ejemplo se ha buscado mostrar la integración de medidas de posición de sensores relativos (S1) y absolutos (S2 y S3). Cada uno de ellos representa un sensor de posición de los tipos indicados en el capítulo 2, como son sensores odométricos (S1), sensores de orientación (S2) y balizas artificiales (S3).

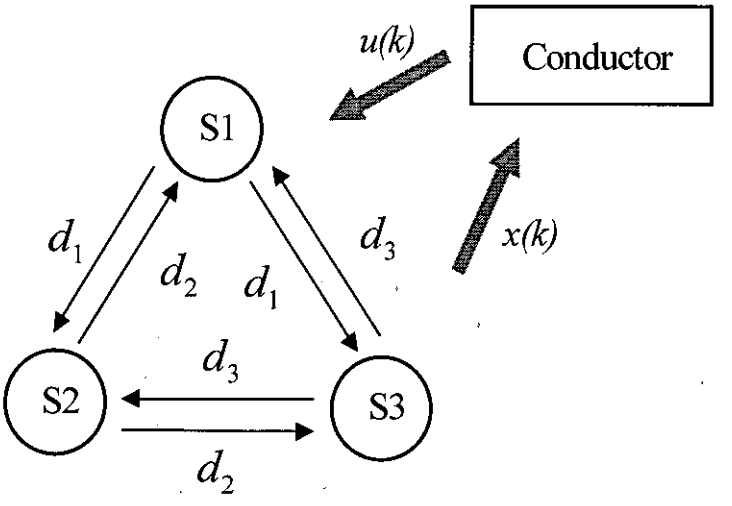


Figura 3-5. Sensores distribuidos. $d_i \equiv \{t^k, X_i(t^k), Y_i(t^k), H_i(t^k)\}$

3.5.1 Modelo del sistema y de los sensores

El modelo de estado del móvil para cada sensor, donde el vector de estado del móvil representa su posición y orientación, $\vec{x}_i(t) = (x(t), y(t), \vartheta(t))$, es el mostrado en (3.10).

Si se define $V(t)=\frac{v^{izq}+v^{der}}{2}$ (velocidad solicitada) y $\gamma(t)=\frac{v^{izq}-v^{der}}{b}$ (giro demandado, b es la separación entre las dos ruedas) como las señales de control al móvil $u(t)=[V(t),\gamma(t)]$ se obtiene el siguiente modelo no lineal:

$$\begin{aligned}\dot{x}_i(t)&=V\cdot\cos\vartheta_i(t)\\ \dot{y}_i(t)&=V\cdot\operatorname{sen}\vartheta_i(t)\\ \dot{\vartheta}_i(t)&=\gamma\end{aligned}\tag{3.79}$$

O, en forma discreta es:

$$\begin{aligned}x_i(k+1)&=x(k+1)+D(k)\cdot\cos\vartheta_i(k)\\ y_i(k+1)&=y(k+1)+D(k)\cdot\operatorname{sen}\vartheta_i(k)\\ \vartheta_i(k+1)&=\vartheta(k)+\Delta\vartheta(k)\end{aligned}\tag{3.80}$$

donde $D(k)=\frac{v^{izq}\cdot T+v^{der}\cdot T}{2}=\frac{d^{izq}+d^{der}}{2}$ y $\Delta\vartheta(k)=\gamma\cdot T=\frac{d^{izq}-d^{der}}{b}$; y la señal de control queda $u(k)=\left[D(k),\Delta\vartheta(k)\right]$.

Modelo de medida de los sensores:

El modelo de observación de cada sensor se considera lineal para simplificar el ejemplo y son:

a) Brújula magnética:

La brújula magnética puede considerarse como un sistema de primer orden con una constante de tiempo τ de 4 seg.

$$\tau\cdot\dot{\vartheta}_c(t)=-\vartheta_c(t)+\vartheta_i(t)\tag{3.81}$$

b) Receptores de balizas:

Las balizas artificiales utilizadas proporcionan la posición absoluta del móvil directamente. Así, puede modelarse añadiendo ruido blanco al estado del móvil y no necesita utilizar ningún estado adicional.

c) Odómetro:

El modelo de los odómetros vienen dados por la adición de un random walk a la lectura de la variación de posición estimada.

$$\dot{\vec{x}}_e(t)=\vec{w}(t)\tag{3.82}$$

A partir de los modelos anteriores, se tienen dos opciones:

(1) Se utilizan como estados los tres indicados anteriormente que representan las variables de interés en el robot móvil y se realiza un aumento de estados debido a los estados adicionales de la brújula y el encoder. En este caso tendremos un vector final de estados como,

$$\vec{x}_i(t_i) = (x(t_i), y(t_i), \vartheta(t_i), \vartheta_c(t_i), x_e(t_i), y_e(t_i)) \tag{3.83}$$

Este vector de estados y el modelo de transición de estados correspondiente (una matriz de tamaño 6x6) puede utilizarse tanto de forma distribuida como centralizada.

Esto tiene la ventaja de que los resultados del filtro distribuido son directamente comparables con los resultados del filtro centralizado. Sin embargo, también existen serios inconvenientes como son que el tamaño de las matrices utilizadas es muy grande; y que si se desea incluir un nuevo sensor o suprimir uno ya existente es necesario modificar el vector de estados y por tanto cambiar todas las matrices del sistema en cada uno de los sensores. Las matrices utilizadas en este caso en el filtro de Kalman se pueden encontrar en el Apéndice D.

(2) Se utiliza un modelo distinto para cada sensor. Cada sensor tendrá en cuenta las variables de estado de interés que deben comunicarse entre sí y aquellas necesarias debido al modelo del propio sensor. Aunque el vector de estado interno del sensor tenga más estados la única información que se transmite a otros sensores es el vector de estados común, $\vec{x}_i(t) = (x(t), y(t), \vartheta(t))$, y su matriz de covarianzas asociada.

Esto reduce el tamaño de las matrices utilizadas y permite que cada sensor sea independiente de los demás. Las matrices necesarias en el filtro de Kalman se encuentran descritas en el Apéndice D.

3.5.2 Simulaciones

Se realizarán distintas simulaciones sobre una misma trayectoria para poder compararlas entre sí. Esta trayectoria consta de movimientos con velocidades entre 1 m/s y 0.5 m/s, véase la figura 3-6, donde se mantiene movimientos rectilíneos y se

realiza una curva de 180 grados. De esta forma se puede observar el efecto del filtro en el momento más comprometido, cuando se realiza una curva con velocidad máxima. Las simulaciones distinguen dos casos claramente diferenciados:

La primera simulación muestra el funcionamiento del sistema de fusión cuando todos los sensores funcionan correctamente. En esta simulación se pone de manifiesto el comportamiento del filtro subóptimo diseñado.

La segunda simulación presenta el comportamiento del sistema ante distintos fallos de los sensores, de forma que se pueda observar la evolución y comportamiento del sistema de fusión ante distintos fallos. Se han estudiados dos tipos de fallos: cuando un nodo del filtro deja de funcionar debido a que no puede utilizar su sensor y cuando falla su comunicación con los demás sensores y por tanto sólo puede utilizar su propia estima.

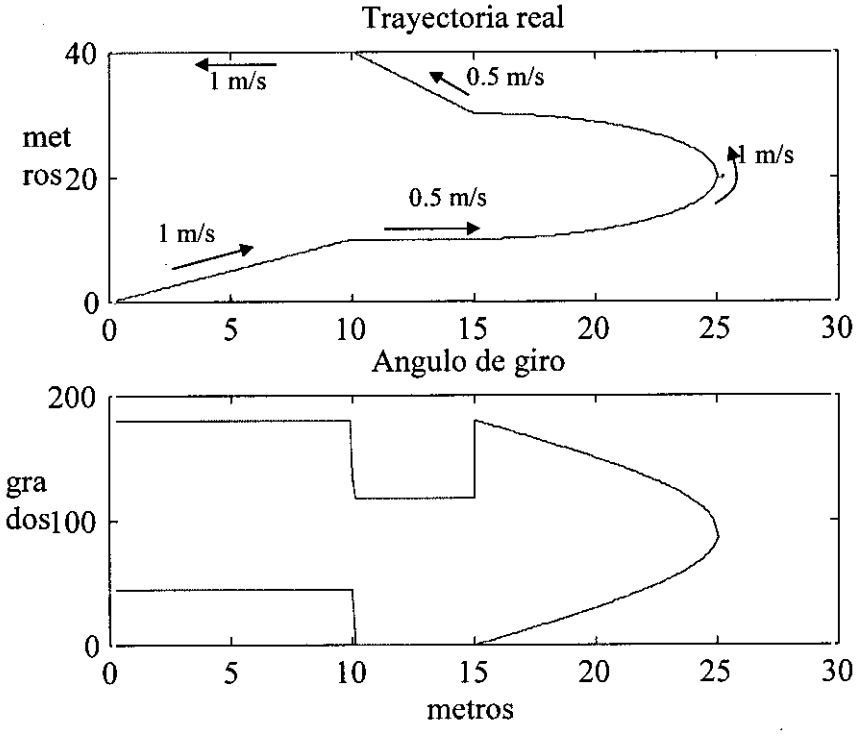
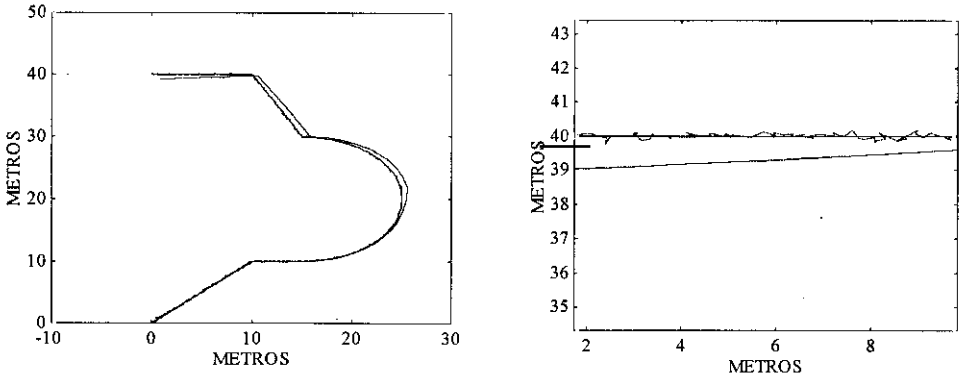


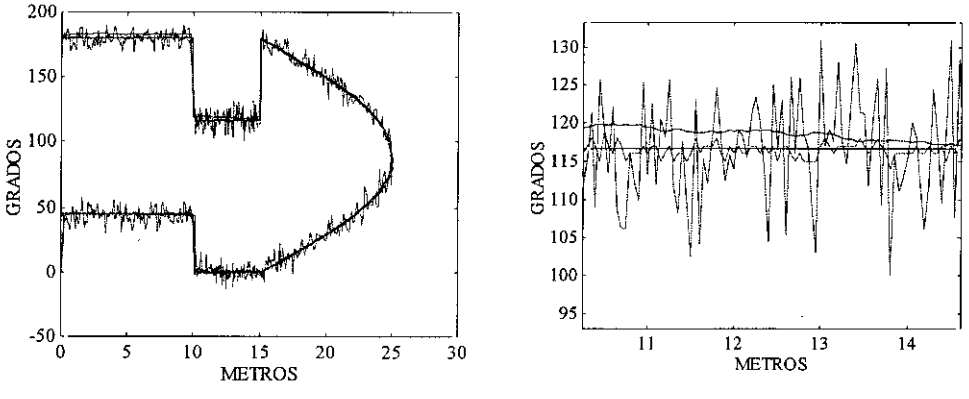
Figura 3-6. Datos de la trayectoria sin ruidos

En la figura 3-7 se muestran las medidas de los sensores en la trayectoria seguida por el móvil para ambos casos. En la figura 3-7a se muestran las medidas de posición (x,y) observadas por la baliza y por los odómetros; y en la figura 3-7b se presenta la orientación obtenida por los tres sensores. A la derecha de cada figura se incluye una ampliación de la figura para destacar el error que comete cada uno

de los sensores. Puede observarse como el error de posición de la odometría crece con el tiempo, el error de orientación de la brújula es del orden de uno o dos grados de error y el error de orientación de los receptores de baliza es muy grande, y por tanto no debería utilizarse la orientación indicada.



a) Posición medida por la baliza y los odómetros



b) Orientación medida por cada sensor.

Figura 3-7. Posición y orientación medidos por cada sensor.

Con estas medidas se realizan los siguientes experimentos:

Caso 1: Funcionamiento sin fallos de los sensores

En este caso no se producirá ninguna anomalía en la información obtenida por los sensores ni en la transmisión de esta información. Los únicos errores que pueden aparecer en el cálculo de la trayectoria correcta serán debido a uno de estos efectos: (1) desviaciones de los sensores de la trayectoria real debido a su error en la medida; (2) no validación (interna o externa) de las estimas de los sensores; y (3), rechazo de estimas debido al asincronismo entre los nodos, es decir,

a la no aceptación de una estima procedente de otro sensor cuando llegue retrasada un tiempo mayor del considerado como permisible.

En la figura 3-8 puede observarse la trayectoria estimada por cada sensor a partir de su propia observación y de la información recibida de los otros sensores. Esta trayectoria es comparada con la que calcularía un filtro centralizado (óptimo) en esas mismas condiciones y puede observarse como la estima distribuida prácticamente no se aparta de la centralizada.

Conviene destacar que con este método todos los sensores conocen el estado completo del móvil, aunque por sí solos únicamente pudiesen estimar parte de ese estado. Por ejemplo, la brújula magnética sólo observa la orientación del móvil. Sin embargo, gracias a la información recibida, es capaz de estimar el estado completo del móvil, que puede utilizar en posteriores estimas. De esta forma se evita, como buscábamos, el uso de un nodo de fusión central que realice la fusión de las medidas de todos los sensores y el posterior envío del resultado de vuelta a todos los sensores.

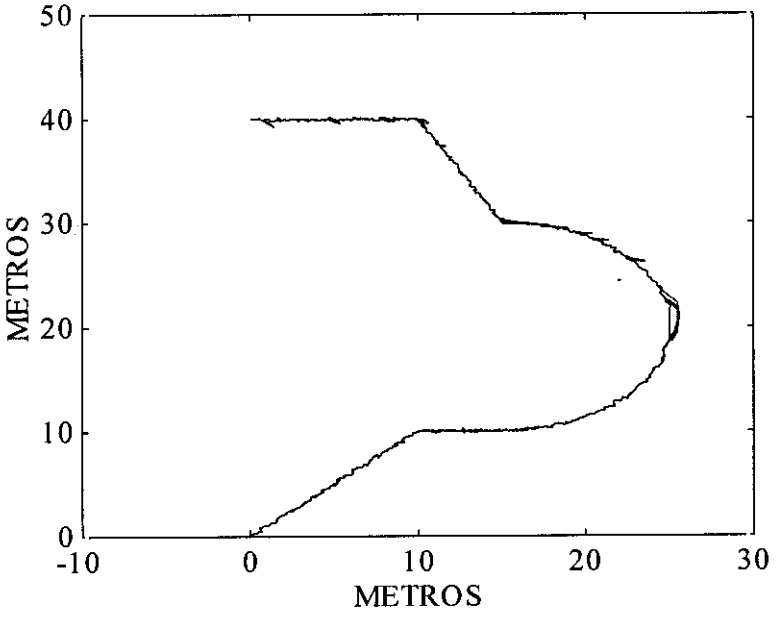
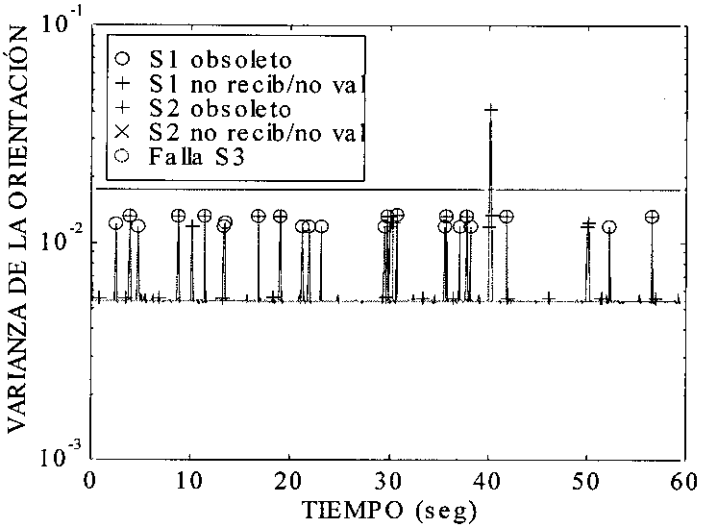


Figura 3-8 . Trayectoria estimada

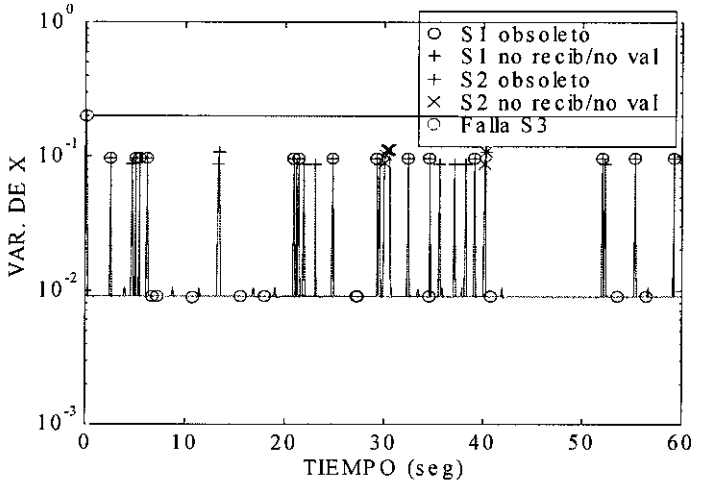
En la figura 3-9a se compara la varianza de las estimas de orientación del móvil (estado 3) realizadas por la brújula, con la varianza de la observación del sensor y con la covarianza estimada en el caso síncrono centralizado. Aquí puede verse cómo, cuando no ocurre ningún fallo en el sensor ni en la validación de los

datos estimados, la covarianza de la estima se encuentra por debajo de la varianza propia del sensor pero por encima de la obtenida en el caso centralizado. Será igual a la del caso centralizado cuando todos los sensores validan sus medidas y las recibidas por los otros.

En la figura 3-9b se muestra el mismo caso para los odómetros. Aunque el filtro es subóptimo y el filtro centralizado obtiene una estima más precisa (estima óptima), en esta figura se muestra también que la covarianza de la estima del filtro distribuido asíncrono es menor que la varianza de la medida y, por tanto se consigue mejorar la estima de un sensor aislado.



Brújula magnética (S1)



Odómetros (S3)

Figura 3-9. Varianza de la orientación. Caso centralizado y distribuido.

Así podemos observar que cuando un sensor asimila toda la información procedente de los otros sensores, las estimas de cada sensor serán las mismas entre sí, y muy próximas a la del filtro centralizado (estima óptima). Pero cuando se producen rechazos de la información que se recibe debido al asincronismo del sistema se producen aumentos en la incertidumbre de la estima calculada. Estos picos en la covarianza se pueden ver en las figuras 9a y 9b. También se pueden observar otros picos producidos por el rechazo de la observación (validación interna) y el rechazo de la información recibida (validación externa).

Para comprender mejor los instantes en los que cada sensor rechaza las medidas en una validación de datos se representan en la figura 3-10 el lugar donde se ha producido un fallo para la situación mostrada en la figura 3-9a.

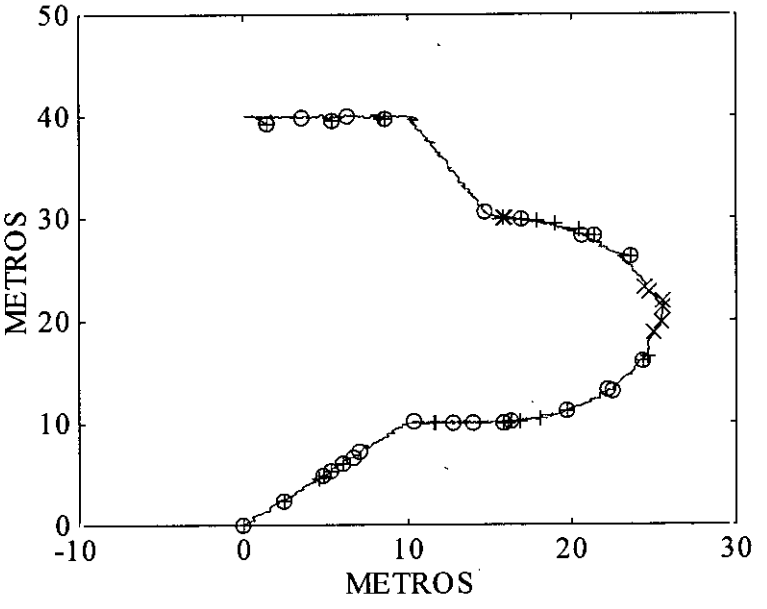


Figura 3-10. Situación de los fallos en el sensor brújula (S1).

En las tablas siguientes se muestra el número de validaciones y fallos ocurridos en cada sensor para el ejemplo presentado anteriormente:

a.- La tabla 3-1 muestra el número de medidas realizadas y cuantas son rechazadas por el sensor al realizar la validación de sus propias observaciones (validación interna). Como puede verse son muy pocas las medidas rechazadas.

b.- La tabla 3-2, presenta las validaciones externas que realiza cada sensor, es decir, el número de datos que acepta y rechaza procedente de otros sensores.

c.- La tabla 3-3 es un complemento de la tabla 3-2 y muestra cuantos datos de los no validados por los sensores son debidos a que han llegado tarde y se consideran obsoletos, tengamos en cuenta que el número de datos obsoletos debe ser pequeño en relación con el número de medidas, lo contrario significaría que el sistema sensorial no está correctamente diseñado. Por ejemplo en el caso mostrado es aproximadamente de un 4%.

En el ejemplo mostrado todos los sensores realizan 599 medidas. Para comprender el conjunto de las tablas se véa la siguiente interpretación de las tablas 3-1, 3-2 y 3-3. La relación del sensor 1 con respecto al sensor 2 es: S1 no recibe de S2 5 estimas, S1 valida de S2 565 de las estimas recibidas, no valida 6 y considera 23 estimas procedente de S2 como obsoletas debido a retrasos en la comunicación. El total es, como es lógico, 599. Lo mismo es aplicable a los demás sensores.

Tabla 3-1: Validación interna

	Total	Errores
Sensor 1	599	1
Sensor2	599	5
Sensor 3	599	0

Tabla 3-2: Validación de datos externos

Total/no recibido		Valida			NO valida		
	Si / Sj - Sk	S1	S2	S3	S1	S2	S3
S1	599 / 5 – 0	-	565	573	-	6	0
S2	599 / 1 – 0	570	-	567	0	-	1
S3	599 / 1 – 5	572	565	-	0	3	-

Tabla 3-3: Dato obsoleto

	S1	S2	S3
S1	-	23	26
S2	28	-	31
S3	26	26	-

Caso 2 : Fallo en el funcionamiento de un sensor

La ventaja del sistema distribuido es que en caso de malfunción de un sensor, el sistema podrá continuar funcionando satisfactoriamente, siempre que existan medidas redundantes claro está. Esto lleva a preferir este tipo de sistema frente a los centralizados a pesar de que su precisión sea un poco menor. Para mostrar la robustez del sistema frente a distintos fallos de los sensores se realizan dos experimentos:

- *Experimento 1: Fallo en la medida del sensor*

Uno de los sensores no puede obtener ninguna medida por parte de su sensor físico, de forma que no puede actualizar su estado. Por ejemplo, supongamos que el receptor de las señales de la baliza (sensor S2) entra en una zona de sombra y no puede recibir dichas señales. Como es lógico será incapaz de estimar su posición por sí solo.

En este caso, si funcionase sólo sin tener en cuenta las medidas de los otros sensores (ya sea porque no está trabajando dentro del filtro o porque no es capaz de asimilar la información de los otros sensores) sería imposible que el sensor S2 pudiese estimar la posición del móvil, véase la figura 11a. En este caso el sensor S2 sólo puede estimar su posición mediante el modelo del sistema que posee, por tanto su incertidumbre aumenta indefinidamente con el paso del tiempo como puede verse en la figura 3-11b.

En el momento que el sensor pueda recibir la información de sus emisores de señal podrá localizarse de nuevo.

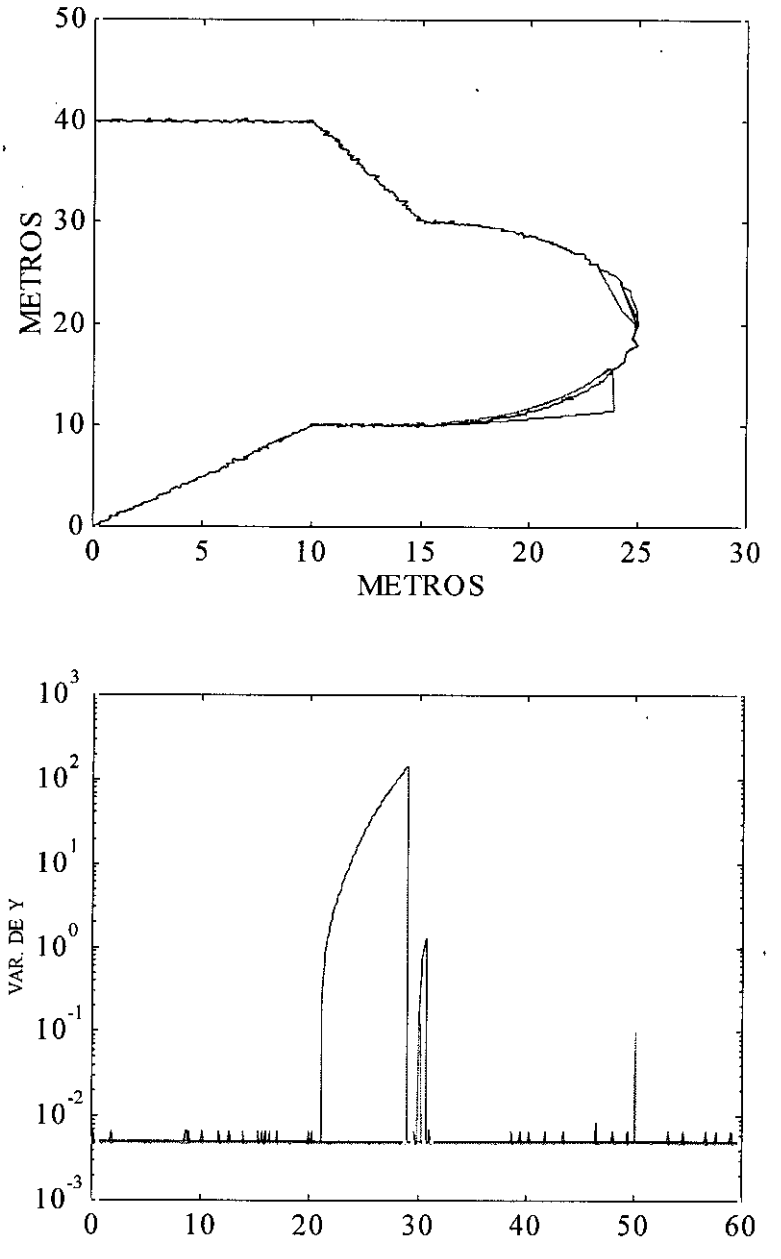
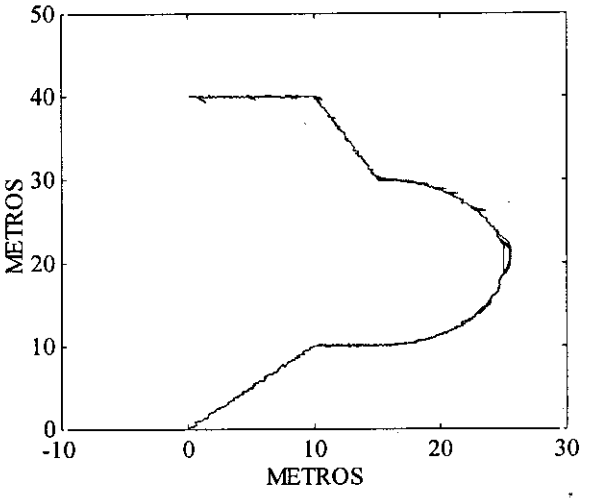


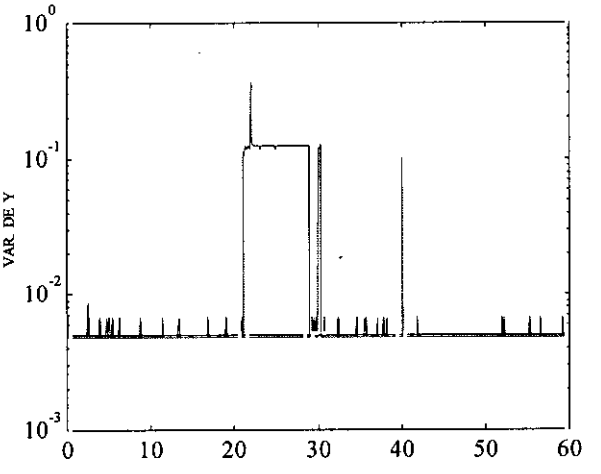
Figura 3-11. Fallos en la adquisición de medidas y en la recepción en sensor Baliza (S2).

Sin embargo, gracias a la utilización del filtro de Kalman distribuido este caso no afecta al funcionamiento del sistema, mostrando de esta manera su robustez ante fallos del sensor o pérdidas momentáneas de señal.

En la figura 3-12 puede verse que, aunque el sensor S2 deja de recibir información de su emisor de baliza y no puede localizarse por sí mismo, puede reconstruir perfectamente su posición gracias a la información de los otros sensores.



a) Posición estimada por los sensores



b) Covarianza del sensor baliza

Figura 3-12 Fallo en la adquisición de medidas del sensor Baliza (S2).

En la figura 3-12b se muestra que la varianza del sensor S2 aumenta al no disponer de una de las fuentes externas de información, pero su aumento está acotado y la precisión de su estima se mantiene gracias a la estima que recibe de los demás sensores.

Las tablas 3-4, 3-5 y 3-6 muestran el comportamiento de cada uno de los sensores para el caso mostrado. Su interpretación es la misma que en el caso 1, pero destacar aquí que el sensor S1 y el sensor S3 no reciben un número alto de medidas procedentes del sensor S2 (84 en concreto) debido a que son debidas a la no validación interna o a que el sensor propio no responde y por tanto no se tiene información que enviar.

Tabla 3-4: Validación interna		
	Total	Errores
Sensor 1	599	1
Sensor2	599	5
Sensor 3	599	0

Tabla 3-5: Validación de datos externos							
Total/no recibido		Valida			NO valida		
	Si / Sj - Sk	S1	S2	S3	S1	S2	S3
S1	599 / 84 – 0	-	489	573	-	6	0
S2	599 / 1 – 0	570	-	567	0	-	1
S3	599 / 1 – 84	572	491	-	0	3	-

Tabla 3-6: Dato obsoleto			
	S1	S2	S3
S1	-	20	26
S2	28	-	31
S3	26	21	-

- *Experimento 2: Fallo en la comunicación entre sensores*

En este ejemplo se produce un fallo en la comunicación y el sensor 2, el receptor de balizas artificiales, no es capaz de recibir información para mejorar su propia estima. Sin embargo sí puede estimar el estado del móvil gracias a la información que obtiene de su propio sensor físico.

En este caso se puede observar cómo la covarianza de los estados es mayor que en el caso de asimilación de datos externos, figura 3-13, puesto que al disponer únicamente de su propia observación no existen medidas redundantes que permitan mejorar su estima.

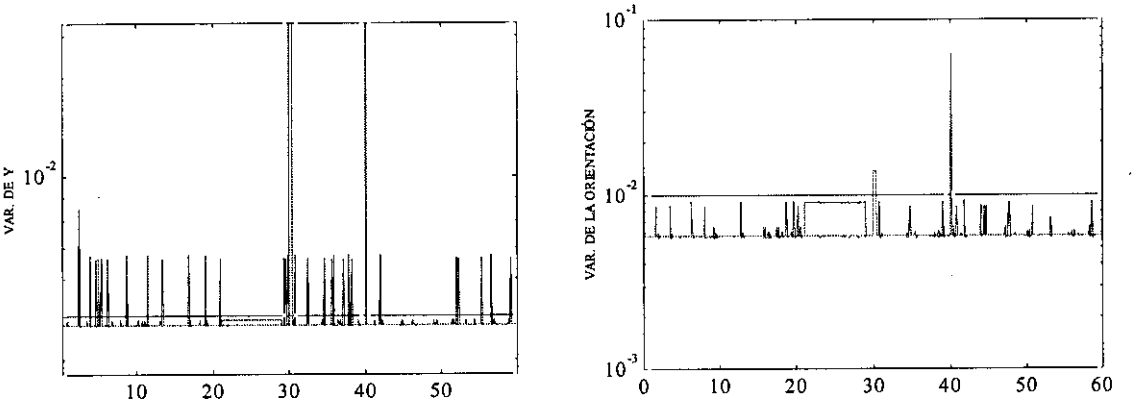


Figura 3-13. Fallo en comunicación (Baliza).

Las tablas 3-7, 3-8 y 3-9 muestran el funcionamiento de este ejemplo. En la tabla 3-5 se puede observar que el sensor S2 no recibe información de ninguno de los otros sensores pero aun así el sensor funciona correctamente.

Tabla 3-7: Validación interna

	Total	Errores
Sensor 1	599	1
Sensor2	599	7
Sensor 3	599	0

Tabla 3-8: Validación de datos externos

Total/no recibido		Valida			NO valida		
	Si / Sj - Sk	S1	S2	S3	S1	S2	S3
S1	599 / 7 – 0	-	569	579	-	4	0
S2	599 / 80 – 79	497	-	500	0	-	6
S3	599 / 1 – 7	556	553	-	0	9	-

Tabla 3-9: Dato obsoleto			
	S1	S2	S3
S1	-	19	20
S2	22	-	14
S3	33	30	-

3.5.3 Decisión de la estima mejor

El último paso en el sistema de fusión es la elección de la mejor estima de los distintos sensores distribuidos. Cada estima de posición calculada por un sensor es enviada a un nodo "central" que se encarga de llevar el registro de la posición del móvil y ofrecer esta información a toda tarea que lo necesite. Este nodo no realiza ningún tipo de fusión, sino que toma las medidas de posición de los distintos sensores y envía al robot aquella que considera mejor, es decir, la más precisa. En teoría, todos los sensores deberían tener la misma estima, y con la misma precisión, cuando no existe ningún rechazo en las medidas (validación interna y validación externa); es decir, el comportamiento del filtro distribuido es semejante al del centralizado y por tanto no existe ningún problema en la elección de una medida, cualquiera sería válida al ser todas estimaciones similares. El problema surge cuando alguna de las estimas del sensor es distinta a las demás (normalmente será peor que la que indique la mayoría) debido a un error o fallo en la validación o en el sensor. Para evitar tomar una medida incorrecta se debe realizar un test que nos indique qué estima final es conveniente escoger.

Existen multitud de tests de decisión, tests de hipótesis o distintos tipos de medidas para escoger entre cierto número de elementos. Por ejemplo se dispone de la regla de decisión bayesiana, la regla de decisión bayesiana de coste mínimo, el test de Neyman-Pearson, el test Minimax, coeficientes de correlación y medidas de concordancia, técnicas de votado, plantillas, medidas de distancia, etc. Estos tipos de tests están orientados, por ejemplo, a la asociación de datos [Hall 92], la clasificación o reconocimiento de patrones [Fukunaga 90; Fukunaga 93] o la detección, distribuida o no, [Varshney 96].

Tengamos en cuenta que ya han sido evaluados los errores debido a desviaciones de un sensor respecto a los otros y rechazadas aquellas medidas que

no se ajustan al modelo de medida ni a la predicción del estado del móvil. Esto es debido al uso del filtro de Kalman distribuido, donde ya se han realizado tests Chi-cuadrado para eliminar tales errores, por lo tanto no es necesario realizar de nuevo decisiones complejas.

Sin embargo, si se quieren eliminar errores en la comunicación o en el cálculo de los nodos sensoriales debería realizarse algún tipo de test que tenga en cuenta la mayoría de las opiniones de los sensores. Por ejemplo, si un sensor debido a una causa indeterminada, como por ejemplo un malfuncionamiento o un error en la transmisión, envía al nodo "central" un estado muy distinto a la de los otros sensores pero con una varianza menor que la de los demás será escogido, incluso en contra de la estima de los demás nodos incluso aunque la válida sea otra. Para eliminar este tipo de errores se debe escoger un test relativamente sencillo de realizar para evitar una carga de cálculo excesiva y redundante.

Para realizar el test de decisión sólo se dispone de la estima y covarianza de cada sensor. Teniendo en cuenta la escasa información disponible y la necesidad de no realizar excesivos cálculos se escoge un test de distancia entre las estimas y una referencia de forma que se puede diseñar una clasificación de las medidas y se escoge la mejor. Para calcular la distancia necesitamos una referencia común con respecto a la cual comparar las medidas recibidas. Para ello se utiliza la mejor estima posible de las medidas, en cierto sentido probabilístico, como es la media o primer momento y la varianza o segundo momento de las estimas recibidas. Así, la estima más próxima a la media será la mejor estima posible.

El test de distancia realizado necesita utilizar una métrica que tenga en cuenta la proximidad a la media y la precisión de la medida (la covarianza). Entre las métricas disponibles, la Euclídea no es válida puesto que no tiene en cuenta la covarianza de las medidas. Tampoco lo será la métrica de Mahalanobis puesto que aunque incluye la covarianza en el cálculo de la distancia a la media, no comprueba la semejanza de las covarianzas de la media y de las estimas. La distancia más indicada en este caso es la *distancia de Chernoff*, véase la tabla 3-7, que representa una cota superior al error de Bayes, pero evita el gran esfuerzo de calculo que se requiere para su cálculo. A pesar de ser una cota superior puede ser directamente utilizable puesto que se desea ordenar las medidas con respecto a su proximidad a la medida óptima y, si todas tienen el mismo bias, el orden obtenido será el mismo.

Sin embargo, el test escogido es la medida de distancia de Bhattacharyya [Fukunaga 90]. Esta medida es un caso especial de la distancia de Chernoff (se

utiliza s=0.5, véase tabla 3-7) en el que la distancia es mucho más sencilla de calcular. Esta métrica mide la diferencia entre dos distribuciones y consta de dos términos: el primer término mide la separación de cada medida de la media pero ponderada por la varianza de la medida (es un término equivalente a la distancia de Mahalanobis); el segundo término representa un incremento en la distancia de las medidas debido a la diferencia entre sus covarianzas. Una comparación de las métricas indicadas y mostradas en la tabla 3-7 puede encontrarse en [Fukunaga 93].

Euclidea	$(X-M_i)^T \cdot (X-M_i)$
Mahalanobis	$(X-M_i)^T \cdot \Sigma^{-1} \cdot (X-M_i)$
Bhattacharyya	$\frac{1}{8}(X-M_i)^T \cdot \left(\frac{\Sigma_x+\Sigma_i}{2}\right)^{-1} \cdot (X-M_i) + \frac{1}{2}\ln \frac{\left\|\frac{\Sigma_x+\Sigma_i}{2}\right\|}{\sqrt{\|\Sigma_x\| \cdot \|\Sigma_i\|}}$
Chernoff	$\frac{s(1-s)}{8}(X-M_i)^T \cdot [s\Sigma_x+(1-s)\Sigma_i]^{-1} \cdot (X-M_i) + \frac{1}{2}\ln \frac{\|s\Sigma_x+(1-s)\Sigma_i\|}{\|\Sigma_x\|^s \cdot \|\Sigma_i\|^{1-s}}$

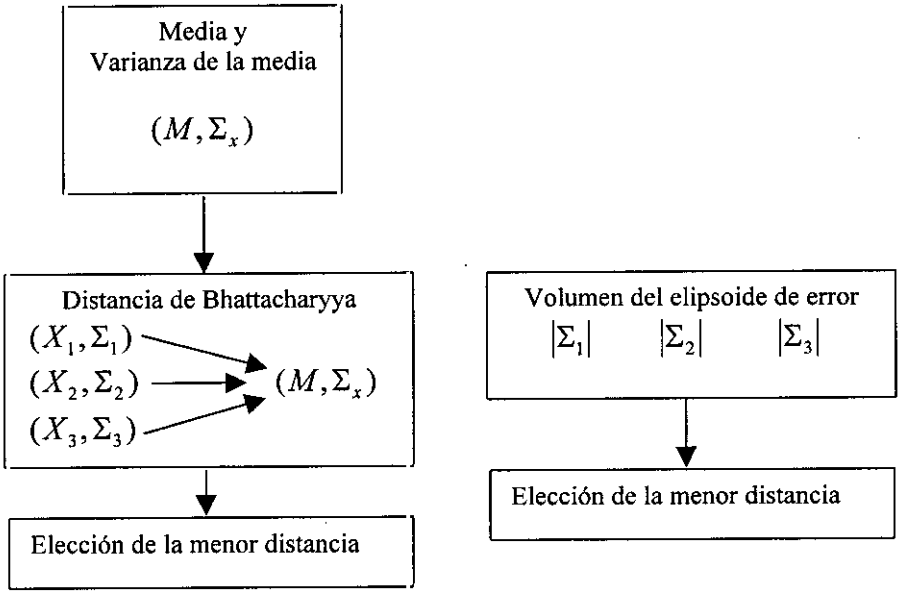
Tabla 3-7. Ejemplos de medidas de distancias.

Por lo tanto mediante la distancia de Bhattacharyya se calcula la proximidad de cada una de las estimas a la media y se escoge la menor: aquella que es la más próxima a la media (lo que opina la mayoría) y la más próxima a la varianza de la media (menor error posible).

Téngase en cuenta que la obtención de esta distancia, aunque es un test que necesita muy poco esfuerzo de cálculo comparado con otros test, requiere un calculo excesivo si tenemos en cuenta que ya se han realizado anteriormente distintos test de error y, por lo tanto, las medidas no pueden diferir demasiado. Además, al provenir de un filtro de Kalman, las medidas que poseen menor covarianza son aquellas que están más próximas a la medida correcta (salvo errores que no tendremos en cuenta) por lo que sería suficiente con escoger aquella estima que posea menor error, es decir, aquella que posee una varianza menor.

Para encontrar la estima de menor varianza se calcula el determinante de la matriz de covarianzas de cada una de las medidas y se toma aquella que sea menor. Se hace así ya que el determinante es proporcional al volumen del elipsoide de

error [Duda 73] y por tanto un volumen menor indicará una mayor precisión de la medida. Como se puede observar en la figura 3-14, ahora se ha reducido de forma apreciable el cálculo necesario para determinar la estima que se debe seleccionar. En este caso sólo es necesario calcular un determinante para cada estima recibida.

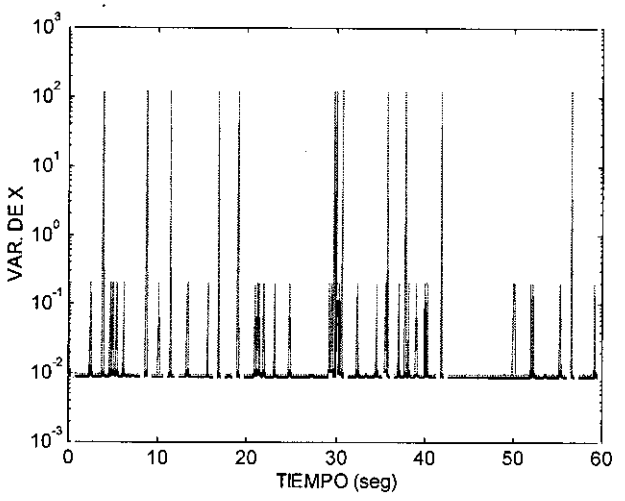


a) Medida de semejanza con la media b) Cálculo de error mínimo

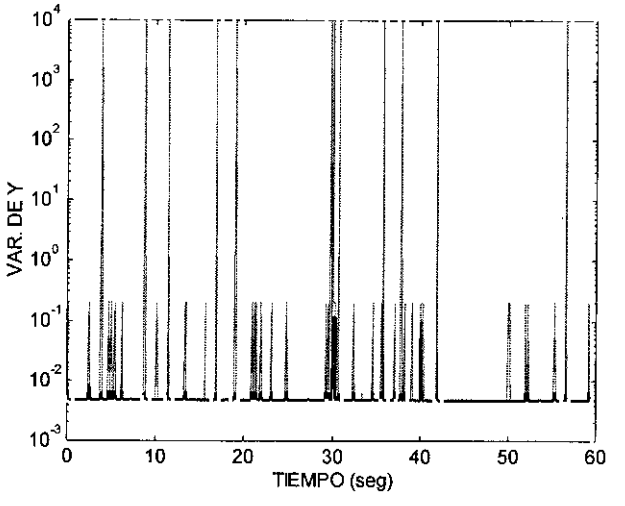
Figura 3-14. Operaciones realizadas en los dos test seleccionados.

Por ejemplo, en la figura 3-14 se muestra la covarianza de cada uno de los sensores junto con la covarianza de la medida considerada en cada momento (con un trazo más grueso). Las gráficas corresponden al Caso 1 mostrado en la sección 3.5.2. En estas gráficas se puede observar que la covarianza de las estimas consideradas es casi siempre más pequeña que la de las estimas enviadas, y en el instante que es mayor que la de alguna de las estimas enviadas es debido a que el peso de la mayoría ha sido mayor que la precisión considerada. Recordemos que esto es válido porque las estimas proceden de un filtro de Kalman, en caso contrario no se puede utilizar un test tan simple.

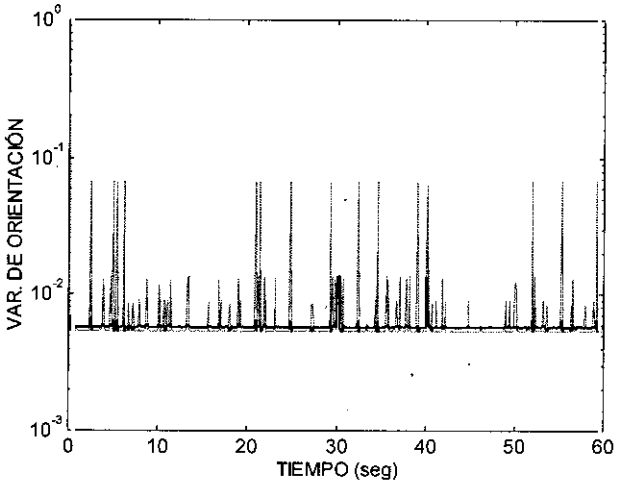
El cálculo del volumen del elipsoide de error requiere un tiempo de cálculo muy pequeño debido a que sólo necesita el calculo del determinante de cada una de las matrices de covarianzas recibidas, lo que lo hace muy adecuado para sistemas en tiempo real.



a)



b)



b)

Figura 3-15. Varianza de la medida aceptada.

3.6 Resultados del capítulo

Se han mostrado las ecuaciones necesarias para tratar los distintos casos que aparecen en un sistema de fusión de medidas de posición, debido a su funcionamiento asíncrono y distribuido. Se ha mostrado el funcionamiento de un sistema distribuido asíncrono para fusión multisensorial. Este sistema puede funcionar de forma distribuida y asíncrona con todos sus sensores trabajando en paralelo. Estos sensores pueden estar distribuidos geográficamente en caso de que así se desee.

Como se ha mostrado con distintos ejemplos, este sistema es robusto a cualquier variación o fallo en el funcionamiento de los sensores. Incluso será prácticamente insensible al malfuncionamiento de un sensor siempre que existan medidas redundantes, como se ha puesto de manifiesto en las simulaciones realizadas.

Este sistema se ha utilizado para la fusión de información de posición de un robot móvil, pero de igual forma puede usarse para la estima de cualquier otro vector de estado de un sistema o para el seguimiento de multiples objetos.

Capítulo 4

Fusión multisensorial de medidas de entorno

La integración y fusión de medidas de entorno en robots móviles es uno de los campos sobre el que dirigen su atención muchas de las técnicas actuales de inteligencia artificial. Existen multitud de métodos distintos que normalmente están orientados a la resolución de un problema concreto y con unos sensores determinados. En este Capítulo se describe un método que permite utilizar distintos sensores de entorno sin necesidad de cambios en el sistema de fusión. Este método obtiene, a partir de la estima local de los distintos sensores, una única estima más precisa de las características de los objetos detectados que rodean a un robot móvil.

Como ejemplo de su aplicación se ha utilizado la fusión de las estimas de la existencia o no de un objeto. El resultado de la fusión se usa en la construcción de mapas de creencia o mapas de ocupación, comparando su funcionamiento con el de los métodos clásicos, como los mapas de creencia o los histogramas de campo.

La red bayesiana utilizada para la fusión multisensorial permite la utilización asíncrona de los sensores, pero se ha extendido su uso a una red bayesiana dinámica para modificar su comportamiento temporal, lo que permite la adaptación dinámica del sistema al entorno. Como ejemplo del uso temporal de la red se han introducido dos factores de olvido: sobre la información a priori

y sobre la información externa sensorial, que permiten ajustar de forma fina el funcionamiento del sistema de fusión de entorno.

4.1 Introducción

Los diseñadores de robot móviles suelen utilizar diferentes tipos de robots debido a las ventajas y limitaciones que cada uno presenta en distintas aplicaciones, entornos, situaciones, etc. Aunque existe la posibilidad de cambiar el robot, en muchos casos ciertas limitaciones como la falta de resolución en determinados entornos, la velocidad de muestreo o restricciones económicas o técnicas, suelen resolverse mediante el uso de nuevos o mejores sensores y actuadores, o la adición de sensores redundantes y complementarios. En este Capítulo se desarrolla un método sencillo y eficaz que permite la fusión de medidas de entorno de diversos sensores sin necesidad de grandes modificaciones o adaptaciones en el sistema de fusión. De esta forma, un mismo sistema puede utilizar unos sensores determinados y en cualquier momento suprimir, cambiar o añadir uno nuevo. Así, sin necesidad de realizar cambios importantes en el sistema de control del robot, se puede hacer un estudio o desarrollo de distintos conjuntos de sistemas sensoriales dependiendo de la disponibilidad de los sensores y del entorno o uso que tendrá el robot móvil, o configurar el sistema sensorial de forma que se obtenga el máximo rendimiento posible de los sensores existentes.

Por supuesto, toda generalización requiere un mayor grado de abstracción y por tanto una pérdida de la información observada, pero ésta se suple mediante la mejora producida al integrar información de distintas fuentes.

Existen varias arquitecturas utilizadas en fusión multisensorial de información [Chong 90; Viswanathan 97], entre ellas hemos escogido una red distribuida de sensores, al igual que en la fusión de la información de estado, con dos niveles distintos de abstracción:

1.- Preprocesamiento y extracción de características: El tratamiento de las medidas de los sensores físicos es realizada por parte del usuario mediante cualquiera de las técnicas utilizadas comúnmente [Hall 92; Linas 90; etc.] y el resultado de su observación, el vector de características que interese al diseñador, será proporcionado al sistema de fusión para que sea fusiónado con los resultados obtenidos por otros sensores.

2.- Sistema de fusión: El sistema de fusión de identidad será diseñado mediante una red bayesiana [Pearl 88; Neapolitan 90] donde se

integrarán las estimas de los sensores; permitiéndose que el sistema sea fácil de configurar y que pueda paralelizarse en caso necesario. La información a fusiónar, el modo de fusiónarla y la información que el sistema proporciona pueden ser elegidas y diseñadas de forma sencilla por el usuario.

A continuación se muestrán distintos métodos de fusión multisensorial de medidas de identidad, sus ventajas y sus inconvenientes. Después de esta revisión, en la sección 4.2, se describe el método escogido (las redes bayesianas) para desarrollar un nuevo método de fusión y se describe cómo puede implementarse. En la sección 4.3, se muestra cómo nuestro método funciona igual que los mapas de ocupación [Elfes 86] y los histogramas de campo [Borestein 89] y cómo puede adaptarse para trabajar con otros distintos tipos, y formas de actualización, de los mapas de entorno. Una vez estudiado el funcionamiento de las redes bayesianas y su utilización en fusión sensorial, en la sección 4.5 se muestra un ejemplo de la utilización del sistema desarrollado para la fusión de medidas de entorno de dos sistemas de sensores: un cinturón de ultrasonidos y un sistema de visión estereoscópica. Una vez mostrada su capacidad para fusionar sensores distintos es necesario dotar al sistema de un cierto comportamiento temporal y dinámico. Esto se realiza en la sección 4.6, donde por medio de redes bayesianas dinámicas se adapta el sistema de fusión para que evolucione en el tiempo. Como ejemplo se implementan distintos factores de olvido y se muestran distintos casos para resaltar su comportamiento temporal y eficacia. Por último, en la sección 4.7 se realizan una serie simulaciones con datos reales y simulados para mostrar la eficacia y el comportamiento del sistema ante errores y ruido en las medidas.

4.2 Algoritmos de fusión de identidad

En un caso general tendremos múltiples sensores de diferentes tipos que observan una entidad física, objeto o suceso. Cada sensor realiza un preprocesamiento que incluye la extracción de características y la declaración de identidad de los objetos observados. La declaración de identidad de cada sensor debe ser procesada mediante un mecanismo de asociación a fin de agrupar las observaciones en conjuntos que representen el mismo objeto o entidad. Los métodos para fusiónar estas declaraciones de los sensores se pueden clasificar en: inferencia clásica, inferencia bayesiana, método de Dempster-Shafer, teoría generalizada de Thomopoulos y otros métodos directos como la lógica difusa o las redes neuronales artificiales.

4.2.1 Inferencia clásica y modelo probabilístico

La probabilidad es el método más extendido para representar la incertidumbre en la afirmación de cierta hipótesis. En la inferencia probabilística existen cuatro interpretaciones distintas del término probabilidad [Papoulis 90]: Axiomática (definición de la probabilidad de cada suceso), frecuentísta (empírica a partir del número de sucesos observados), clásica (a partir del número de sucesos posibles) y subjetiva (medida de creencia de un suceso).

Los modelos probabilísticos más utilizados para mostrar incertidumbre en la toma de decisiones son el frecuentísta y el subjetivo. Un modelo probabilístico es el método que proporciona una conexión entre los datos observados y la población existente.

La inferencia frecuentísta tradicional presupone que la probabilidad de un suceso A en un experimento dado se aproxima a su probabilidad verdadera cuando el número de pruebas que se hacen es muy grande, por lo que podemos decir que mide su repetibilidad. Sin embargo, en la interpretación subjetiva el valor que se asigna a un suceso A es una medida del estado de conocimiento o creencia en la ocurrencia de ese suceso. Así, cuando existe una total ignorancia en un conjunto n de sucesos, la probabilidad de que ocurra uno de ellos será igual para todos, es decir $1/n$. Esta interpretación es la que será utilizada en este trabajo.

En ambas aproximaciones se asume que los datos observados corresponden a uno de los N posibles objetos existentes (universo de discurso). Entonces surge la

cuestión siguiente: ¿podría encontrarse una correspondencia razonable entre los datos observados y el objeto *j*?. Para contestar a esta pregunta se supone que se dispone de un conjunto de hipótesis exhaustivas y mutuamente exclusivas que pueden explicar los datos observados, y que se debe realizar algún tipo de test que nos indique si los datos justifican la afirmación señalada. Estos test se denominan *contrastes de hipótesis* y en ellos se procede de la siguiente forma: se asume que la hipótesis H_0 es cierta y se realiza el contraste de esa hipótesis; si se supera el test, entonces las observaciones tienen una gran probabilidad de ser obtenidas si H_0 es cierta, es decir, los datos se toman como no contrarios a H_0. Al contrastar una hipótesis H_0 podemos adoptar dos decisiones, aceptarla o rechazarla; y además dicha hipótesis puede ser cierta o falsa, véanse las distintas definiciones en la tabla 4-1.

Verdadero positivo (sensibilidad): *Aceptar* la hipótesis cuando *es cierta*	Acierto: SI hay objeto	$P(+s\|+e)$
Falso negativo: *Rechazar* la hipótesis cuando *es cierta*	Fallo: error de 1ª especie (error tipo I)	$P(-s\|+e)$
Falso positivo: *Aceptar* la hipótesis cuando *es falsa*	Fallo: error de 2ª especie (error tipo II)	$P(+s\|-e)$
Verdadero negativo (especificidad): *Rechazar* la hipótesis cuando *es falsa*	Acierto: NO hay objeto	$P(-s\|-e)$

Tabla 4-1. Distintas denominaciones de los errores y aciertos de un sensor *s*.

Aunque los errores que pueden cometerse son iguales en cuanto a que son aleatorios, pueden ser distintos en cuanto a sus repercusiones. Parece lógico admitir que será de mayor alcance o gravedad el error de tipo I que el error de tipo II, pues rechazar una hipótesis que resulta ser cierta posee, al menos en principio, mayor transcendencia que aceptar una hipótesis que resulta ser falsa, debido a que se supone que en este último caso podrá ser rechazada en posteriores pruebas, mientras que si se rechaza nunca más volverá a ser utilizada. Por ejemplo, en el caso de sensores detectando obstáculos, será más grave rechazar la existencia de un obstáculo cuando realmente existe (error I) que suponer que existe un obstáculo cuando no hay nada (error II).

Puesto que no pueden hacerse los dos errores a la vez tan pequeños como queramos, es conveniente acotar el error de tipo I e intentar hacer lo más pequeño

posible el error de tipo II. Esto no es otra cosa que elaborar una regla que nos permita dilucidar la aceptabilidad o no de la hipótesis considerada, es decir, un contraste. Existen distintas reglas o contrastes para tomar dicha decisión [Duda 73; Hall 92], como son el máximo a posteriori, el de máxima verosimilitud, el test de Neyman-Pearson, el minimax o un test bayesiano.

A pesar de su alta utilización en multidud de campos, la inferencia probabilística clásica tiene serios inconvenientes en su utilización para la fusión de información sensorial, como son:

- La inferencia clásica puede generalizarse para incluir datos multidimensionales de diversos sensores, pero requiere un conocimiento a priori y un cálculo de funciones de probabilidad, lo que no es factible en muchas situaciones reales.
- Sólo se pueden contrastar dos hipótesis al mismo tiempo (la hipótesis nula H_0 y la hipótesis alternativa H_1).
- La complejidad aumenta enormemente con datos multivariables.
- La inferencia clásica no tiene en cuenta la ventaja de un conocimiento a priori, imprescindible cuando se trabaja con información sensorial.
- La probabilidad frecuentísta sólo es definible para sucesos repetibles. Esto conlleva ciertas limitaciones en las aplicaciones de fusión de datos puesto que incluyen sucesos únicos o poco repetibles.
- No indica si una hipótesis es cierta o falsa, sino más bien la verosimilitud de que los datos serán observados si la hipótesis es cierta, lo cual no es directamente aplicable al caso de fusión de datos.
- Los expertos, y en general los seres humanos, tienen dificultades a la hora de asignar valores numéricos a observaciones y juicios, la mayor parte de ellos subjetivos. Para evitar "obligar" al experto a dar un valor entre 0 y 1, algunos sistemas dan una escala numérica más amplia. Por ejemplo el sistema PROSPECTOR [Duda 79] permitía al experto expresar su grado de confianza entre -5 y 5.

- Las reglas de cálculo de la teoría de probabilidad (sumas y productos) favorecen el aumento del error en la propagación de incertidumbre a lo largo de una cadena de razonamiento.
- Es necesario considerar un gran conjunto de alternativas exhaustivo y mutuamente exclusivo, lo cual es en general bastante difícil.

Todos estos inconvenientes en la utilización de modelos probabilísticos e inferencia clásica han llevado a utilizar métodos alternativos que salven estas dificultades en su aplicación en la fusión multisensorial.

4.2.2 Inferencia bayesiana

La inferencia bayesiana resuelve algunas de las dificultades de la probabilidad clásica. Puede utilizarse tanto con probabilidades clásicas como con subjetivas (no son necesarias funciones de densidad de probabilidad). Gracias al teorema de Bayes, el concepto de probabilidad subjetiva, siempre que se utilice con cuidado, puede ser muy útil en el proceso de inferencia de la fusión de datos. El concepto fundamental de las estadísticas bayesianas es el de probabilidad condicional:

$$P(H|E) \tag{4.1}$$

La expresión anterior se puede leer como sigue: la probabilidad de la hipótesis H dada la evidencia E. Supongamos H_1, H_2,... H_k hipótesis mutuamente exclusivas y exhaustivas que pueden "explicar" un suceso E. Para calcular la probabilidad (condicional) de una hipótesis H_i se utiliza el teorema de Bayes:

$$P(H_i|E) = \frac{P(E|H_i)\cdot P(H_i)}{\sum_{n=1}^{k} P(E|H_n)\cdot P(H_n)} \quad ; \quad \sum_{i=1}^{k} P(H_i) = 1 \tag{4.2}$$

donde,

$P(H_i|E)$, la probabilidad de que la hipótesis i, H_i, sea verdad dada la evidencia E.

$P(E|H_i)$, la probabilidad de que se observe la evidencia E dada la hipótesis i como verdadera.

$P(H_i)$, la probabilidad a priori de que la hipótesis i sea cierta, independientemente de cualquier evidencia específica.

K, el número total de hipótesis posibles.

La formulación bayesiana posee básicamente dos ventajas sobre la formulación clásica:

- Proporciona una forma para determinar la probabilidad de que una hipótesis sea verdadera dada la evidencia y no para calcular, como ocurría en la clásica, la probabilidad de que una observación pueda ser debida a un objeto o suceso supuesta una hipótesis.
- Incluye la posibilidad de incorporar conocimiento a priori al calcular la probabilidad de que la hipótesis sea cierta. Esta propiedad permite su utilización en procesos de inferencia multisensorial puesto que no se necesitan funciones de densidad de probabilidad. Sin embargo, no se debe olvidar que la salida de tal proceso de fusión será tan bueno como sean los datos de entrada (las probabilidades a priori).

La ecuación (4.1) representa la probabilidad de H condicionada a la evidencia E. Sin embargo, si existen otras evidencias, y éstas no son independientes, no pueden sumarse sus efectos: es necesario representar la probabilidad condicionada que surge de su conjunción. Así, si aparece una nueva evidencia e, es necesario realizar la siguiente operación en la ecuación (4.2), obteniendo:

$$P(H|e,E) = P(H|e) \cdot \frac{P(E|e,H)}{P(E|e)} \qquad (4.3)$$

En la figura 4-1, se muestra el uso de la integración de datos de identidad mediante Bayes [Hall 92]. En ella se muestra cómo múltiples sensores observan datos paramétricos de un objeto cuya identidad es desconocida. Cada uno de los sensores proporciona una hipótesis sobre la identidad del objeto. Para cada sensor, un dato a priori proporciona una estima de la probabilidad de que el sensor declare el objeto como de tipo i y que de hecho sea el objeto j, $P(O_j|declara\ i)$. Estas declaraciones de los sensores se combinan mediante el teorema de Bayes para obtener una probabilidad conjunta de la posibilidad de que sea el objeto j:

$$P(O_j \mid D_1,...,D_n); \qquad j = 1...M \qquad (4.4)$$

Siendo D_i la declaración del sensor i. La declaración conjunta es la elección del valor más alto de la ecuación (4.4) denominándose regla de decisión de la probabilidad máxima a posteriori.

Cuando no existe información a priori relativa a la verosimilitud de H_i, se utiliza el principio de indiferencia que emplea el mismo valor de $P(H_i)$ para toda i.

A pesar de su amplia utilización, existen reservas en su aplicación por parte de los seguidores de la escuela estadística frecuentísta debido a dos causas [Jefferys 92]:

- Argumentan que si sólo una hipótesis, H_i, puede ser cierta no tiene sentido hablar de la probabilidad de H_j. Esto es lógico desde el punto de vista frecuentísta, pero no si se interpreta la probabilidad como un grado de la confianza en una hipótesis.
- No hay un modo universalmente aceptado de asignar las probabilidades a priori $P(H_i|E)$, necesarias en el teorema de Bayes. Además, los mismos datos pueden llevar a conclusiones diferentes. A pesar de ello, los defensores de la inferencia bayesiana sostienen que no hay nada básicamente erróneo en la probabilidad subjetiva, puesto que la estadística frecuentísta no es mucho más objetiva.

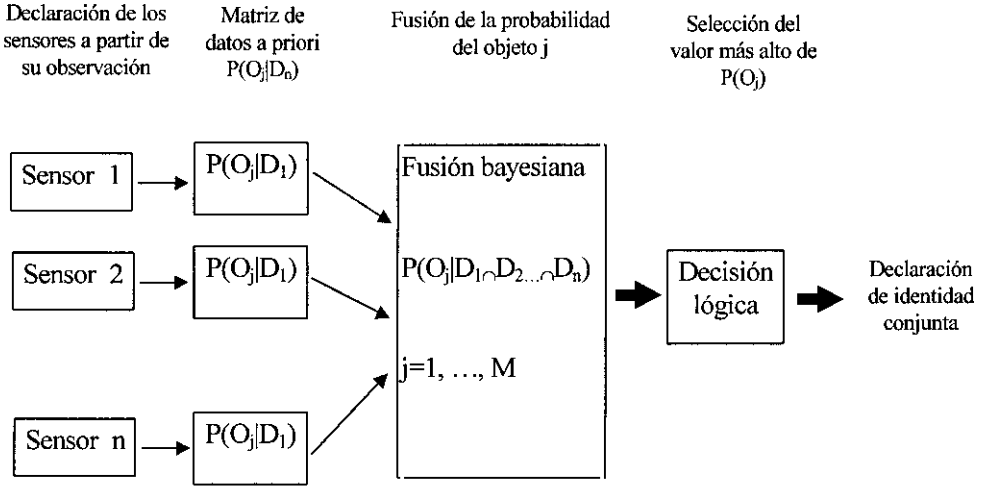


Figura 4-1. Integración de datos de identidad mediante Bayes.

Un ejemplo de un sistema experto donde se ha utilizado la formulación de Bayes con éxito es en el programa PROSPECTOR [Duda 79], aplicado en la localización de yacimientos de minerales.

Desafortunadamente, debido a que el mundo es enormemente complejo, el tamaño del conjunto de probabilidades combinadas que es necesario tener en

cuenta crece de forma 2^n, donde n es el número de hipótesis a considerar. Así, el teorema de Bayes es inaplicable directamente porque:

- Son necesarias demasiadas probabilidades. Además las personas son malas estimadoras de probabilidades condicionales.
- El tiempo empleado en calcular las distintas probabilidades posibles es demasiado grande para aplicaciones en tiempo real.

A pesar de todos estos problemas, la estadística bayesiana proporciona un interesante sistema para razonar bajo incertidumbre. Por ello se han desarrollado distintos mecanismos que consiguen que el problema sea tratable, como son la inclusión de factores de certeza a las reglas y las redes bayesianas.

4.2.2.1 Factores de certeza

La forma más intuitiva de expresar incertidumbre en una base de conocimiento es introduciendo factores numéricos de probabilidad de certeza en cada premisa. Estos factores numéricos, que están basados en la lógica bivaluada y en la teoría de probabilidad, presentan graves limitaciones (algunas demostraciones de ello pueden encontrarse en [Cuena 86]). El factor numerico/probabilístico más conocido son los *factores de certeza*.

Fueron definidos en el sistema MYCIN [Shortliffe 75; Buchanan 94]. Este sistema es un sistema experto bayesiano puro utilizado para recomendar terapias apropiadas en pacientes con infecciones bacterianas. Representa la mayor parte de su conocimiento en forma de reglas, cada una de las cuales tiene asociado un factor de certeza, que representa una medida sobre la evidencia que existe de que la conclusión sea cierta cuando se verifica el antecedente.

Un factor de certeza, *CF(H,E)*, toma sus valores entre -1 (completamente falso) y +1 (completamente cierto) y refleja un equilibrio entre la evidencia a favor y la evidencia en contra. En el trabajo original [Shortliffe 75] se da una definición de cómo calcular *CF(H,E)* pero era incompatible con la visión bayesiana de la probabilidad condicionada. Posteriormente, en [Heckerman 86] se realizaron unas pequeñas modificaciones en su definición para lograr que sí fueran compatibles.

Una de las principales ventajas que proporciona la modularidad del sistema de reglas de MYCIN es que permite considerar las relaciones individuales antecedente/consecuente de forma independiente unas de otras, mientras que en un sistema puramente bayesiano, si existe otra evidencia que es relevante para

deducir la hipótesis H es necesario considerar la combinación de probabilidades. Esta modularidad de sus reglas provoca también su principal desventaja: siempre existe el peligro de que las justificaciones de una creencia sean importantes para la determinación de sus consecuencias, provocando entonces deducciones cíclicas. Esto debe evitarse a toda costa. Por ejemplo, si una regla describe una relación de causalidad inversa, es decir, la evidencia se deriva de un indicio a partir de su causa y otra regla describe la causa a partir de la observación del indicio; se debe procurar que la evidencia se derive de una forma que no se utilice de nuevo en otra regla, para no volver a deducir la información inicial, que daría lugar a deducciones erróneas [Rich 94].

4.2.2.2 Redes bayesianas

Este otro planteamiento conserva el formalismo del razonamiento bayesiano. Para realizar la simplificación que requiere la inferencia bayesiana pura se confía en la modularidad del mundo. La idea principal es que la mayoría de los sucesos del mundo son condicionalmente independientes de los demás y por lo tanto no es necesario representar explícitamente las probabilidades existentes entre todos los sucesos (muchas de ellas serían cero) sino sólo la de los grupos que interaccionan.

La relación o dependencia causa/efecto entre las variables se realiza en forma de una red de restricciones. Se utiliza un grafo acíclico dirigido donde los arcos representan las relaciones de causalidad entre las variables, mientras que los nodos del grafo representan los conceptos (X, Y, Z, W,...) o variables correspondientes. Para poder utilizarlo como base de razonamiento probabilístico es necesaria más información: para cada valor de un nodo padre (causa) necesitamos conocer qué evidencia existe sobre los valores que pueden tomar los nodos hijos (efecto). Esta información se sitúa fuera del grafo, en tablas de probabilidades condicionales.

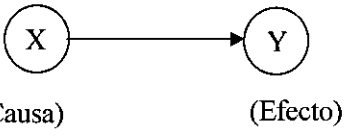


Figura 4-2. Relación causa - efecto.

Para que este sistema sea útil en la resolución de problemas, es necesario proporcionar un mecanismo que calcule la influencia de un nodo arbitrario sobre otro cualquiera. Para realizar esto es necesario que el grafo acíclico dirigido se

convierta en un grafo acíclico no dirigido y cuyas aristas se utilicen para transmitir probabilidades en cualquier dirección dependiendo de dónde provenga la evidencia, de forma que para el razonamiento sobre uno de los atributos se usará la causa o el efecto según convenga. El método clásico para transmitir probabilidades es mediante paso de mensajes π-λ (véase el Apéndice A) [Pearl 88]. En general, se tendrá información sobre los nodos de los bordes y se querrá saber algo sobre los nodos intermedios.

Una de las restricciones fuertes que deben existir sobre la estructura de la red es que debe ser una red sin ciclos, es decir, se debe separar de cualquier atributo aquello que es causa de lo que es efecto. Cuando se produce dicha separación pueden distinguirse distintas estructuras de red, como son las redes con estructura en cadena, en árbol y en poliárbol, que dan lugar a complejidades de cálculo cada vez mayores, véase el Apéndice A.

4.2.3 Teoría de Dempster-Shafer

También conocida como teoría de la evidencia, fue propuesta como una alternativa a la teoría bayesiana [Dempster 67; Shafer 76]. Aunque ambas teorías producen idénticos resultados cuando todas las hipótesis consideradas son mutuamente exclusivas y el conjunto de hipótesis es exhaustivo [Waltz 90], su forma de razonamiento es completamente diferente.

El método de Dempster-Shafer, o razonamiento evidencial, modela la forma que tienen los humanos de asignar evidencia a proposiciones hipotéticas. Dempster argumenta que los humanos no asignan evidencia (probabilidades) a un conjunto de hipótesis mutuamente exclusivas y exhaustivas, sino que asignan estimas de creencia a combinaciones de hipótesis. Así, en el razonamiento evidencial, se distingue entre hipótesis y proposiciones (conjunto de hipótesis). Ésta es la gran diferencia con las técnicas bayesianas, que consideraban proposiciones individuales, cada una con un grado de creencia. Sin embargo, la teoría de Dempster-Shafer considera conjuntos de proposiciones (universo exhaustivo de hipótesis mútuamente excluyentes, conocido como *marco de discernimiento*) y les asigna a cada uno de ellos un intervalo [Shafer 76] del tipo [*Creencia, Verosimilitud*]. A cada marco se le asigna un peso conocido como *función de densidad de masa* de forma que la suma de todos los subconjuntos sea 1.

La Creencia (normalmente se denota por Bel, *belief*) representa la fuerza de la evidencia en favor de un conjunto de proposiciones. El rango va de 0 (evidencia nula) a 1 (certeza absoluta).

La Verosimilitud (PL, *plausibility*) se define como Pl(s) = 1 - Bel(¬s). Su rango también va desde 0 hasta 1 y representa el alcance con que la evidencia en favor de ¬s deja lugar a la creencia s. Por ejemplo, si se tiene evidencia cierta a favor de ¬s, entonces Bel(¬s) es 1 y Pl(¬s) es 1. Esto indica también que Bel(s) es 0 y Pl(s) es 0.

Ejemplos de aplicaciones y el afianzamiento de la teoría de la evidencia pueden encontrarse en la colección de artículos de [Yager 94]. Distintas modificaciones y adaptaciones de la teoría han sido realizadas para resolver distintos problemas como por ejemplo en diagnóstico de fallos en sistemas físicos [Gertler 92], ejemplos de distintas aplicaciones para identificar objetos en un sistema multisensor [Bogler 87], fusión multisensorial para reducción de ambigüedades en el reconocimiento de objetos [Hutchinson 92], formalizaciones para su razonamiento cualitativo [Wong 90] o ampliaciones de la teoría para toma de decisiones [Yang 94a; Yang 94b].

La amplia utilización de esta técnica es debida a que el intervalo de creencia-verosimilitud no sólo mide el nivel de creencia sobre algunas proposiciones, sino también la cantidad de información que se posee. Por ejemplo, si se dispone de tres hipótesis y ninguna información adicional, la probabilidad de que sean ciertas para cada una de ellas está en el rango [0, 1] y conforme se acumule evidencia sobre las hipótesis el intervalo se irá estrechando. En la probabilidad bayesiana pura todas tendrían 0.33 al comenzar, pero podrían llegar, después de analizar la evidencia sobre ellas, al mismo valor, siendo imposible distinguir ambos casos. Esta diferencia puede ser importante si se pretende decidir sobre si se reúne más evidencia o se actúa sobre la que ya hay.

Sin embargo también tiene ciertas desventajas:

- Si el marco de discernimiento contiene n elementos, existen 2^n subconjuntos. La asignación de la función de densidad de masa m a cada subconjunto debe ser tal que la suma de todos los subconjuntos sea 1. Para evitar que el reparto de todos los valores sea inabordable, a aquellos conjuntos que no son relevantes para el problema se les asigna el valor 0. Sin embargo, la utilización del

factor de escala para normalización a 1 da lugar a resultados incorrectos que no se corresponden con la realidad [Zadeh 79].

- La evaluación del grado de credibilidad y verosimilitud requiere un tiempo que crece exponencialmente con la cardinalidad del conjunto de hipótesis. Barnett [Barnett 81] consiguió reducir la complejidad al caso lineal a costa de introducir restricciones adicionales acerca del tipo de evidencia a considerar, pero no es factible en muchos casos.
- Por último, se observa [Waltz 90; Buede 97] que, en el tratamiento de la incertidumbre, la teoría probabilística converge a una solución de forma mucho más rápida que la teoría de la evidencia. Esto la hace menos recomendable en el caso de sistemas que requieren una rápida respuesta.

A pesar de estas desventajas esta técnica es muy utilizada en fusión de información sensorial y en [Murphy 98] se propone una nueva forma de utilización de dos aspectos de la teoría de Dempster-Shafer (la *métrica de conflicto*, que cuantifica la discordancia entre conjuntos de evidencias; y la *función de aumento*, que permite incorporar conocimiento en el proceso de inferencia de modo natural) que son muy útiles para la función sensorial y que abre nuevas perspectivas de futuro en su utilización como una importante técnica de fusión sensorial.

4.2.4 Teoría generalizada de Thomopoulos

Es una nueva generalización de la teoría bayesiana propuesta por Thomopoulos [Thomopoulos 90]. Esta teoría se centra básicamente en la asignación de evidencia a las hipótesis o proposiciones y la relación de estas aseveraciones con las decisiones de fusión. Explícitamente separa las hipótesis de las decisiones y destaca la elección de distintos test de hipótesis para cuantificar los diferentes niveles de los datos.

El método del procesamiento generalizado de la evidencia es análogo al de Dempster-Shafer en cuanto a que cada sensor recoge cierta evidencia que representa mediante masas de probabilidad. Pero se diferencia del método de Dempster-Shafer en que combina las masas de probabilidad con datos de masa basados en la probabilidad condicional a priori de las hipótesis involucradas. El

espacio de decisiones se divide de forma que se optimiza un cierto criterio (por ejemplo, minimizar falsas alarmas, minimizar una función de riesgo, etc.)

Con este método, Thomopoulos proporciona una extensión de las reglas de combinación para múltiples hipótesis y una formulación para varios criterios de optimización. Esta formulación permite una fusión y decisión general que busca la optimización de criterios tales como la máxima probabilidad de detección de una falsa alarma. La investigación sobre el tema está dirigida a su implementación, a la comparación con técnicas de Dempster-Shafer y con las formulaciones bayesianas y en su aplicación en sistemas de fusión.

4.2.5 Métodos heurísticos

Hasta ahora no se han modificado los fundamentos de la lógica ni de la teoría de conjuntos, sino que se han ampliado utilizando construcciones adicionales de la teoría de probabilidad.

Sin embargo, existen una gran cantidad de métodos alternativos para la decisión y fusión de identidad que tratan de dar un enfoque distinto al probabilístico en los métodos utilizados.

Unos métodos se basan en considerar el problema de la fusión de identidad como si se tratase de un grupo de humanos, donde cada uno representase un sensor. En este caso se pueden utilizar técnicas de toma de decisiones en grupo, entre las que pueden destacarse la votación, los modelos de puntuación, el método de la clasificación, etc. Una breve descripción de ellos se encuentra en [Hall 92].

Otros enfoques de gran actualidad distintos al probabilístico son: (1) modificar la teoría de conjuntos (como por ejemplo la noción de pertenencia a un conjunto) y las operaciones lógicas sobre ellos (como pertenencia, unión,...), que conducirá a la lógica difusa y a los conjuntos difusos; o (2), la utilización de funciones no lineales distintas a las probabilísticas por medio de redes neuronales artificiales, de forma que a partir de un aprendizaje sobre una serie de entradas sensoriales se dispone una red capaz de responder correctamente a distintas entradas sensoriales.

4.2.5.1 Lógica difusa

La lógica difusa, o borrosa, [Zadeh 75; Zadeh 78] es una opción a la representación de conocimiento incierto. Surge para especificar cómo de bien un

objeto satisface una descripción vaga (propia de los expertos) o incertidumbre sobre una definición o un término lingüístico. Mientras que la teoría de conjuntos tradicional define la función de pertenencia a un conjunto con un predicado booleano (0 ó 1, pertenece o no pertenece), la teoría de conjuntos difusos representa dicha función como una distribución de probabilidades (un grado de pertenencia a un conjunto).

Uno de los problemas de la lógica difusa es su inconsistencia con la lógica de primer orden [Cuena 86]. Otra de las grandes desventajas es la existencia de multitud de álgebras difusas con distintas propiedades dependiendo de las funciones que se utilicen para construir las funciones básicas AND y OR. Si además interesa que dicha álgebra sea conmutativa y asociativa es necesario utilizar funciones matemáticas que verifiquen estas propiedades (*t-normas*, para construcción de la intersección, y *t-conormas*, para construcción de la unión). Con estas funciones se pueden construir hasta nueve álgebras difusas distintas con diversas propiedades. Pero todas tienen el inconveniente de que ninguna de ellas presenta todas las propiedades del álgebra de Boole, excepto cuando se aplican a conjuntos clásicos.

Una vez definidos los conjuntos borrosos, mediante las funciones de distribución que representan el grado de pertenencia de un elemento a un conjunto y las funciones que permiten operar sobre ellos, se pueden definir sistemas de razonamiento que estén basados en estas definiciones. Ejemplos de integración de información sensorial utilizando estas técnicas son [Dubois 92], [Agapito 93], [Chee 96] o [Gasós 99].

A pesar de la amplia utilización de los conjuntos difusos en problemas reales y de los intentos de generalizar la teoría de Dempster-Shafer para tratar conjuntos difusos [Yen 90], esta forma de tratar la incertidumbre tiene muchos detractores y, como se señala en [Russell 95], algunos autores opinan que la teoría de conjuntos borrosos no es un método apropiado para el razonamiento en condiciones de incertidumbre.

4.2.5.2 Redes neuronales artificiales

Una alternativa a los métodos estadísticos de reconocimiento de patrones es el uso de las redes neuronales artificiales [Pao 93; Bishop 95]. El elemento básico de las redes neuronales es el perceptrón; y su comportamiento dependerá de la función de evaluación de cada neurona y el modo de conexión entre neuronas

[Haykin 94; Hilera 95]. El perceptrón multicapa tiene la ventaja de que no necesita ninguna suposición sobre la forma de las distribuciones asociadas a los datos sensoriales, evitando así el típico uso de distribuciones gaussianas y permitiendo el uso de funciones no lineales.

La gran ventaja de las redes neuronales artificiales es su potencia de cálculo debido a: su estructura, que permite cálculos distribuidos y masivamente paralelos; y a su habilidad para aprender y por lo tanto de generalizar (al ser capaces de producir salidas razonables a entradas que no fueron utilizadas en el proceso de aprendizaje).

En la práctica, las redes suelen integrarse en un sistema más complejo, es decir, el problema complejo a tratar se descompone en un número de tareas relativamente simples y las redes neuronales son asignadas a un subconjunto de estas tareas que se ajustan a sus inhenerentes capacidades (por ejemplo, reconocimiento de patrones, clasificadores, memorias asociativas, control, ...).

El uso de redes neuronales artificiales ofrece las siguientes propiedades y capacidades [Haykin 94; Hilera 95]: no linealidad, correspondencia entrada-salida, adaptativilidad, respuesta a la evidencia (clasificación), información contextual, tolerancia a fallos, implementación VLSI, uniformidad en el diseño y análisis, y analogía con neurobiología. Todas estas propiedades y capacidades han dado lugar a una gran cantidad de aplicaciones. Las aplicaciones más relacionadas con la fusión de datos son las de clasificación o reconocimiento de imágenes. Por ejemplo ALVINN (*Autonomous Land Vehicle In a Neural Network*) para integración de dos sensores en navegación autónoma [Pomerleau 93], pero limitado a casos muy concretos; en el "matching" para visión estereoscópica [Cruz 95]; en análisis dinámico de imágenes (para resolver ambigüedades en segmentación de imágenes y en el proceso de análisis de movimiento del robot) [Huntsberger 92] o como fusión de datos para control y guiado mediante técnicas neurofuzzy [Doyle 95].

Las mayores desventajas para integración multisensorial radican precisamente en su principal característica: la necesidad de entrenamiento de la red para obtener un funcionamiento adecuado para unas entradas definidas. En caso de utilizar otro sensor para el cual no ha sido entrenada la red será necesario volver a entrenar la red para dicho sensor. A pesar de que no son directamente aplicables como integradores de información con incertidumbre, su funcionamiento como clasificadores es inmediato y se encuentra muy extendido.

4.3 Fusión de medidas de entorno mediante redes bayesianas

La idea intuitiva para realizar la fusión de medidas de entorno mediante técnicas bayesianas es encontrar la probabilidad a posteriori de la existencia de un atributo a partir de la probabilidad a priori de la existencia de ese atributo y de su estimación por parte de cada uno de los sensores. Para ello se utiliza la formulación de Bayes y la probabilidad condicional de la forma:

$$P(O|S1,S2,...) = P(S1|O)\,P(S1|S2,S3,...) + P(S2|O)\,P(S2|S1,S3,...) + ... \quad (4.5)$$

Donde $P(O|S1,S2,...)$ es la probabilidad a posteriori de que la celda esté ocupada (hipótesis, el atributo en este caso es ser parte de un objeto o no), $P(Si|O)$ probabilidad estimada por el sensor i de que la celda esté ocupada sabiendo que está ocupada y $P(Si|Sj,\ i\neq j)$ probabilidad de que del sensor i enuncie esa hipótesis sabiendo como se encuentran los demás sensores. Esto no es un modelo factible debido a la gran cantidad de combinaciones que es necesario tener en cuenta. Para simplificar este procedimiento de calculo de la probabilidad a posteriori del estado de una celda utilizaremos las redes bayesianas.

Antes de realizar la fusión de medidas, es conveniente ver cómo funcionan las redes bayesianas, es decir, cual es la base teórica en la que se apoyan.

4.3.1. Funcionamiento de las redes bayesianas

En nuestro sistema de fusión, al igual que en otras técnicas de Inteligencia Artificial, es muy importante separar el conocimiento de la inferencia realizada. Esto nos permitirá ampliar o depurar el conocimiento que se posee sobre los sensores o el entorno sin tener que modificar el algoritmo de inferencia. Para ello es conveniente separar el *motor de inferencia* de la *base de conocimiento*.

4.3.1.1. Base de conocimiento

Toda base de conocimiento de una red bayesiana, y por lo tanto del sistema de fusión desarrollado, se compone de dos elementos: la red y las tablas de probabilidades condicionales (matrices bayesianas), que representan las relaciones entre conceptos o atributos el sistema y el efecto que cada uno ejerce sobre los demás.

Red

Es una relación semántica entre distintos atributos. Los arcos de dicha red representan la relación causa/efecto entre los atributos (nodos) que componen la red. Un atributo puede tener una única causa, como en la figura 4-3a donde un atributo, *X*, de un objeto (que puede tomar *n* valores exclusivos y exhaustivos) es observado por un sensor *S*, o múltiples causas, como por ejemplo en la figura 4-3b donde un sensor *S* observa varios atributos o aspectos de un objeto o entorno.

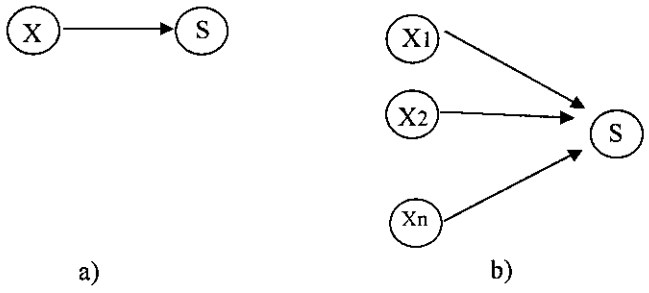


Figura 4-3. Relación causa - efecto.

Tengamos en cuenta que si un objeto posee cierta propiedad o atributo, ésta es causa de una medición concreta en el sensor *S*.

Existen distintas estructuras de red dependiendo de las relaciones existentes entre las variables o atributos considerados, véase el apéndice A para ver un ejemplo de cada una de ellas. Estas pueden ser:

- Estructura en cadena (silogismo), donde cada causa tiene como máximo un único efecto, véase la figura A-2.
- Estructura en árbol, en este tipo de relaciones causa/efecto cada nodo puede poseer una causa y tener varios efectos, figura A-3.
- Red simplemente conexa (poliárbol), en la red existen múltiples causas y múltiples efectos, figura A-4.

Matrices bayesianas

Representan el modelo de las relaciones entre los nodos de la red. En nuestro caso, pueden considerarse como el peso que cada uno de los distintos atributos (causas) ejerce sobre sus efectos. Esto es, el peso o "seguridad" con que un sensor observa (efecto) cada uno de los valores posibles de un determinado atributo de un objeto (causa).

Por ejemplo, en el caso de una relación del tipo X *implica* Y (una única flecha llega a Y), como en la figura 4-3a, la matriz de relación entre ambos conceptos se muestra en la tabla 4-2.

X\Y	Y_1	Y_2	...	Y_k	...	Y_m
X_1				:		
X_2				:		
...				:		
X_j	.. .. ..	.. .. ..	.. .. ..	$P(Y_k/X_j)$		
...						
X_n						

Tabla 4-2. Modelo de probabilidad para una causa.

En caso de que se tengan relaciones del tipo mostrado en la figura 4-3b, la matriz se representa como una tabla con tantas filas como combinaciones posibles de los atributos de $V_1, V_2, \ldots V_n$ tengamos (valores $v_1, v_2, ..v_m$). Véase la tabla 4-3.

V_1	V_2	.....	V_n	X	P
v_1	v_2	.....	v_n	x_j	$P(x_j/v_1,v_2,v_3,\ldots v_n)$

Tabla 4-3. Modelo de probabilidad de múltiples causas.

En situaciones más complejas (como en poliárboles), las tablas tienen múltiples entradas y no es posible representarlas de forma sencilla.

4.3.1.2. Motor de inferencia

Además de indicar la forma en que interaccionan causas y efectos, es necesario señalar cómo realizar los razonamientos necesarios para propagar las estimas por la red diseñada. Es decir, el procedimiento por el cual, conocidas las distribuciones de probabilidad en unos puntos, se obtienen las distribuciones de probabilidad en los puntos requeridos a partir de las tablas de relaciones causa/efecto. Esto se conoce como motor de inferencia.

Para obtener el motor de inferencia se parte de la idea de causa-efecto, con ella se podrán propagar las estimas (probabilidades) por la red para poder definir la credibilidad de un atributo. Se trata de, primero distinguir qué premisas son causales y cuales son efecto; y después, de propagar certeza sobre los efectos y sobre las causas de la red: el conjunto de ambos definirá la credibilidad. Los dos conceptos básicos para comprender cómo se propaga la evidencia son la independencia condicional y la separación direccional [Pearl 88]. Una breve descripción de ambos puede encontrarse en el Apéndice A.

Los métodos y algoritmos para propagar las estimas dependerán de los tipos de estructuras de la red (relaciones) existentes. El primer algoritmo importante para la evaluación de redes de decisión fue diseñado por [Shachter 86], pero el algoritmo más representativo es el diseñado por Pearl para redes en forma de árboles [Pearl 82] que utiliza un procedimiento de paso de mensajes (mensajes π-λ) para la comunicación de evidencia. Posteriormente [Kim 83] generalizó el algoritmo de Pearl para poliárboles. En el apéndice A, se muestra cómo se calculan los mensajes π y λ para propagación de evidencia en distintas redes, como la red en cadena y el poliárbol.

4.3.2. Implementación de las redes bayesianas

Existen multitud de programas comerciales y no comerciales que permiten trabajar con redes bayesianas y realizar inferencias y toma de decisiones sencillas (una descripción y ejemplos de algunos de ellos se pueden encontrar en el apéndice A). La implementación de las redes bayesianas puede ser distribuida o no, dependiendo del tipo de programación utilizado. Un ejemplo clásico de programación no distribuida para el cálculo de redes bayesianas para árboles basado en el algoritmo de paso de mensajes π-λ de Pearl es mostrado en [Morawski 89]. Un algoritmo para poliárboles puede encontrarse en [Russell 95] donde utiliza un algoritmo de encadenamiento hacia atrás para calcular la probabilidad a posteriori de un nodo. Este algoritmo puede modificarse para que funcione con encadenamiento hacia delante, siendo su implementación en computadoras paralelas muy sencilla y consiguiendo así un cálculo distribuido de la red.

La implementación de la red utilizada en nuestro sistema será distribuida, donde cada nodo es un proceso independiente y realizándose la transmisión de los mensajes π y λ, mostrados en la sección 4.3.1, por medio de mensajes entre

procesos. El algoritmo utilizado es el presentado en [Diez 94] y se encuentra esquematizado en el apéndice A. El algoritmo no necesita un mecanismo global de control del funcionamiento de la red, por lo que puede implementarse como una red distribuida y cada nodo (proceso) de la red sabe qué mensajes debe enviar a partir de los mensajes que recibe. La única restricción es que es necesario que la red no presente ciclos. Estos no aparecen en los ejemplos mostrados anteriormente (árboles y poliárboles), sino sólo en redes múltiplemente conexas (aquella cuyos nodos están conectados por más de un camino, conocido como *ciclo*). Si se utiliza como diseño del sistema de fusión una red de conexión múltiple, estos algoritmos no serán válidos y se deberá recurrir a algoritmos que eliminen los ciclos. En cualquier caso, los ciclos normalmente son eliminados "off-line", es decir, antes de utilizar la red por el sistema de fusión, por lo que el método empleado no será tenido en cuenta en nuestro caso, puesto que es un preprocesamiento que se realiza después de que el diseñador especifique su sistema de fusión y previo al funcionamiento del sistema. Por lo tanto la fusión multisensorial es ajena al método utilizado para la eliminación de ciclos. Los algoritmos para el tratamiento de ciclos se pueden clasificar en:

- Métodos de *agrupamiento*, transforman la red a un poliárbol probabilísta equivalente mediante la eliminación de los nodos que dan lugar a los ciclos (*árbol de cliques*) [Lauritzen 88]. Estos métodos han llegado a ser algoritmos estándar para inferencia en redes bayesianas.

- Métodos de *condicionamiento*, efectúan la transformación de la red a poliarbol dando valores concretos a las variables, y a continuación evaluan un poliárbol por cada concretización efectuada. Puede ser *global*, desarrollado por [Pearl 88], algoritmo poco eficiente que no ha sido implementado en un sistema real; o *local*, desarrollado por [Diez 96], donde se busca un condicionamiento local de las variables que forman el ciclo y se evita construir un árbol de cliques.

- Métodos de *simulación estocástica*, utilizan la red para producir una gran cantidad de modelos concretos de dominio congruentes con la distribución de la red. Proporcionan una aproximación de la evaluación exacta. Este método transforma la red en una cadena de Markov. La idea consiste en aislar probabilísticamente un nodo de

la mayoría de los otros nodos de la red. Los algoritmos estocásticos se ejecutan con rapidez en la práctica, pero puede que no produzcan resultados totalmente correctos. Un ejemplo de este método lo podemos ver en [Pradhan 94], sistema experto para medicina interna.

Téngase en cuenta que cuando se utilizan aproximaciones en el cálculo de la red, el sistema de fusión multisensorial será ajeno al método utilizado en el sentido de su utilización, pero no en cuanto al resultado obtenido. Así, es útil tener resultados empíricos o teóricos para comprobar qué aproximaciones producen los resultados correctos. Esto se conoce como análisis de sensibilidad. Esto está relacionado con el análisis numérico y comprueba la sensibilidad del problema, es decir, si ligeros cambios en los datos producen grandes cambios en la respuesta (en ese caso se denomina *mal-condicionada*).

Pero, independientemente del método de cálculo de la red, el cálculo exacto de una red bayesiana general sin ningún grado de restricción es un problema NP complejo [Cooper 90; Dagum 93]. Por lo que es aconsejable que el modelo sensorial que se desarrolle sea simple y, en la medida de lo posible, evite la existencia de conexiones múltiples.

4.4. Utilización de la red bayesiana en ejemplos clásicos

Antes de integrar la información sensorial es necesario definir una forma de soportar y almacenar las estimas de los distintos sensores y la fusión de éstos. Una de las formas más ampliamente utilizadas para almacenar información multisensorial es un enrejado multidimensional de celdas donde cada celda corresponde a una región concreta del espacio, y pueden dividirse en dos vertientes:

1.- Los Mapas de certeza (*certainty grid)* [Moravec 88] o el muy utilizado Mapa de ocupación (*occupancy grid)* [Elfes 87; Matthies 88; Elfes 90; Menezes 95; Dam 96]. Su característica principal es que cada medida realizada por el sensor provoca la variación de probabilidad de un amplio conjunto de celdas del mapa. Las celdas actualizadas dependerán del modelo de sensor utilizado.

2.- Los Histogramas de campo (*field histogram grids*) [Borenstein 89; Borenstein 91a], y variantes como mapas de Certeza de Presencia de Obstáculo (CPO) [Schneider 96]. En este caso las celdas que sufren una variación de su probabilidad son aquellas en las que el sensor detecta un objeto o están relacionadas con ésta. En ciertos casos puede llegar a ser simplemente la actualización de una única celda. Este método funciona debido a la alta frecuencia de muestreo de los sensores utilizados. En caso de que se actualice únicamente las celdas medidas será necesario la inclusión de un factor de olvido.

Muchas son las ventajas de la representación de la información multisensorial en un mapa basado en un enrejado o conjunto de celdas. Entre éstas cabe destacar [Luo 92]: (1) que permiten una automática correspondencia espacial de los atributos detectados; y (2) permiten la fusión y procesamiento en cada celda antes de realizar un tratamiento de más alto nivel (lo cual es necesario en muchas aplicaciones en tiempo real, debido al excesivo tiempo requerido en inferencias de alto nivel). La principal desventaja de esta representación reside en su implementación, ya que el almacenamiento del mapa requiere gran cantidad de memoria.

El enrejado, o conjunto de celdas, (de 2 o 3 dimensiones) puede estar centrado en el vehículo o en un sistema de referencia externo, dependiendo de las

necesidades, utilización o requerimientos del sistema. En cada celda los atributos considerados son representados por variables aleatorias, expresadas en términos de la probabilidad de que esa celda contenga algún atributo del entorno. El atributo más utilizado en cualquier representación es el grado de ocupación, es decir, si la celda está o no ocupada por un objeto. Este es un atributo básico en la integración sensorial cuando el móvil necesita detectar y esquivar objetos. Así, este tipo de representaciones del entorno se encuentra directamente asociado a funciones de control de un robot como la planificación de caminos y la esquiva de obstáculos. Ejemplos de este uso son [Elfes 90; Borenstein 91b].

Antes de mostrar el diseño del sistema de fusión de entorno en nuestro sistema sensorial, vamos a ver cómo funcionan las redes bayesianas en la fusión de información y compararemos su funcionamiento con el método de Elfes (Mapa de certeza con un modelo complejo del entorno) y con el método de Borenstein (Histograma de campo) para mostrar que el método de fusión presentado en esta Tesis es equivalente a ambos, adaptándose a cualquiera de los dos conceptos dependiendo de los intereses del usuario o las cualidades del robot utilizado.

4.4.1. Comparación de las red bayesiana con el Mapa de certeza de Elfes

Se ha escogido el método de Elfes porque utiliza la formulación de probabilidad condicional de Bayes para actualizar la información de cada celda a partir de la información de los sensores y un modelo complejo de éstos. También es interesante comparar el funcionamiento de nuestro método de fusión con el modelo completo de entorno que utiliza Elfes, porque es el primer método que utiliza mapas de ocupación. Además, es la fuente de inspiración en la que se basan muchos otros métodos que utilizan mapas de ocupación y que básicamente añaden diversas variantes en el cálculo del modelo utilizado para los sensores disponibles.

Elfes [Elfes 87; Matthies 88] aplica el teorema de Bayes, ecuación (4.6), para actualizar el estado de una celda s que puede tomar uno de los n posibles valores disjuntos, s_i, suponiendo que todas las celdas son independientes.

$$P(s_i \mid e) = \frac{P(e \mid s_i) \cdot P(s_i)}{\sum_j P(e \mid s_j) \cdot P(s_j)} \tag{4.6}$$

donde, e es la evidencia obtenida, $P(s_i)$ la probabilidad de que la celda se encuentre en el estado s_i y $P(e \mid s_i)$ la probabilidad de que la evidencia e esté presente cuando el sistema se encuentra en el estado s_i.

A partir de la ecuación (4.6) y teniendo en cuenta que las celdas corresponden a un mapa de ocupación y que sus estados son ocupada (*ocu*) y vacía (*vac*) se obtiene la ecuación (4.7). Donde además se incorpora la evolución temporal en el cálculo del estado de una celda, puesto que incluye el conjunto de evidencias hasta el instante actual t, $\{r\}_t$, en $P(ocu/\{r_t\})$ y la nueva evidencia r_{t+1} en $t+1$ (que da lugar al conjunto de evidencias $\{r\}_{t+1}$) $P(r_{t+1}/ocu)$:

$$P(ocu \mid \{r\}_{t+1}) = \frac{P(r_{t+1} \mid ocu) \cdot P(ocu \mid \{r\}_t)}{P(r_{t+1} \mid ocu) \cdot P(ocu \mid \{r\}_t) + P(r_{t+1} \mid vac) \cdot P(vac \mid \{r\}_t)} \quad (4.7)$$

El problema de esta ecuación son las probabilidades condicionales $P(r_{t+1} \mid ocu)$ y $P(r_{t+1} \mid vac)$ de la ecuación (4.7). Éstas son calculadas a partir del modelo del sensor. Una definición formal de estas probabilidades no es nada simple y en general dependen del estado completo del entorno y no pueden calcularse independientemente para cada celda. En [Elfes 90; Elfes 92] se indica una expresión para su cálculo a partir del modelo del sensor y utilizando el teorema de Kolmogoroff:

$$p(r \mid S(C_i) = ocu) = \sum_{\{G_s(C_i)\}} p(r \mid S(C_i) = ocu, G_s(C_i)) \cdot P(G_s(C_i) \mid S(C_i) = ocu) \quad (4.8)$$

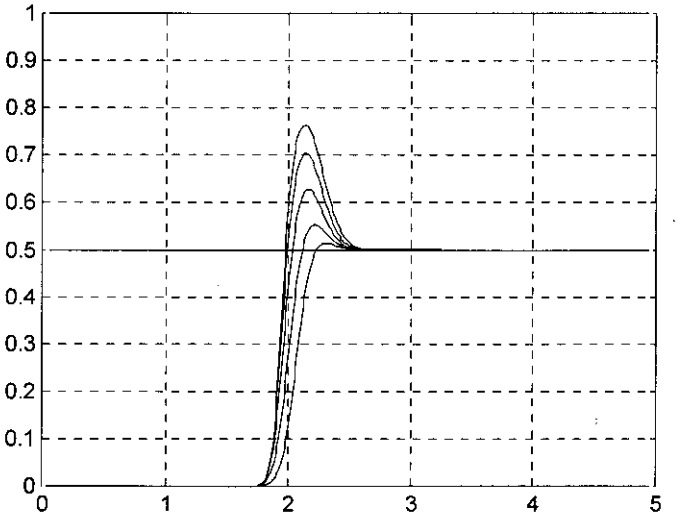
donde $G_s(C_i) = \{S(C_i) = s_1, ..., S(C_i) = s_{i-1}, S(C_i) = s_{i+1}, ..., S(C_i) = s_n\}$ representa una de las configuraciones posibles de todas las celdas del enrejado excepto la C_i.

El cálculo de la ecuación (4.8), implica la suposición de un modelo de medida del sensor, $p(r \mid z)$, y el cálculo de la probabilidad $P(G_s(C_i) \mid S(C_i) = ocu)$, que se realiza a partir de las probabilidades de estado a priori de las celdas y de medidas experimentales de las áreas de interés o de otras consideraciones sobre los estados posibles de las celdas. Esto lleva a la necesidad de que el sistema de fusión posea un modelo del sensor utilizado, lo que no permite la versatilidad y abstracción del sensor ni la posibilidad de construir sistemas generales. Además, toda suposición sobre el entorno no debería hacerla el sistema de fusión, sino los sensores (si tienen posibilidad de hacerlo) para evitar la dependencia de los sensores por parte del sistema de fusión, como se vio en el capítulo 2.

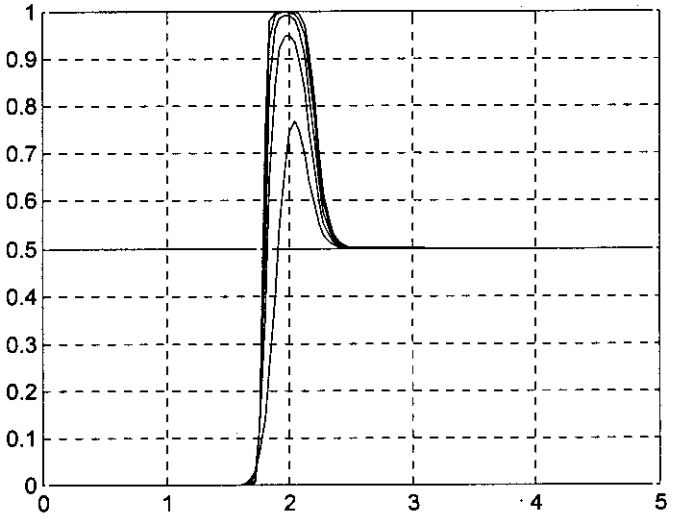
Utilizando la ecuación (4.8) para calcular la estima del sensor y la ecuación (4.7) para la fusión de las distintas medidas, se puede mostrar cómo aumenta la creencia de la ocupación de la celda al repetir la detección de dicha celda. Para ello, y como ejemplo ilustrativo, supondremos un mapa de ocupación de una dimensión y un modelo gaussiano para la detección del sensor, como el expresado en la siguiente ecuación:

$$p(r \mid z) = \frac{1}{\sigma\sqrt{2\pi}} e^{\left[\frac{-(r-z)^2}{2\sigma^2}\right]} \tag{4.9}$$

Con este modelo, supongamos un objeto situado a una distancia de 2 metros (z=2) y una varianza para el modelo de medida del sensor de σ=0.12. El valor de $P(S(C_i) = ocu \mid \{r\}_{t+1}))$ para t=0,1,...,5 es el mostrado en la figura 4-4.



a) Tamaño de celda de 5 cm.



b) Tamaño de celda de 20 cm.

Figura 4-4. Probabilidad de ocupación mediante el método de Elfes.

La figura 4-4a muestra la fusión de 5 medidas iguales sobre un enrejado de 5 cm., mientras que en la figura 4-4b el enrejado es de 20 cm. Puede observarse que la precisión escogida para la rejilla da lugar a una certidumbre que coincide con el ancho de celda elegido. La certeza en la estima se debe al número de celdas existentes y al ancho del modelo del sensor, por lo que cuanto más ancha sea la celda mayor será la probabilidad de ocupación (para un mismo modelo de sensor).

Veamos ahora qué resultados se obtendrían con una red de Bayes:

Como en el ejemplo sólo se ha utilizado un sensor la red de Bayes comparable al ejemplo anterior es muy sencilla, véase figura 4-5. Esta red sólo dispone de un nodo que representa el atributo considerado (Ocupado o Vacío) y el efecto que la medida (o estima) del sensor produce sobre esa celda.

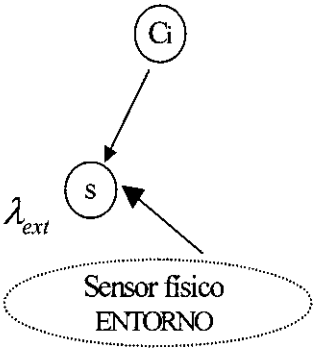


Figura 4-5. Red bayesiana sencilla para el Mapa de ocupación.

A continuación podemos desarrollar las ecuaciones (A.13) y (A.14), que para el caso estudiado quedan unas expresiones muy sencillas, con el fin de estudiar su semejanza con la ecuación (4.7) de Elfes,

$$P(S(C_i) = ocu \mid +s) = \alpha \cdot \pi(S(C_i) = ocu) \cdot \lambda(S(C_i) = ocu) \tag{4.10}$$

$$\begin{aligned} \lambda(S(C_i) = ocu) &= \lambda_s(S(C_i) = ocu) = \\ &= \lambda_{ext}(+) \cdot \lambda_{ant}(+s) \cdot P(+s \mid S(C_i) = ocu) + \\ &+ \lambda_{ext}(-) \cdot \lambda_{ant}(\neg s) \cdot P(\neg s \mid S(C_i) = ocu) \end{aligned} \tag{4.11}$$

Y tenemos,

$$\pi(S(C_i) = ocu) = P(S(C_i) = ocu) \tag{4.12}$$

$$\lambda(S(C_i) = ocu) = p(+s_{t+1} \mid S(C_i) = ocu) \tag{4.13}$$

$$\begin{aligned} \lambda(S(C_i) = ocu) = \lambda(+s_{t+1}) \cdot \lambda(+s_t) \cdot P(+s_{t+1} \mid S(C_i) = ocu) + \\ + \lambda(\neg s_{t+1}) \cdot \lambda(\neg s_t) \cdot P(\neg s_{t+1} \mid S(C_i) = ocu) \end{aligned} \tag{4.14}$$

donde $\lambda(s_{t+1})$ es la estima actual de ocupación del sensor para esa celda, $\lambda(s_t)$ la estima anterior del sensor y $P(s_{t+1} \mid S(C_i) = ocu)$ sería equivalente a la fiabilidad del sensor, es decir, la probabilidad de detectar un objeto por parte del sensor cuando se sabe que la celda está ocupada.

Como se observar, para que las ecuaciones (4.7) y (4.10) sean semejantes (teniendo en cuenta las ecuaciones 4.10 y 4.14), la estima del sensor $P(r_{t+1} \mid ocu)$ y $P(r_{t+1} \mid vac)$ deben ser equivalentes a $\lambda(+s_{t+1})$ y $\lambda(\neg s_{t+1})$. Además, la matriz de relación entre el sensor y la celda tiene que ser considerada la matriz identidad: $P(s_t \mid S(C_i)) = I$.

Así, se obtiene el resultado de la figura 4-6 que, como podemos ver, es exactamente el mismo que se obtenía con Elfes.

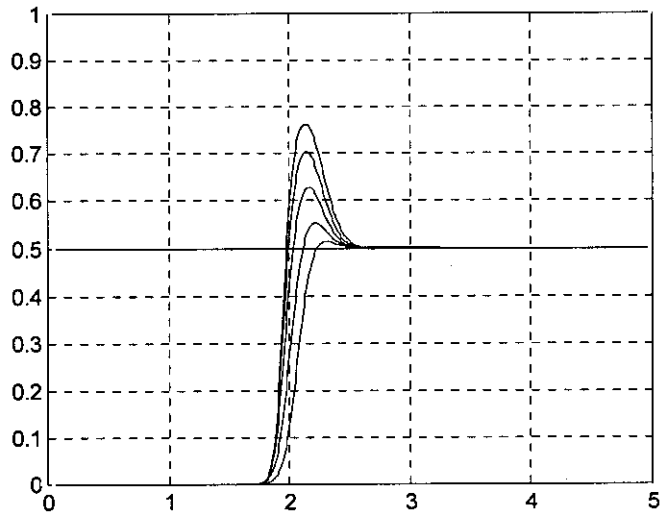


Figura 4-6. Probabilidad de ocupación a posteriori mediante redes bayesianas.

La forma en la que se calcula $\lambda(+s_{t+1})$ y $\lambda(\neg s_{t+1})$, por parte del sensor, permite ajustar el funcionamiento del sistema de fusión de entorno al comportamiento de las distintas aproximaciones del método de Elfes existentes. Por ejemplo, si se calcula por cada medida del sensor un conjunto de $\lambda(s_{t+1})$ correspondientes a todas las celdas afectadas según el modelo de sensor utilizado, obtendremos un mapa de ocupación como el obtenido en el mapa de ocupación original de Elfes. Sin embargo, si el modelo no incluye todas las celdas del mapa sino unas determinadas, se puede adaptar el método de fusión para que actúe de la forma que más interese al usuario y a la aplicación a la que está dirigida.

4.4.2. Comparación de las redes bayesianas con el Histograma de campo

En este caso no se utiliza un modelo probabilístico del sensor sino que se asigna un valor determinado a ciertas celdas por cada medida del sensor, normalmente de ultrasonidos, y se considera una celda ocupada cuando la cuenta (debido a reiteradas medidas) alcanza un valor determinado. Por ejemplo en la figura 4-7 se muestra la cantidad a añadir a la cuenta almacenada en cada celda cuando se observa un objeto con un sensor de ultrasonidos. En este caso la actualización de las celdas es mucho más rápida y se utiliza principalmente para

detectar y evitar obstáculos con rapidez [Borenstein 91a; Borenstein 91b]. Además en este caso es necesario aplicar un factor de olvido a las celdas para eliminar medidas espurias.

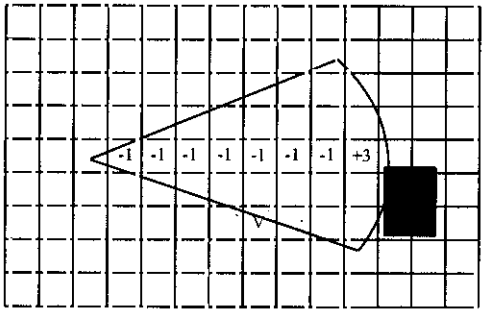
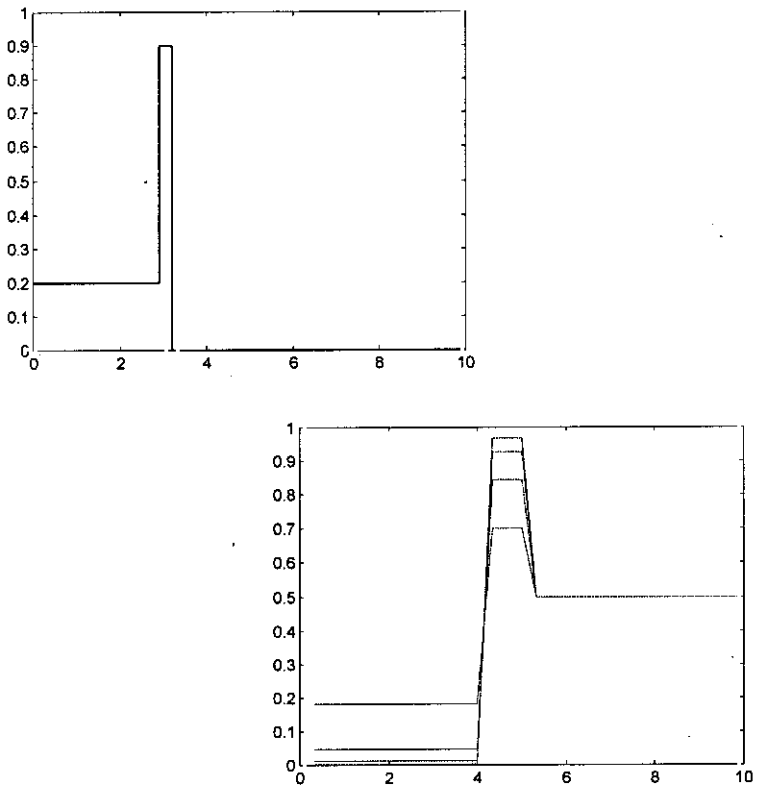


Figura 4-7. Valores asignados a las celdas por medida en el Histograma de campo.

Para emular un funcionamiento semejante al histograma de campo con la red bayesiana diseñada, únicamente es necesario utilizar una única $\lambda(s_{t+1})$ (o un pequeño conjunto de lamdas en el ejemplo de la figura 4-7) para cada medida del sensor, lo que permite actualizar una única celda, o un conjunto reducido de celdas. Por ejemplo supongamos un sencillo modelo sensorial en el que se supone que cuando se detecta un objeto las celdas de delante no están ocupadas, figura 4-8a. En este caso, la red bayesiana de la figura 4-5 producirá un resultado como el de la figura 4-8b, donde la probabilidad de que estén ocupadas las celdas anteriores al objeto disminuye y la del objeto aumenta.



a) Modelo de sensor. b) Fusión de medidas

Figura 4-8 . Fusión de celdas observadas.

4.4.3. Modelo de los sensores en redes bayesianas

Las situaciones de las secciones 4.4.1 y 4.4.2 son casos extremos: desde el cálculo de la probabilidad de todas las celdas del mapa en cada medida (modelo) del sensor hasta la actualización de una única celda (o un conjunto mínimo de celdas).

En estos casos se han utilizado las redes bayesianas con un modelo del sensor que corresponde a los modelos de medida del sensor de Elfes o de Borenstein y no se aprovechan la versatilidad ni las ventajas que nos ofrece la utilización de redes bayesianas. En los casos anteriores, y sobretodo en el primer caso, el sensor ofrece una estima de la celda que tiene en cuenta la situación del enrejado, el modelo del sensor y su fiabilidad, lo que deja únicamente al sistema sensorial la función de acumulador de estimas. Esto es delegar demasiada responsabilidad en la estima que ofrece el sensor físico, complicando excesivamente los cálculos que puede realizar este. Además, no todos los sensores pueden actuar de igual manera, por lo

que el método puede variar de un tipo de sensor a otro. Por lo tanto, sería mejor descargar parte de la responsabilidad de la estima del sensor en el sistema de fusión.

Para descomponer el trabajo de fusión entre el sensor y el sistema de fusión, el modelo del sensor se divide en dos partes: una correspondiente a la medida sensorial (estima del sensor) y la otra correspondiente a la confianza que se tiene de esa medida, que equivale a un peso en la integración de las medidas. Esto es equivalente al modelo presentado en [Elfes 92], ecuaciones (4.15) y (4.16), que como se puede ver es más similar al modelo considerado en las redes bayesianas.

$$p(r \mid \exists T(z)) = p(r \mid \det(T,z)) \cdot P_d(z) + p(r \mid \neg \det(T,z)) \cdot (1 - P_d(z)) \quad (4.15)$$

$$p(r \mid \nexists T(z)) = p(r \mid \det(T,z)) \cdot P_f(z) + p(r \mid \neg \det(T,z)) \cdot (1 - P_f(z)) \quad (4.16)$$

Donde $p(r \mid \det(T,z))$ es el modelo de medida del sensor; *det(T,z)* representa la detección de la característica T en la posición z; $P_d(z)$ es el modelo del comportamiento del sensor, que modela la probabilidad de detección de un objeto por parte del sensor cuando el objeto existe en z; y $P_f(z)$ es la probabilidad de detección de un objeto por parte del sensor cuando el objeto no está en z (probabilidad de falsa alarma).

Esta forma de modelar el comportamiento del sensor es mucho más intuitiva y es la que se utiliza en la red de Bayes, donde se puede observar una correspondencia de los términos de la ecuación (4.15) con los términos de la ecuación (4.14) (lo mismo ocurre con la probabilidad condicionada a la no existencia de características). Y podemos encontrar las siguientes equivalencias:

$$\lambda(+s_{t+1}) = p(r \mid \det(T,z)) \qquad \lambda(\neg s_{t+1}) = p(r \mid \neg \det(T,z)) \quad (4.17)$$

$$P(s_{t+1} \mid S(C_i) = ocu) = \begin{pmatrix} P_d(z) & (1 - P_d(z)) \\ (1 - P_f(z)) & P_f(z) \end{pmatrix} \quad (4.18)$$

Estas matrices son equivalentes al modelo de cada uno de los sensores y representan el peso que da cada sensor a sus estimas en el sistema de fusión global.

Una vez que la información sensorial es convertida en probabilidad (correspondiente a la presencia de algún atributo en la celda) [Elfes 87], es necesario el uso de técnicas de estimación bayesiana para fusiónar información de múltiples sensores. Por ejemplo, Elfes [Elfes 90; Elfes 92] además de integrar

distintas medidas correspondientes a un mismo sensor, extendió su uso a la integración de múltiples sensores.

En nuestro sistema, *la fusión de medidas de entorno de los sensores se realizará sobre un mapa de ocupación mediante redes de Bayes*. El uso de estas redes permite, además, la utilización, la adición o supresión, la configuración y la expresión de dependencias de distintos sensores de forma sencilla, gráfica y fácilmente comprensible. Presenta claramente los atributos que puede estimar y permite modificar su actuación cuando se desee sin necesidad de reconfigurar el sistema completo.

Los atributos considerados en cada celda dependen de la aplicación del robot. En el caso de robot móviles terrestres, los más importantes son el estado de una celda (ocupada o vacía) y el reconocimiento de marcas (naturales o artificiales) del terreno, como pueden ser rincones, esquinas, paredes, puertas, etc., que pueden aprovecharse en la navegación.

Por último indicar que para que el sistema funcione con un modelo de sensor semejante a los histogramas de campo es necesario dotar al sistema de fusión de la posibilidad de utilizar un factor de olvido. Esto también puede incluirse y utilizarse de forma sencilla en las redes bayesianas como se mostrará en la sección 4.5.3.

4.5. Red bayesiana de fusión multisensorial en nuestro robot

Una vez se ha mostrado el método para propagar evidencias en la red, cómo representar el conocimiento que se posee del sistema y su funcionamiento como sistema de fusión frente a otros métodos clásicos, en esta sección se describe su funcionamiento como sistema de fusión en un robot móvil. Para ello veamos primero cómo se ha diseñado el sistema general de fusión de entorno para un robot móvil autónomo terrestre. A continuación se muestra cómo se diseña el sistema de fusión, cómo definir el modelo sensorial desde el punto de vista de las redes bayesianas y su funcionamiento en distintos ejemplos.

4.5.1. Modelo de fusión sensorial mediante redes bayesianas

El principal propósito de nuestro sistema de fusión es que sea general. Por general se entiende que será válido para cualquier sensor que se utilice para medir las características consideradas del entorno del robot. Para su correcto funcionamiento es necesario que se sigan ciertos requisitos: diseñar el modelo causa-efecto de la red bayesiana o en su defecto del sensor correspondiente; y, diseñar el interfaz necesario entre la información del sensor y la utilizada por la red para llevar a cabo el proceso de fusión.

Si estos requisitos se cumplen correctamente, el sistema de fusión puede ser utilizado con cualquier tipo de sensor que sea capaz de estimar las características de interés de los objetos que sean observados. Tengamos en cuenta que cada sensor no tiene por qué observar todas las características contempladas en el sistema de fusión, sino sólo aquellas para las que esté capacitado en cada momento, llegando incluso a observar espacios complementarios (en este caso no existirá fusión, sino integración).

La arquitectura para implementar este sistema de fusión será una red de procesos distribuidos, donde cada proceso representa una etapa de fusión o nodo en la red bayesiana. Estos nodos los clasificamos en:

- *Nodo sensor*, véase la figura 4-9, que corresponde al nodo hoja de la red bayesiana. Se encarga de recibir evidencia externa procedente de la sensorización, es decir, se ocupa de convertir la información sensorial del sensor físico en información comprensible por la red. Realiza una fusión de la información externa recibida y de la

procedente de nodos de fusión superiores en un mapa local. También se encarga de interpretar las órdenes que le son enviadas para realizar una determinada función o de enviárselas al sensor físico. Este nodo sensor corresponde a la clase *SensorLógico* definida en el capítulo 2.

- *Nodo fusión*, que corresponde a nodos que no son hoja en la red. Se encarga de realizar la fusión de las características de distintos sensores. Al contrario que en los nodos sensor que siempre son uno por sensor existente (o unidad sensorial, es decir, el conjunto de transductores de ultrasonidos se puede considerar un único sensor), los nodos fusión pueden ser tantos como se necesite en el modelo causa-efecto diseñado por el usuario, véase figura 4-9. Estos nodos de la red corresponden a la clase FusiónEntorno mostrada en el capítulo 2.

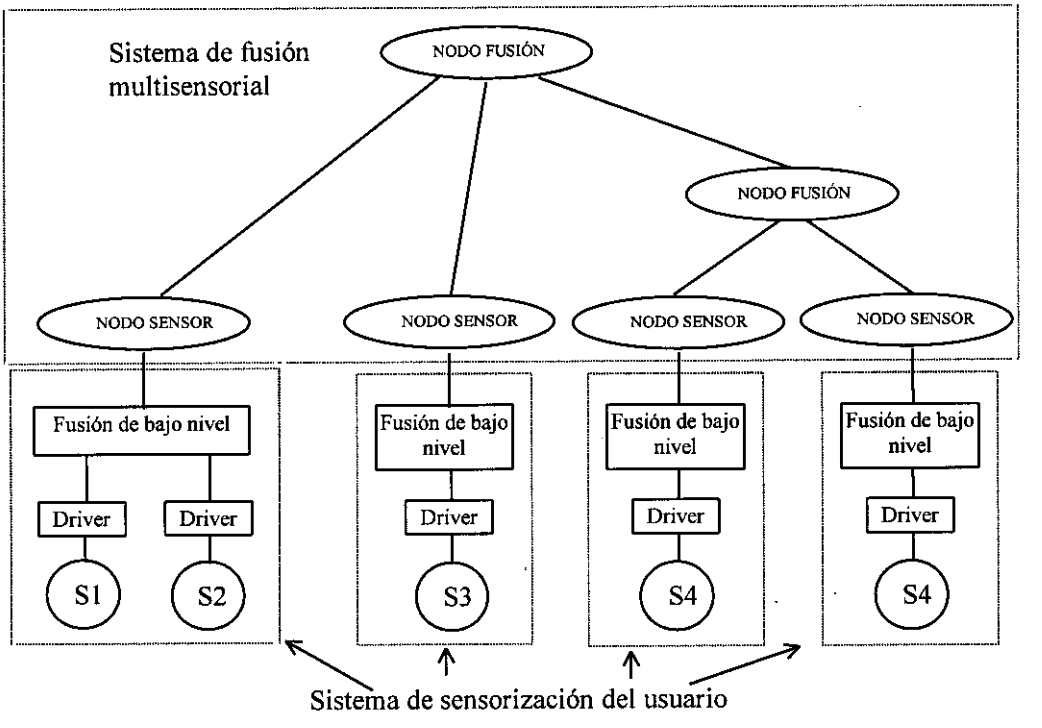


Figura 4-9. Sistema de fusión de entorno.

En la figura 4-9, además del sistema de fusión formado por los nodos sensor y nodos fusión, se muestra la parte sensorial correspondiente al usuario, como son:

- *Sensor físico* (S_i), que corresponde a los sensores reales. Los datos que obtienen son en bruto y deben ser procesados e interpretados.
- *Drives*, que son las rutinas que permiten leer y manejar los datos que se obtienen directamente de los sensores. Incluyen procesamiento A/D de señales, almacenamiento temporal de los datos, etc.
- *Fusión de bajo nivel*, que corresponde al tratamiento de los datos recibidos a fin de obtener las características buscadas. En este nivel se encuentra el modelo físico del sensor y reduce el número de medidas observadas a un conjunto de estimas válidas, realiza la asociación de datos, actualiza observaciones, etc. Este nodo corresponde al método de fusión o interpretación de los datos que el usuario utiliza sin intervención del sistema de fusión y está estrechamente relacionado con el nodo sensor o sensor lógico, que se mostró en el Capítulo 2.

4.5.2. Sistema de fusión para un robot móvil

Una red para un robot autónomo terrestre puede, por ejemplo, diseñarse en la forma que se muestra en la figura 4-10.

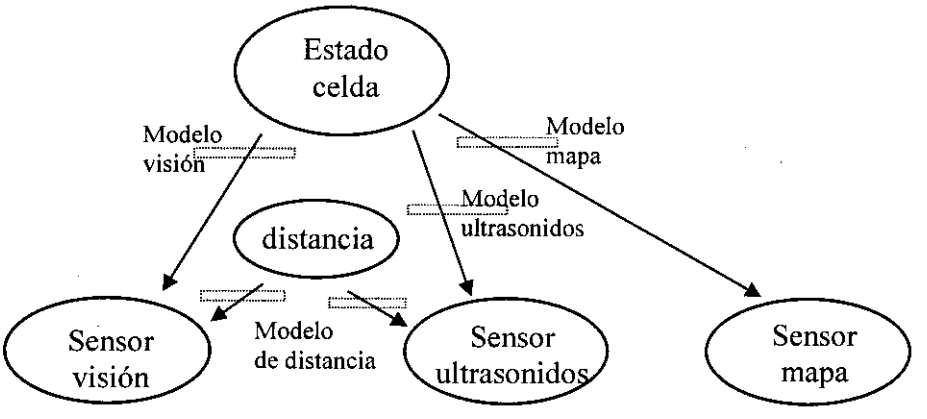


Figura 4-10. Red de Bayes completa.

Este diseño posee un ciclo (Estado celda - Sensor visión – distancia – Sensor ultrasonidos). Antes de aplicar alguna de las técnicas de eliminación de ciclos, se debe simplificar aprovechando cualquier determinismo en las variables observadas. Por ejemplo, la distancia que es determinista, puesto que es conocida para cada celda (la posición del vehículo se conoce con bastante precisión), puede ser eliminada como variable aleatoria e introducirse en el modelo del sensor

correspondiente. Esto es, la distancia puede codificarse como una función que irá incluida en el modelo de cada uno de los sensores.

Ahora queda una red bayesiana mucho más simple, figura 4-11, que puede implementarse directamente sin la necesidad de aplicar técnicas de eliminación de ciclos.

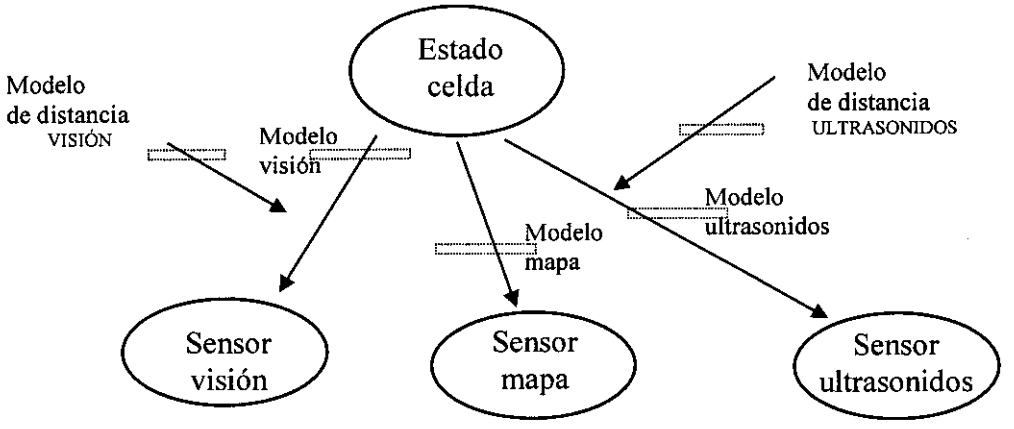


Figura 4-11. Red de Bayes simplificada.

Para simplificar las expresiones que se mostrarán a continuación, se utiliza un único atributo para el nodo celda, que esté o no ocupada la celda, dejando el tipo de celda para posteriores versiones más complejas, aunque el mecanismo es semejante al descrito a continuación.

Para realizar la fusión de datos de visión y ultrasonidos es necesario, por tanto, definir la probabilidad de acierto en función de la distancia , r, y del ángulo que forma el objeto con el sensor, θ, para cada uno de los sensores que se estén utilizando. Esto es:

$$\begin{aligned} P(+u|+e) &= f_u(r,\theta) \\ P(-u|-e) &= f_u(r,\theta) \end{aligned} \tag{4.19}$$

donde u representa el sensor de ultrasonidos y e el estado de la celda que se está estudiando. Los signos (+ y -) representan el caso afirmativo (sensar objeto u ocupación de la celda) y el caso negativo (no se sensa ningún objeto o no está ocupada la celda).

Para el caso de visión sería lo mismo:

$$\begin{aligned} P(+v|+e) &= f_v(r,\theta) \\ P(-v|-e) &= f_v(r,\theta) \end{aligned} \tag{4.20}$$

Con estas funciones se pueden construir el modelo del sensor de ultrasonidos, ecuación (4.21), y el del sensor de visión, ecuación (4.22), respectivamente que nos dé las matrices que relacionan las celdas con respecto a lo que observan los sensores,

$$M(ocupado|Sens\_ultras) = \begin{pmatrix} P(+x|+u) & P(-x|+u) \\ P(+x|-u) & P(-x|-u) \end{pmatrix} \tag{4.21}$$

$$M(ocupado \mid Sens\_visión) = \begin{pmatrix} P(+x \mid +v) & P(-x \mid +v) \\ P(+x \mid -v) & P(-x \mid -v) \end{pmatrix} \tag{4.22}$$

En cualquier caso se deberán hacer estudios sobre el funcionamiento de cada sensor y del conjunto para sintonizar correctamente la red.

Recordemos que las probabilidades de las ecuaciones (4.21) y (4.22) se corresponden con las probabilidades de acierto y de fallo (errores de tipo I y tipo II) que se mostraron en la tabla 4.1

A continuación veamos el funcionamiento de la red para un ejemplo concreto. Utilizaremos dos sensores de entorno: ultrasonidos y visión.

Lo primero es especificar el modelo de los sensores, es decir, las funciones que corresponden a los modelos de probabilidades de los ultrasonidos y visión en función de la distancia a la que se encuentra un objeto. O lo que es lo mismo, debemos encontrar mediante pruebas experimentales las ecuaciones (4.19) y (4.20).

Como primera aproximación se han utilizado los modelos probabilísticos de los sensores que pueden verse en la figura 4-12: en la figura a) se muestra la probabilidad de que un sensor, ya sea de ultrasonidos o visión, detecte un objeto cuando sabemos que el objeto se encuentra allí (probabilidad de acierto), $P(+u|+e)$ o $P(+v|+e)$; y en la figura b) se muestra la probabilidad de que visión, o ultrasonidos, detecte un objeto cuando sabemos que el objeto no está allí (error de primera especie),), $P(+u|-e)$ o $P(+v|-e)$.

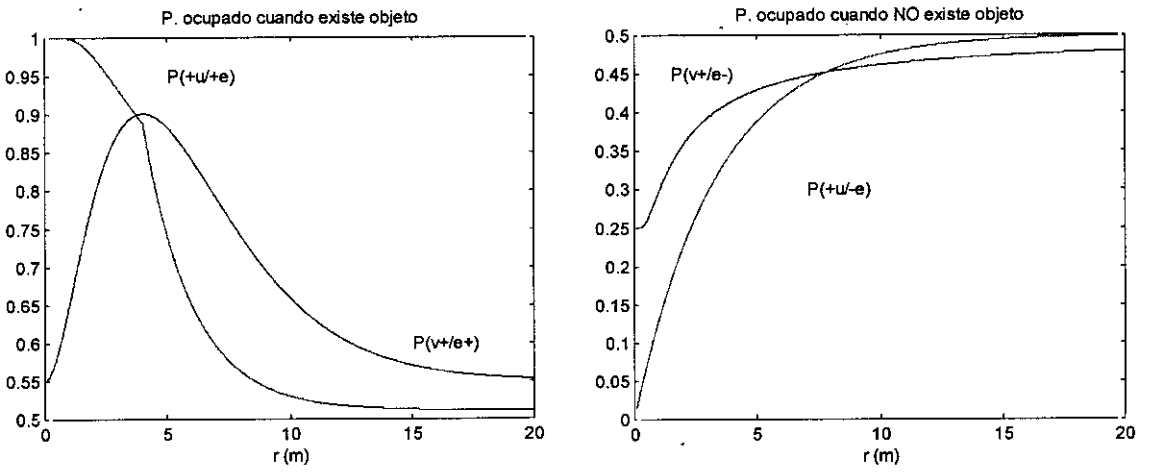


Figura 4-12. Modelo de Ultrasonidos y visión

Estos modelos de los sensores sólo han sido definidos en función de la distancia al objeto (aunque podrían añadirse más dependencias). Como puede verse, para distancias cortas el mejor sensor (más fiable) es el sistema de ultrasonidos y para distancias largas el de visión. Cuando la distancia aumenta la probabilidad tiende a 0'5, es decir desconocido.

Estas funciones dependen de la distancia del objeto y son propias de cada sensor. Pero siempre se debe tener en cuenta que $P(+u|+e) > P(+u|-e)$; y lo mismo ocurre para visión, como puede verse en la figura 4-12. Se observa también que para distancias cortas el sensor de ultrasonidos es mejor que el de visión, pero a partir de aproximadamente los 5 metros se invierten los términos.

Normalmente lo que se busca en un sensor es que la diferencia entre $P(+u|+e)$ (sensibilidad) y $P(+u|-e)$ (especificidad) sea la mayor posible, es decir, que aumente la probabilidad de acertar y disminuya la de cometer un falso negativo (error de primera especie). Sin embargo cuando se intenta aumentar una, aumenta también la otra con lo que debemos llegar a un cierto compromiso. En la figura 4-12 se muestra la especificidad y la sensibilidad de los sensores de ultrasonidos y de visión considerados.

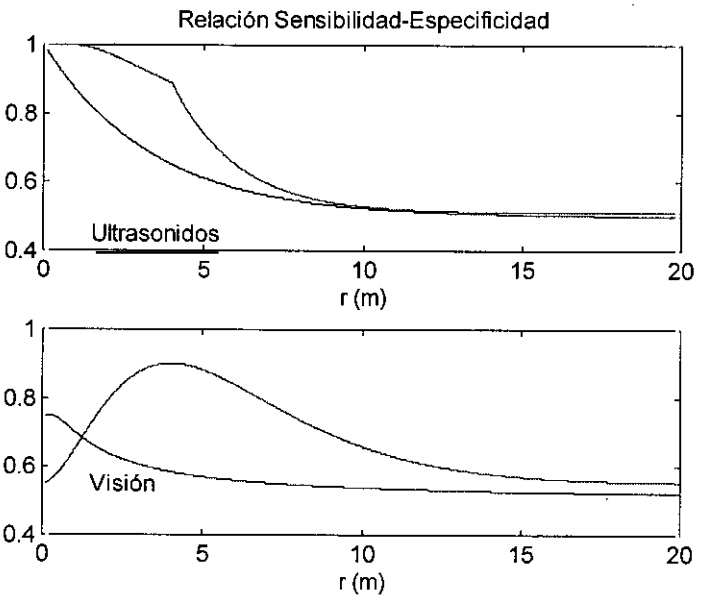


Figura 4-13 . Sensibilidad y especificidad de los sensores utilizados.

4.5.3. Ejemplo del comportamiento del sistema de fusión

Veamos el funcionamiento de la red de Bayes y cómo se realiza la fusión de las medidas de sensores de entorno. Para ello se utilizan los sensores de la sección anterior. Se pueden distinguir varios casos como son:

- Caso I: Fusión de medidas de un único sensor. Permite comprobar que el sistema es capaz de evolucionar aunque sólo funcione un único sensor (robusto frente a fallos). Además muestra la credibilidad de cada sensor dependiendo de la distancia al objeto para mostrar el efecto de la sensibilidad y especificidad definidas.
- Caso II: Fusión de la estima de los dos sensores. En este caso se muestra como al fusionar la estima de dos sensores se obtiene un resultado más preciso.
- Caso III: Fusión de medidas contradictorias. Indica que la fusión de los sensores es una medida similar a la ponderación de las medidas individuales.
- Caso IV: Fusión de medidas en el tiempo. Permite observar que la estima final aumenta su certeza a medida que se repite la observación. Pero esta estima no aumenta hasta llegar a 1, certeza absoluta, sino que tiende a un valor límite que viene dado por la matriz del sensor correspondiente.

Caso I: Fusión de medidas sensoriales procedentes de un único sensor

En primer lugar se prueba el sistema con un único sensor actuando cada vez. El otro sensor puede suponerse desactivado, estropeado o que simplemente no existe.

La probabilidad de la estima del atributo en cuestión (únicamente se considera si la celda está ocupada o vacía) obtenida a partir de la información medida por cada uno de los sensores será la misma para poder comparar el comportamiento de cada uno de ellos. En un caso los ultrasonidos estiman que la celda se encuentra ocupada con una probabilidad de acierto de 0'7 y que se encuentra vacía con una probabilidad de 0'2 mientras que la visión no obtiene ninguna información del entorno. Por otro lado, en otro caso distinto, se procede al contrario, los ultrasonidos no dan ninguna información y el sensor de visión es el que estima la certeza de ocupado y vacío de cada celda en la misma proporción que ocurría con los ultrasonidos en el primer caso (0'7 y 0'2 respectivamente).

En la figura 4-14, se presenta la probabilidad a posteriori de que la celda se encuentre ocupada (la probabilidad a priori es 0'5) a partir de la información de los ultrasonidos. En cada uno de los casos se han utilizado las celdas a la misma distancia (10 celdas ocupadas a distancias R=[0.3 0.5 1 1.5 2 3 4 5 7 10]) para mostrar cómo la distancia introducida en el modelo condiciona la estima final de cada uno de los sensores y cómo el modelo que ofrezca más fiabilidad dará más confianza a la información del sensor. Es decir, se da más peso al sensor que sea más fiable en cada situación.

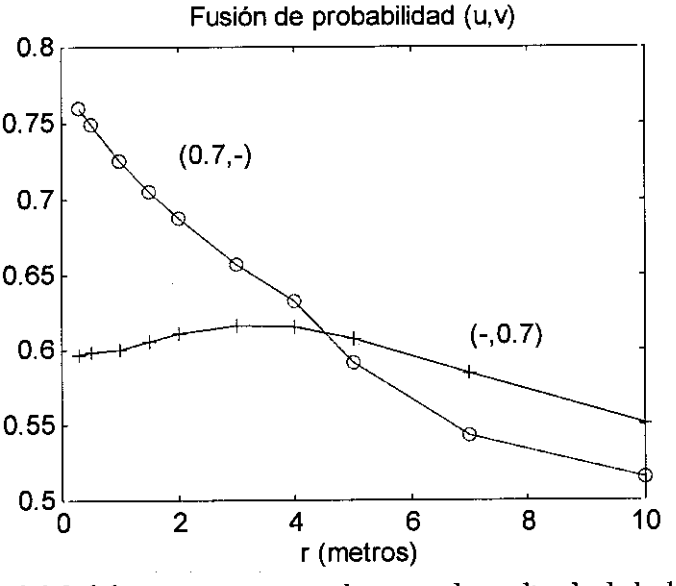


Figura 4-14. Probabilidad de ocupación para cada sensor dependiendo de la distancia de la celda.

(o ultrasonidos, + visión)

En la figura 4-14 se indica la creencia positiva de cada sensor en un par de elementos, siendo el primer elemento del par la creencia del sensor de ultrasonidos y el segundo la creencia del sensor de visión. Por ejemplo, (-,0.7) representa que el sensor de ultrasonidos no funciona o no realiza ninguna afirmación y el de visión indica la existencia de un objeto con una probabilidad del 0.7.

Caso II: Fusión de medidas de ambos sensores

En este ejemplo se pretende poner de manifiesto el funcionamiento correcto del sistema de fusión. Para ello utilizaremos el sensor de ultrasonidos y el de fusión informando simultáneamente sobre la misma celda. La estima de la probabilidad de ocupado y vacío para cada uno de los sensores es la misma y es igual a la utilizada en caso anterior (0'7 y 0'2).

En la figura 4-15 se muestra (para las mismas celdas que en el caso anterior) la probabilidad a posteriori de que cada celda se encuentre ocupada cuando se realiza la fusión de la estima de ambos sensores junto con la obtenida en caso de que cada uno de los sensores actuase solo.

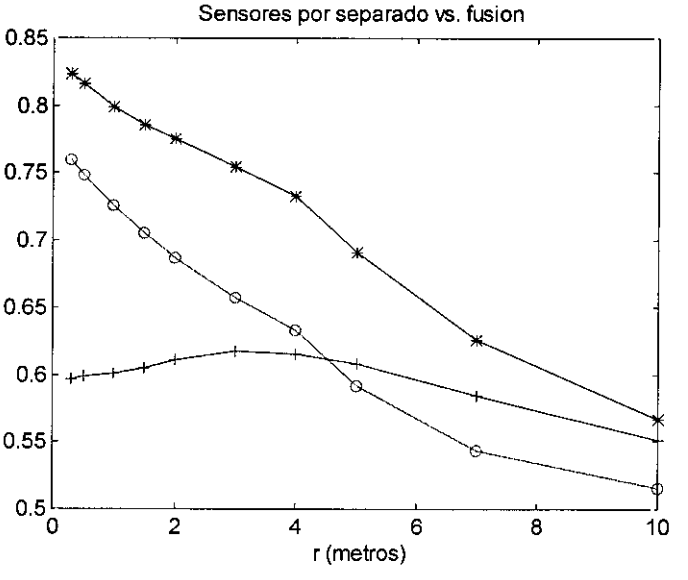


Figura 4-15 . Fusión de certeza en la medida
(o ultrasonidos, + visión, * fusión de ambas)

En la figura 4-15 se puede ver que la fusión de la estima de los dos sensores siempre da lugar a una certeza mayor de que la celda se encuentre ocupada que en el caso de utilizar un único sensor.

Por supuesto, en el caso de que la estima de los sensores sea contraria a la existencia de un objeto en la celda en cuestión, la fusión de estimas de este tipo dará lugar a un reforzamiento de la creencia a posteriori sobre la no ocupación de esa celda. Es decir, obtendremos una mayor certeza de que la celda esté vacía, por lo que la probabilidad a posteriori conjunta será siempre menor que la calculada por los sensores individualmente, como puede verse en la figura 4-16. El aumento de la creencia con la distancia no quiere decir que esté menos seguro de la existencia de un objeto, sino que a medida que aumenta la distancia, la incertidumbre sobre la existencia de un objeto aumenta (por eso tiende a 0.5). Recordemos que se indica la probabilidad de existencia de un objeto señalada por cada sensor con *(ultra,vis)*, y se pone un guión, -, si el sensor no está funcionando.

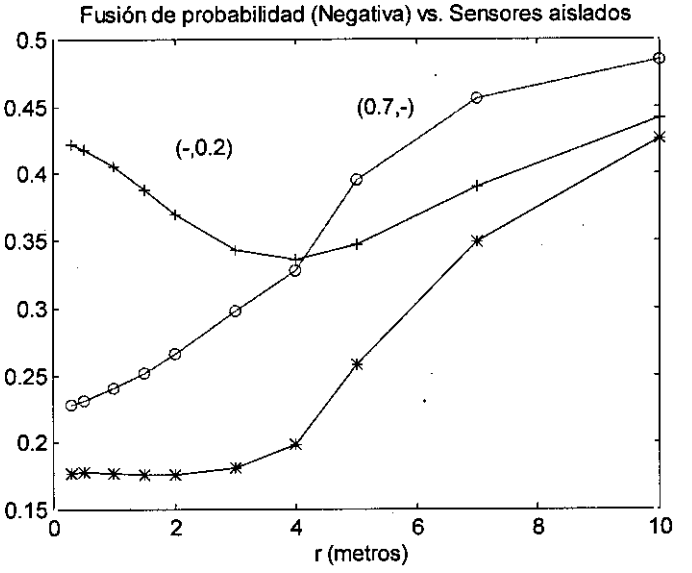


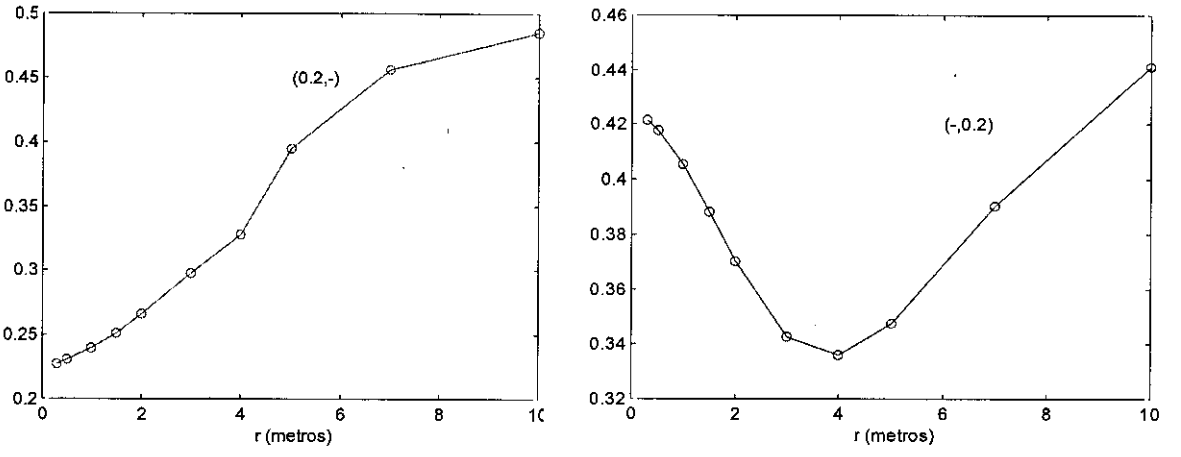
Figura 4-16. Fusión de probabilidad negativa.
(o ultrasonidos, + visión, * fusión de ambas)

Caso III: Fusión de medidas contradictorias

Con el fin de comparar los resultados de este caso con los anteriores se continua utilizando las mismas celdas (distancias entre la celda del mapa y el robot) que en los ejemplos anteriores. En este caso se fusionan datos de los dos sensores, al igual que en el caso anterior, pero las certezas de ambos son contradictorias (0'7

y 0'2 para ultrasonidos y 0'2 y 0'7 para visión) con lo que las opiniones sobre la existencia, o no, de un objeto están enfrentadas.

En el caso de que un sensor informe negativamente sobre la existencia de un objeto, la probabilidad a posteriori tenderá a un valor inferior a la incertidumbre (0'5), y esto dependerá de la estima del sensor y del modelo del sensor para la celda en cuestión. En la figura 4-17 se muestra la probabilidad a posteriori para la fusión de medidas de un único sensor (0'2,0'7).



a) Fusión de medidas de sensores de ultrasonidos b) Fusión de medidas de sensores de visión

Figura 4-17. Probabilidad a posteriori cuando la información es negativa

Cuando se dispone de medidas contradictorias, la probabilidad a posteriori siempre se encuentra entre la que se calcula con cada uno de los sensores aisladamente, pero con una tendencia a uno de ellos dependiendo del peso que tengan. Es decir, la información que predomina es la de aquel sensor cuya fiabilidad es mejor, pero ponderado por la opinión del otro. En la figura 4-18 se muestra un ejemplo de este caso.

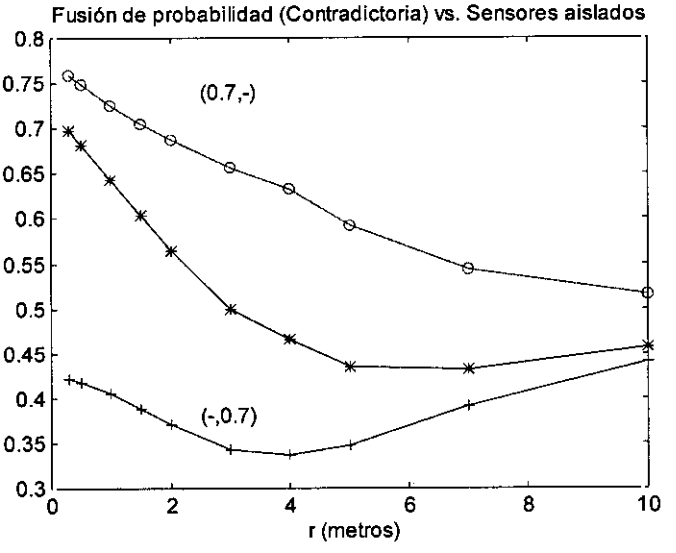
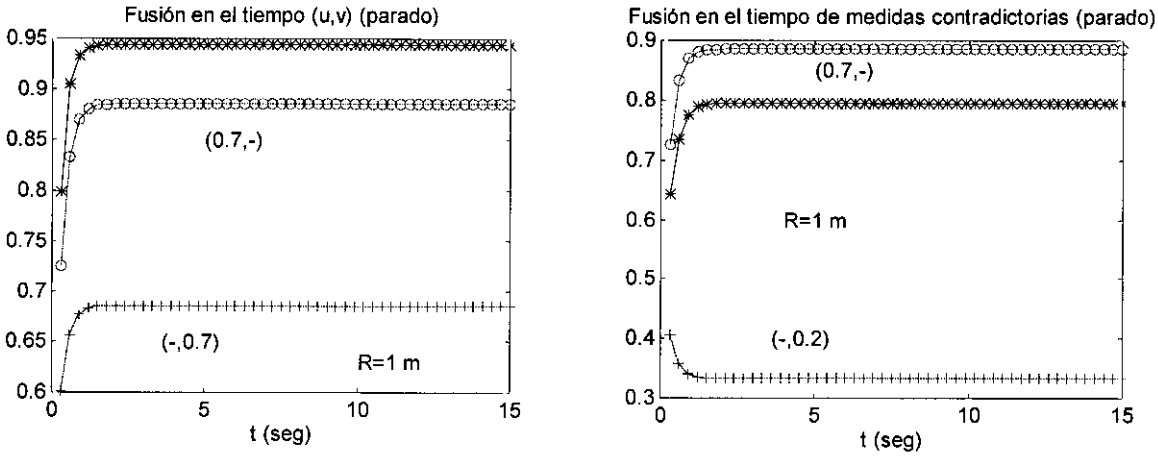


Figura 4-18. Fusión de probabilidad con certeza contradictoria (o ultrasonidos, + visión, * fusión de ambas)

Caso IV: Fusión de medidas a lo largo del tiempo

Se puede mostrar el funcionamiento del sistema de entorno en función del tiempo. Para ello se presenta el caso de la fusión de información sobre una misma celda a lo largo del tiempo. Las estimas de cada uno de los sensores es la misma que en los ejemplos anteriores (0'7 y 0'2) y se repetirá constantemente. Se realizan dos ejemplos: uno el robot se encuentra a una distancia fija de un objeto y otro cuando se aproxima a éste.

En primer lugar, se supone que el robot se encuentra parado, a una distancia de 1 metro y se toman 50 medidas de dicha celda. En la figura 4-19 se muestra la probabilidad a posteriori cuando sólo actúa uno de los sensores y la probabilidad a posteriori conjunta (procedente de la fusión de las estimas de ambos sensores).



a) Medidas en el mismo sentido

b) Medidas contradictorias

Figura 4-19. Fusión de medidas en el tiempo.
(o ultrasonidos, + visión, * fusión de ambas)

En la figura 4-19a se muestran los casos correspondientes a afirmaciones iguales y en la figura 4-19b a afirmaciones contradictorias de los sensores. En la primera situación (como ocurría en el caso II) la probabilidad final siempre es mejor que la de los dos sensores actuando independientemente. En la situación con afirmaciones contradictorias, figura 4-19b, la probabilidad conjunta a posteriori se encontrará siempre entre la de los dos sensores (como ocurría en el caso III). En la figura 4-19ª las dos medidas de los sensores crecen porque se refuerza la creencia a favor de que la celda esté ocupada, mientras que en la figura 4-19b la medida de visión decrece puesto que se refuerza la creencia en contra de la existencia de un objeto. La creencia crece hasta alcanzar un valor determinado (no tiene porqué llegar a 1) que depende del conocimiento a priori y la distancia (modelo del sensor).

En segundo lugar, se puede medir la misma celda de igual forma que en el caso anterior (un sólo sensor en cada ocasión y la fusión de ambos) pero moviéndonos hacia la celda que contiene el objeto observado, figura 4-20. Para ello se supone que se parte de una distancia a la celda de 8 metros y se acerca a ella a una velocidad de 0.3 m/s hasta llegar a una distancia de 2 metros de ella.

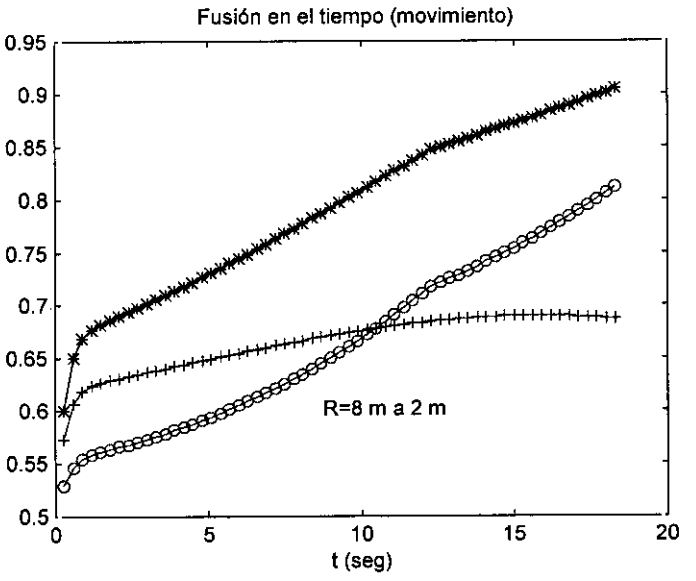


Figura 4-20. Evolución de la probabilidad a posteriori en el tiempo (o ultrasonidos, + visión, * fusión de ambas)

En la figura 4-20 se puede ver que la probabilidad de acierto aumenta hasta alcanzar el máximo en la posición más cercana a la celda. En este caso la creencia obtenida queda por debajo de la de la figura 4-19a porque se encuentra a una distancia de 2m. También se observa que la velocidad con la que aumenta la creencia de cada sensor depende del modelo utilizado.

4.6. Funcionamiento temporal

El diseño de una red como la mostrada en la sección anterior permite la actualización, a partir de la información que cada uno de sus sensores ofrece, de cualquier característica (celda/objeto) del entorno del robot.

Sin embargo, se debe tener en cuenta que en cualquier modo de funcionamiento de los sensores en el que sólo informen de características o atributos de objetos detectados (como en el caso del histograma de campo) y no de todas las celdas observadas por el sensor (como ocurre en mapas de ocupación) el algoritmo de fusión no funciona correctamente. Esto se debe a que en estos casos si el sensor comete un error en la lectura o en su estima (por ejemplo un falso eco) y esa celda no vuelve a ser observada, el atributo estimado permanecerá indefinidamente. Lo mismo ocurrirá con objetos en movimiento, puesto que al ser observado durante su trayecto dejará un rastro que no se eliminará. Para evitar esto, tenemos dos opciones:

- *Información sobre celdas vacías y ocupadas*. Los sensores pueden informar sobre la certeza de todas las celdas observadas (tanto las estimadas ocupadas como las vacías), véase la figura 4-21, de modo que se actualicen continuamente, por ejemplo en [Menezes 95]. Con esta opción se consigue evitar el problema de la detección de objetos en movimiento pero no el de aquellas celdas lejanas que hayan sufrido un falso eco. Además otro inconveniente de este procedimiento es que requiere una mayor carga de cálculo y una sobrecarga en la comunicación entre procesos que no es factible en muchos sistemas debido a sus necesidades de tiempo.

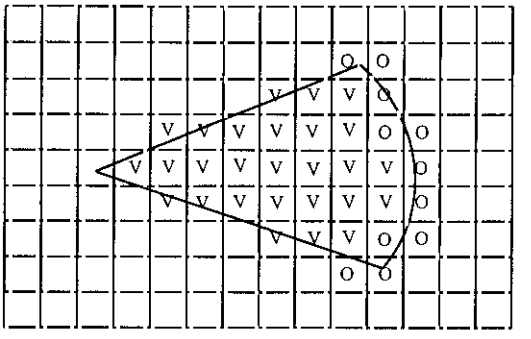


Figura 4-21. Múltiples celdas ocupadas y vacías

- *Factor de olvido*. Se realiza un decremento temporal de la certeza de cada una de las celdas. Esto equivale a dotar de un factor de olvido al robot, de forma que hace que la certeza de una celda tienda a volverse desconocida (0'5, 0'5) cuando no es observada durante cierto tiempo. Este tiempo o *factor de* olvido, τ_0, debe ser sintonizado en función de la aplicación para la que se utilice el robot. Depende de la velocidad de muestreo de los sensores, de la velocidad de los objetos móviles detectados y de la aplicación particular del robot (construcción de mapas, ir de un lugar determinado a otro, etc.).

Como la finalidad del sistema de fusión de medidas de entorno es ser independiente del tipo de sensores utilizados, es conveniente dotar al sistema de la posibilidad de utilizar ambos mecanismos. Por tanto, se adaptará el sistema para permitir la actualización de las celdas con un factor de olvido como el indicado en la segunda opción, aunque el usuario es libre de configurar el sistema para actuar como en el primer caso sin más que disponer de las estimas de las celdas adecuadas y no utilizar el factor de olvido.

Para encontrar el factor de olvido es necesario diseñar un modelo del comportamiento temporal de la creencia de una celda, teniendo incluso en cuenta que no todos los atributos tienen porqué tener un mismo factor de olvido. En general, puede suponerse que el estado actual de una celda (atributo), X_t, estimado por el robot dependerá de su estado anterior, X_{t-1}, y de la acción que tome el robot, A_t.

$$P(X_t \mid X_{t-1}, A_t) \tag{4.23}$$

Para simplificar el modelo, podemos suponer que el robot no interactua con su entorno, sus sensores no afectan a las celdas que observa (no tiene sensores activos que pudiesen modificar el estado de las celdas al tomar medidas) ni su movimiento afecta al estado de una celda observada. Con esta simplificación, la ecuación (4.23) queda,

$$P(X_t \mid X_{t-1}) \tag{4.24}$$

Por tanto, lo que nos interesa es construir una red bayesiana con un nodo por cada estado y variable sensor en cada etapa temporal (véase la figura 4-22). Esto se conoce como red de Bayes dinámica. Las redes de bayes dinámicas pueden verse como una codificación distribuida de un proceso de Markov. Fueron utilizadas

inicialmente por Dean y Nicholson [Dean 89; Nicholson 92; Kjaerulff 92 o Nicholson 94] entre otros.

Por medio de esta red se puede estimar la probabilidad de un estado en el momento actual *t*. Hay que tener en cuenta que, aunque no será utilizado en nuestro sistema, se puede predecir el estado futuro de una celda (lo que se conoce como proyección probabilista), aunque para ello es necesario presuponer unas ciertas entradas sensoriales futuras.

Aunque la predicción puede parecer inútil en este caso, puede llegar a ser una herramienta muy potente. Por ejemplo, mediante una red de creencia dinámica se puede diseñar un filtro de Kalman con sólo una red dinámica con dos periodos de tiempo y tres etapas (predicción, observación y estimación) [Russell 95].

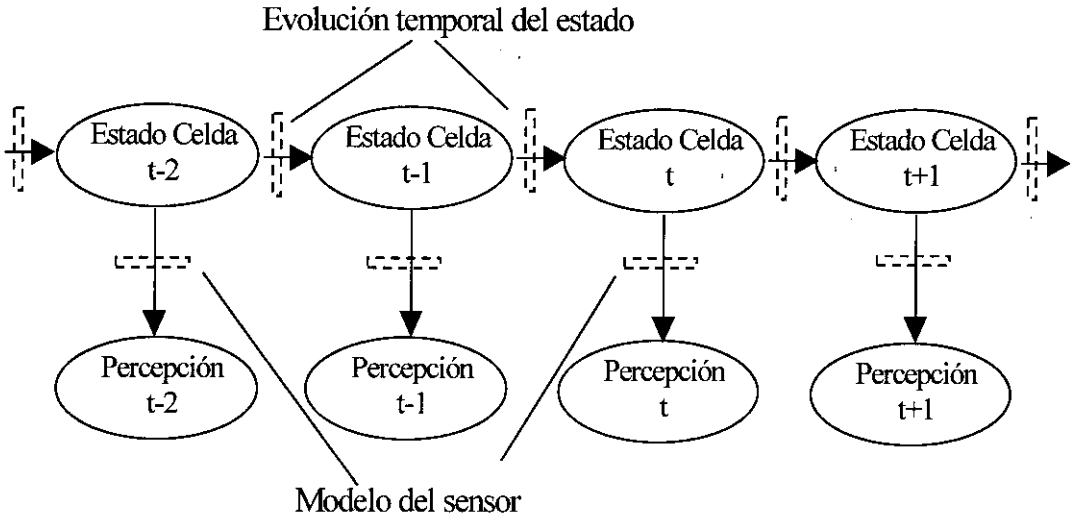


Figura 4-22. Estructura genérica de una red bayesiana dinámica.

Puesto que se considera un modelo para la evolución temporal que depende de instantes anteriores al estado actual, la figura 4-22 es suficiente para representar cualquier sistema sensorial en el tiempo. Sin embargo, puesto que no nos interesa guardar la evolución temporal de la celda ni la secuencia de observaciones realizadas, es suficiente con mantener un modelo con el estado actual y el estado siguiente y realizar tres fases cíclicamente, véase la figura 4-23. Esta fases son semejantes al caso de un filtro de Kalman: predicción, observación y actualización.

- Predicción: En este paso se calcula el estado que tendrá la celda antes de realizar la observación de una medida por parte del sensor y corresponde a la evolución temporal del estado de la celda. Esto equivale a aplicar el factor de olvido sobre las celdas.
- Observación: El sensor adquiere una medida referente a una celda y calcula su estima.
- Actualización: En esta fase se aplica el mecanismo estándar de actualización de la creencia de la red bayesiana y se obtiene el nuevo estado de la celda a partir de la observación realizada. En este momento la red está lista para comenzar un nuevo ciclo.

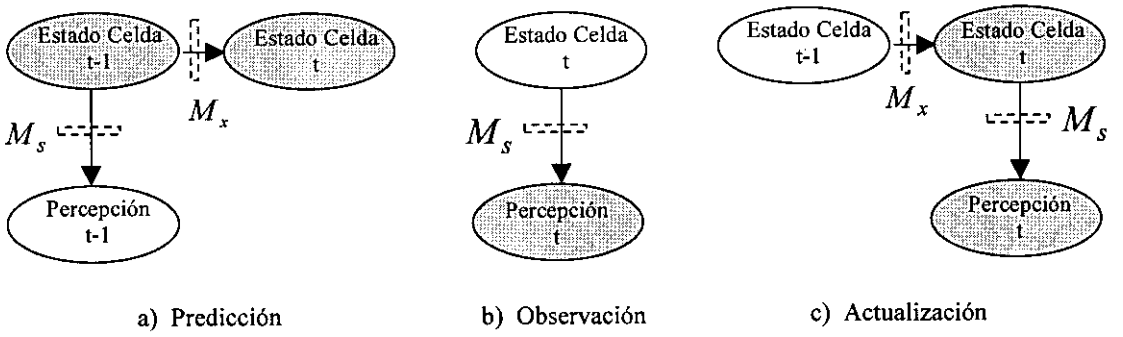


Figura 4-23. Pasos para actualizar nuestra red dinámica.

En la figura 4-23, se muestra cada una de las etapas en el cálculo del estado de la celda en el instante *t*, en ella se muestran sombreados los nodos que se encuentran involucrados en cada una de las fases. En la figura también se muestra el modelos sensorial, *Ms*, y el modelo de transición de estado, *Mx*.

El modelo temporal del estado de las celdas puede ser variable sin más que modificar adecuadamente la matriz de transición de estados, *Mx*. Ésta puede depender de aquellas variables que se considere que afectan al factor de olvido. En nuestro caso consideraremos que el factor de olvido es independiente de la celda en cuestión, por lo que se utilizará una matriz que sólo depende del periodo de refresco,

$$M_x = \begin{pmatrix} e^{-\frac{t}{\tau_0}} & 1-e^{-\frac{t}{\tau_0}} \\ 1-e^{-\frac{t}{\tau_0}} & e^{-\frac{t}{\tau_0}} \end{pmatrix} \tag{4.25}$$

Por supuesto τ_0 debe ser calculado experimentalmente y dependerá del tipo de sensor utilizado, su velocidad de muestreo, el movimiento del robot, de los objetos que le rodean, etc.

Por lo tanto ya tenemos definida la red dinámica y su comportamiento temporal. Una red ejemplo con dos sensores puede verse en la figura 4-24.

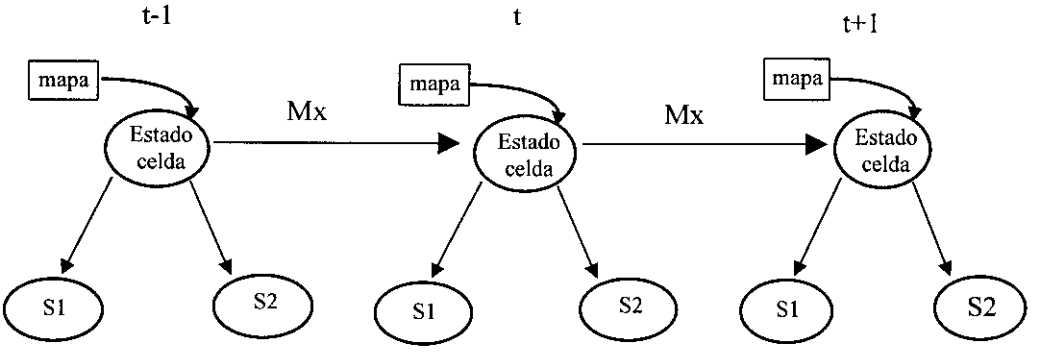


Figura 4-24. Red de fusión temporal.

A continuación se mostrarán varios ejemplos del funcionamiento temporal de dicha red para mostrar la utilización del factor de olvido en distintos casos.

4.6.1. Ejemplos de funcionamiento temporal

A continuación se presentan distintos ejemplos del funcionamiento temporal del sistema de fusión de entorno para mostrar cómo se comporta la creencia de la red bayesiana dinámica y el factor de olvido del sistema. En todos estos ejemplos se utiliza la matriz de transición de estados mostrada en la ecuación (4.25) con un valor para τ_0 de 5.85s., que para un periodo de refresco de 0'3 s da la matriz constante,

$$M_x = \begin{pmatrix} 0.95 & 0.05 \\ 0.05 & 0.95 \end{pmatrix} \tag{4.26}$$

Debe tenerse en cuenta que la red bayesiana dinámica engloba a la anteriormente definida y que se obtiene con sólo dejar de aplicar el factor de olvido, es decir, sin tener en cuenta ningún cambio temporal del estado de los nodos. Esto se consigue dejando la matriz de transición, *Mx*, como la matriz identidad.

La dependencia temporal del sistema de fusión puede ser muy variada dependiendo de su aplicación y del entorno donde se utilice. En estos ejemplos se utiliza la capacidad dinámica de la red para incorporar un factor de olvido. El factor de olvido incluido en nuestra red de bayes dinámica permite dos opciones diferentes:

- *Olvido de la información a priori*, lo que permite que la información a priori se utilice inicialmente pero a medida que se obtiene información externa sólo se utilice la información sensorial.
- *Olvido de la información sensorial*, lo que permite incluir un factor de olvido en la información obtenida por los sensores de igual forma que se utiliza en los histogramas de campo y otros problemas.

4.6.1.2 Factor de olvido en la información a priori

Para mostrar el efecto del factor de olvido sobre la creencia a priori de una celda, se parte de un valor a priori para la creencia de ocupación de la celda de 0'8 (por ejemplo debido a un mapa externo), es decir, inicialmente la celda se cree ocupada. Téngase en cuenta que si el valor a priori de la celda es de total incertidumbre (0'5), el factor de olvido no afecta, puesto que su cometido es hacer olvidar el conocimiento a priori de esa celda, es decir, se olvida el conocimiento inicial para que sólo permanezca el debido a la entrada sensorial. Si no se dispone de ninguna entrada sensorial sobre la celda y no hubiese factor de olvido, está permanecerá a 0'8 indefinidamente. Pero si se configura el sistema para que posea un factor de olvido como el anterior y no existe ninguna evidencia externa, su certeza descenderá hasta la incertidumbre total (0'5), como puede verse en la figura 4-25.

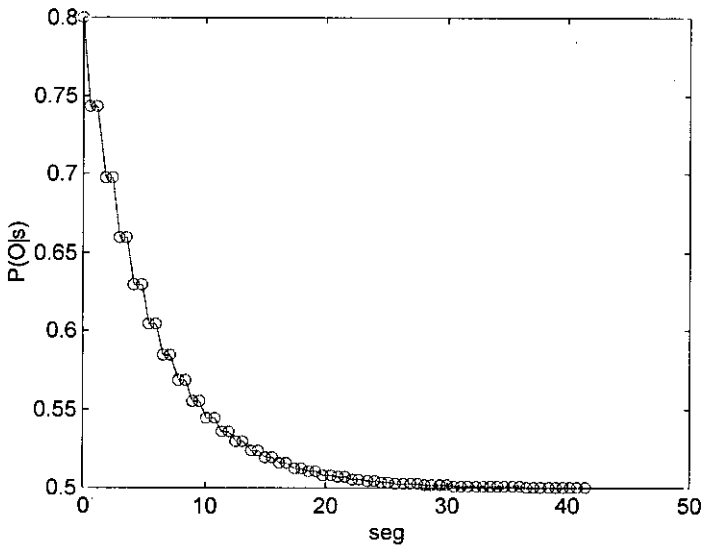


Figura 4-25. Decremento de la creencia a priori de una celda en el tiempo.

También es interesante mostrar el funcionamiento del factor de olvido cuando el sensor está observando un objeto. Así, en el caso de una celda a una distancia R= 1m. (como en el caso anterior) y una estima del sensor de (0'7, 0'3) podemos ver como, en la figura 4-26, la creencia aumenta hasta alcanzar un valor de 0.97, momento en el que no puede aumentar su certeza puesto que la fiabilidad considerada para el sensor a esa distancia lo impide. Cuando se activa el factor de olvido se observa como la certeza cae hasta un valor por debajo del cual no puede disminuir. Esto es debido a que el factor de olvido se aplica al conocimiento a priori y el valor que permanece constante es el debido a la entrada sensorial - la celda no tiene en ese momento ningún conocimiento a priori (0'5, 0'5). Su alta creencia se debe a un seguro conocimiento de la existencia de un objeto provocado por a una entrada sensorial afirmativa durante mucho tiempo – la λ procedente del sensor tiende a (1,0).

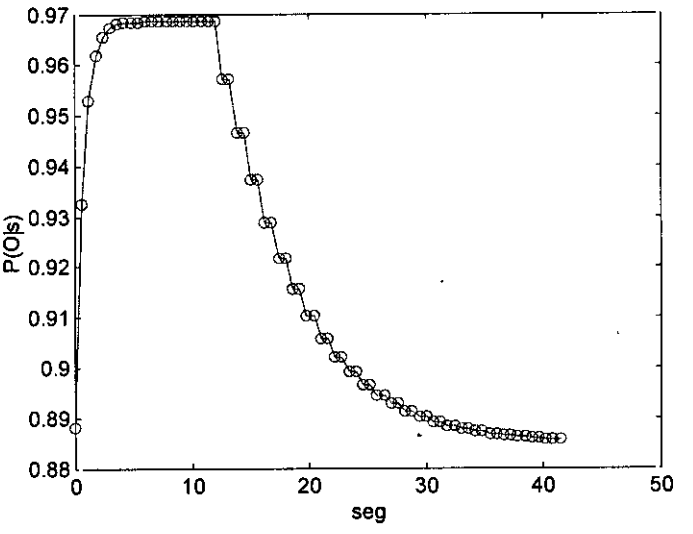
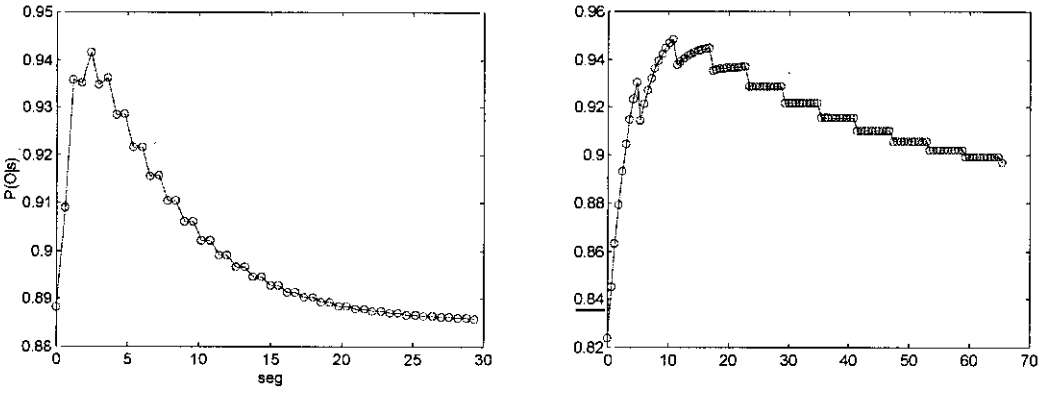


Figura 4-26. Creencia de una celda: olvido de la probabilidad a priori y entrada sensorial.

Por supuesto, el tiempo de olvido debe sintonizarse según el funcionamiento de los sensores y del entorno en el que actúa el sistema. En la figura 4-27 pueden observarse dos factores de olvido distintos. En ambas figuras se observa la disminución de la certeza debido al factor de olvido, pero con una velocidad de actuación diferente. Esto permite que sea se pueda configurar dependiendo de las necesidades del problema a tratar.

En al figura 4-27b puede observarse como llega un momento en que no puede estimarse una probabilidad mayor porque la certeza en el sensor tiende a (1,0). Por lo tanto lo único que se consigue al utilizar el factor de olvido es variar el valor de la probabilidad a priori de la celda. Es decir, se olvida la información disponible a priori puesto que puede ser errónea, pero no se ve afectada la creencia procedente de nuevas medidas.



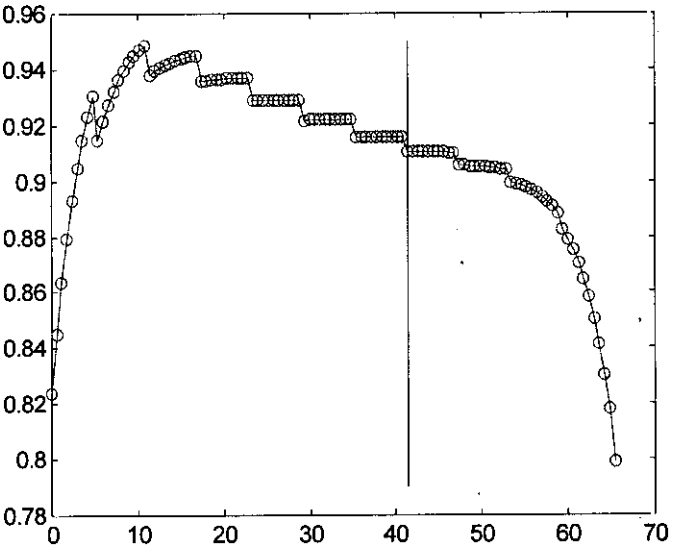
a) Tiempo de olvido rápido b) Tiempo de olvido lento.

Figura 4-27. Entrada sensorial y factor de olvido.

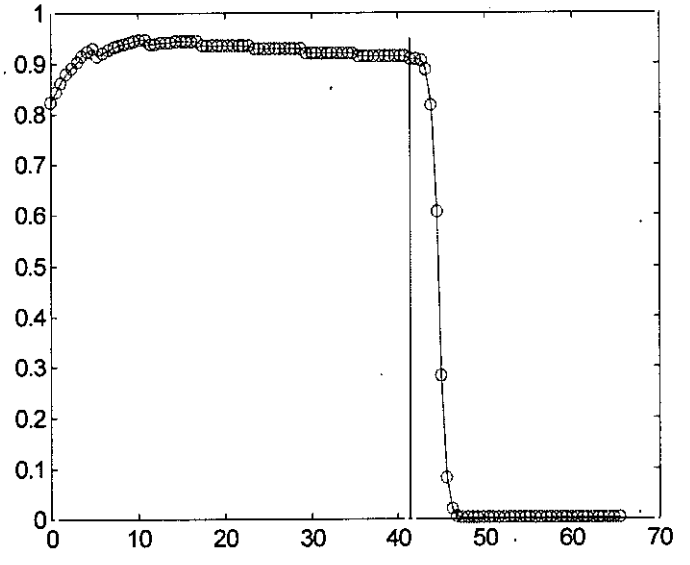
Esto es lógico, puesto que inicialmente se utiliza la información a priori mientras se recoge evidencia externa, pero una vez que se ha obtenido información del entorno la información a priori no es tan relevante y se puede olvidar. Sin embargo, aunque este funcionamiento puede ser deseable, en la figura 4-27b se puede observar que el cambio en la certeza de la celda es debido al olvido de la información a priori y no a medidas del sensor, puesto que este alcanza una certeza absoluta debido a la reiteradas medidas. Esto plantea la siguiente cuestión: ¿será el sistema capaz de reaccionar a una variación de la entrada sensorial cuando ésta ha permanecido constante durante mucho tiempo?. O, por el contrario, ¿la inercia del sistema de fusión es tan grande que no será capaz de reaccionar rápidamente ante objetos imprevistos?. Si esto último fuese cierto el sistema sensorial sería inválido en un sistema reactivo, puesto que no puede permitirse un tiempo de reacción excesivo ante cualquier imprevisto.

Para comprobar su capacidad de reacción se realizan dos pruebas:

- En la primera el sistema sensorial recibe una entrada del sensor de ultrasonidos, λ=(0'6, 0'5), durante mucho tiempo. En un instante determinado el sensor comienza a estimar la ocupación de la celda de forma contraria a la anterior, λ=(0'5, 0'6), véase la figura 4-28a.
- En la segunda prueba el caso es semejante, pero en el momento de detectar un cambio, el sensor lo hace con gran seguridad en su certeza, λ=(0'2, 0'9). Obsérvese la figura 4-28b.



a) Afirmación débil.



b) Afirmación fuerte.

Figura 4-28. Cambio de opinión en la entrada sensorial del sensor.

En el caso de que el sensor no esté muy seguro del cambio percibido y emita una información contraria a la anteriormente emitida pero de forma débil, el tiempo de reacción es grande (unas 15 medidas), figura 4-28a. Esto es lógico puesto que si no está muy seguro del cambio en el entorno, necesita cerciorarse con varias medidas para compensar su inseguridad. En el otro caso, figura 4-28b, debido a que el sensor se encuentra muy seguro de la inexistencia de un objeto

(afirmación fuerte), se puede ver como en muy pocas observaciones (2 medidas) se recoge el cambio percibido.

Por lo tanto, el sistema es capaz de reaccionar adecuadamente a las entradas sensoriales, siendo su respuesta a un cambio en el entorno proporcional a la certeza con la que se detecta dicho cambio. La diferencia entre los instantes en los que comienza a cambiar la certeza debido a la entrada sensorial en cada uno de los casos puede observarse en la figura comparativa 4-29, donde se han representado juntas las figuras 4-28a y 4-28b y se ha ampliado la zona en cuestión.

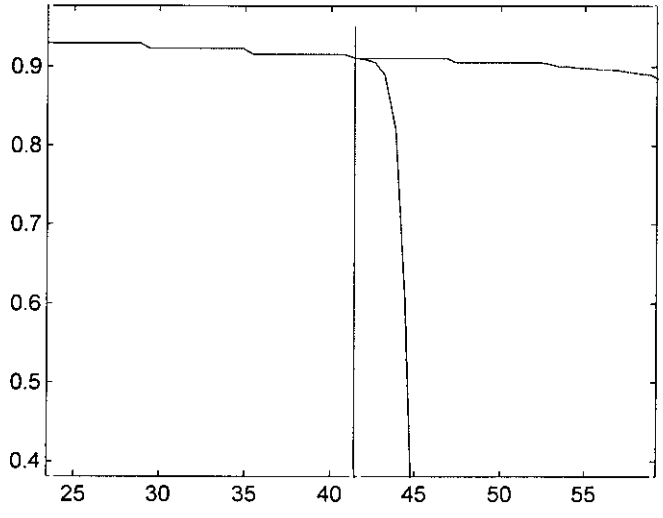


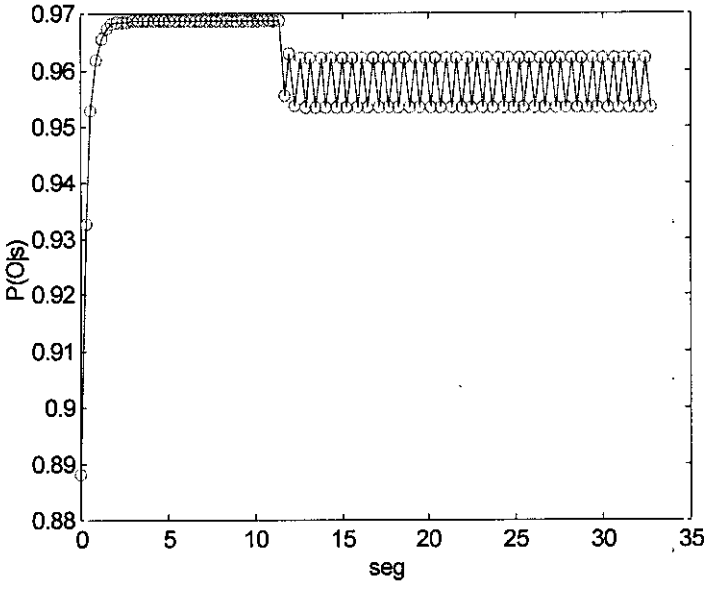
Figura 4-29. Comparación del tiempo de respuesta del sensor.

Siempre que se utilice información de un mapa como información a priori, puede ser eliminada mediante este factor de olvido, evitando así errores en la lectura de la información del mapa o en la utilización de información incorrecta.

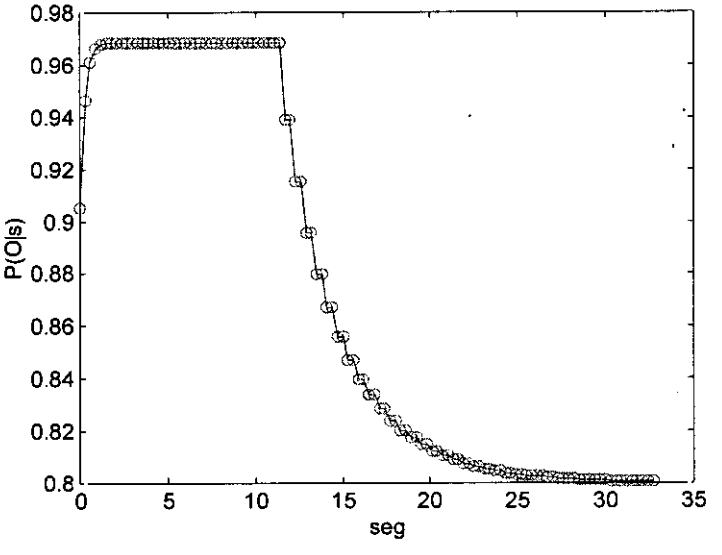
4.6.1.2. Factor de olvido en las medidas sensoriales

Otro tipo de información que es recomendable olvidar en muchas ocasiones es la referente a la información sensorial. Cuando un sensor realiza una estima a partir de una cierta medida, ésta es susceptible de errores y por tanto deber tenerse en cuenta esta posibilidad. La forma de hacerlo, debido a que estos errores suelen ser aleatorios, es disminuyendo su certeza de forma que al cabo de un cierto tiempo sólo queden aquellas medidas que han sido repetidas cierto número de veces.

La forma de hacer esto es incluyendo un factor de olvido a la entrada sensorial de cada uno de los sensores (el factor de olvido de cada sensor será distinto). Este factor de olvido tiende a hacer que las dos medidas sean iguales, lo que indica un desconocimiento total por parte del sensor.



a) Olvido con medidas sensoriales.



b) Olvido sin medidas sensoriales

Figura 4-30. Factor de olvido en las medidas sensoriales.

Por ejemplo, para ver su efecto, supongamos un sensor que percibe un objeto, estima de λ=[0'9, 0'3] y un conocimiento a priori de la celda de (0'8, 0'2), y en un momento determinado comienza a actuar el factor de olvido sobre la observación sensorial. En la figura 4-30a, el sensor continua percibiendo la existencia de un objeto en esa celda mientras actúa el factor de olvido. En este caso la creencia en la ocupación de esa celda se mantiene (aunque disminuirá en un pequeño factor). En la figura 4-30b se muestra un caso distinto. El sensor deja de realizar medidas sobre esa celda. En este caso se puede ver como cuando comienza a actuar el factor de olvido la creencia sobre la celda debido a la entrada sensorial disminuye hasta olvidarse por completo y la creencia final de la celda será la que estaba disponible a priori (no está activo el factor de olvido para la información a priori).

Tengamos en cuenta que el cálculo del factor de olvido, tanto a priori como sensorial, se realiza de forma empírica puesto que es muy dependiente de la calidad de los sensores, la frecuencia de muestreo y actualización de la información, del tipo de entorno y de las características del robot (su aplicación, velocidad de desplazamiento, etc.). Por ello no se ha indicado un método para su cálculo sino se ha mostrado su sencilla utilización y versatilidad para adaptarlo a cualquier aplicación. El cálculo de τ_0 o de la matriz de transición de estados, así como su frecuencia de aplicación, debe ser sintonizada para cada aplicación concreta.

Recordar, por último, que si la información del sensor permanece constante durante cierto tiempo se puede alcanzar la creencia externa por parte del sensor de (1,0), o al contrario. Sin embargo, esta creencia no se ve reflejada de igual forma en la creencia de la celda, puesto que el sensor se encuentra ponderado por la matriz del modelo del sensor y la creencia final no tiende a 1, sino a un valor inferior. Ejemplos en los que se puede ocurre esto se pueden observar en las figuras 4-27b y 4-28a, donde al cabo de un cierto tiempo de permanecer la entrada sensorial fija el sensor alcanza una creencia externa de (1,0) y por tanto la creencia de la celda permanece constante ya que no hay ningún cambio en el estado sensorial. Sólo se pueden producir cambios debido a una opinión en contra de este sensor, o de otro, o a la disminución, o aumento, de la información a priori. En las figuras 4-27b y 4-28a los escalones observados se deben a la disminución de la información a priori.

4.7 Resultados experimentales

Un vez se ha indicado con detenimiento el funcionamiento del sistema de fusión de medidas de entorno, en esta sección se muestra su comportamiento en un robot en movimiento. Para ello se realizan dos pruebas distintas. La primera realiza la fusión de medidas reales de ultrasonidos tomadas durante un recorrido del robot en nuestro laboratorio. La otra prueba consiste en simular el movimiento del robot en ese mismo laboratorio para fusionar la medida de dos sensores distintos: visión estereoscópica y cinturón de ultrasonidos. En este caso ambos sensores son simulados.

Hasta el momento sólo se ha mostrado la evolución de la creencia de una celda de forma aislada de las demás, ahora se pretende señalar el efecto del sistema de fusión en la construcción de un mapa de creencias para su utilización por parte de un robot móvil.

4.7.1. Modelo sensorial y entorno utilizados

Lo primero que se debe realizar, como se señaló en la sección 4.5.1, es el diseño de la dependencia causa-efecto de los sensores de entorno disponibles. Se debe tener en cuenta el efecto de cada uno de los sensores y cómo están relacionados entre sí y con respecto al nodo de fusión. El nodo de fusión se encargará de obtener el resultado conjunto de las estimas de los vectores de características que hemos definido. En nuestro caso sólo estudiaremos el estado de ocupación de la celda (existencia o no de un objeto), aunque podrían tratarse otros aspectos como detección y reconocimiento de esquinas, rincones, paredes, puertas, etc.

Para realizar la fusión de las medidas de ocupación de cada celda, se diseña una red bayesiana semejante a la de la figura 4-11. Nuestro diseño de la red de fusión se muestra en la figura 4-31, donde se consideran dos sensores independientes entre sí (ultrasonidos y visión) y la información del mapa como información a priori.

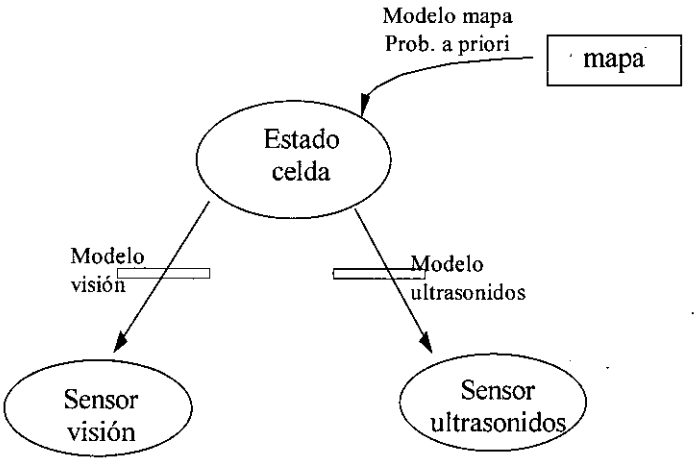


Figura 4-31. Red bayesiana para integración multisensorial en el ejemplo práctico.

El modelo de sensibilidad y especificidad de cada sensor debe ser calculado por el usuario realizando distintas pruebas experimentales sobre sus sensores para comprobar como afectan distintas circunstancias sobre el modelo de detección de sus sensores. El modelo diseñado por el usuario puede ser tan complejo como se desee. En nuestro caso utilizaremos el modelo de los sensores indicado en la sección 4.4.2, modelo que es dependiente de la distancia al objeto detectado.

Para mostrar el funcionamiento del sistema de fusión sobre un entorno real el robot se desplaza por nuestro laboratorio, véase figura 4-32, midiendo el entorno que le rodea. Las pruebas realizadas se presentan a continuación en dos ejemplos distintos: medidas reales de ultrasonidos y medidas simuladas de ultrasonidos y visión.

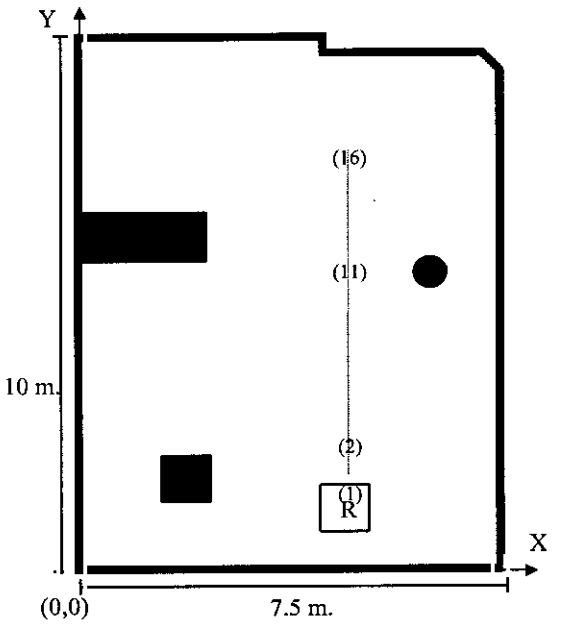


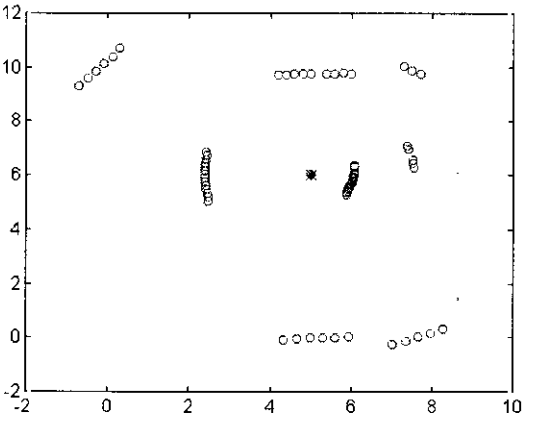
Figura 4-32. Mapa del laboratorio y camino del robot.

4.7.2. Fusión de medidas reales de ultrasonidos

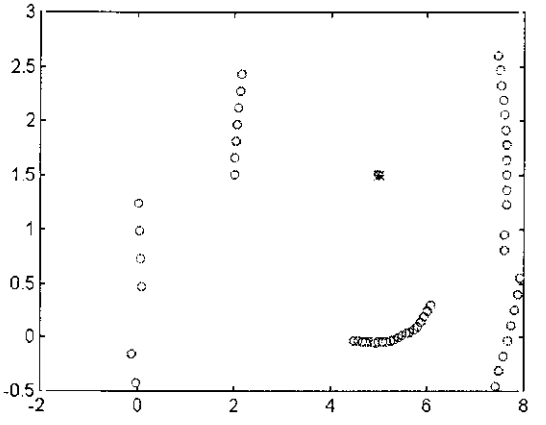
En este caso únicamente se utilizan medidas del cinturón de ultrasonidos durante el movimiento del robot. El modelo de red presentada en la figura 4-31 puede ser utilizada igualmente, aunque las únicas medidas utilizadas son las obtenidas por el sensor de ultrasonidos (el sensor de visión no está activo y además no se utiliza ningún mapa como información a priori).

En este ejemplo, el cinturón de ultrasonidos es implementado mediante una torreta de ultrasonidos, véase el Apéndice D, que puede girar 360 grados. Aquí se utiliza para tomar datos de su entorno cada 15 grados a fin de detectar cualquier objeto alrededor del robot.

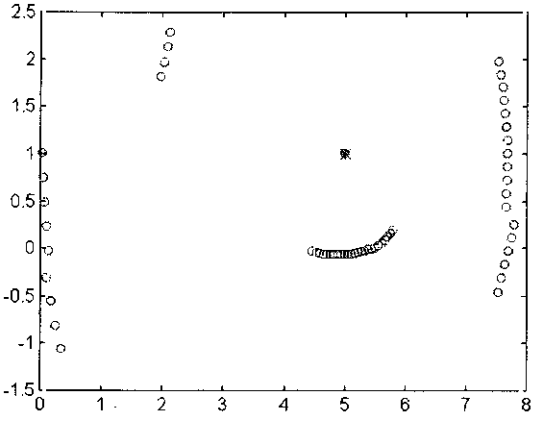
El robot realiza el trayecto mostrado en la figura 4-32 y observa su entorno en unos instantes determinados, en particular realiza una media del entorno cada 0.5 m., lo que da lugar a 16 observaciones. Como ejemplo, en la figura 4-33 se muestran las observaciones obtenidas por el sensor en las tres posiciones (1), (2) y (11) señaladas en la figura 4-32. Las marcas corresponden a las posiciones de las celdas del mapa de ocupación indicadas por el sensor como ocupadas y utilizadas para incrementar la certeza acumulada en el mapa de ocupación (en el ejemplo las celdas son de 10 x 10 cm.). La señal correspondiente a un asterisco (*) representa la posición del robot.



a) Posición 1



b) Posición 2



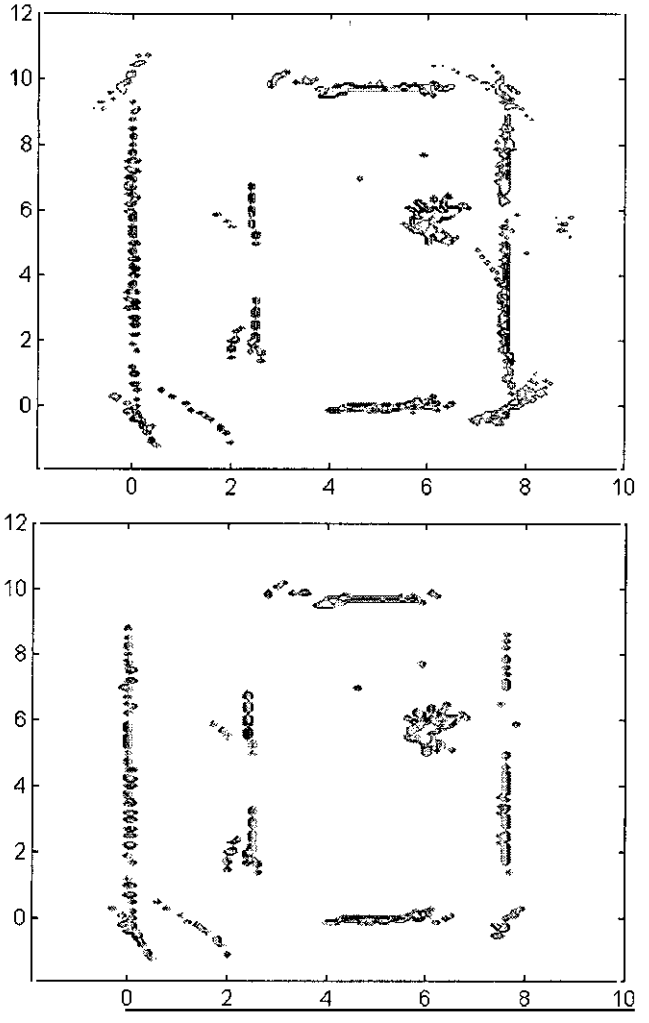
c) Posición 11.

Figura 4-33. Medidas 1,2 y 11 del sensor de ultrasonidos en el trayecto elegido.

El sensor utilizado es el más simple posible. No se define ningún tipo de modelo del entorno y sólo se informa sobre celdas ocupadas con una información de certeza de (0'8, 0'2). Inicialmente se han considerado todas las celdas del mapa con una certeza de ocupación de 0'5 (totalmente desconocidas, puesto que no se ha utilizado ningún mapa como conocimiento a priori).

En la figura 4-34a se muestra el mapa resultante al integrar todas las medidas observadas por los ultrasonidos sin tener en cuenta ningún factor de olvido. En este caso se muestran todas las celdas que tienen una creencia de ocupación superior a 0'5. Como se puede ver, al no existir ningún factor de olvido, cualquier medida obtenida (aunque hayan sido falsos ecos) permanece almacenada aunque la creencia en su ocupación es pequeña.

En la figura 4-34b se muestra el mismo resultado de fusión pero sólo para las celdas que considera el robot como ocupadas. Esto es un factor experimental que depende de la frecuencia de medida de los sensores, la certeza de estos, etc.; en este ejemplo se han considerado como ocupadas (aquellas celdas cuya creencia es superior a 0'64). Se puede observar que las medidas erróneas o espurias han sido observadas una o muy pocas veces y por tanto no ha crecido su certeza hasta ser consideras como ocupadas, por lo que se obtiene un mapa más preciso. Por supuesto si existen errores en la medida que se repiten, como el caso de la columna o de las esquinas, no podrán eliminarse. La única forma de corregir estos errores es con la ayuda de otro sensor.



a) Mapa de creencia obtenido b) Mapa de celdas consideradas ocupadas

Figura 4-34. Fusión de medidas reales de sensores de ultrasonidos.

4.7.3. Fusión de medidas de ultrasonidos y visión simuladas

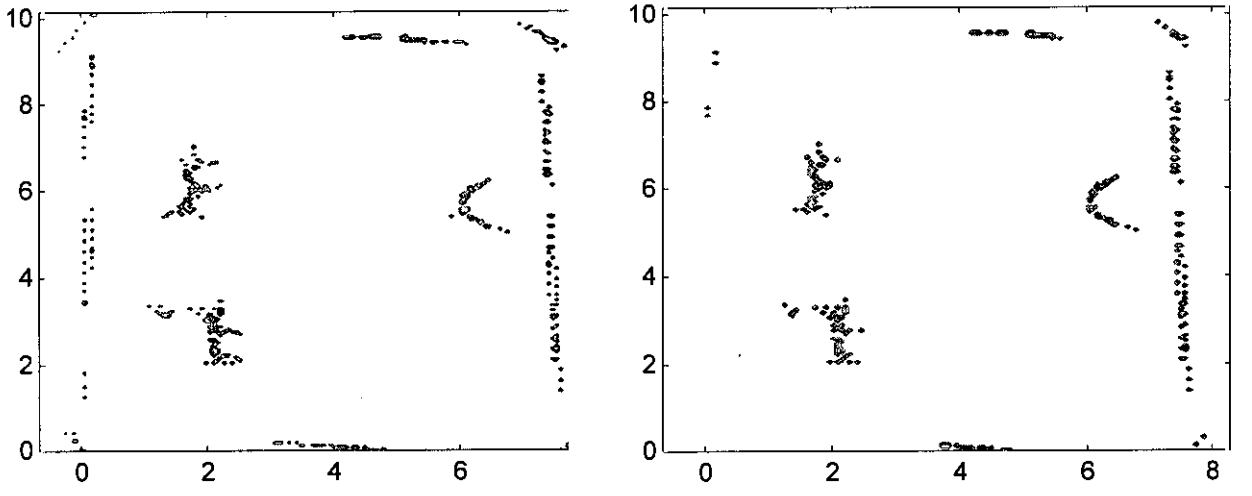
En este caso se utilizan los dos sensores de entorno indicados en la figura 4-31: cinturón de ultrasonidos y visión estereoscópica. Las medidas de ambos sensores son simuladas a partir de la posición del móvil y del mapa de entorno y, como en el ejemplo anterior, no se utiliza información a priori correspondiente a un mapa de entorno.

Para simular los ultrasonidos se utiliza un modelo de los sensores semejante al desarrollado por Leonard para el sonar [Leonard 90; Leonard 92]. El modelo de visión es un modelo propio desarrollado a partir de nuestra experiencia en visión estereoscópica y tiene en cuenta la distancia al objeto, el ángulo de visión, la

apertura de campo y los parámetros de las cámaras como tamaño de pixel, distancia focal, etc. Con él se simula si el robot reconoce o no un objeto y, en caso afirmativo, a qué distancia estima su posición.

El recorrido del robot es el mismo que el mostrado en la figura 4-32 con el fin de compararse con el ejemplo anterior. En este recorrido se realizan distintas pruebas: usando un único sensor (o ultrasonidos o visión) o utilizando los dos a la vez.

En la figura 4-35a se muestra el mapa de creencia obtenido mediante la fusión de las medidas ultrasonidos cuando sólo se simula el funcionamiento del cinturón de ultrasonidos y en la figura 4-35b las celdas consideradas ocupadas. Aquí conviene hacer notar que, como se puede observar comparando con la figura 4-34, las medidas simuladas son más imprecisas que las medidas reales pero como lo que nos interesa es comparar la fusión de medidas de un sensor con respecto a la de dos sensores es suficiente con esta simulación.

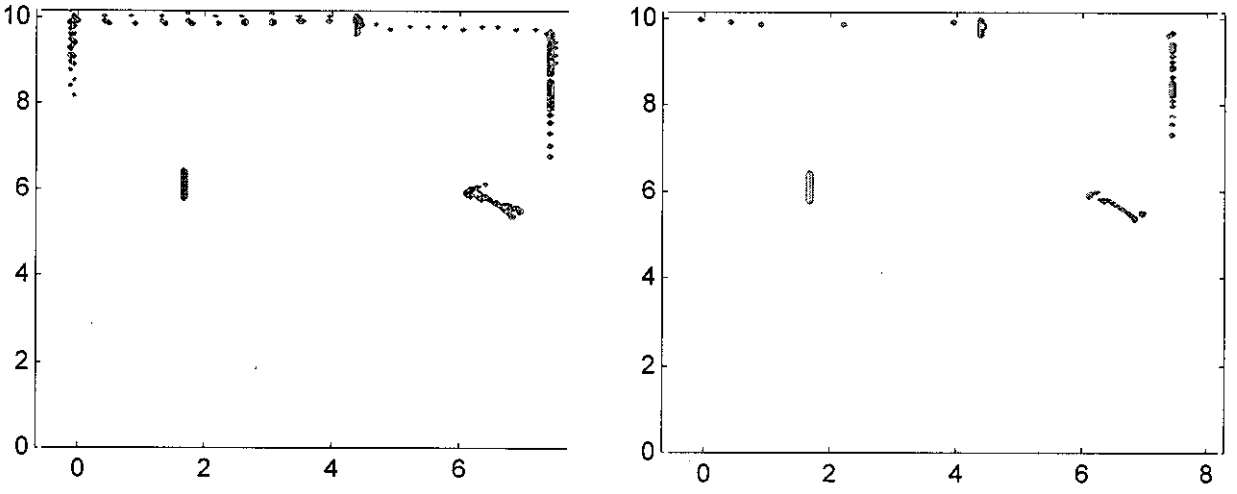


a) Mapa de creencia obtenido b) Mapa de celdas consideradas ocupadas

Figura 4-35. Fusión de medidas del sensor de ultrasonidos.

En la figura 4-36a se muestran las observaciones del sensor de visión estereoscópica y en la figura 4-36b las celdas que se consideran ocupadas, como en el caso de ultrasonidos, pero ahora sólo se simula el funcionamiento del sensor de visión. Sólo se ve la parte superior porque el robot únicamente tiene un ángulo de visión hacia delante en cada cámara de 45 grados. Como el robot no ha girado en ningún momento la visión sólo se ha observado la parte delantera. Además debido a que para detectar una superficie necesita un cierto contraste, sólo se puede

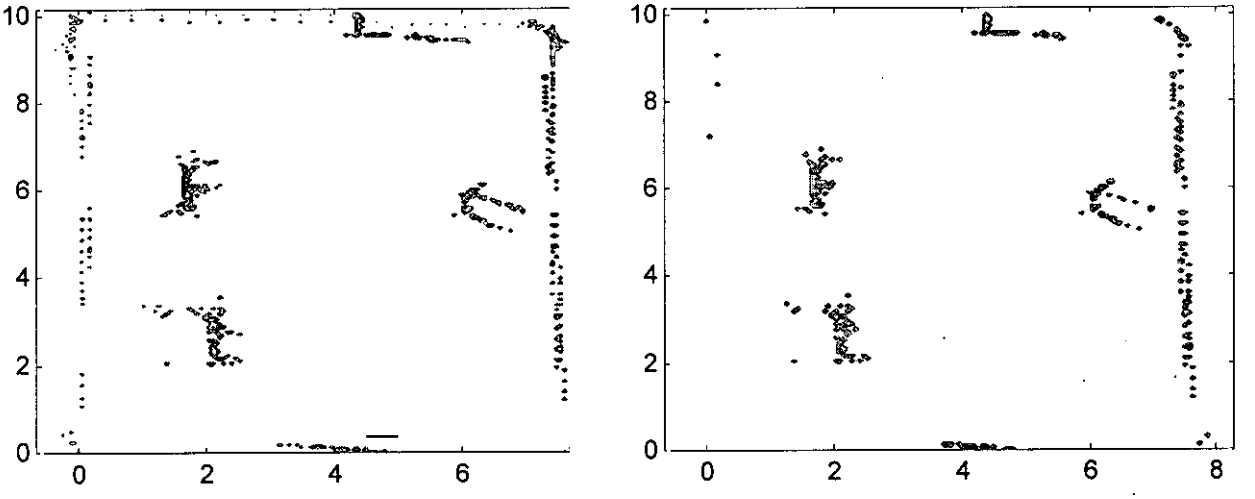
reconocer con precisión el borde del objeto de la parte superior (el que queda en su ángulo de visión) y las esquinas superiores, sobre todo la esquina y la pared que se encuentran justo enfrente del robot.



a) Mapa de creencia obtenido b) Mapa de celdas consideradas ocupadas

Figura 4-36. Fusión de medidas de sensores de visión estereoscópica.

La fusión de ambas medidas da lugar a una estima mejor del entorno, véase la figura 4-37. En el caso mostrado, existe un error entre las medidas de ultrasonidos y las de visión en el cálculo de la posición de la pared de enfrente, lo que impide su correcta fusión. Sin embargo, podemos ver como se reconoce con mayor precisión las esquinas del objeto que es observado por los dos sensores y se detecta la esquina que se encuentra enfrente del robot y que no se podría estimar con ultrasonidos.



a) Mapa de creencia obtenido b) Mapa de celdas consideradas ocupadas

Figura 4-37. Fusión de medidas de sensores de ultrasonidos y visión.

Conviene indicar que siempre se podría mejorar la simulación para obtener mejores resultados, pero se han dejado éstos para hacer hincapié en que la fusión de las observaciones de distintos sensores no es una panacea y sólo da lugar a una observación mejor cuando estas observaciones están de acuerdo o se complementan. En caso contrario, ni pueden reforzarse y ni dan lugar a una mejora de las estimas obtenidas individualmente. Además, se debe resaltar que se ha procurado simular la dificultad de tomar datos por parte de visión, principalmente en su lentitud en el procesamiento de una imagen, lo que nos ha llevado a tomar muy pocas observaciones (cuatro imágenes), por ello no ha sido posible confirmar la existencia de la pared del fondo, si se desease una estima mejor es necesario reiterar las medidas para que aumente su creencia en la existencia del objeto, como se ya ha mostrado en las secciones anteriores.

4.8 Resultados del capítulo

En este Capítulo se han descrito distintos métodos para fusionar características y declaraciones de identidad de distintos sensores. Entre ellos se han escogido las redes bayesianas, o redes de creencia, porque permiten un diseño gráfico de la utilización de los sensores y una sencilla configuración del sistema de fusión. Esto permite que sea muy adecuado para su utilización por distintos usuarios y en distintas aplicaciones. Además las redes permiten su uso de modo distribuido, paralelo y asíncrono, características buscada en el diseño del sistema sensorial de esta tesis. Todas estas propiedades han sido mostradas para un ejemplo sencillo: la creación de mapas de creencia.

Para comprender mejor el funcionamiento del sistema se ha comparado su utilización con los mapas de creencia diseñados por Elfes, pero además se ha mostrado su capacidad para ampliar y modificar los modelos utilizados, así como la posibilidad de su uso en otro tipo de mapas como los histogramas de campo de Borenstein o la utilización de ambos métodos simultáneamente según las necesidades del entorno, sensores o problema planteado.

La utilización de las redes bayesianas para crear histogramas de campo llevan a la necesidad de diseñar redes temporales. Las redes dinámicas planteadas resuelven el problema del uso del sistema de fusión de entorno cuando se necesitan propiedades temporales, como por ejemplo es la inclusión de un factor de olvido. Se han incluido en el sistema diseñado dos tipos de factores de olvido: uno para olvidar la información a priori y otro para olvidad la información externa (sensorial) obtenida, mostrando distintos ejemplos de utilización.

Capítulo 5

Implementación del Sistema de Fusión

En este capítulo se expone cómo se utiliza el sistema de fusión multisensorial desarrollado en los capítulos anteriores dentro de un sistema de control de un robot real. Para ello, siguiendo la misma filosofía anterior, se realizan diseños lo más generales y versátiles posibles, se estudian las arquitecturas de control existentes y se propone una arquitectura de control general que permita modificar de forma sencilla las tareas necesarias (e incluso la inserción de otras nuevas) en cualquier robot móvil autónomo.

Las tareas escogidas son muy generales para que el sistema sea utilizable en diferentes robots. Su implementación es muy sencilla y se deja la posibilidad de cambiar los algoritmos utilizados por otros más complejos. Por supuesto para que este sistema sea fácilmente reutilizable es necesario realizar un diseño muy claro que sea de fácil comprensión y cuyo interfaz esté claramente especificado. Esto se ha realizado utilizando el lenguaje de especificación gráfica orientado a objetos en tiempo real, ROOM.

Finalmente, una vez desarrollado todo el sistema de control, se simulan diferentes situaciones en las que el sistema puede encontrarse: se presentan situaciones de fallo de alguno de los sensores, esquiva de obstáculos imprevistos en la trayectoria fijada o utilización del robot para exploración de su entorno y construcción de mapas.

Introducción

Para implementar el sistema de fusión multisensorial en un robot móvil es necesario definir su relación o interfaz con otras tareas necesarias para el correcto funcionamiento del robot móvil. Por ello, la mejor forma de mostrar su utilización en un sistema robótico es presentar su funcionamiento dentro de todo el conjunto de tareas existentes en un robot móvil.

Por supuesto, el tipo, número y función de cada una de las tareas que componen un robot autónomo móvil es diferente y depende del diseñador y de la labor que desarrollará el robot. Sin embargo, todos presentan ciertas características comunes que ayudan a comprender la importancia del sistema de fusión dentro de un robot móvil.

Por lo tanto, siguiendo la filosofía de generalidad que se ha buscado en el sistema de fusión multisensorial, se define una organización del sistema robótico en tareas que pueden ser fácilmente comprensibles y modificables por el usuario.

La organización de cualquier sistema robótico determina su capacidad para llevar a cabo tareas y reaccionar a eventos. Su organización da lugar a la arquitectura de control del robot, que se pueden clasificar en dos grandes filosofías:

- Imitación de las capacidades de los seres vivos, es decir, son sistemas que se inspiran en la biología y presentan sistemas basados en respuesta a distintos estímulos [Brooks 91; Beer 91].
- Orientados a Inteligencia Artificial, utilizan representaciones simbólicas del mundo y realizan razonamientos sobre estas representaciones; poseen distintos niveles de representación, conocimiento y control [Elfes 86; Simmons 94; Alami 98].

Las distintas funciones que componen un control del robot (planificación de movimientos, control de movimiento, localización, visión, etc.) se denominan de diferentes formas, como son módulos [Fleury 94; Alami 98], agentes dinámicos [Maes 91], agentes inteligentes [Russell 95; Müller 96], agentes reactivos [Brooks 86], comportamientos y esquemas motores [Arkin 91], etc. Estas funciones representan distintas acciones de control o tareas a realizar en nuestro sistema. Cada una dependerá de otras a fin de poder llevar a cabo sus objetivos y los objetivos del sistema completo. El modo en el que se relacionan las distintas funciones, sus capacidades y la toma de decisiones da lugar a distintas

organizaciones conocidas como *arquitecturas de control*. En nuestro caso usaremos el término "módulos" u "objetos" como una forma neutra de referirnos a ellos, y como actores cuando sean especificados mediante ROOM.

Una vez se escogida la arquitectura de control, es necesario definir un método que sea claro en el diseño de cada una de las tareas de forma que su especificación sea clara y de fácil comprensión por parte de cualquier usuario, para que pueda adaptarlo a su robot sin dificultad. Por ello se escoge la metodología y lenguaje de especificación orientada a objetos en tiempo real, ROOM [Selic 95], que permite especificar a cualquier nivel de detalle el funcionamiento del sistema sin perder generalidad.

Así, se diseña una arquitectura de control abierta, capaz de utilizar eficientemente el sistema de integración multisensorial y que puede ser utilizada en diferentes robot sin necesidad de grandes modificaciones, y aquellas que sean necesarias serán identificadas con claridad a partir del diseño realizado.

A continuación, en la sección 5.1, se describen las arquitecturas de control existentes, las tareas necesarias y su organización en robot móviles autónomos. A partir de esta discusión se presenta la arquitectura de control escogida: las tareas necesarias para su buen funcionamiento y su organización. En la sección 5.2 se desarrolla cada una de las tareas especificadas, se describe su motivación, su interfaz, protocolo y funcionamiento interno; y, por último, el modelo de tiempo real orientado a objetos utilizado. En la sección 5.3 se describe el modelo ROOM de cada una de las tareas definidas para el funcionamiento del robot y se muestra un ejemplo de su funcionamiento. Finalmente, en la sección 5.4, se muestra un ejemplo del funcionamiento del sistema, de su capacidad de modificación, robustez y versatilidad mediante la simulación de diversas situaciones en las que se puede encontrar o utilizar un robot móvil terrestre.

5.1. Arquitectura de control propuesta

Uno de los principales inconvenientes en el diseño de robots móviles autónomos es la gran cantidad de potencia de cálculo que se necesita, tanto para interpretar el entorno que le rodea como para planificar un cierto comportamiento. Por ello, el primer objetivo en el diseño de la arquitectura de nuestro móvil es que sea paralelizable en la medida de lo posible, de forma que pueda realizar todas las tareas de planificación y sensorización de forma paralela y distribuida en distintas máquinas.

Normalmente en la mayoría de los robots móviles las responsabilidades de los diferentes aspectos a tratar para conseguir su correcto funcionamiento se dividen en módulos. Estos módulos representan las distintas acciones de control o tareas a realizar en el sistema. Cada uno de los módulos dependerá de otros a fin de poder llevar a cabo sus objetivos y los objetivos del sistema completo.

La responsabilidad para tomar decisiones en un robot móvil, como en todo sistema en tiempo real, puede estar dividida en tareas especializadas y cuidadosamente ordenadas o como cooperaciones sueltas entre tareas. A este conjunto de módulos, según el grado de responsabilidad en la decisión de sus tareas, se le denomina arquitectura de control. Se clasifican en centralizadas y descentralizadas (o distribuidas). En la práctica muy pocos sistemas caen completamente en uno u otro extremo y normalmente presentan un comportamiento mixto con predominio de una u otra filosofía.

En una arquitectura centralizada, la responsabilidad de los módulos de alto nivel es generar especificaciones de las tareas; y la de los módulos de bajo nivel ejecutar dichas especificaciones tan fielmente como sea posible. En las arquitecturas distribuidas, los módulos de bajo nivel pueden responsabilizarse de la realización de metas de alto nivel. En este caso la distinción entre los módulos de bajo y alto nivel está basada en la variabilidad de las metas. Las metas que pueden estar diseñadas en los niveles más bajos, no necesitan ser manipuladas por los niveles superiores; mientras que las metas que pueden cambiar considerablemente de una aplicación a la siguiente podrían ser manejadas mejor por los niveles más altos. Así, dependiendo de qué niveles tomen las decisiones, y de los altos o bajos que sean, tendremos una arquitectura más o menos centralizada o distribuida, siendo lo más normal las arquitecturas mixtas.

Control centralizado: En este tipo de arquitectura el software del robot está separado en distintos módulos: sensorización, razonamiento y control. Es decir, se dispone de un módulo que controla el vehículo, otro que controla los sensores y un tercero de control de la modelización del entorno, la planificación, etc. Ejemplos de esta arquitectura son [Elfes 86] o [Simmons 94]. La ventaja de esta descomposición en tareas es que permite una fácil división en áreas de investigación separadas. Sin embargo, esta aproximación no permite reflejos de propósito especial, ni la alta velocidad que requieren los sensores para ser atendidos y razonar sobre sus medidas.

Control distribuido: Si se permite que el controlador de movimiento pueda tomar una serie de decisiones, estaremos pasando de un sistema centralizado a uno distribuido. El número e importancia de la toma de decisiones que pueda realizar el controlador dependerá del diseño del planificador y del controlador. Por ejemplo, el caso extremo viene representado por [Brooks 86], donde se permite que cada agente calcule sus propias ordenes de guiado y de velocidad directamente a partir de los datos sensados. Eso puede realizarse satisfactoriamente con unas duras restricciones temporales y tiene la ventaja de que, al observar directamente los datos de los sensores, es extremadamente sensible a los cambios del entorno [Baumgartner 94]. Esta capacidad para actuar rápidamente en función de la entrada sensorial, le ha dado la denominación de arquitectura reactiva. La gran desventaja de este tipo de arquitectura es su dificultad de mantenimiento, adaptación a distintos sistemas y su incapacidad para la planificación de tareas complejas. Otros ejemplos de arquitecturas reactivas son los esquemas de Arkin [Arkin 87] y la planificación reactiva [Payton 91].

Ninguno de los dos extremos es ideal para el diseño del sistema buscado. Esto se debe a que se necesita un sistema descompuesto en módulos fácilmente reemplazables por el usuario, versátil y sencillo. Pero a su vez debe ser capaz de realizar tareas complejas y permitir comportamientos reactivos, delegando, cuando sea posible, distintas responsabilidades de control. Por ello se buscará una arquitectura mixta que aproveche lo mejor de ambas. Antes de mostrar el diseño realizado, veamos cuales son las distintas tareas a realizar en un robot móvil autónomo para mostrar qué módulos son necesarios en nuestro robot.

5.1.1. Tareas necesarias en un robot móvil autónomo

Ya sea el control centralizado o distribuido, en el sentido de quién, qué tipo y qué rango de decisiones pueden tomarse en el sistema, la implementación de ese control se realiza siempre, puesto que es la forma natural de hacerlo, mediante un conjunto de tareas distribuidas, ya sea en uno o varios procesadores.

En navegación autónoma y en robot móviles es necesario resaltar la diversidad y compleja información que debe ser procesada, la diversidad de las tareas físicas necesarias para llevar a cabo el control y a la planificación requerida para realizar acciones complejas. Por ello, cualquier sistema como el buscado puede llegar a volverse de una complejidad desconcertante. Decidir qué sensor, actuador o acción de procesamiento se realizará a continuación (sincronización de varias actividades) necesita un planificador a nivel de tareas para proporcionar la secuencia apropiada. Un entorno semejante también necesita múltiples agentes o tareas que se ejecuten semi-independientemente y que cooperen de una forma coherente. Todas estas actividades se pueden clasificar, véase la figura 5-1, en una serie de grupos o niveles. Estos niveles pueden variar de nombre y ordenarse de diferentes formas según las tareas que se le encomiendan, en [Elfes 86] se presenta otra clasificación distinta a la que se presenta a continuación:

I. *Control del Robot:* Este nivel se encarga del control físico de los diferentes sensores y actuadores del vehículo: primitivas para el movimiento, control de sensores (internos y externos), adquisición de datos, etc. Actúa como interfaz entre los niveles superiores del sistema y los detalles de bajo nivel.

II. *Interpretación sensorial:* Los datos adquiridos por los sensores son interpretados y transformados en información utilizable por los niveles superiores. Esto es realizado por los "Sensores lógicos", que se especializan en una clase específica de información de los datos sensados.

III. *Integración y fusión sensorial:* Se encarga de integrar la información proveniente de sensores cualitativamente diferentes, como visión estereoscópica, ultrasonidos, GPS, etc. En este nivel se compara la información redundante de los distintos tipos de sensores, y se extrae una información más completa o precisa.

IV. *Modelización del mundo real:* Para llevar a cabo cualquier grado de autonomía, el robot debe tener un modelo, cuanto más rico mejor, de todo cuanto le rodea. Este modelo se basa en afirmaciones sobre los datos integrados de varios sensores, y se encarga de contrastar los datos obtenidos y proponer una hipótesis adecuada.

V. *Control:* Es responsable de la sincronización de las diferentes actividades y de combinar la planificación realizada con la conducción física del móvil, de modo que se obtenga un comportamiento coherente.

VI. *Planificación:* Se ocupa de la resolución de problemas tales como la planificación de caminos a largo y corto plazo, esquiva de obstáculos, detección de emergencias, etc.

VII. *Navegación:* Para llevar a cabo una meta global propuesta, este nivel proporciona planificación a nivel de tareas para la generación autónoma de secuencias de acciones del actuador, sensor y procesamiento. También incluye simulación, detección de errores, diagnóstico y reactivación y replanificación en caso de situaciones inesperadas o fallos.

VIII. *Supervisor:* Supervisa las distintas actividades y proporciona un interfaz para el usuario humano.

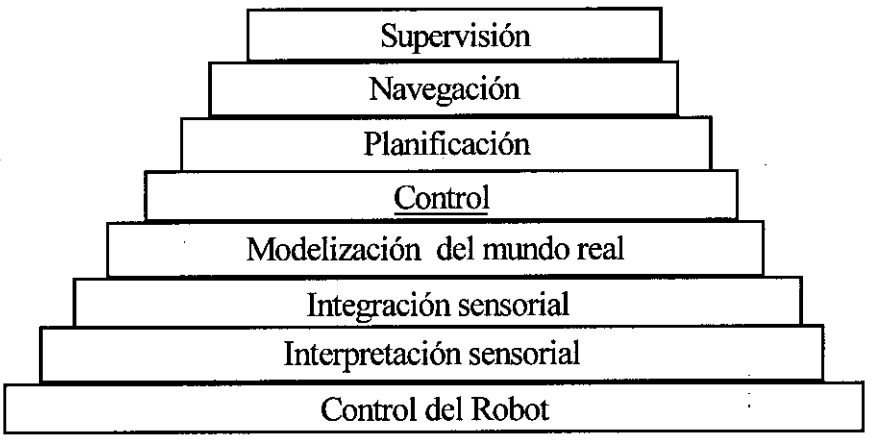


Figura 5-1. Niveles de actuación en todo robot móvil

La representación en forma piramidal de la figura 5-1, sólo pretende mostrar el aumento del grado de complejidad y abstracción de cada uno de los niveles, pero

la interconexión no es siempre entre módulos adyacentes, sino que, por el contrario, en sistemas reales existen interconexiones complejas con varios subsistemas distintos y con múltiple flujo de control y de datos.

Recordemos que un módulo (como por ejemplo Visión estereoscópica) puede ser por sí mismo un sistema complejo con sofisticadas actividades de control, planificación y resolución de problemas. Además, en ningún sistema real se encuentran todos los niveles descritos anteriormente, aunque sí se sigue una estructura semejante.

Por último indicar que esta división de responsabilidades en capas es válida para ambos modos de control o arquitecturas: centralizado y distribuido. Veamos a continuación la relación entre los distintos niveles de actuación para ambas arquitecturas.

Arquitectura centralizada. También conocida como descomposición vertical del problema, se refiere a la separación de las tareas en segmentos verticales. Estos segmentos forman un canal a través del cual la información fluye desde el entorno del robot, vía sensación, por el robot y de vuelta al entorno, vía acción, cerrando el lazo de realimentación en el sentido que muestran las flechas en la figura 5-2. Ejemplos de esta arquitectura son [Elfes 86], [Evans 92], o el ALV [Olin 91] en el que la percepción de los sensores asciende en abstracción por los niveles de percepción (Local, Mundo, Misión y Monitor del sistema) y desciende para actuar con el sistema por los niveles de planificación (Misión, Ruta, Local y Reflejo). Cada pieza debe ser construida a fin de ejecutar el robot como un todo. Por supuesto, en este caso, los cambios posteriores en cualquier parte (para mejorar o extender alguna función) deben hacerse de modo que las interfaces entre módulos no cambien o los efectos del cambio deben propagarse a los niveles vecinos cambiando también su funcionalidad. Cada uno de estos pedazos representan los niveles en los que se descomponen las tareas de un robot móvil.

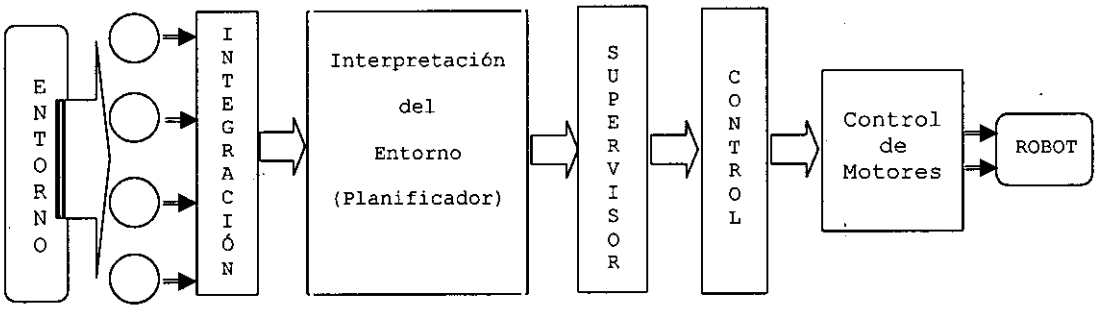


Figura 5-2. Descomposición vertical de las tareas

Arquitectura descentralizada: Cada módulo cubre un rango completo desde una entrada sensorial hasta el control de la salida; sus módulos se dividen en una jerarquía de funciones, cada una actuando sobre niveles inferiores. Es decir, en lugar de fragmentar el problema a partir de una serie de tareas que lleven a la solución (descomposición vertical), se divide el problema basándose en manifestaciones externas deseadas para el control del robot (descomposición horizontal). La arquitectura de este sistema puede verse en la figura 5-3, donde el primer módulo (nivel) observa datos del sensor y mueve el vehículo lejos de los obstáculos. La capa siguiente mueve el vehículo aleatoriamente, a menos que la capa inferior tome el control para evitar un obstáculo. Las capas superiores añaden propósito al movimiento (moverse hacia puertas abiertas, por ejemplo), buscar objetos de interés, etc.

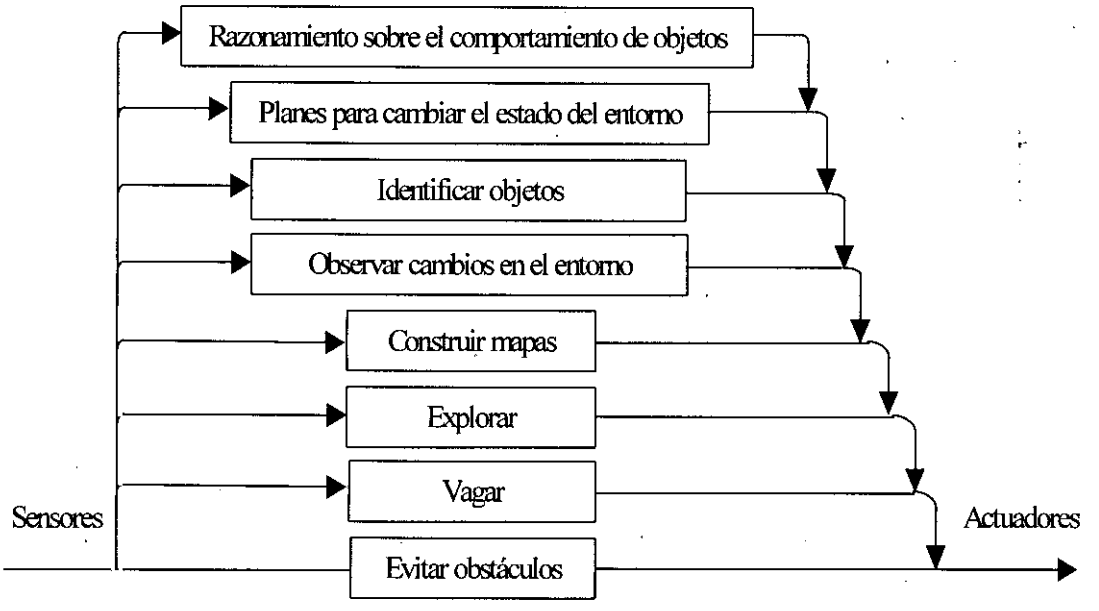


Figura 5-3 Descomposición horizontal de las tareas

Esta arquitectura, típicamente distribuida, denomina a cada módulo como *niveles de competencia* [Brooks 86] y son una especificación informal de una clase de comportamiento para un robot sobre todos los entornos que encontrará. En la figura 5-3, se puede observar que cada nivel de competencia incluye como un subconjunto de herramientas a cada nivel anterior de competencia. La ventaja de los niveles de competencia es que podemos construir capas de un sistema de control correspondientes a cada nivel de competencia y simplemente añadir una nueva capa a un conjunto existente para desplazarlo al nivel superior más próximo. Por esto, a esta arquitectura se le llama también *arquitectura asumida*.

La gran desventaja de estas arquitecturas es que la complejidad de estos sistemas y la estrecha relación entre niveles lleva a la imposibilidad de construir niveles de complejidad elevada, lo que restringe su utilización a sistemas reactivos elementales.

5.1.2. Esquema general

Nuestro sistema constará de una arquitectura de control centralizada, con una descomposición vertical de las tareas del problema. Los agentes representan capas verticales equivalentes a los niveles clásicos de control del robot, como Interpretación sensorial, Integración sensorial, Navegación,... La utilización de esta arquitectura tiene ciertas ventajas en nuestro planteamiento como son:

- Una mejor descomposición de los módulos en distintos campos de trabajo, permitiendo así que varios grupos de trabajo puedan investigar en diversas áreas.
- Expresión natural de la cooperación entre tareas, lo que lleva a una rápida comprensión de las tareas de cada sistema y una fácil actualización de cualquier función para dotar al sistema de distintas capacidades.
- Correspondencia directa entre cada tarea y los objetos de procesamiento independientes, como pueden ser agentes, procesos, objetos, ...
- Facilidad para distribuir las tareas entre distintos procesadores y expresar el paralelismo característico de estas tareas. Además permite especificar clara y sencillamente el interfaz entre los distintos módulos.

Dentro de la multitud de posibilidades en el diseño de una arquitectura centralizada de control se puede destacar la organización de la figura 5-4. En ésta los módulos de bajo nivel maestro-esclavo se corresponderían con los drivers de los sensores de localización y reconocimiento, y a los drivers de los sensores de movimiento. El preprocesamiento representa el tratamiento de los datos y el paso de mensajes entre los módulos. La integración multisensorial realiza un nivel superior de tratamiento de los datos y se encuentra relacionado directamente con el planificador.

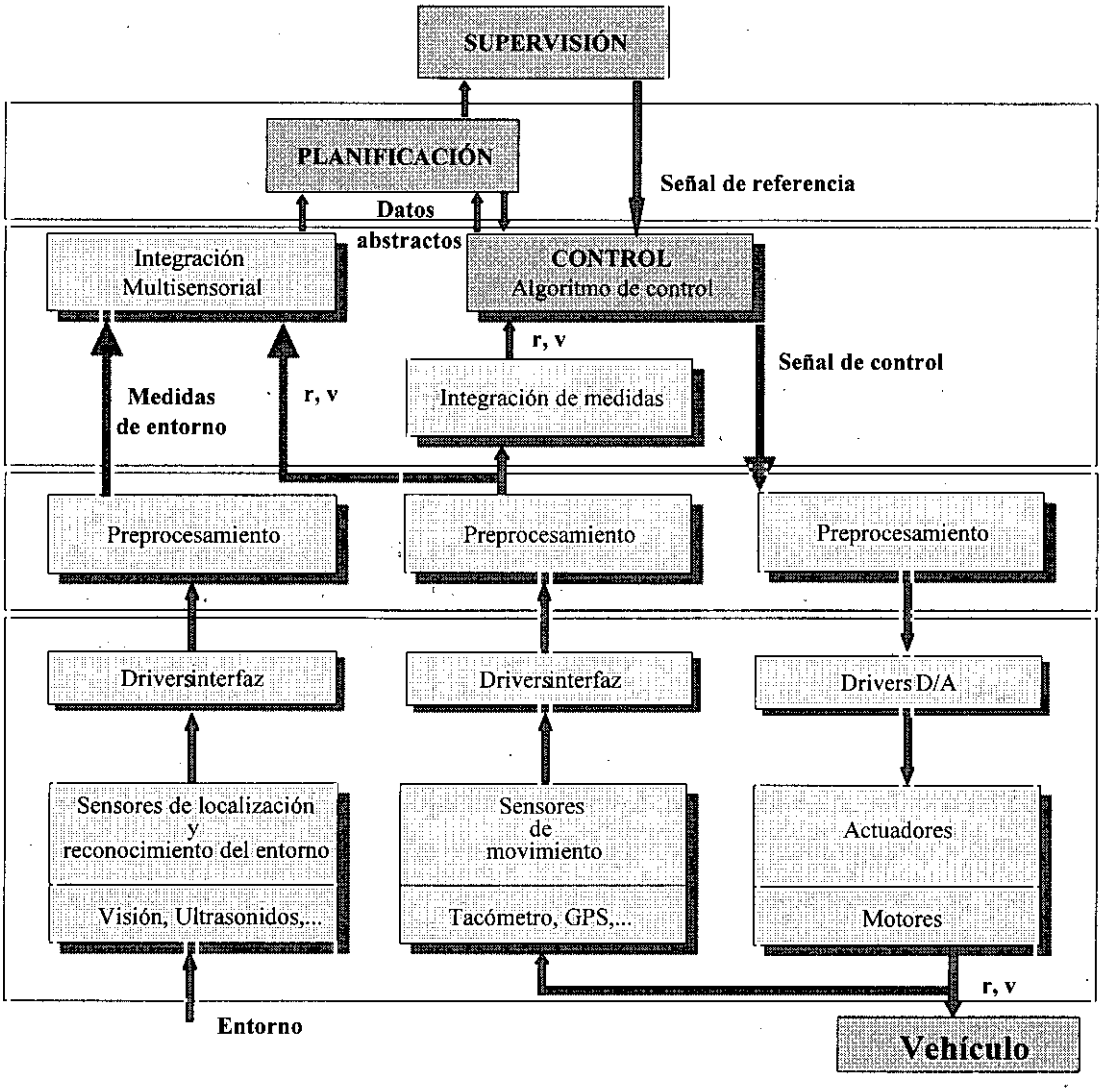


Figura 5-4. Esquema general de un sistema de navegación.

En todo sistema de control de un robot móvil autónomo inteligente a menudo se pueden identificar varias restricciones y necesidades. Veamos como podemos solventarlas con el esquema de la figura 5-4 y cómo deberá ser el modelo propuesto:

Metas múltiples: A menudo aparecerán en un robot metas múltiples, algunas en conflicto. Por ejemplo, puede intentar orientar el vehículo hacia el destino final y a su vez tener que esquivar un obstáculo, variando entonces su orientación; o se puede intentar seguir determinada pared para alcanzar un destino, pero ser prioritario el mantener el equilibrio del vehículo. Para ello hemos dividido el sistemas en módulos (Supervisor, Planificador y Algoritmo de Control principalmente) que dotan al sistema de distintas perspectivas en la consecución

del plan; y una posibilidad de actuaciones reflejas (módulos de bajo nivel como los Actuadores) aunque sólo en situaciones urgentes o peligrosas como en un cambio brusco y peligroso del entorno o en un fallo del sistema.

Sensores diversos: Un robot móvil necesita una gran cantidad de sensores de distintos tipos (cámaras de TV, encoders, sistema GPS, barómetro, brújula, sistema de navegación inercial, etc.) que le permita adquirir y manipular un modelo de su entorno rico y completo. La estructura propuesta dispone de la integración e interpretación (módulos Planificación e Integración y fusión sensorial) de los datos provenientes de todos los sensores disponibles, sin que los demás módulos tengan conocimiento de la existencia de estos sensores.

Robustez: Cuando alguno de los sensores falla o algún tipo de los sensores no está operativo debido a cambios del entorno, el robot debe ser capaz de adaptarse y seguir funcionando perfectamente con los restantes sensores. Esto es posible por medio del módulo Integración y fusión sensorial, puesto que todas la medidas son filtradas y tratadas estadísticamente según su error actual, uniéndose en un único resultado.

Extensibilidad: Poder ampliar las posibilidades del robot añadiendo sin problemas nuevos sensores con mejores capacidades que aparezcan en el mercado. La adición de nuevas capacidades en nuevos algoritmos también debe ser contemplada. La posibilidad de cambiar un algoritmo por otro es muy simple. Sencillamente se crea el nuevo algoritmo, se prueba y depura externamente y entonces se sustituye por el módulo antiguo sin prejuicio en el funcionamiento de los demás módulos. Los nuevos sensores u otros del mismo tipo de alguno de los existentes también pueden ser añadidos sin problemas dada la estructura en módulos independientes del sistema.

5.1.3. Modelo propuesto

Las responsabilidades en la toma de decisiones y las restricciones de tiempo real son divididas en tareas como en la figura 5-2 y teniendo en cuenta los módulos sombreados del esquema general de la figura 5-4 su funcionamiento es el siguiente:

- *Algoritmo de Control*, tiene una función crítica en tiempo real para guiar al robot suavemente a lo largo de una trayectoria especificada.

En nuestro planteamiento se prevé un mecanismo de seguridad para el caso de que no reciba más mensajes de consigna.

- *Planificador de movimiento*, necesita detalles precisos de los datos de los sensores para poder construir un modelo del mundo real y descubrir distintos caminos a partir de dicho modelo, por ello necesita una integración de todos los datos disponibles de los sensores (*Sistema de Integración Multisensorial*). Sus restricciones temporales son algo más blandas, menos críticas y más negociables que las del controlador de movimiento. Nuevas trayectorias deben calcularse a una velocidad razonable, pero si una trayectoria no está disponible el resultado no será catastrófico. El Planificador puede calcular trayectorias más largas para tener más tiempo para futuros cálculos, pero si ocurre una variación del entorno y la trayectoria calculada no es apropiada, debe ser lo suficientemente rápido para garantizar que se actuará en consecuencia: calculando una nueva trayectoria o deteniendo el vehículo.
- *Supervisión*, realiza el control global del sistema y contiene el interfaz de usuario. El interfaz de usuario recibe las órdenes o consignas para la navegación y muestra el estado del sistema, como por ejemplo la posición actual, el entorno observado, la carga de trabajo de cada tarea o distintas descripciones sobre el funcionamiento del sistema.

Una vez comentado en términos generales las necesidades y los distintos módulos en que se puede dividir de forma natural el sistema pasemos a realizar un estudio más detallado de cada módulo:

Posicionadores

Son los elementos necesarios para encontrar el estado y el entorno en el que se encuentra el móvil, figura 5-5. Podemos destacar distintos módulos según la información que utilizan y su nivel de abstracción.

Posicionador 1

Conoce el estado del móvil y en concreto su posición absoluta en el entorno. También dispone de la posición de los obstáculos previstos o baliza naturales como paredes, columnas, puntos de bifurcación, etc.

La posición exacta del carro se conoce a partir de los datos procedentes de la integración de medidas de posición (encoders, GPS, ...) y de la recalibración periódica mediante búsqueda de balizas naturales o artificiales (ultrasonidos, visión, ...).

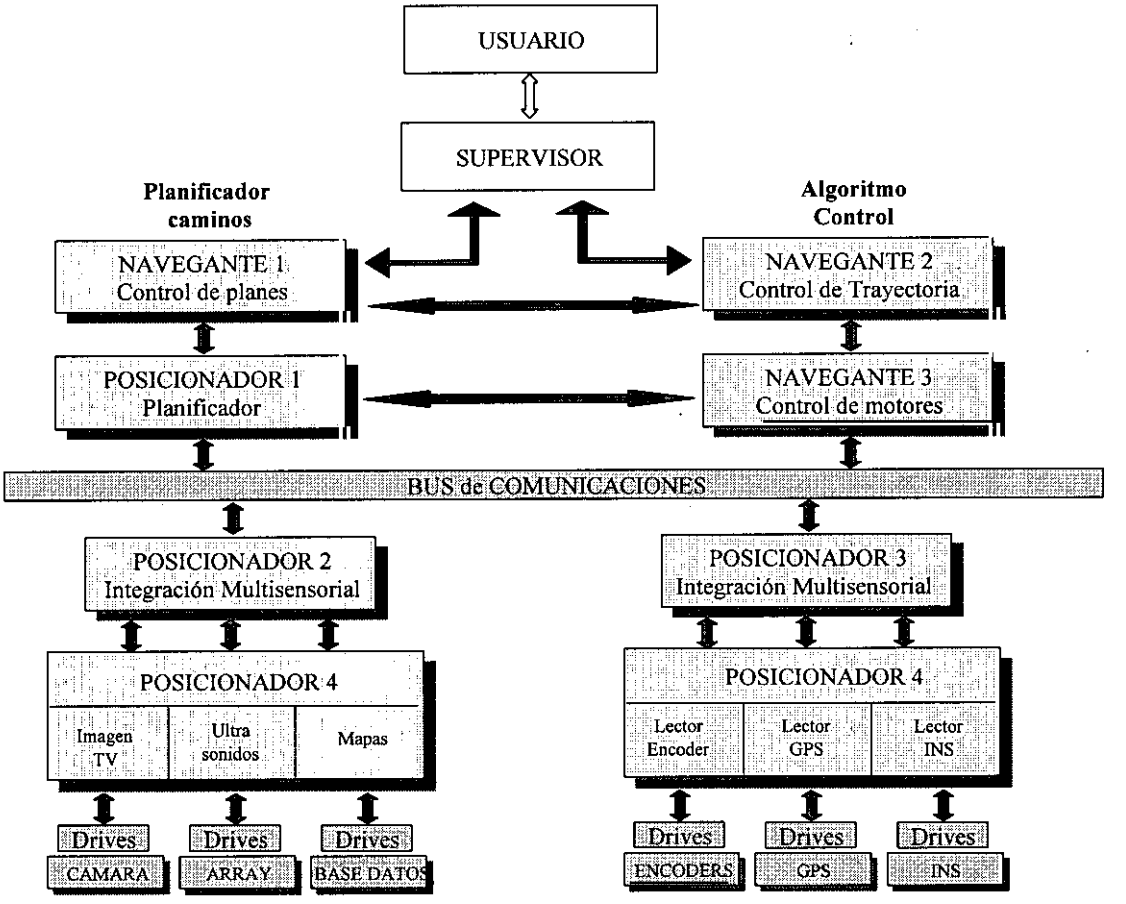


Figura 5-5. Modelo propuesto

El entorno del móvil se almacena en un mapa de ocupación. Éste consta de pequeños cuadrados (rejilla) en la que se situarán los datos de los sensores de entorno. Esta rejilla estará siempre orientada de la misma forma en el plano, el robot podrá moverse en ella en cualquier dirección y siempre mantendrá su forma con respecto al robot.

Posicionador 2

Su misión es integrar los datos procedentes de los sensores de detección de obstáculos y de entorno procedentes del Posicionador 4 (descrito más adelante). Es decir, se encarga de detectar cualquier obstáculo existente en el entorno del robot

de modo que pueda calcularse la trayectoria óptima para alcanzar la meta y comunicárselo al Posicionador 1. Los sensores que trata son sensores capaces de detectar objetos y sus características, como cámaras de visión, ultrasonidos, mapas, etc.

Este módulo utiliza una red bayesiana para fusionar la información proveniente de distintos sensores sobre la existencia de un mismo objeto (visión, ultrasonidos, radar, ...). De esta forma se puede dar un tratamiento cuantitativo a la información recibida para realizar un juicio concreto sobre cada zona del espacio y sobre el funcionamiento correcto o no de cada sensor.

Posicionador 3

Recibe datos del estado del móvil (posición y velocidad del carro) a partir de los datos obtenidos por el Posicionador 4 (de sensores como los encoders, GPS, ...) y datos de posición procedentes del Posicionador 1 (a partir de la correspondencia entre las balizas detectadas y su posición predefinida) y se encarga de integrarlos en una estima de posición lo más precisa posible. Esta estima se realiza a partir de un filtro de Kalman, integrando datos referentes a la posición absoluta del robot y datos de posición relativa. Los sensores tratados por este módulo son los sensores de posición como odómetros, giróscopos, acelerómetros, GPS, captadores de balizas, etc.

Posicionador 4

Su misión es tratar la información en bruto de los sensores para que sean directamente utilizables por los Posicionadores 2 y 3. Este grupo trata información tanto cualitativa como cuantitativa. Se encarga de obtener y transformar la información procedente de todos los sensores disponibles, tanto los que indican donde se encuentra el móvil con respecto a un mapa global, como los que indican la existencia y tipo de un determinado objeto en cierta zona del espacio. Este módulo se encarga de realizar preprocesamiento, asociación de datos, etc.

Tratan los distintos tipos de sensores adaptables en el carro. Éstos se pueden dividir en dos grandes categorías, que ya hemos visto clasificados en el Capítulo 2.

Existe un tipo de sensor que realmente no pertenece a ninguna de las dos clasificaciones anteriores: el cinturón de contacto que actúa de forma reactiva en una situación límite. Consiste en mandar un mensaje de emergencia al Algoritmo de Control cuando el carro contacta con un obstáculo que no ha sido detectado por

los otros sensores. En este caso el Control detiene el carro y se lo comunica a todos los elementos de la navegación.

Navegantes

Son los módulos que determinan las acciones a realizar por el vehículo. Toman decisiones sobre el camino a seguir, la existencia o no de obstáculos, la forma de seguir correctamente la trayectoria y de esquivar los obstáculos imprevistos. También se dividen en varios tipos según su grado de decisión:

Navegante 1

Es el que conoce la misión a realizar, comunicada a partir del Supervisor. Éste es el que se encarga de estudiar el mapa global para determinar la trayectoria a seguir para alcanzar la meta final a partir de las premisas indicadas por el Supervisor.

También es el que planifica la trayectoria sobre un mapa local para llevar a cabo el movimiento inmediato. Si en la trayectoria calculada inicialmente en el plano global se encuentra un obstáculo, calcula sobre el mapa local (mapa más preciso) la trayectoria que permita esquivarlo y volver a recuperar la trayectoria global trazada. De esta forma actúa de manera reactiva, teniendo en cuenta la situación actual.

Navegante 2

Determina los puntos de consigna concretos que debe enviar al control de los motores (Navegante 3) para que siga lo más fielmente posible la trayectoria trazada por el navegante 1, teniendo en cuenta a su vez las restricciones físicas y del entorno para mantener la seguridad en el trayecto del móvil.

Además controla que la trayectoria que realmente sigue el móvil se corresponda con la solicitada y que no corre ningún riesgo cuando se desvía o cuando surgen objetos en cualquier zona próxima al camino previsto.

Navegante 3

Representa el control directo de los motores de las ruedas. Por lo tanto es el responsable de producir las salidas de control necesarias para controlar los motores, de forma que se consiga llevar la velocidad, dirección o radio determinados. Para ello debe compensar errores de desplazamiento lateral debido

a diferencias geométricas o físicas de las ruedas y controlar por ejemplo las dos ruedas de forma diferente en caso de poseer controles separados.

También realiza un control sobre la trayectoria del robot para seguir correctamente la trayectoria fijada por el Navegante 2. Recordemos que en cualquier caso existe un nivel superior (el Navegante 2) que previene posibles desviaciones de la trayectoria fijada, pero siempre que estas sean considerables.

5.2 Sistema de control orientado a objetos en tiempo real

Los sistemas en tiempo real [Laplante 93; Benett 94] normalmente poseen sensores y actuadores de forma que el sistema mide y cambia el estado de su entorno. Se dice que el sistema es de tiempo real cuando es capaz de reaccionar a distintos sucesos dentro de unos límites de tiempo especificados.

Por tanto el software de tiempo real es aquel que controla el comportamiento del sistema en tiempo real, es decir, debe satisfacer las necesidades de tiempo exigidas. Según sean estas necesidades los sistemas se clasifican en sistemas de tiempo real duros y sistemas de tiempo real blandos.

Además de ROOM (*Real-Time Object-Orient Modeling*) [Selic 95] que es el utilizado, existen otros métodos de análisis y diseño de sistemas en tiempo real que utilizan la metodología orientada a objetos como por ejemplo CODARTS [Gomaa 93], OCTOPUS [Awad 96] o extensiones de tiempo real de notaciones orientadas a objetos como el *Real-Time* UML (*Unified Modeling Language*) que permite analizar, diseñar y modelar sistemas en tiempo real utilizando UML [Powel 98; Selic 98].

CODARTS se basa en el análisis y diseño estructurado. Tiene un componente denominado modelo del dominio [Gomaa 95] que define varios puntos de vista para el análisis del sistema. Destaca el uso de objetos concurrentes y su enfoque en tareas posee muchas características del lenguaje ADA.

El método OCTOPUS está basado en los populares métodos OMT y Fusion, y también utiliza notación UML. Está orientado para desarrollo de software orientado a objetos para sistemas empotrados de tiempo real, cubriendo las necesidades de especificación, arquitectura del sistema, análisis y diseño de subsistemas y especifica métodos de implementación y análisis de eficacia. Permite el trabajo en distintos procesadores y resuelve la unión de procesos con objetos mediante un procedimiento sistemático y eficiente que proyecta objetos en procesos. Pero su principal desventaja es que está diseñado para trabajar con requerimiento de tiempo real blandos y si es necesario trabajar con restricciones de tiempo real duras sólo se puede realizar en una parte limitada del sistema.

ROOM es un lenguaje de especificación gráfica, orientado a objetos, para modelar sistemas de tiempo real. Sus principales características son:

- Orientado a eventos, el modelo se adapta perfectamente a los sistemas reactivos.

- Concurrencia, la propia definición de los elementos básicos de ROOM permiten la concurrencia de éstos. Esta propiedad es básica en cualquier sistema empotrado de tiempo real. En sistemas robóticos es imprescindible debido a la propia naturaleza del sistema.
- Distribución, la abstracción de los objetos como máquinas lógicas y la creación de una máquina virtual permiten la distribución de las tareas entre distintas máquinas y su diseño sin tener en cuenta donde se ejecutarán. Esto es una facilidad aconsejable en robots móviles debido a la gran necesidad de cálculo que requieren sus funciones y que obligan a distribuir su procesamiento para poder cumplir las restricciones necesarias.
- Recursividad, el lenguaje permite desarrollar un modelo con varios niveles de detalle que especifiquen por completo un sistema complejo. La recursión puede darse a dos niveles: a nivel de objetos, objetos complejos pueden ser modelados con otros objetos; y a nivel de estados, el comportamiento de un objeto puede ser modelado con una máquina de estados jerárquica en la que los estados se descomponen en subestados hasta el nivel de profundidad necesario.
- Modelado incremental, permite la expresión y verificación de las propiedades del modelo del sistema de modo incremental.
- Reutilización, gracias al paradigma de la programación orientada a objetos, en general, y a la herencia, en particular, la reutilización de distintos objetos es directa y sencilla.

ROOM soporta el paradigma de los objetos que potencia la reutilización. En este trabajo se aprovechará la parte más básica del lenguaje de modelado ROOM, suficiente para modelizar las tareas y sistemas de control de nuestro robot. El lenguaje completo es realmente potente y permite modelizar sistemas complejos como por ejemplo la pila del protocolo ATM [Gullekson 97].

En el Apéndice B se presenta una pequeña descripción del lenguaje gráfico de ROOM y de los conceptos y notaciones gráficas que se utilizarán a continuación para la descripción del sistema desarrollado. Para una mayor descripción de la metodología y del lenguaje de especificación, véase [Selic 95].

5.2.1. Niveles de integración

El esquema de control presentado en la figura 5-5 está referido a la organización y relación entre las tareas necesarias en el robot. Pero nuestro problema se centra en la integración de múltiple información sensorial, por lo que será conveniente especificar los módulos indicados en la figura 5-5 en distintos niveles de abstracción y de tratamiento de la información disponible. Por ello en la figura 5-6 se muestra un esquema del sistema de integración completo desde el punto de vista del grado de procesamiento que sufre la información. En él se pueden ver los distintos niveles de tratamiento de la información que se tendrán en cuenta y su correspondencia con las tareas mostradas en la figura 5-5.

Los niveles de integración mostrados en la figura 5-6 representan:

- *Plan del usuario*: Se encarga de recibir la consigna o plan a ejecutar por parte del usuario. Incorpora el interfaz de usuario y un supervisor del funcionamiento correcto del sistema.
- *Supervisión de planes*: Se ocupa del control del robot a nivel inteligente, es decir, interpreta las consignas del usuario y se encarga de comunicárselas al planificador y también de obtener las consignas necesarias para el control de los motores. Además supervisa el funcionamiento global del sistema.
- *Control y planificación*: Encuentra el camino óptimo en todo momento (esquivando obstáculos imprevistos) y se encarga de conducir el carro para seguir correctamente la trayectoria. Puede soportar el Nivel II de integración definido en el Capítulo 1.
- *Integración multisensorial*: Fusiona los datos de los distintos sensores de forma que se obtenga una estima más precisa que las opiniones individuales. Realiza la fusión del estado del robot y obtiene un mapa global de la información de entorno. Correspondería al Nivel I de integración comentado en el Capítulo 1.
- *Preparación de datos*: Realiza todo el preprocesamiento de los datos antes de realizarse la fusión de datos. Esta preparación incluye alineamiento, asociación, extracción de características, etc.

- *Drives e interfaces*: Corresponden a las rutinas que permiten leer y tratar los datos que se obtienen directamente de los sensores. Incluyen manipulación de datos, procesamiento A/D, etc.
- *Entorno físico*: Corresponde al propio móvil y a los sensores que lleva incorporados.

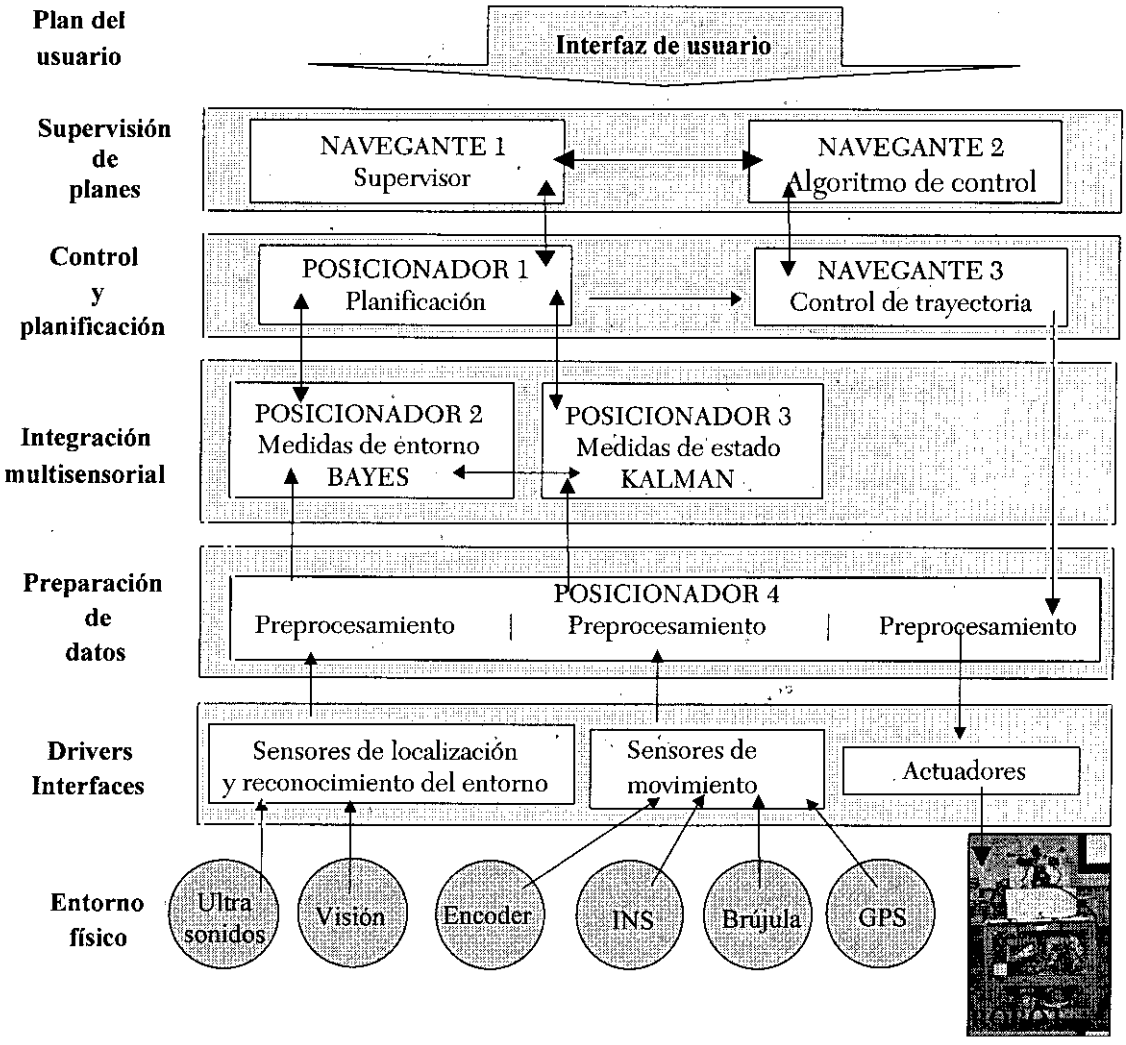


Figura 5-6 . Niveles de integración

5.2.2. Tareas del robot y modelo ROOM

En el esquema de la figura 5-6 se puede observar cómo fluye la información entre las distintas tareas o niveles de abstracción. Todas estas tareas pueden agruparse en distintos agentes que procesan independientemente la información que reciben y conducen a la consecución de un plan común: el funcionamiento correcto del robot y la consecución del objetivo marcado. A partir de este esquema y de la información obtenida en la sección 5.1 se ha optado por una división de las tareas del robot como la mostrada en la figura 5-7.

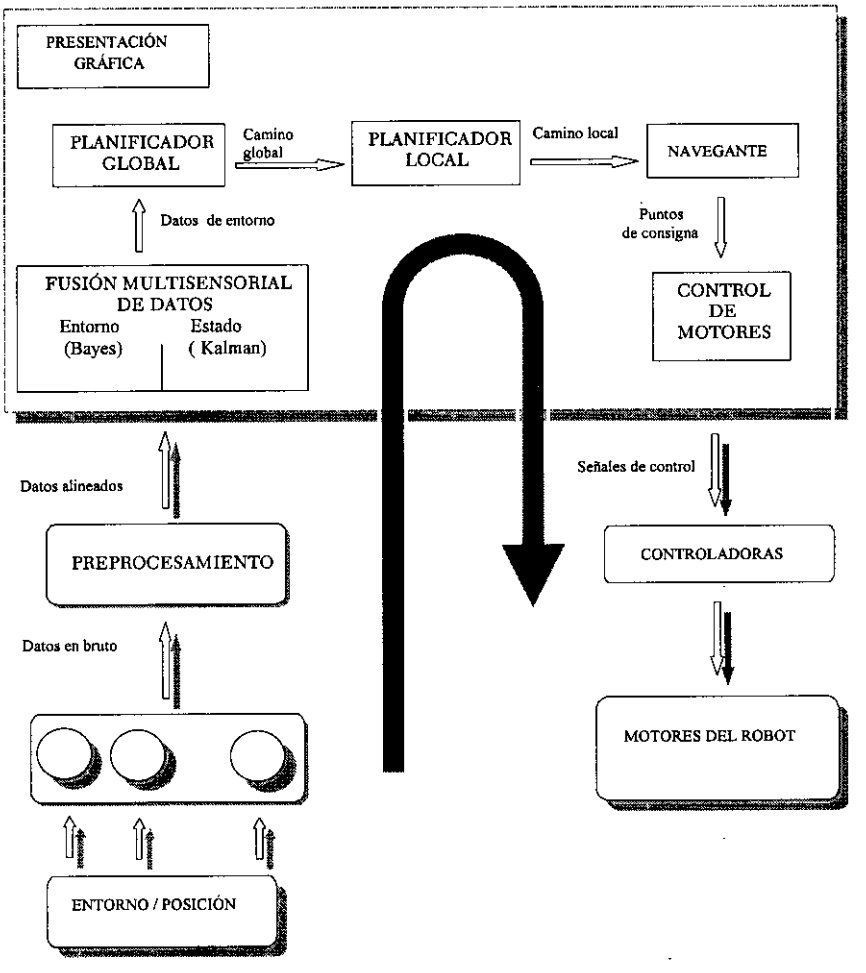


Figura 5-7. Tareas del robot y flujo de la información.

Una vez introducidos los niveles de tratamiento de la información por parte del robot en la sección 5.2.1 y la descomposición en tareas necesaria para ello en la sección 5.2.2, se está en disposición de especificar mediante la metodología

ROOM [Selic 95]. Así, se considerará al robot y a cada una de sus tareas como actores.

En la figura 5-8 se muestra su interfaz con el exterior. El actor mostrado en la figura, actor ROBOT, presenta una serie de puertos para comunicarse con el exterior y sólo podrá hacerlo a través de ellos. En esta figura cabe destacar que el actor ROBOT es independiente de los sensores que se utilicen, de la plataforma donde se ejecute (se ha utilizado C++ y la representación gráfica es un actor independiente del robot para evitar particularidades) y del método de conducción utilizado. Esto siempre es posible si los drivers correspondientes satisfacen el interfaz especificado.

El actor ROBOT de la figura 5-8 presenta 4 puertos externos que representan el interfaz entre el sistema de control del robot y el exterior y son:

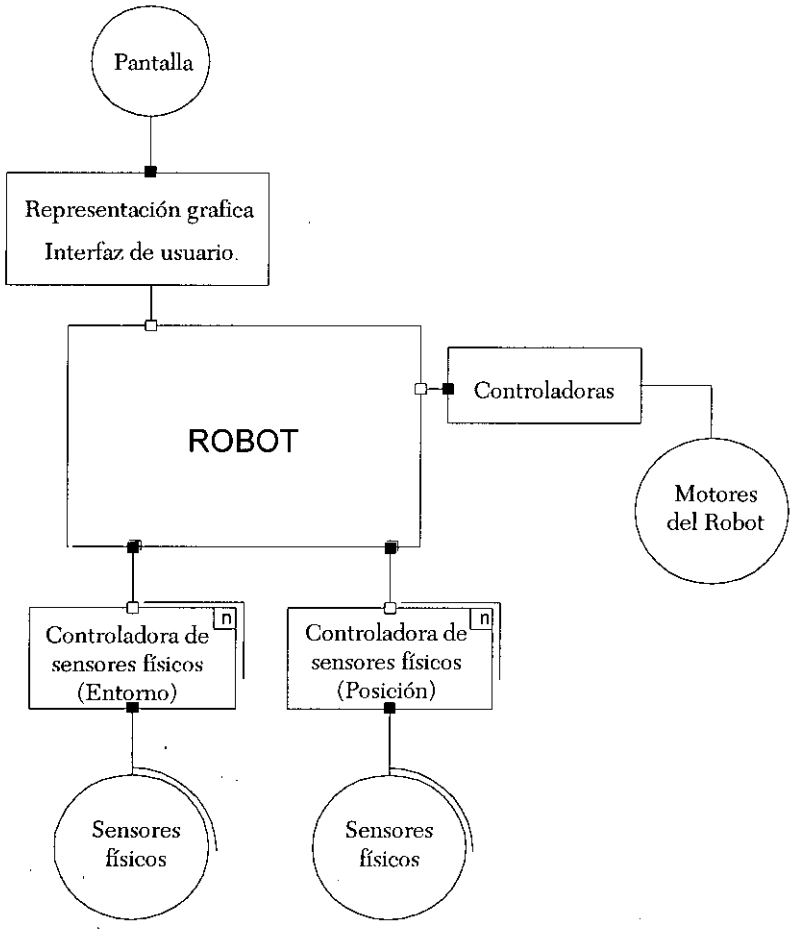


Figura 5-8. Relación del actor ROBOT con el sistema sensorial.

1.- Un puerto para conectar un actor gráfico, encargado de mostrar al usuario toda la información que los actores internos deseen mostrar y que envíen a ese puerto. De esta forma se abstrae el sistema de representación gráfica y el interfaz de usuario del sistema diseñado, haciéndolo más independiente de la máquina donde se ejecute.

2.- Un puerto que conecta al robot con un actor que maneja el sistema motriz del robot, de forma que un mismo diseño pueda ser utilizado por distintos robots con las mínimas modificaciones posibles.

3.- Distintos puertos que se conectan a los actores que actúan como interfaz de cada uno de los sensores utilizados. El número de puertos será variable y dependerá del número de sensores (de entorno y de estado) disponibles.

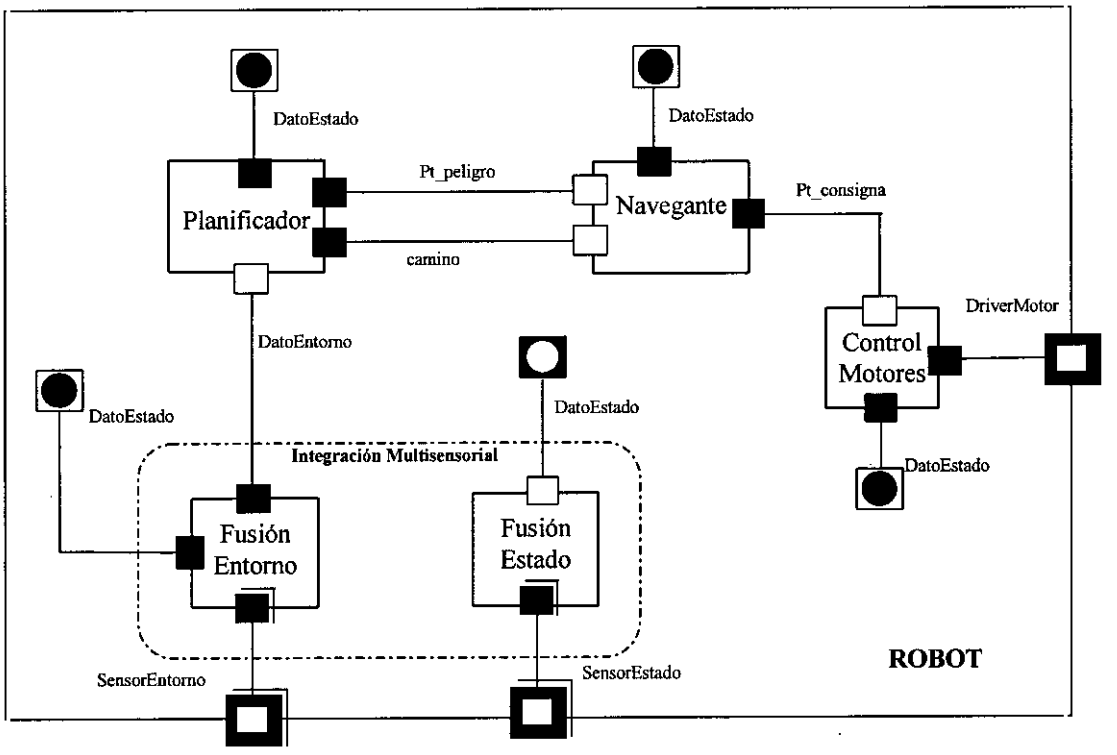


Figura 5-9. Interconexión entre los actores componentes del actor ROBOT.

Los actores que componen el actor ROBOT equivalen a los agentes mostrados en la figura 5-7. Una representación de la estructura de conexión entre ellos puede verse en la figura 5-9, donde conviene destacar los puertos externos correspondientes al interfaz que controla los motores del robot, *DriverMotor*, y los

correspondientes a los sensores de estado, *SensorEstado*, y de entorno, *SensorEntorno*. Estos dos puertos pueden replicarse en función del número de sensores que se conecten al robot. El puerto externo correspondiente a la salida gráfica se ha suprimido para simplificar el esquema.

De las conexiones entre actores de la figura 5-9, cabe destacar que el estado calculado por el módulo fusión de estado es enviado a todos los actores (por el puerto de comportamiento denominado *DatoEstado*) ya que todos necesitan conocer la posición del móvil. El puerto *Pt_peligro* permite enviar mensajes de alta prioridad desde el Planificador al Navegante para que decida que se debe hacer en caso de un objeto en la trayectoria calculada. Otros puertos son *DatoEntorno*, por el que circulan datos de objetos detectados para que el Planificador construya su modelo del mundo; *camino*, por el que se comunica al Navegante el mejor camino de acuerdo a la elección del usuario; y *Pt_consigna*, por el que circulan la consigna que el navegante envía al Control de los Motores para que guíe al robot por la trayectoria indicada.

5.3. Tareas en el control del robot móvil autónomo

5.3.1. Sistema sensorial

El sistema sensorial será totalmente distribuido, siendo independiente del número y tipo de sensores acoplados al robot. Esto permite que el móvil evolucione correctamente aunque alguno de los sensores falle.

Por supuesto, el sistema tendrá ciertas restricciones. En primer lugar deben distribuirse los sensores en dos tipos, como vimos en el Capítulo 2, según la información que son capaces de proporcionar: sensores de estado y sensores de entorno.

5.3.1.1. Sensores de estado

El sistema de fusión de medidas de posición será distribuido, y, como se mostró en el diseño realizado en el Capítulo 2, consta de dos clases claramente diferenciadas: la clase *SensorLógico*, que se encarga de tomar datos de los sensores físicos, procesarlos y transmitirlos al nodo encargado de fusionar su estima con la de otros sensores; y la clase de fusión de medidas de estado (en nuestro caso únicamente posición del móvil), *FusiónEstado*, que por medio de un filtro de Kalman distribuido (Capítulo 3), se ocupa de realizar la estima global correspondiente a la fusión de la estima de cada uno de los sensores.

En la figura 5-10, se muestra un esquema del funcionamiento y de la distribución de los actores correspondientes al sistema de fusión de medidas de estado. En ella se muestra el flujo de datos y la información transmitida. Como puede verse, los Sensores Lógicos envían la información procesada, *datosN*, a los actores encargados de su fusión, los cuales intercambian entre sí información correspondiente al estado estimado y a su error (eX y eP). Todos los nodos de fusión envían también a otro nodo, *distri*, su estima del estado, X, y su error, P, y reciben de éste la acción de control, u, que el robot a aplicado a los motores para que cada nodo la utilice en su modelo de estado. El nodo *distri* se encarga de seleccionar la mejor estima y enviarla al actor ROBOT para que la utilice convenientemente.

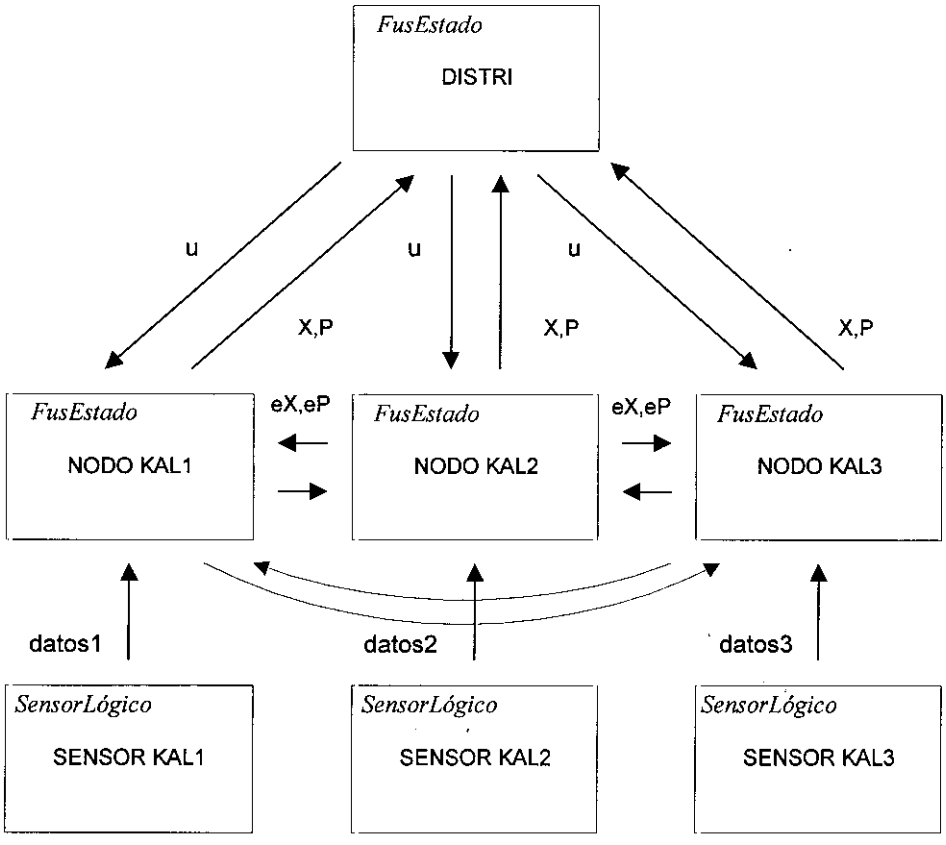


Figura 5-10 . Información transmitida en el sistema de fusión de estado.

Este esquema de funcionamiento se puede sintetizar en la clase actor *FusiónEstado*, figura 5-11, donde se puede ver que se compone de dos actores: *fusión* que pertenecen a la clase *FusEstado* y el actor *sensorFusion* de tipo *SensorFusion*. El actor *fusion* se encarga de recoger la estima de todos los sensores y comunicar aquella que sea "mejor" al planificador. Se entiende por mejor aquella estima de posición más precisa o más actual.

La clase *SensorFusion* se compone a su vez de dos actores, el actor *sensorlogico* correspondiente a la clase *SensorLógico* encargado de temporizar e interpretar las lecturas sobre el dispositivo físico y el actor *nodoKalman* perteneciente a la clase FusEstado, que recibe las estimas de los otros nodos y calcula las suyas a partir de éstas y de las medidas obtenidas. Para cada sensor utilizado se replica un actor del tipo *SensorFusion* para el tratamiento de sus datos. El actor *FusiónEstado* puede utilizar múltiples sensores de estado que pueden conectarse en el puerto externo *sensorEstado*.

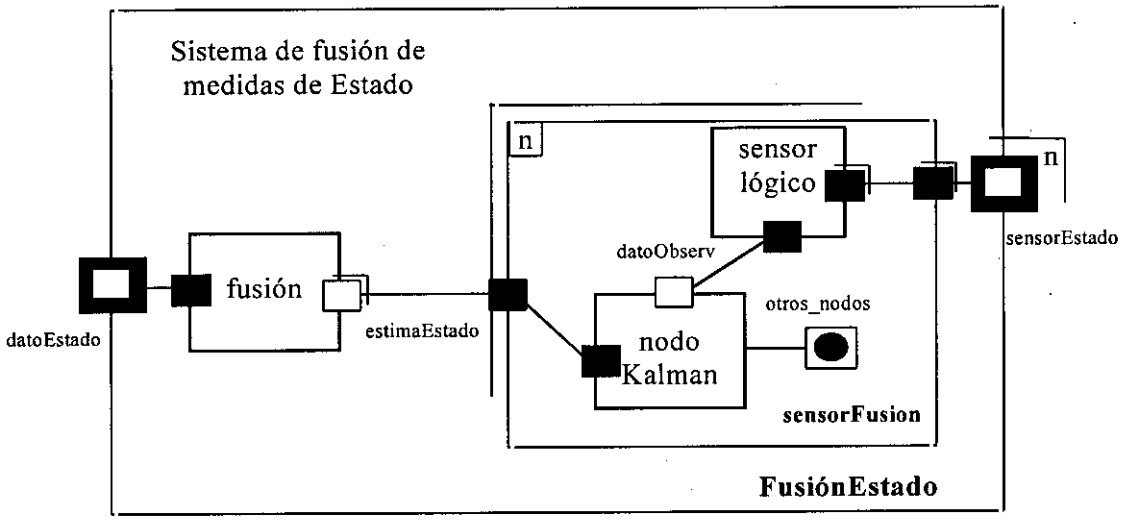


Figura 5-11. Clase actor *FusiónEstado*.

A continuación se muestra el protocolo correspondiente a los actores y puertos que componen el actor *FusiónEstado*.

```
Protocol class SensorKal_a_NodoKal
     In: { {mensaje_fin, null},{reset, tipo_estado} }
     Out: { {mensaje_misensor, dato_medido} }

Protocol class NodoKal_a_NodoKal
     In: { {mensajeXP, info_XP} }
     Out: { {mensajeXP, info_XP} }

Protocol class NodoKal_a_Fusion
     In: { {mensaje_u, u},{mensaje_fin, null},{reset,tipo_estado} }
     Out: { {medidasKal, info_nodo} }

Protocol class Fusion_a_Motor
     In: { {mensaje_fin, null}, {mensaje_u, u} }
     Out: { {nuevo_est, tipo_estado } }
```

Para entender mejor el funcionamiento de este actor a continuación se muestra el comportamiento de cada uno de sus actores componentes.

Así, en la figura 5-12, se muestra el diagrama de estados (denominado *ROOMchar*) que describe el funcionamiento del actor *sensorlógico*. Como se puede ver se encarga de pedir la información cada ciertos instantes marcados por un timer (que el usuario programa dependiendo del tipo de sensor que se utilice) y de procesar la información pedida antes de enviarla al nodo de fusión de Kalman.

En la figura 5-13, se muestra el diagrama de estados del actor de fusión (*nodoKalman*) en el que cabe destacar los pasos típicos en un filtro de Kalman distribuido, ya descritos en el Capítulo 3: predicción, actualización y asimilación.

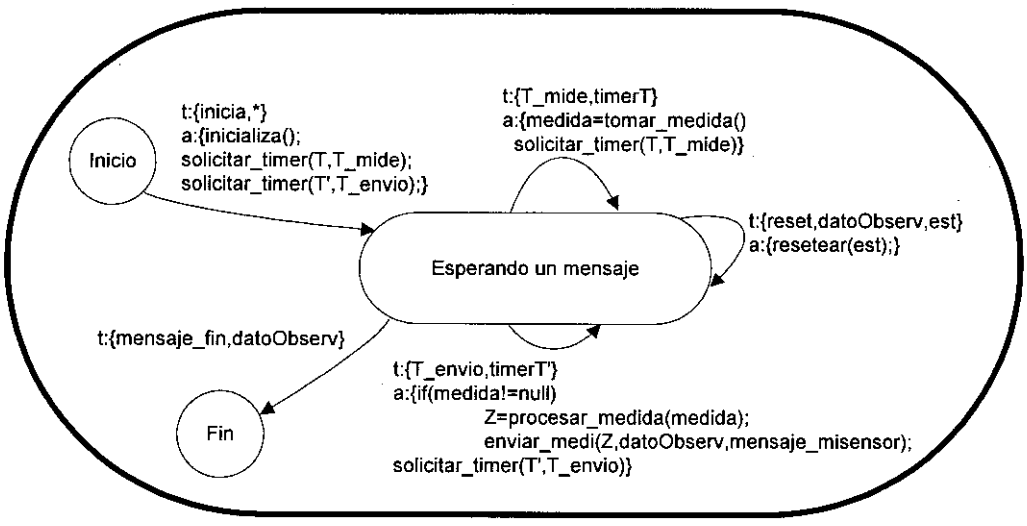


Figura 5-12. Comportamiento del actor *sensorlógico*.

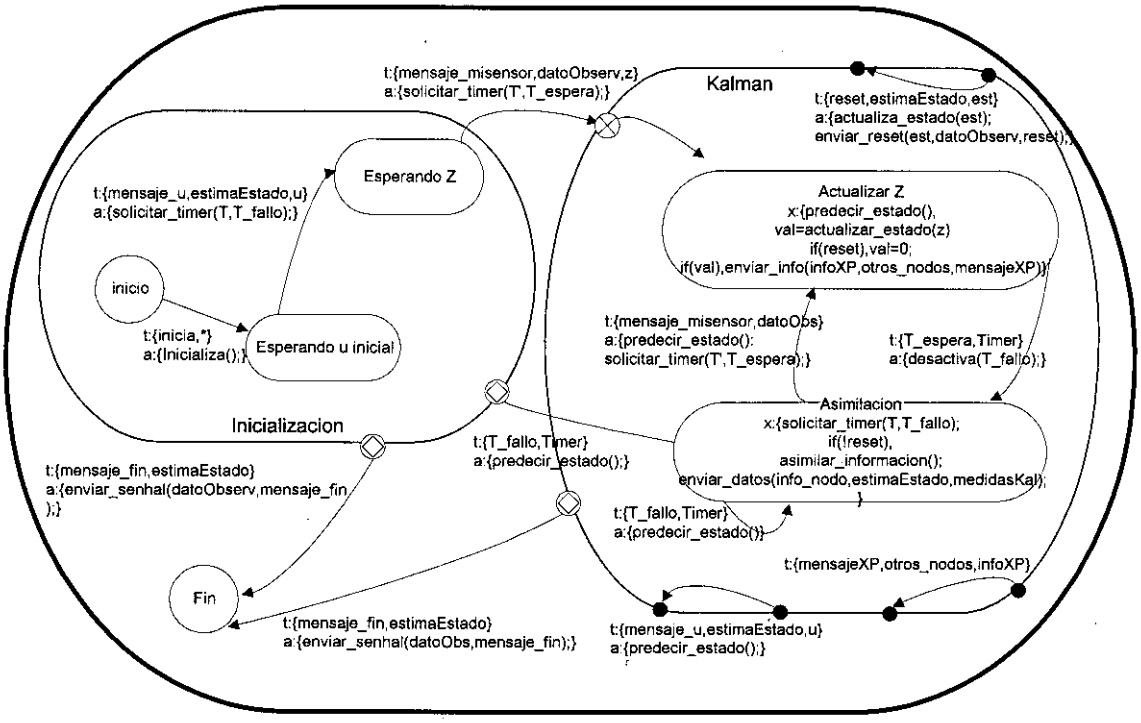


Figura 5-13. Comportamiento del actor nodo*Kalman*.

Finalmente, en la figura 5-14 se muestra el actor *fusion* que se encarga principalmente de recibir las estimas de los sensores y enviar al exterior la estima final para que sea utilizada por otros actores del actor ROBOT. Además se encarga de recibir la señal de control y de enviarla a todos los nodos de fusión. El comportamiento de este actor se podría incluir en el comportamiento de la clase actor *FusionEstado* pero por claridad se ha utilizado un actor distinto.

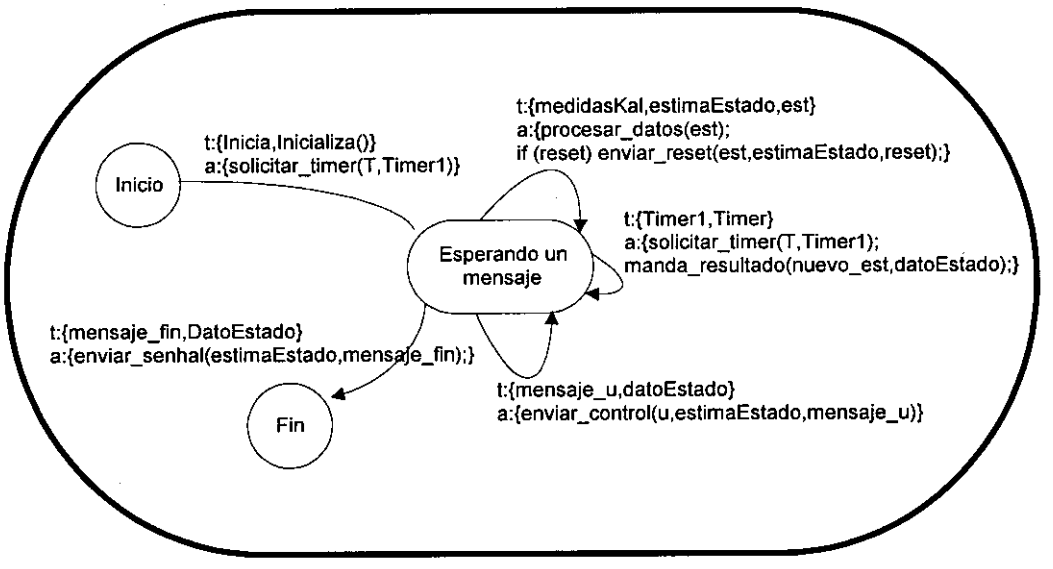


Figura 5-14. Comportamiento del actor *Fusion*.

5.3.1.2. Sensores de entorno

De igual modo que ocurre con los sensores de estado, los sensores de entorno se tratan de igual forma. La especificación de los actores como clases será a partir de la definición mostrada en el Capítulo 2. Utilizando su diseño y aplicando el algoritmo desarrollado en el Capítulo 4 se obtiene el sistema de fusión de medidas de entorno.

Un esquema sencillo donde se muestra el flujo de datos entre dos actores de un hipotético sistema de fusión de entorno puede verse en la figura 5-15.

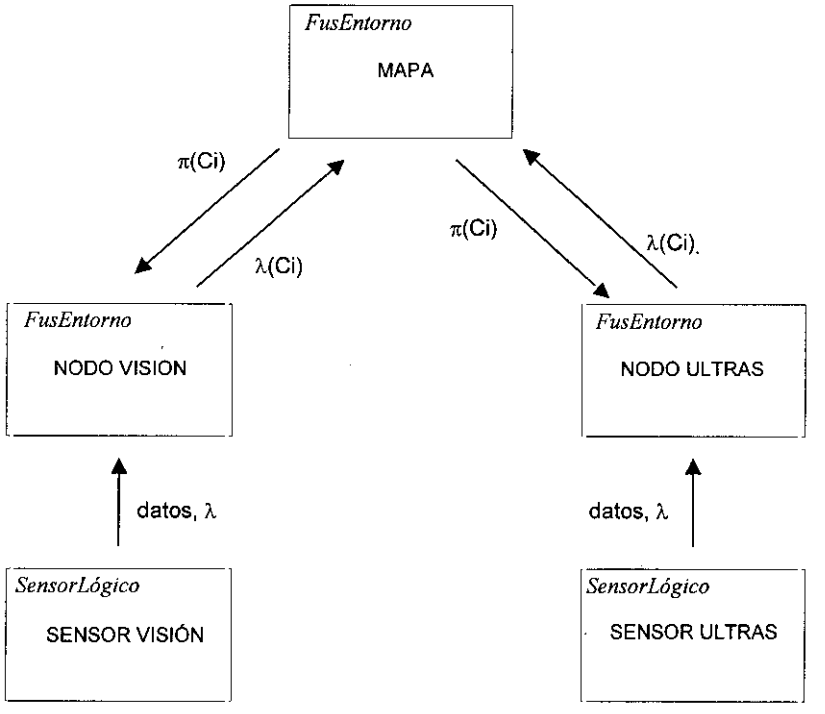


Figura 5-15. Esquema y flujo de información en el sistema de fusión de entorno.

El actor de fusión de medidas de entorno definido tiene la misma estructura que el actor de fusión de medidas de estado, véase la figura 5-16, pero el tratamiento que realiza sobre la información recibida es completamente distinto. Conviene señalar que el mensaje *FuncionModo* que se envía el Planificador al Nodo de fusión de Bayes hace referencia a las órdenes que envía, y de igual manera para los demás nodos y sensores lógicos, para modificar su funcionamiento, petición de estado, etc.

```
Protocol class SensorBay_a_NodoBay
     In: { {mensaje_fin, null},{FuncionModo,modo} }
     Out: { {nuevos_ptos, datos_sen} }

Protocol class NodoBay_a_FusionBay
     In: { {mensaje_fin, null}, {nueva_pos, pos}, {mensaje_bayes,
          datos_bayes},{FunciónModo,modo} }
     Out: { {datos_nodo, datos_celda} }

Protocol class FusionBay_a_Planificador
     In: { {mensaje_fin, null},{FuncionModo,modo} }
     Out: { {nuevas_celdas, celdas_N} }
```

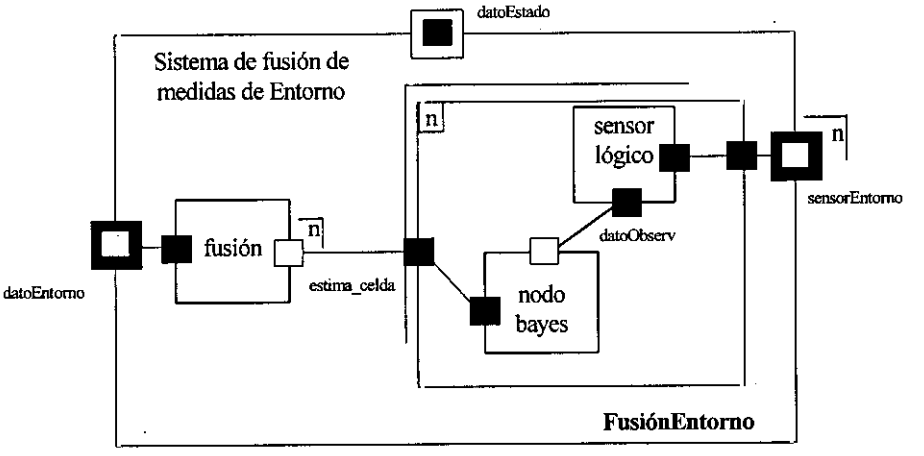


Figura 5-16. Actor *FusionEntorno*.

Cada uno de los sensores lógicos almacena su estima local en un mapa de ocupación local y todas las estimas globales se almacenan en un mapa de ocupación global que utiliza el Planificador para calcular una trayectoria segura.

Esto puede verse como una serie de planos transparentes que se van complementando para obtener una imagen completa y mejorada del entorno, véase la figura 5-17.

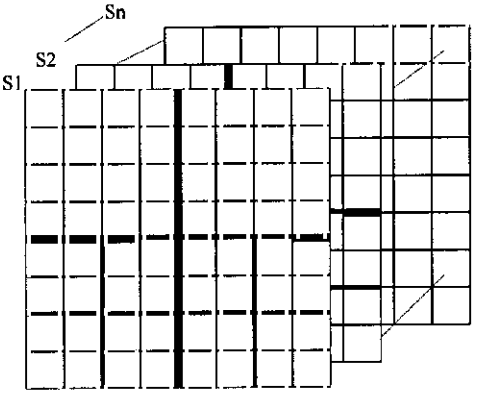


Figura 5-17 . Representación del entorno para cada sensor.

El comportamiento del sensor lógico de entorno (*sensorlogico*) es el mismo que en el caso anterior. En la figura 5-18 se observa que su función principal es la de tomar datos del sensor en ciertos instantes de tiempo y preparar esa información para que sea procesada en el nodo encargado de la fusión

(*nodoBayes*). Su modo de actuación vendrá dada por las órdenes que haya recibido mediante el mensaje *FuncionModo*.

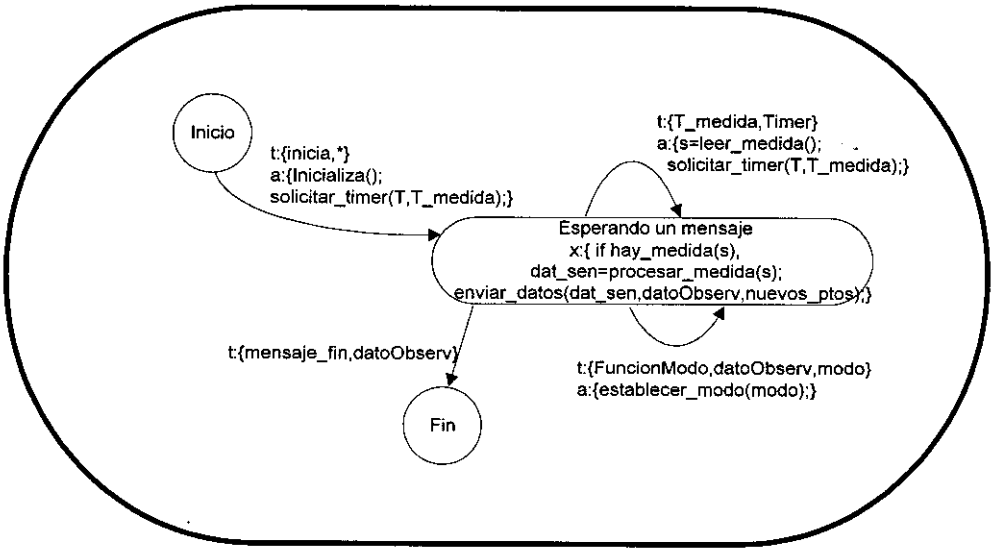


Figura 5-18. Comportamiento del actor *sensorlógico*.

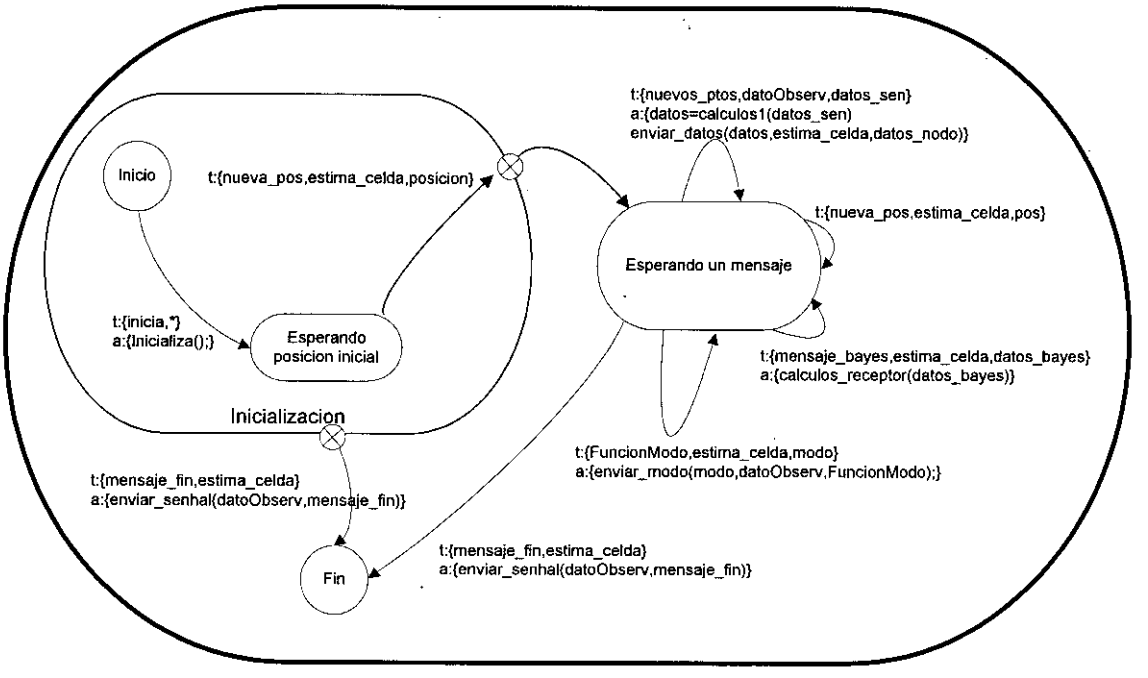


Figura 5-19. Comportamiento del actor *nodoBayes*.

Aunque los actores encargados de la fusión de medidas de entorno, poseen la misma estructura que los de estado, el método de fusión es completamente distinto: utiliza técnicas bayesianas. Por ello, el actor encargado de la fusión local de la información, *nodoBayes*, es de tipo *FusEntorno*. Este actor realiza una fusión de las medidas propias del sensor y construye un mapa local con la estima realizada como puede verse en el diagrama de su comportamiento de la figura 5-19.

El mapa global, donde se fusiona la estima de todos los sensores, se realiza en el actor *fusiónbayes*, véase figura 5-16 y la figura 5-20 para una descripción de su comportamiento. Este actor es el que mantiene la base de conocimiento del entorno y ofrece la información necesaria al Planificador. Una diferencia importante con el otro tipo de sistema de fusión, a parte del almacenamiento de un mapa del entorno, es la aplicación de un factor de olvido sobre las medidas de los sensores y sobre el conocimiento a priori del entorno.

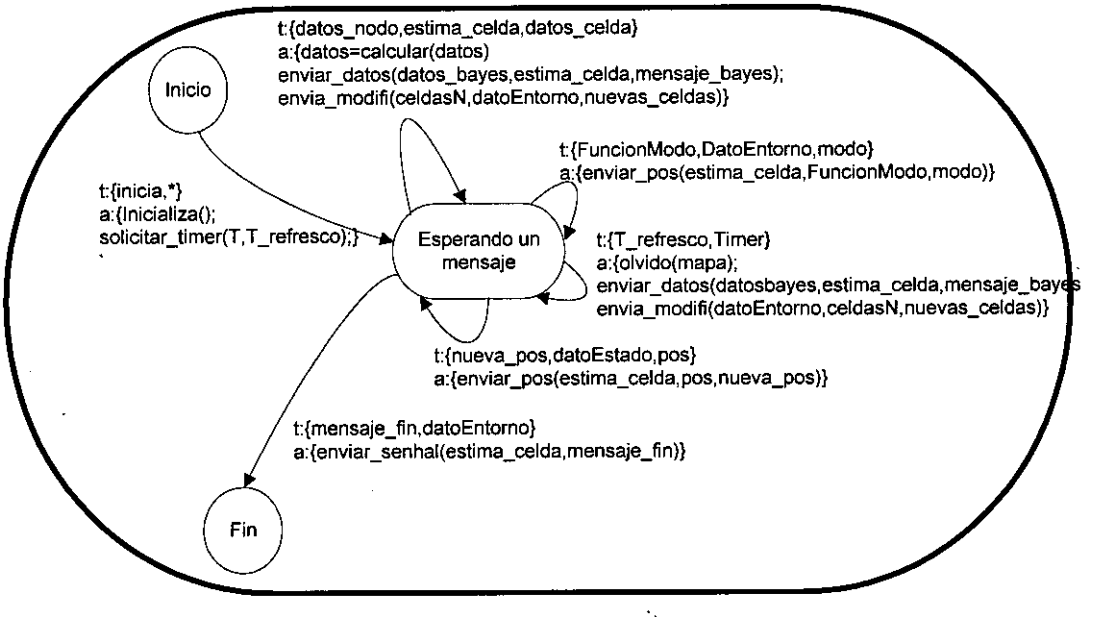


Figura 5-20. Comportamiento del actor *FusionBayes*.

5.3.2. Control de los motores

El control de los motores es una máquina lógica (actor) muy dependiente del robot móvil que se utilice. Esto se debe a que se dispone de distintos algoritmos y métodos de control dependiendo del tipo de robot utilizado. Principalmente el control depende del tipo de tracción motriz y ruedas guía disponibles, por ejemplo: cuatro ruedas (dos traseras motrices y dos delanteras guía) como en los

automóviles; tres ruedas (dos traseras motrices y una delantera guía), como los triciclos; dos ruedas motrices independientes, como en los tanques,.... Pero no sólo pueden utilizarse ruedas como elemento de tracción, sino que pueden utilizarse otros mecanismos como patas, cadenas, etc. Un análisis de la cinemática y distintas topologías de robots puede verse en [Kelly 94; Campion 96].

La arquitectura de control del robot está basada en un Planificador que se encarga de encontrar el camino más adecuado para lograr los objetivos fijados por el usuario. Así, el control de los motores se debe encargar de seguir los puntos de consigna de la forma más eficiente y fiel posible.

En nuestro caso los puntos de consigna son posiciones (x,y,ϑ) por las que se desea que pase el robot, y el control debe encargarse de dar las órdenes necesarias a las ruedas para que realicen el trayecto solicitado. Por lo tanto, primero será necesario calcular la trayectoria que el robot debe seguir para pasar por los puntos de consigna marcados por el Planificador. Así, tenemos dos tareas principales: generación de trayectoria y control de trayectoria.

5.3.2.1 Generación de la trayectoria

A partir de la posición actual del robot y de los puntos de consigna que el Planificador le manda al control de los motores, se calcula la trayectoria de referencia. Esto es, el conjunto de valores (x,y,ϑ) – posición y orientación – que debe seguir el robot en cada instante de muestreo. Esta trayectoria debe seguir lo más fielmente posible los puntos de consigna marcados, pero además debe cumplir unas serie de restricciones como que el movimiento del robot sea suave, sea físicamente realizable por el móvil o la minimización de ciertas variables como la energía consumida [Delingette 95], la curvatura de la trayectoria [Kanayama 96; Bissé 95], etc.

La solución a la generación de trayectoria dependerá del tipo de robot utilizado. La mínima restricción que debe cumplir es que sea continua, pero existen métodos para calcular una curva de ajuste C^1 o incluso C^2. Ejemplos de curvas de este tipo son las curvas de Bezier o los B-Splines, que pueden encontrarse, por ejemplo, en [Prieto 95].

En nuestro caso se calcula una trayectoria de referencia a partir de las consignas de posición generadas por el Navegante. La cinemática de nuestro robot, con dos ruedas motrices independientes, viene dada por la ecuación (5.1). Teniendo en cuenta esta ecuación, puede demostrarse que la expresión para la

curvatura, k, de cualquier trayectoria es la mostrada en la ecuación (5.2) [Rintanen 91]. Lo que indica que si la velocidad del robot se puede hacer tender a cero, el radio de curvatura, r, de la trayectoria puede hacerse cero. Es decir, la curva puede ser continua, C^0, con lo que el cálculo puede reducirse y hacerse muy sencillo.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\vartheta} \end{bmatrix} = \begin{bmatrix} \frac{V_{der} + V_{izq}}{2} \cos(\theta) \\ \frac{V_{der} + V_{izq}}{2} \operatorname{sen}(\theta) \\ \frac{V_{der} + V_{izq}}{d} \end{bmatrix} \tag{5.1}$$

$$k = \frac{\vartheta}{V} = \frac{1}{d} \cdot \frac{V_{der} - V_{izq}}{V} \quad ; \quad r = \frac{1}{k} \tag{5.2}$$

5.3.2.2. Control de trayectoria

Una vez que se dispone de la trayectoria de referencia es necesario garantizar que el robot sigue correctamente esa trayectoria, modificando las señales de control que se envían al robot (velocidades de las ruedas) para corregir cualquier desviación de la trayectoria.

Estas desviaciones pueden deberse a muchas causas como son imperfecciones en las ruedas, deslizamientos o patinado de las ruedas, imperfecciones en el terreno, etc.

Así el control de trayectoria será semejante a cualquier sistema de control, véase la figura 5-21, con un control (el módulo control de trayectoria), un sistema físico (robot), una señal de referencia (la trayectoria generada) y su lazo de realimentación.

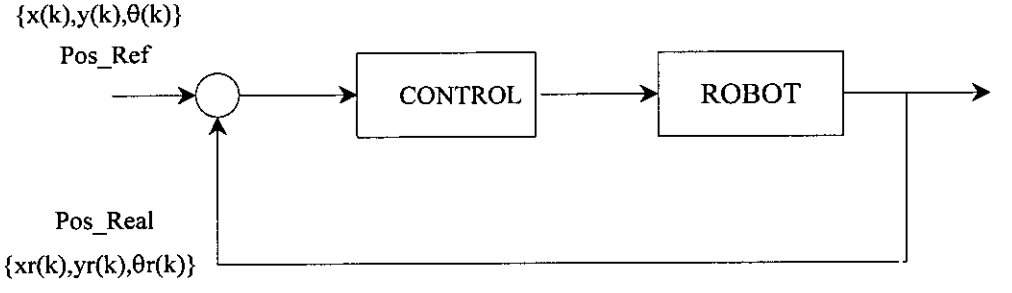


Figura 5-21. Esquema del control del robot.

Existen distintos tipos de control que pueden ser aplicados al robot para controlar el correcto seguimiento de una trayectoria de referencia, desde los más simples como PID, hasta adaptativos, predictivos, robustos, borrosos, etc. El control puede ser diseñado por el usuario y en nuestro caso se ha desarrollado un control proporcional a los errores en la trayectoria, cuya principal ventaja es que requiere muy poca carga de procesador y permite ser utilizado de forma directa en sistemas multitarea en tiempo real. Las ecuaciones (5.3) y (5.4) muestran las entradas de control a las ruedas del robot, donde f_1 y f_2 son la acción de control en cada instante.

$$V_{izq}^{Control} = V_{izq}^{Ref} + f_1(e_{tan}, e_{nor}, e_g) \tag{5.3}$$

$$V_{der}^{Control} = V_{der}^{Ref} + f_2(e_{tan}, e_{nor}, e_g) \tag{5.4}$$

La ley de control necesaria para calcular f_1 y f_2 se basa en minimizar el error cometido en la trayectoria del robot [Nelson 88]. Como se puede ver en la figura 5-22, el error se puede dividir en tres tipos:

- error de orientación, e_g, que se define como la diferencia entre la orientación de referencia del robot y la orientación real que tiene.
- error normal, e_n, que representa la distancia, perpendicular a la trayectoria de referencia, que separa la posición de referencia y la posición real del robot.
- error tangencial, e_t, se define como el error de seguimiento de la trayectoria, es decir, cuánto se aleja el robot de la posición de referencia si no existiese error normal ni error de orientación.

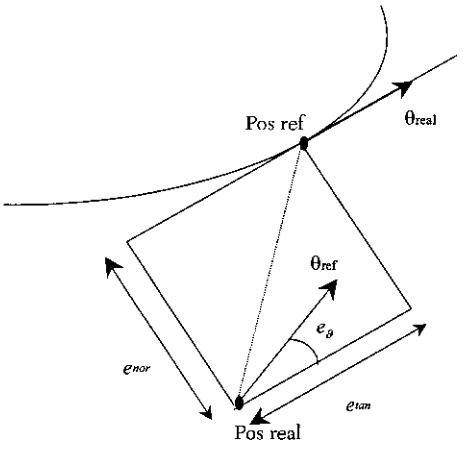


Figura 5-22. Errores de la trayectoria real con respecto a la definida.

Así se pueden obtener las expresiones (5.5), (5.5) y (5.7) para cada uno de los errores:

$$e_{\vartheta} = \vartheta_{\mathrm{Re}f} - \vartheta_{real} \tag{5.5}$$

$$e_{tan} = (X_{\mathrm{Re}f} - X_{real})\cos(\vartheta_{\mathrm{Re}f}) + (Y_{\mathrm{Re}f} - Y_{real})\mathrm{sen}(\vartheta_{\mathrm{Re}f}) \tag{5.6}$$

$$e_{nor} = -(X_{\mathrm{Re}f} - X_{real})\mathrm{sen}(\vartheta_{\mathrm{Re}f}) + (Y_{\mathrm{Re}f} - Y_{real})\cos(\vartheta_{\mathrm{Re}f}) \tag{5.7}$$

En la figura 5-23 un esquema del funcionamiento del control de los motores.

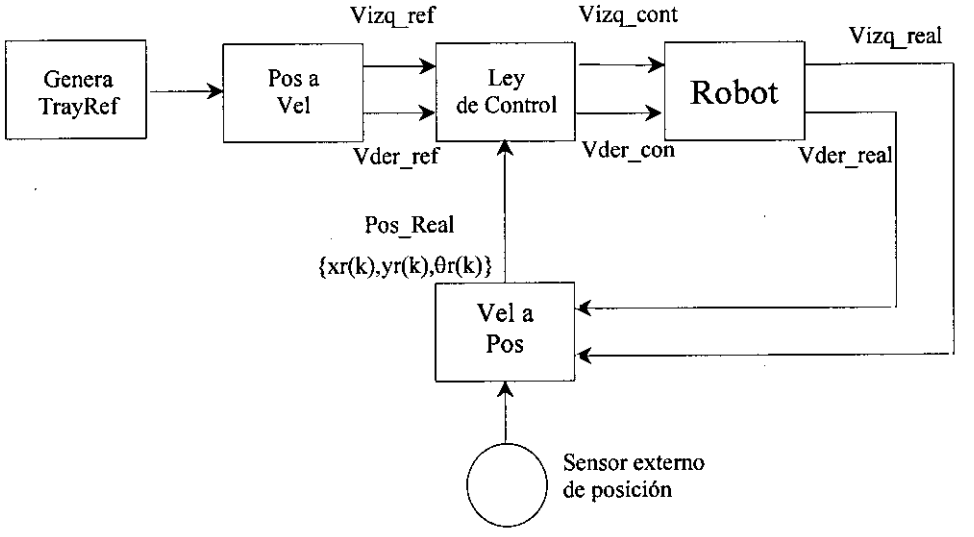


Figura 5-23. Esquema del control de los motores.

5.3.2.3. Modelo ROOM

Para realizar el control del robot se construye la clase actor *ControlMotores* cuyo interfaz y protocolo es el mostrado en la figura 5-24.

```
Protocol class Navegante_a_ControlMotor
     In: { {nueva_consigna, info_consigna}, {F_emergencia, null} }
     Out: {  }


Protocol class Fusion_a_Motor
     In: { {nuevo_est, tipo_estado } }
     Our: { {mensaje_fin, null}, {mensaje_u, u} }


Protocol class Motor_a_Driver
     In: { {lectura_señal, tipo_lectura } }
     Our: { {señal_control, tipo_control} }
```

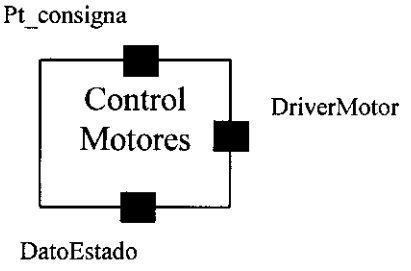


Figura 5-24. Clase actor *ControlMotores*.

El comportamiento de la clase se puede ver en su diagrama de comportamiento, figura 5-25. Destacar que si se desea cambiar el control y los cambios no afectan a la estructura del sistema, su incorporación es muy simple. Por ejemplo, para utilizar otro método de generación de trayectorias o un control diferente sólo es necesario modificar las rutinas *gen_la_tray()* y *control()* respectivamente, sin necesidad de modificar el comportamiento del actor ni definir una nueva clase de actor.

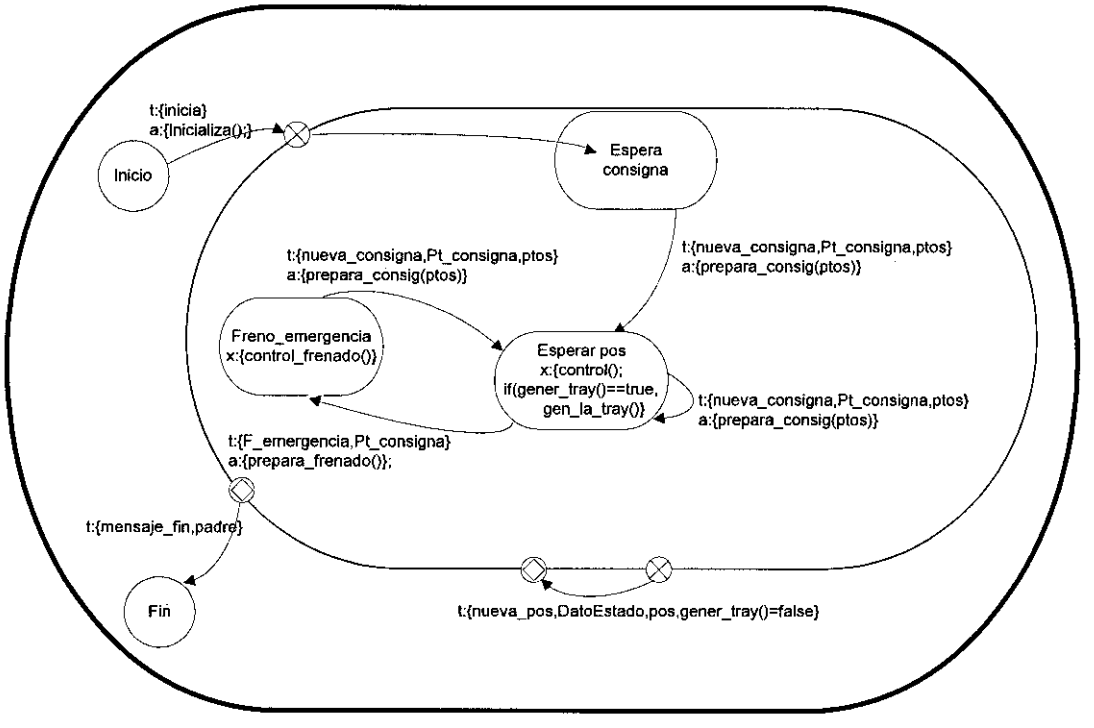


Figura 5-25. Diagrama de comportamiento del actor *ControlMotores*.

5.3.3. Planificador de caminos

En un vehículo autónomo, uno de los módulos clave es el Planificador de caminos. Su objetivo es calcular la trayectoria que se debe seguir para alcanzar la meta indicada por el usuario. Normalmente este Planificador procura encontrar el mejor camino con respecto a una cierta consigna, como podría ser el camino más corto, el más seguro, etc.

Otra función implícita en lo expuesto anteriormente es la función de esquiva de obstáculos, es decir, el Planificador debe comprobar que un objeto detectado se encuentra en la trayectoria inicialmente calculada y modificarla para evitar la colisión con ese obstáculo. Esta "vigilancia" de las modificaciones producidas en el entorno y el consecuente "recálculo" de la trayectoria deben ser lo suficientemente rápidos para poder realizarse en el tiempo de respuesta del móvil. Por lo tanto, la complejidad de los algoritmos no debe ser excesiva.

En este apartado podemos tratar diversos aspectos importantes:

1. *Representación local del entorno*; para poder desplazarnos con seguridad, lo primero que se necesita es construir una imagen del entorno que rodea al robot. Este entorno puede representarse de diversas formas, pero siempre de forma que pueda actualizarse dinámicamente, puesto que se construye a medida que se avanza y se descubren nuevos objetos.
2. *Creación del mapa de carreteras y trayectoria óptima*; siempre que se realiza un movimiento en cualquier entorno, es necesario hacer un estudio de a dónde queremos ir y cómo. Para ello se estudian las posibles trayectorias que nos llevan al destino, o subdestino, desde el punto en que se encuentra el móvil y se escoge el mejor camino entre todos los posibles, según uno o varios criterios especificados.
3. *Actualización del entorno y esquiva de objetos*; debido a que se considera al entorno dinámico, es necesario actualizar la representación del entorno que rodea al robot. Si la trayectoria escogida seguida ya no es válida, ya sea por un error en la estimación del tamaño de un objeto o por la aparición de un obstáculo en la trayectoria prefijada, será necesario recalcular una nueva trayectoria que permita salvar dicho escollo.

A continuación se describen con más detalle Veamos cada uno de estos aspectos de la planificación de caminos.

5.3.3.1. Representación local del entorno

El método de representación del entorno que rodea al robot depende de lo que nos interese guardar, es decir, de lo que consideremos significativo para la planificación del camino óptimo. En nuestro caso sólo nos interesa la existencia o no de *algo* en nuestro camino y los *puntos* por los que deberíamos pasar. Por tanto basta con indicar la zona del espacio, si está ocupada o no; y los puntos del recorrido a seguir. Teniendo esto en cuenta existen diversos métodos para representar el entorno por el que se puede mover el robot:

a.- Mediante un conjunto de nodos conectados entre si, por ejemplo en [Meng 93] donde los nodos representan corredores, intersecciones entre corredores, etc. y en su interior se guarda la información relevante para el movimiento (puertas, tablones,...).

b.- Como regiones poligonales que rodean a todos los objetos. Esta representación es la apropiada para el uso de los diagramas de Voronoi [Edelsbrunner 87; Aurenhammer 91].

c.- Dividir la región del espacio en celdas o cuadrículas. Esta división de celdas puede, según el punto de vista del que se hable, clasificarse en: exacta o aproximada; o en, iguales o desiguales. Por división en celdas iguales indicamos que la celda tiene un objeto o no, por ejemplo en [Gat 90; Elfes 90], pero no puede contener a la vez parte de un objeto y zonas vacías. Por el contrario la división aproximada permite la existencia de celdas que no están completamente vacías o llenas, a estas celdas se las denominan celdas mixtas. Cuando no son iguales, división desigual, se procura que cada celda contenga el mayor espacio posible del mismo tipo, ya sea vacío o lleno.

En nuestro caso se utiliza una división del espacio en celdas desiguales, cuadradas y de tamaño variable.

La división en celdas desiguales se debe a que si una zona del espacio está ocupada o vacía no es necesario que se divida en partes más pequeñas que aportarán la misma información. La descomposición en celdas cuadradas es exclusivamente para simplificar su comparación con las celdas del mapa global (mapa de ocupación). Además de optimizar el espacio ocupado en memoria tomando celdas de diversos tamaños, el tamaño de estas celdas será variable para

realizar un procesamiento del espacio que rodea al robot lo más rápido posible, aumentando la división de las celdas, y por tanto la precisión, sólo cuando sea necesario.

Esta división se conoce como "quadtree" si es en el plano u "octtree" en 3 dimensiones [Jackins 83; Chaudhuri 85]. Quadtree es un tipo de descomposición del espacio que consiste en la división de una región del espacio en cuatro partes iguales, las partes resultantes que están completamente vacías o llenas no se dividen más; pero si es una celda mixta, volvemos a aplicar quadtree a ésta recursívamente hasta que dejen de ser mixtas o alcancen un tamaño mínimo determinado que correspondería a la mejor precisión utilizada.

Esta técnica de descomposición del espacio en rectángulos es un caso particular de la *descomposición en arbol-2^m* , donde m es la dimensión del espacio. Así, el espacio se representa por un árbol de grado 2^m (cada nodo que no sea un nodo hoja, tiene 2^m hijos). Cada nodo del grafo será del tipo Lleno, Vacío o Mixto. Sólo las celdas Mixtas tendrán hijos en un número igual a 2^m. [Latombe 91]. Si la descomposición se hace en dos dimensiones (m=2), el algoritmo se llama quadtree, en caso de una descomposición tridimensional (m=3), el algoritmo se denomina octtree. Ejemplos y algoritmos para la utilización de quadtree y octree aplicados a gráficos, procesamiento de imágenes, GIS (sistemas de información geográfica), etc. pueden encontrarse en [Samet 89].

El almacenamiento en un árbol de las celdas de distinto tamaño permite ahorrar memoria (los datos referentes a cada celda por parte de los sensores también se guardarán en los nodos del árbol) puesto que si las celdas contiguas son iguales la información se engloba en una celda de mayor tamaño. Además la consulta de cualquier zona del espacio será relativamente rápida, siendo su complejidad $O(n \log n)$ [Mitchell 88], donde n es el número de celdas (de un lado) de esa zona. De igual forma, cualquier operación sobre una zona determinada será más rápida que cuando se trabaja sobre elementos individuales. Por ejemplo, la consulta del estado de una celda o zona concreta será del $o(\log n)$ y la obtención del contorno de los objetos de $o(n)$ [Samet 81; Jackins 83].

La profundidad del árbol indica la resolución de la división del espacio. En nuestro caso cada celda constará de la siguiente información:

- Coordenadas X,Y de la esquina superior izquierda.
- Dimensión del lado X y del lado Y.

- Etiqueta de la celda (Vacio, Lleno o Mixto)
- Información adicional (tipo de objeto).

El árbol utilizado es un multiárbol [Langsam 97], como puede verse en la figura 5-26. Y, como puede observarse, se permite la existencia de nodos hoja con la etiqueta Mixto. Esto se hace para impedir una división del espacio excesiva si no es necesaria, lo que permite al vehículo desplazarse con mayor velocidad.

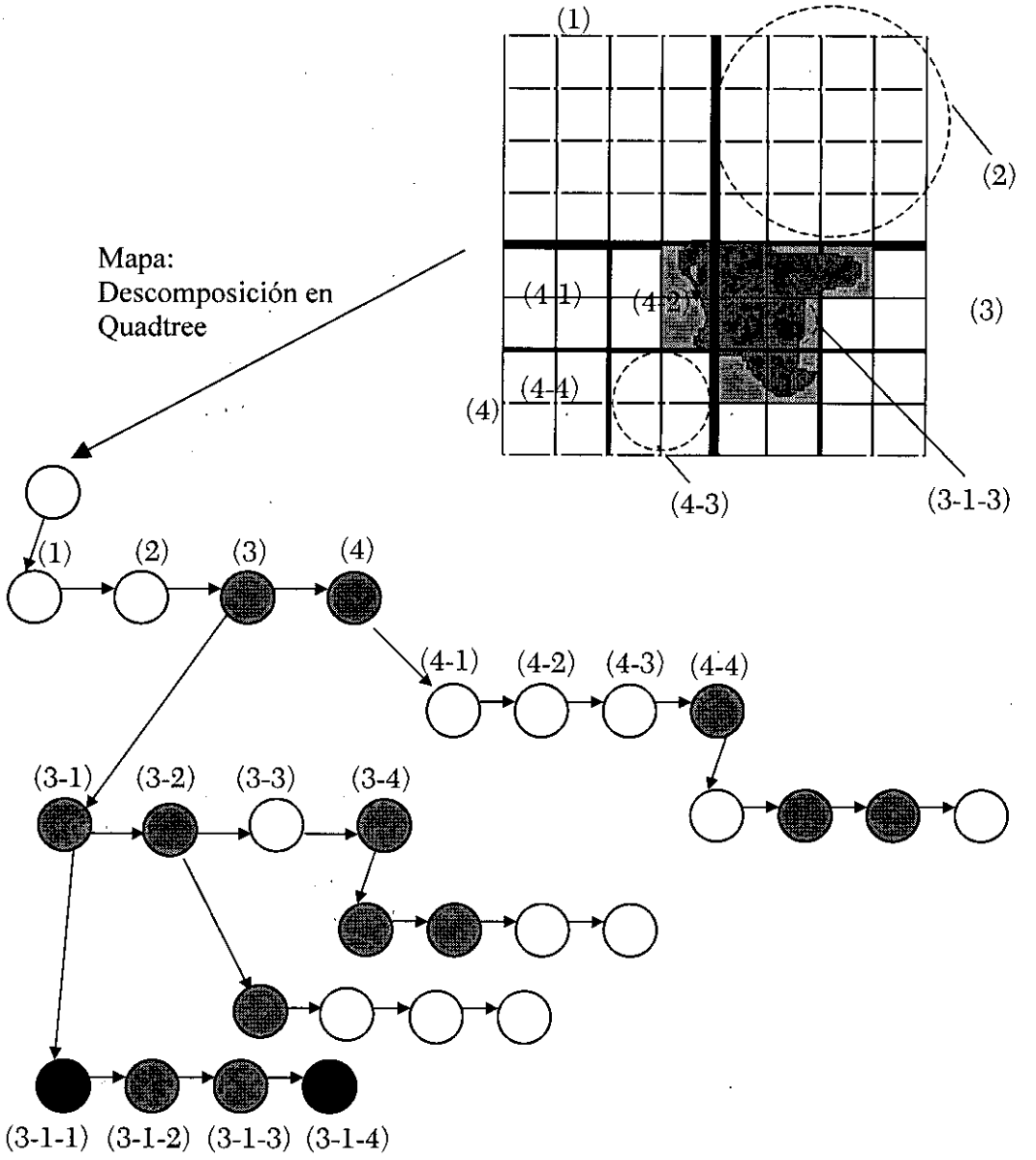


Figura 5-26. Aplicación de la representación "quadtree" al entorno del robot.

En caso de aproximarnos a una zona mixta se puede dividir más dicha zona (expansión de una rama del árbol) para comprobar si pasa sin chocar con el obstáculo, mejorando de forma dinámica la trayectoria seguida. Esto puede verse en la figura 5-27. Como podemos ver la trayectoria se irá calculando automáticamente ajustándose a los nuevos nodos de los objetos observados hasta ese instante. Con este método también podemos esquivar a la vez los obstáculos no previstos en la trayectoria inicial.

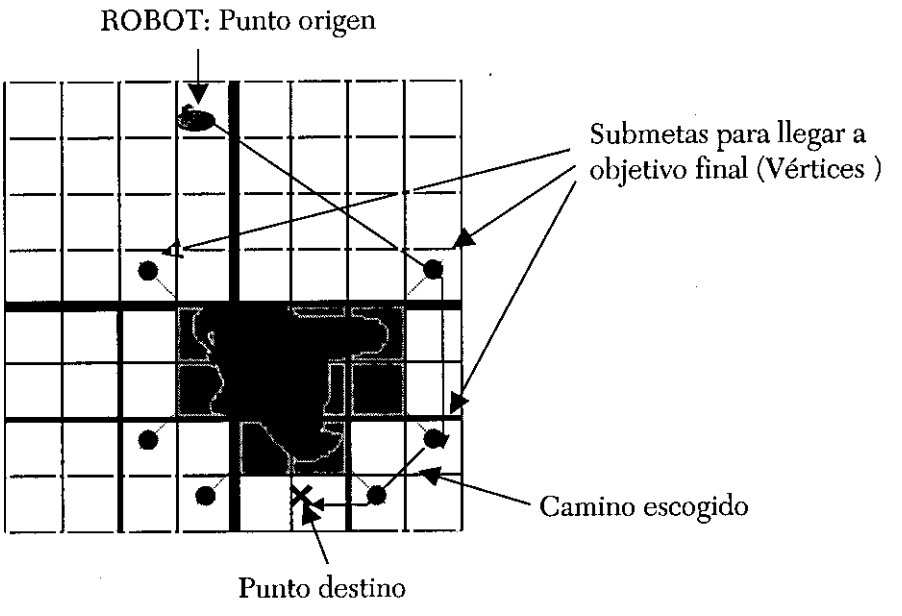


Figura 5-27. Cálculo de la trayectoria inicial y la trayectoria seguida.

5.3.3.2. Creación del mapa de carreteras y trayectoria óptima

Para calcular el camino óptimo se encuentran todos lo posibles caminos como concatenación de una serie de subcaminos o tramos que unen zonas del espacio libre. El camino cuyos tramos cumplan una cierta condición será el escogido.

Existen varios métodos para calcular estos caminos posibles [Latombe 91], entre otros destacamos el grafo de visibilidad, el diagrama de Voronoi, la red de espacios libres y el campo de potenciales.

El grafo de visibilidad no utiliza una descomposición en celdas, sino que calcula todos los caminos (arcos del grafo) entre un vértice de un objeto y cualquier otro (nodos del grafo) utilizado normalmente para obstáculos poliédricos [Lozano-Pérez 79; Mitchell 88]. Los caminos válidos son aquellos que no encuentran ningún obstáculo en su camino.

Otro método diferente, que tampoco utiliza una descomposición de celdas, es el diagrama de Voronoi. Este diagrama es el conjunto de todas las configuraciones vacías cuya distancia mínima al obstáculo es alcanzada al menos por dos puntos de la frontera del obstáculo. La ventaja de este diagrama es que el camino libre tiende a maximizar el espacio libre entre el robot y los obstáculos. Cuando los obstáculos son polígonos el diagrama de Voronoi consta únicamente de segmentos rectos y parabólicos.

La red de espacios libres se aplica a una descomposición en celdas. Existen diversos métodos dependiendo de la descomposición utilizada, pero en general representa un grafo de conectividad que indica si dos celdas libres son contiguas. Así, por ejemplo, si tenemos un descomposición en celdas triangulares y trapezoidales el grafo de conectividad se creará entre celdas adyacentes y el camino se realizará entre el punto medio de dos celdas sucesivas.

Los campos de potenciales [Khatib 86] son técnicas heurísticas locales muy utilizadas. Normalmente se aplican sobre un enrejado de celdas iguales (aunque puede ser continua y no necesitar la división de celdas) y se basan en dotar a cada celda de un potencial (función potencial) basado en la proximidad de los objetos y de su destino. Así, el destino produce un término que tiende a empujar el robot hacia él (potencial atractivo) y los obstáculos tienden a producir un potencial que repele al robot (potencial repulsivo). El camino será aquel que siga un potencial mínimo. Esta técnica tiene el problema de que puede caerse en mínimos locales de los cuales el robot no podrá salir, para evitarlo se utilizan distintas modificaciones como la adición de ruido [Arkin 87].

Nuestro mapa de carreteras esta formado por un grafo de visibilidad, pero en lugar de utilizar los vértices de los objetos se utilizarán los vértices de las celdas Llenas (o en su defecto hojas Mixtas). Para disminuir el número de nodos del gráfo se eliminarán los nodos correspondientes a los vértices cóncavos, puesto que en el cálculo del camino más corto el robot nunca pasará por ellos. Esto permite que el número de nodos del grafo sea significativamente menor que si se considerase un nodo para cada una de las celdas. La distancia entre vértices es "real" y no como ocurre en la red de espacios libres que se calcula la distancia entre los centros de las celdas, que obviamente produce un camino más largo.

Existen multitud de parámetros que pueden optimizarse en el cálculo de la trayectoria, como son el combustible, el tiempo, el espacio, la seguridad, etc. Pero normalmente no todos pueden satisfacerse a la vez. Por ejemplo, si queremos

optimizar el tiempo en llegar al destino, necesariamente debemos utilizar un tamaño de celdas grande para reducir el número de cálculos. En este caso, la seguridad peligra, y obtendremos una trayectoria que no será la más corta (en cuanto a espacio recorrido).

En nuestro ejemplo el tipo de trayectoria que nos interesa es: *optimizar el espacio recorrido*, pero manteniendo la seguridad de que no ocurrirá ninguna colisión, disminuyendo la velocidad del móvil en caso de peligro.

Para encontrar el camino mínimo en el grafo se puede utilizar cualquier algoritmo de optimización existente, como es la programación dinámica [Larson 76], algoritmos genéticos [Goldberg 89; Davis 91; Xiao 97], o algoritmos clásicos de recorrido mínimo en un grafo, etc. En nuestro caso utilizaremos el algoritmo de camino mínimo en un grafo de Dijistra.

En una primera aproximación calcularemos la matriz de distancias, necesaria en el algoritmo de Dijistra, utilizando la distancia euclidea entre los nodos, pero podrían añadirse otros factores. La coordenada de cada vértice se calcula según la ecuación (5.8), donde Xvertice e Yvertice, son la posición del vértice del obstáculo, más en concreto la esquina de la celda correspondiente; Xrobot e Yrobot, son la dimensión del robot; sentidoX, valdrá 1 si es un vértice hacia la derecha (como el de la figura 5-28) o -1 si es un vértice hacia la izquierda; y sentidoY, valdrá 1 si es un vértice hacia abajo o -1 si es un vértice hacia arriba (como el de la figura 5-28).

$$\begin{aligned} \text{Xnodo} &= \text{Xvertice} + \text{sentidoX} * \text{Xrobot} / 2 \\ \text{Ynodo} &= \text{Yvertice} + \text{sentidoY} * \text{Yrobot} / 2 \end{aligned} \tag{5.8}$$

$$D(i,j) = \sqrt{(Xnodo_i - Xnodo_j)^2 + (Ynodo_i - Ynodo_j)^2} \tag{5.9}$$

Puesto que estamos utilizando un grafo de visibilidad, la distancia entre dos vértices i y j viene dada por la ecuación (5.9) si son visibles entre sí. En caso contrario la distancia es infinita. Para comprobar si existe algún obstáculo en la trayectoria recta entre los dos nodos se calcula la recta que los une y se comprueba que ningún punto de esta trayectoria pasa por una celda Mixta o Llena.

Una vez creada la matriz de distancias y el vector de nodos, se procede a aplicar Dijistra al grafo existente de modo que se obtenga la trayectoria más corta. Esta trayectoria será la utilizada inicialmente por el robot para comenzar a moverse, pero puede ser modificada durante el movimiento por la información adicional ofrecida por los sensores que pueden indicar la existencia de obstáculos

no previstos. Incluso aunque no aparezcan obstáculos, puede ser necesario la mayor división de celdas Mixtas debido a su proximidad con el robot, con la consiguiente modificación de los nodos involucrados en la división de las celdas y la modificación de la trayectoria a seguir. Para mantener una coherencia entre las trayectorias, se debe mantener la división de las celdas y la trayectoria que ya se haya indicado al controlador de los motores. Esto indica que no se debe permitir dividir las celdas que estén a una cierta proximidad del robot y que la división de las celdas debe ser de las celdas posteriores a estas, lo que implica la necesidad de una anticipación de la división de las celdas por las que discurre el camino dependiendo de la proximidad de éste a los obstáculos.

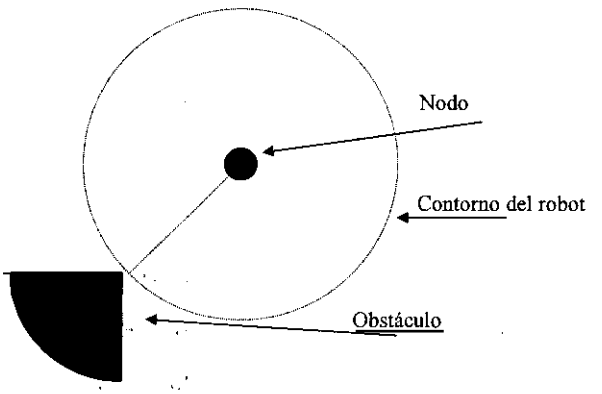


Figura 5-28. Posición de los nodos del grafo.

Para calcular la trayectoria óptima se incluyen dos vértices más: el punto donde se encuentra el robot y el punto destino, entre los cuales hay que calcular el camino más corto.

Una vez calculada la trayectoria inicial se pasa al controlador de los motores los puntos de consigna que debe seguir. Las coordenadas de estos puntos son calculadas a partir de las rectas existentes entre cada nodo y vendrán espaciados según el convenio entre el planificador y el control de los motores.

El planificador siempre ofrecerá una consigna al navegante o al control de los motores correspondiente a una distancia por delante de la trayectoria a seguir que depende de la velocidad del móvil de forma que el planificador pueda responder a la aparición de un obstáculo imprevisto. Esta trayectoria será utilizada por el motor

para controlar la velocidad y orientación de las ruedas para realizar lo más fielmente posible la trayectoria marcada.

5.3.3.3. Actualización del entorno y esquiva de objetos

El entorno viene representado por el árbol señalado anteriormente. En él se integran las medidas de la fusión de todos los sensores por lo que se dispone de una información completa del espacio próximo al robot.

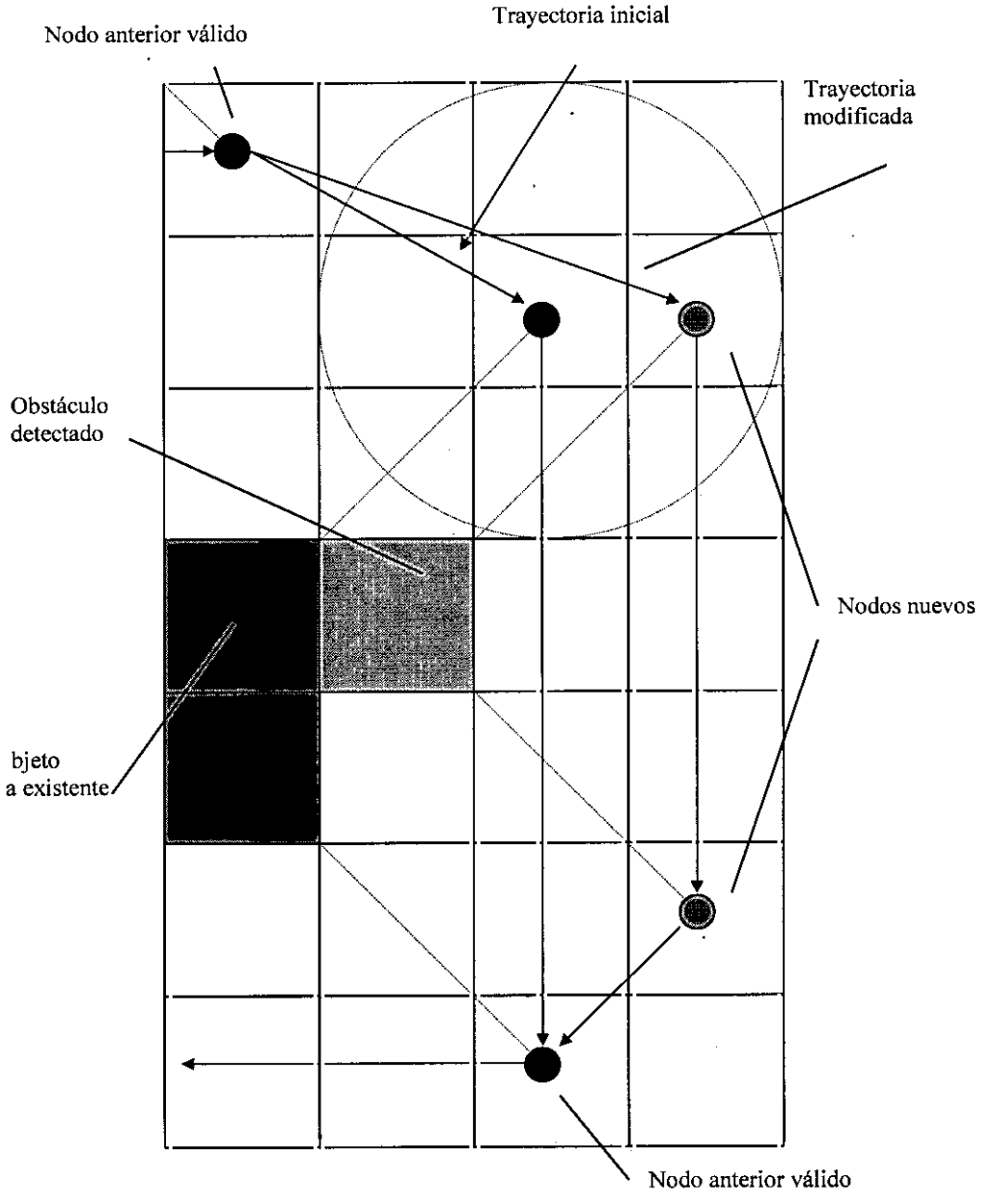


Figura 5-29. Cálculo de la nueva trayectoria

Cualquier modificación del entorno percibida por los sensores es notificada al Planificador, lo que permite detectar la aparición de obstáculos no previstos en la trayectoria inicial. Tengamos en cuenta que la aparición de obstáculos se traduce en la "transformación" de una celda Vacía en Mixta (o Llena). Si estas nuevas celdas "no vacías" no afectan a la trayectoria del robot no será necesario calcular un nuevo camino.

En caso de que el obstáculo obstruya el camino, se debe calcular una nueva trayectoria. El cálculo de la nueva trayectoria será solamente sobre la parte de la trayectoria que permite rodear o desviarse del obstáculo. Este cálculo es relativamente rápido puesto que utiliza únicamente un número muy reducido de nodos, correspondientes a la zona de seguridad preestablecida. Así podemos obtener la trayectoria en tiempo real sin prejuicio del movimiento del robot. En el nuevo cálculo de la trayectoria se utilizan como nodo inicial y nodo final los dos primeros nodos fuera de la zona de seguridad y como nodos intermedios para la nueva trayectoria los nuevos nodos que se encuentran en la zona local. La figura 5-29 muestra un pequeño ejemplo del funcionamiento de este mecanismo.

Tengamos en cuenta que para que el mecanismo funcione es imprescindible una buena actualización del entorno. Esta actualización es realizada por el módulo encargado de la integración multisensorial. Este es el módulo básico del robot, sin él no podría percibir el entorno en el que se mueve y sobre el que trabaja el planificador de caminos.

5.3.3.4. Modelo ROOM

El planificador de caminos se divide en dos actores, figura 5-30, encargados de tratar con entornos distintos: uno es el encargado de planificar una trayectoria sobre el entorno global cumpliendo las consignas y restricciones dadas por el usuario (*planificador Global*); el otro se ocupa de trabajar con el entorno próximo al robot para detectar posibles obstáculos y esquivarlos actuando de forma reactiva (*planificador Local*). Este planificador es el equivalente al Navegante 1 indicado en la sección 5.1.

```
Protocol class PlanG_a_PlanL
     In: { {nuev_CamG, null},{mapa_Local,null} }
     Out: { {mapa_local, celdas,},{nuevo_camino,tipo_camino} }

Protocol class Fusion_a_Motor
     In: { {mensaje_fin, null}, {mensaje_u, u} }
     Out: { {nuevo_est, tipo_estado } }

Protocol class Planificador_Peligro
     In: { {Peligro, ptos_peligro} }
     Out: { {Peligro_real, null}, }

Protocol class Planificador_camino
     In: { {nuevo_cam_glob, null}, {nuev_mapa_centrado, null},
          {nuevo_cam_local,null} }
     Out: { {mensaje_camino, tipo_camino },
          {No_camino_local,null},{fuera_mapa_local,null} }
```

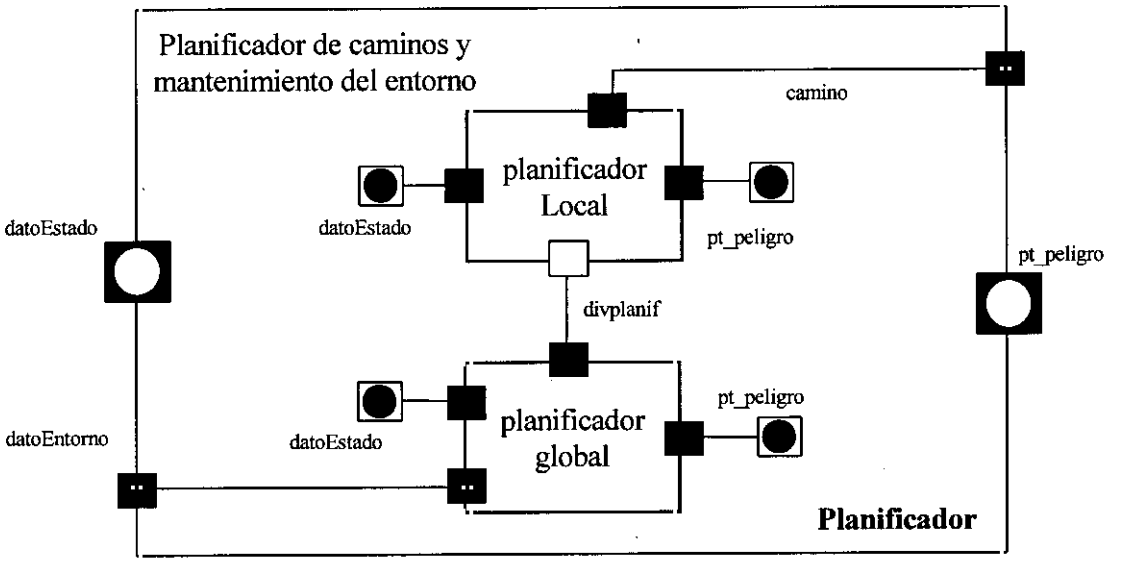
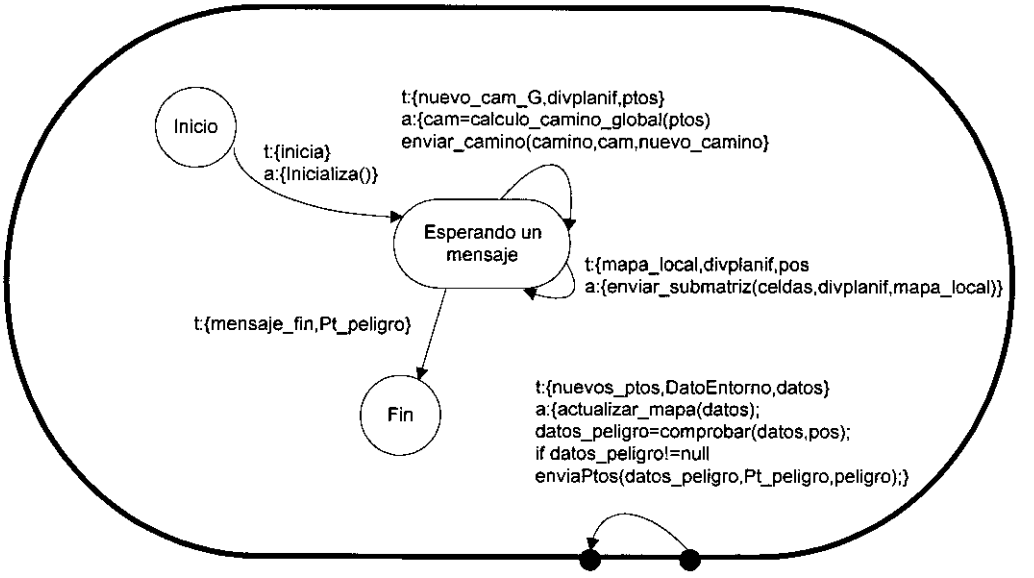
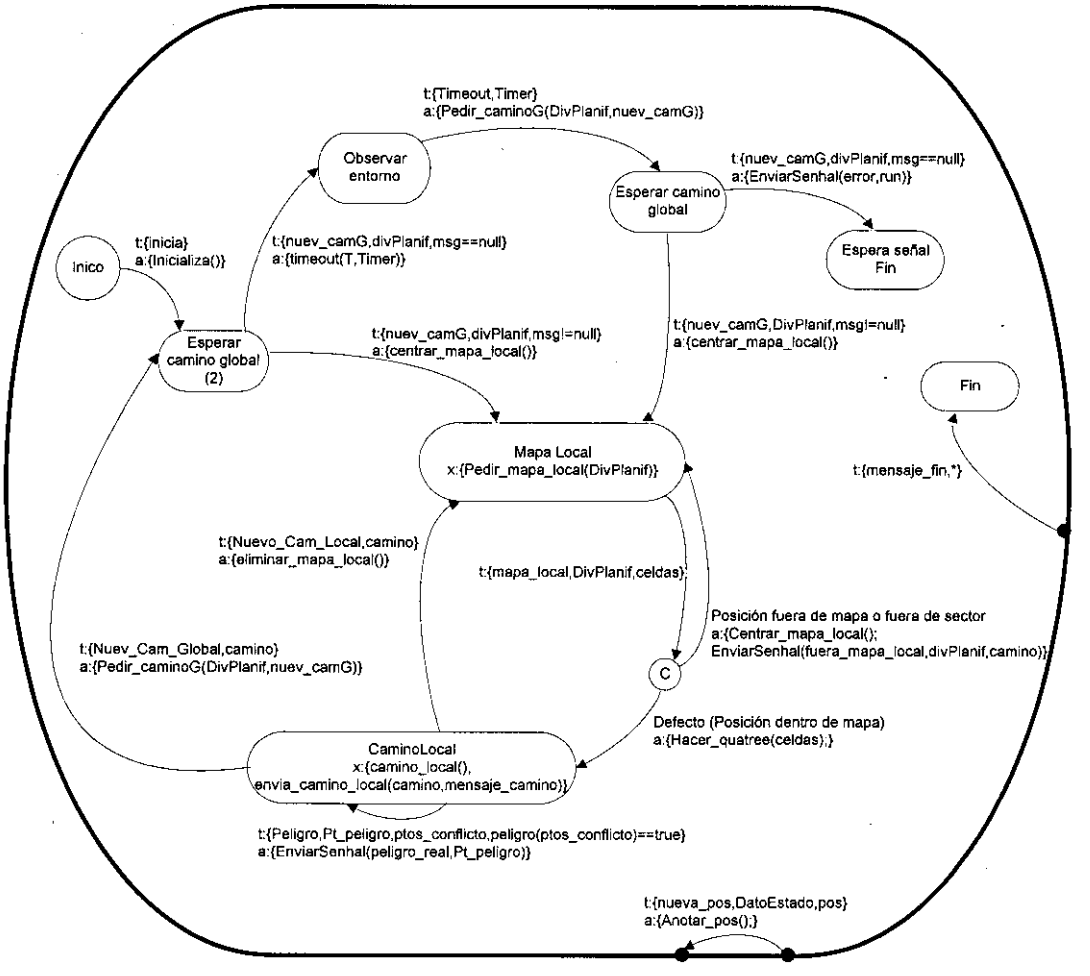


Figura 5-30. Clase actor *Planificador*.

En la figura 5-31 se muestran los diagrama de estado de cada uno de los planificadores utilizados. En la figura 5-31a se muestra cómo actúa el planificador global, pero se ha omitido la actualización dinámica de la consigna por parte del usuario, de forma que sólo es posible introducir una consigna inicial. En la figura 5-31b se muestra el funcionamiento del planificador local, su diálogo con el planificador global y con el navegante, y cómo se representa el entorno local mediante un quadtree.



a) Planificador global



b) Planificador local

Figura 5-31. Comportamiento del Planificador.

5.3.4. El navegante

Este actor se encarga de comprobar que la trayectoria fijada por el Planificador es seguida correctamente por el móvil. Es decir, se ocupa de supervisar el trayecto realizado por el Control de los Motores, de forma que si se desvía de la trayectoria más de un cierto margen, modifica los puntos de consigna dados al Control para rectificar la desviación observada. En caso de que exista un peligro inminente detiene el movimiento.

Este agente, además de recibir los puntos de consigna del Planificador, comprueba si existe peligro de colisión con los nuevos objetos detectados, recibe las restricciones del usuario y comunica la consigna a seguir al control de los motores.

Veamos con más detalle el funcionamiento del Navegante.

1.- Se encarga de procesar las restricciones en el movimiento por parte del usuario, informando al Planificador de cualquier cambio en el tipo de movimiento (creación de mapas, vagar por el entorno, puntos inicial y final, etc.).

2.- Se ocupa de la seguridad del movimiento del robot. Para ello se definen varias zonas dentro del mapa local, véase la figura 5-32:

- Circulo de seguridad. Zona donde se vigila que el movimiento del robot sea seguro e instante donde se debe pedir un nuevo mapa local.
- Margen de maniobra. Zona en la cual se permiten errores en el movimiento, pero cuyo movimiento es totalmente seguro puesto que se vigila que en ese "pasillo" no se encuentra ningún objeto.

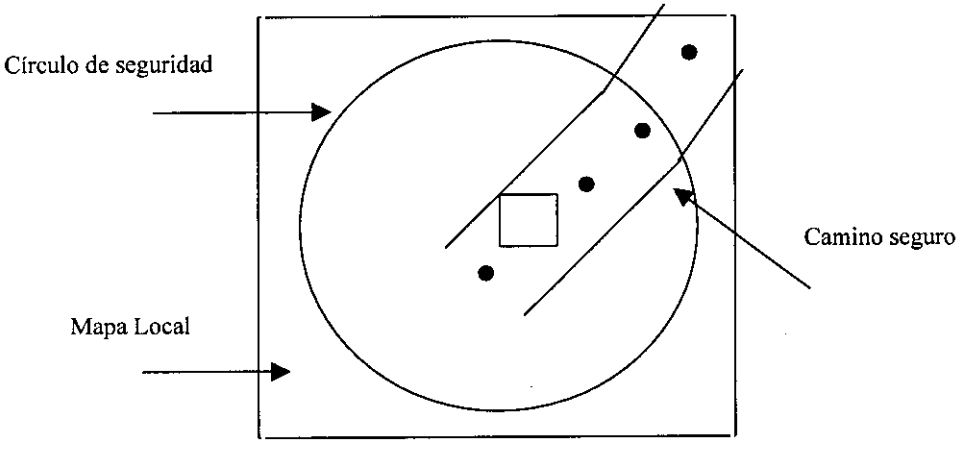


Figura 5-32. Mapa local y zona de seguridad

5.3.4.1. Modelo ROOM

El actor Navegante se ocupa de comprobar que la trayectoria es seguida correctamente. El protocolo de sus puertos ya han sido definido anteriormente en el planificador y Control de motores y será conjugado de estos. El protocolo y la clase pueden verse en la figura 5-33.

```
Protocol class Navegante_a_ControlMotor
     In: { {nueva_consigna, info_consigna}, {F_emergencia, null} }
     Out: {  }
Protocol class Planificador_Peligro
     In: { {Peligro, ptos_peligro} }
     Out: { {Peligro_real, null}, }

Protocol class Planificador_camino
     In: { {nuevo_cam_glob, null}, {nuev_mapa_centrado, null},
          {nuevo_cam_local,null} }
     Out: { {mensaje_camino, tipo_camino },
          {No_camino_local,null},{fuera_mapa_local,null} }
```

Figura 5-33. Actor Navegante

El comportamiento de este actor puede verse en la figura 5-34.

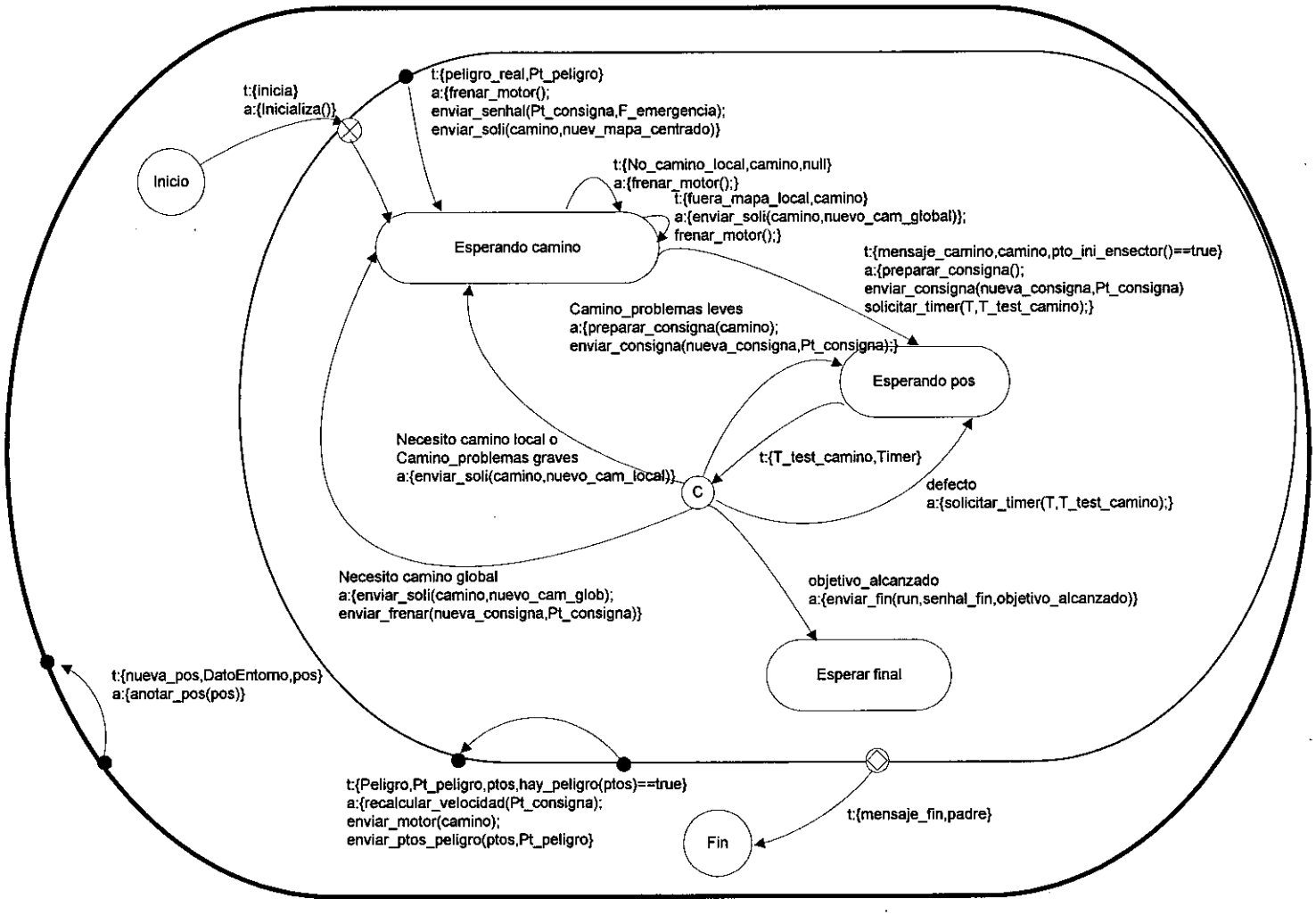


Figura 5-34. Comportamiento del actor *Navegante*.

5.4. Simulación del funcionamiento de los actores

La simulación del sistema permite repetir situaciones que en la realidad serían muy difíciles de reproducir. Con estas experiencias se puede poner de manifiesto la adecuación y robustez de las técnicas descritas en capítulos anteriores. Además, mediante simulaciones se pueden reproducir fallos en los sensores que no podrían realizarse de forma sencilla en un sistema real.

La simulación de los distintos actores permite analizar el comportamiento de los procesos en paralelo, ver cómo se comportan en el paso de mensajes, colaboran entre sí y actúan en condiciones de fallo o error, ya sea debido a un fallo de uno de los sensores o ante errores de comunicación.

Para implementar el sistema de control del robot se ha utilizado PVM (*Parallel Virtual Machine*) como herramienta de implementación y depuración de los distintos procesos involucrados. Esta herramienta permite crear, manejar y comunicar procesos de forma muy sencilla proporcionando las librerías necesarias para su utilización en C++. Su fácil utilización y su interfaz gráfico (XPVM) para observar los procesos y estudiar el correcto funcionamiento del sistema permiten un desarrollo rápido de cualquier sistema con procesos trabajando en paralelo.

A partir del diseño anterior, simulado con PVM, se adaptan los procesos para trabajar con un robot real, que se mostrará en el Capítulo 6. Se ha utilizado RTKernel como sistema operativo de tiempo real que trabaja sobre el sistema operativo MS-DOS. Este sistema operativo en tiempo real es de bajo precio y permite la utilización de PC's para el control de los robots, siendo una alternativa al Linux como plataforma de trabajo.

Las simulaciones que se presentan a continuación muestran el funcionamiento del sistema de integración multisensorial dentro del sistema robótico desarrollado, por lo que se hará hincapié en los procesos existentes y en el funcionamiento global del robot, ya que los resultados del proceso de integración de medidas ya se pusieron de manifiesto en los capítulos 3 y 4. Éstas incluyen:

- Fallo de distintos sensores, para mostrar que el sistema sigue funcionando correctamente y no se detiene.
- Simulación de las tareas del robot junto con el sistema de fusión, para mostrar cómo el sistema de fusión se integra correctamente en el funcionamiento de un robot móvil.

5.4.1. Fallo de distintos sensores

En este apartado se muestra el funcionamiento de los distintos procesos que componen el sistema de fusión multisensorial. En el capítulo 3 se estudió el funcionamiento de los sensores de posición y la respuesta del sistema de fusión ante fallos. Se mostró que el sistema era robusto ante malfuncionamientos de los sensores y se mostraron los resultados del sistema de fusión de estado y la covarianza del error resultante.

Ahora se muestran los mismos ejemplos que en el capítulo 3 pero desde el punto de vista de los procesos involucrados. De esta forma se destaca que el fallo de un sensor no detiene el sistema sino que éste evoluciona correctamente como si no existiese. De igual forma se puede incluir un nuevo sensor o hacer que un sensor deje o vuelva a trabajar con los demás.

Para mostrar el funcionamiento de los distintos procesos involucrados en el sistema sensorial se ha utilizado PVM (*Parallel Virtual Machine*) [Geist 94]. PVM fue desarrollado en 1989 por Vaidy Sunderam y Al Geist en el curso de sus investigaciones sobre computación distribuida hetereogénea, que les condujo al desarrollo del concepto de máquina virtual paralela [Sunderam 90; Geist 92]. Se ha utilizado PVM en lugar de MPI (*Message Passing Interface*) [MPI94; Snir 95] porque PVM es más adecuado cuando las aplicaciones se van a ejecutar sobre redes heterogéneas, lo que ocurre en nuestro caso puesto que debe adaptarse a los distintos equipos disponibles. Esto es posible debido al concepto de máquina virtual, que le permite desarrollar aplicaciones totalmente transportables e independientes de la o las máquinas sobre las que se van a ejecutar. Además tiene otras ventajas que le hacen muy adecuado como son su conjunto de funciones de control y manejo de procesos que permiten que las aplicaciones puedan ser utilizables tanto en estaciones de trabajo como en MPP (*Massively Parallel Processor*), su tolerancia a fallos que permite escribir aplicaciones PVM que pueden continuar ejecutándose cuando un *host* o una tarea fallan o permitir cambiar la carga de trabajo dinámicamente debido a situaciones externas. Para una mayor discusión sobre la conveniencia de la utilización de PVM o MPI véase [Geist 96].

Para la presentación gráfica de esta sección utilizaremos XPVM. XPVM es un interfaz gráfico para PVM. Proporciona un interfaz visual para las funciones de consola de PVM y es una herramienta que permite ver la evolución de los procesos en tiempo real y realizar una depuración o análisis de cualquier programa PVM. Al

igual que PVM, es de dominio público y puede obtenerse por Internet en diversas direcciones. En [Kohl 96] se puede encontrar más información sobre su uso, instalación, etc.

XPVM permite mostrar distintos aspectos de la ejecución de los procesos, como es la estructura de la red, la carga de la red, el estado de las tareas en el tiempo, los mensajes en cola de cada tarea, etc. En nuestro caso nos interesa mostrar el funcionamiento de las tareas y el paso de mensajes entre ellas, por lo que sólo se mostrará la ventana de ejecución de las tareas en función del tiempo. En esta pantalla cada tarea se muestra por una barra horizontal, siendo el eje x el tiempo (común para todos). El color de la barra indica el estado en que se encuentra esa tarea. El color correspondiente a "*Computing*", véase la figura 5-35a, muestra cuando la tarea está realizando cálculos reales, es decir, se encuentra en estado de ejecución o preparada; el color de "*Overhead*" indica que la tarea en cuestión está ejecutando rutinas para comunicación y control de tareas del sistema PVM; el color de "*Waiting*" indica cuándo la tarea está suspendida, por ejemplo si se encuentra en espera de mensajes de otros procesos. Las líneas entre dos barras representan los mensajes intercambiados entre esas tareas. Cada mensaje viene representado por una línea que parte de una barra (mensaje enviado) hasta el instante donde lo recibe la otra tarea (mensaje recibido).

5.4.1.1. Simulación de sensores de estado

En la figura 5-35a se muestran los mensajes intercambiados entre los sensores de medidas de estado cuando no ocurre ninguna anomalía, es decir, en el caso de un funcionamiento correcto de los sensores. En la figura 5-35b se muestra ampliado un ciclo de medida. En este se destacan dos zonas, la zona A y la zona B. La zona A corresponde a la lectura de los sensores físicos, a la predicción y actualización de la estima, al envío de la información de asimilación a los demás sensores y la preparación para su asimilación. Es la encargada de realizar la observación de los sensores y preparase para la asimilación de la información de los demás. La zona B corresponde a la asimilación y al envío de la mejor estima de cada uno de los sensores individuales al sistema de fusión. Y éste, a su vez, comunicará al robot la mejor estima de todas (sería la lectura del estado actual por parte del robot). Es decir, equivale al ciclo de lectura del sensor: en el se realiza la asimilación de las medidas recibidas por cada sensor y su envío al sistema externo que escoge la estima más precisa.

En la figura 5-35c se muestra una ampliación de la zona A, para destacar las distintas fases de cada ciclo de medida:

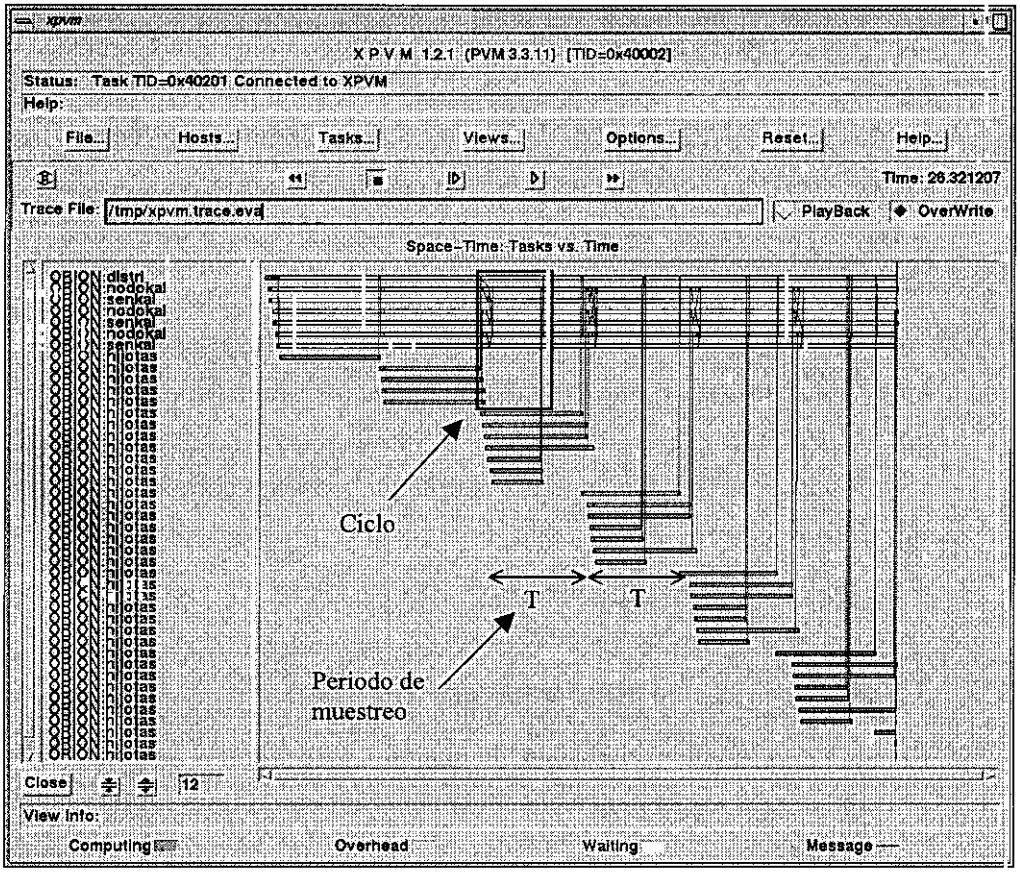
(1) Envío por parte del nodo de fusión, denominado *distri* (véase la figura 5-10), de la señal de control u a cada uno de los nodos de fusión de los sensores. Esta señal puede o no ser necesaria por los sensores dependiendo del modelo utilizado. Previamente al envío, el nodo *distri* recibe una señal que le indica que envíe una señal de control.

(2) Petición de nueva medida al sensor lógico. Esto es la lectura física del sensor y es provocada por la recepción de un timer (*hijotas, hijo de timer asíncrono*) que representa el periodo de medida de cada sensor. Cada sensor puede tener su propio periodo de medida y por supuesto puede realizarse a una frecuencia distinta del resto.

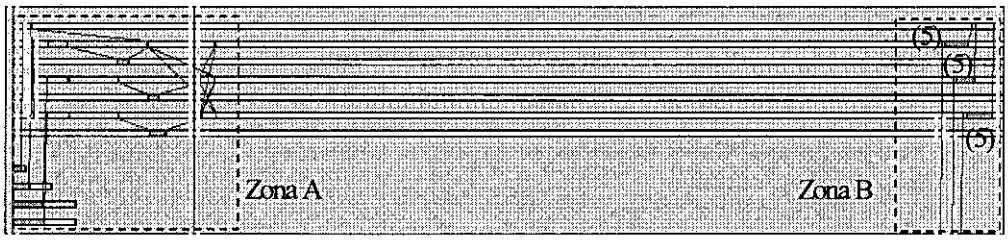
(3) Recepción de la nueva observación y validación, predicción y actualización la estima propia. Una vez realizado esto se envía a los demás la información de asimilación correspondiente.

(4) Recepción de estimas externas. Se encolan en orden de tiempos para su validación externa y asimilación cuando se solicite una nueva lectura del estado.

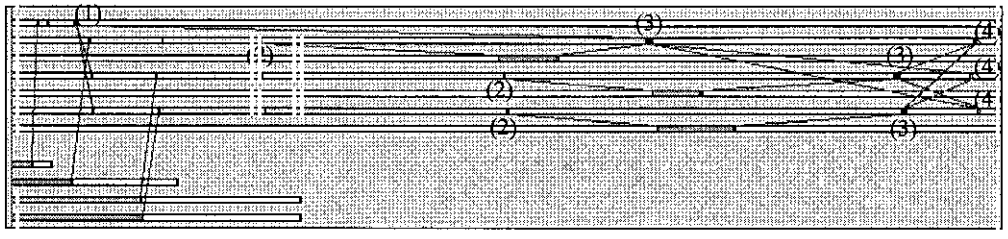
(5) En este momento, figura 5-35b, cada nodo de fusión del sensor recibe la señal de fin del ciclo de lectura por parte de un timer adecuado, es decir, se solicita la estima actual que tiene cada uno de los sensores. En este momento se realiza la validación externa y la asimilación de las medidas acumuladas y se envía la estima final al nodo *distri*, para que este transmita la mejor estima al robot (en teoría todas deberían ser la misma, pero son diferentes debido a fallos, validaciones, ...).



a) Procesos involucrados en Kalman distribuido.



b) Ciclo: medida, asimilación, actualización y envío final.

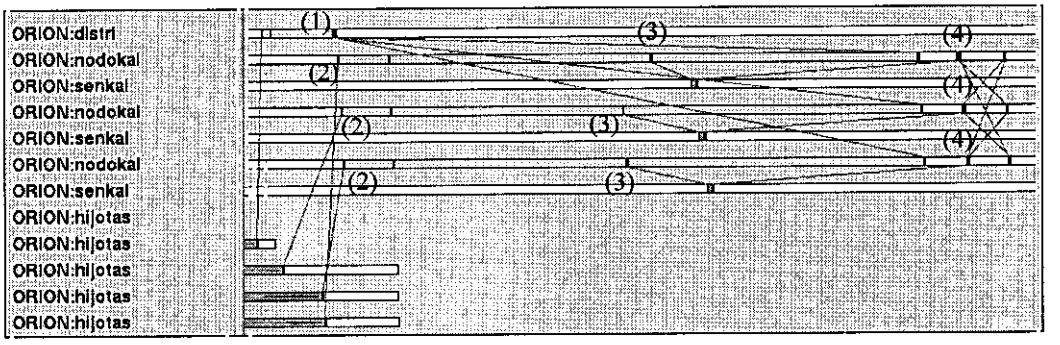


c) Zona A: medida, asimilación y actualización.

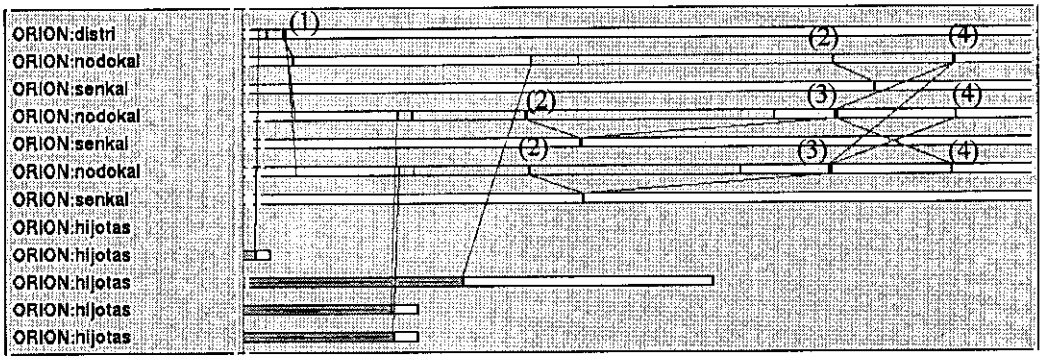
Figura 5-35. Integración de medidas de estado.

También es muy interesante mostrar que cuando se produce un fallo en uno de los sensores y deja de funcionar, el sistema evoluciona sin ningún prejuicio en el paso de mensajes entre los procesos que restan. En la figura 5-36 se muestra un ejemplo en el que ocurre el fallo de un sensor, figura b), frente al mismo caso en el que el sensor actúa normalmente, figura a). En ambas se han incluido la misma numeración que en la figura 5-35c para resaltar la correspondencia entre ellas.

En la figura 5-36a se muestra un ciclo sin fallo como el de la figura 5-35c donde destacar que los sensores trabajan con la misma frecuencia de muestreo (podrían ser distintas). Sin embargo, en la figura 5-36b al fallar el sensor correspondiente al primer *nodokal* (ya sea porque está averiado o porque no puede obtener ninguna lectura del sensor físico), éste no recibe contestación. Debido a esto, este sensor no actúa en los siguientes intercambios de información y por tanto no se realizan las operaciones correspondientes al punto (3). Indicar que sí realiza asimilación de datos externos, en el punto (4), y por lo tanto puede mantener una estima precisa del estado a pesar de no realizar ningún tipo de observación.



a) Ciclo sin fallo del sensor.



b) Ciclo con fallo del sensor.

Figura 5-36. Fallo de un sensor.

5.4.1.2. Simulación de sensores de entorno

En el caso del sistema de fusión de entorno, el mecanismo es similar al descrito para las medidas de estado, aunque los tiempos, señales e información enviada entre los procesos son diferentes, véanse las figuras 5-10 y 5-15 donde se señala un esquema de la información enviada.

La figura 5-37a muestra un ejemplo de los procesos del sistema de fusión multisensorial correspondientes a la fusión de medidas de entorno. En ella se ha indicado cuando se realiza una observación del entorno por parte de cada uno de los sensores (*S1, S2, S3*), para destacar que pueden actuar de forma asíncrona y con frecuencias de muestro distintas.

La figura 5-37b muestra el proceso de actualización de una medida por parte de cada uno de los dos sensores disponibles (visión y ultrasonidos). En ambas observaciones se pueden destacar los siguientes momentos:

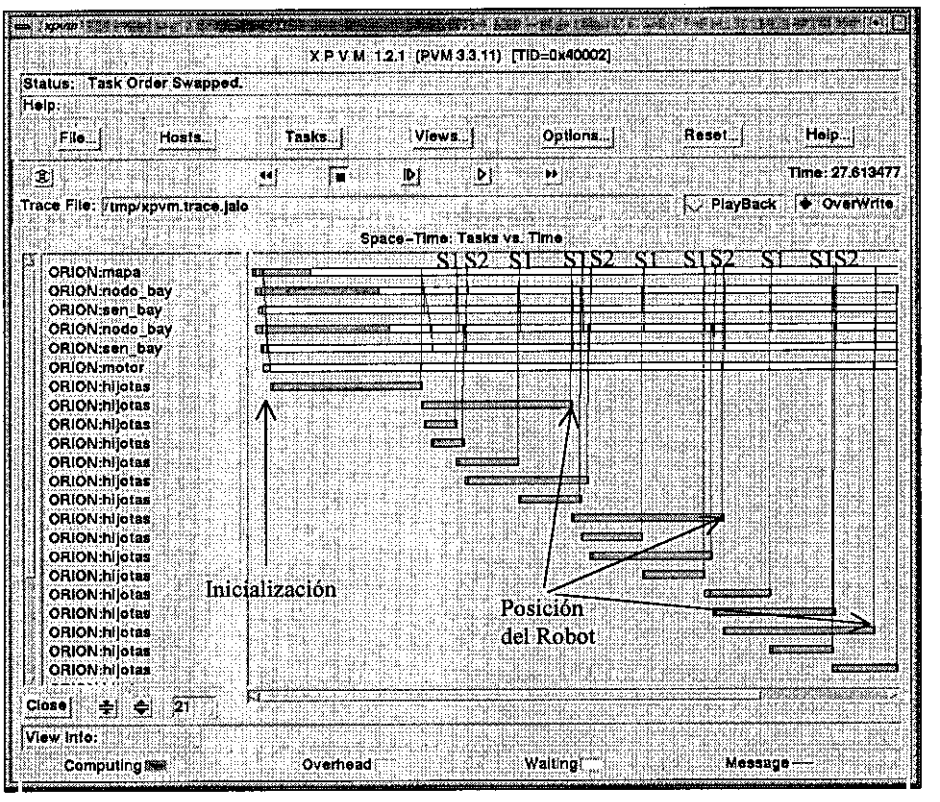
(1) Instante en el que el timer que controla el periodo de muestreo de este sensor le indica que debe realizar una medida de entorno.

(2) Petición al sensor lógico correspondiente (*sen_bay*) una lectura del entorno.

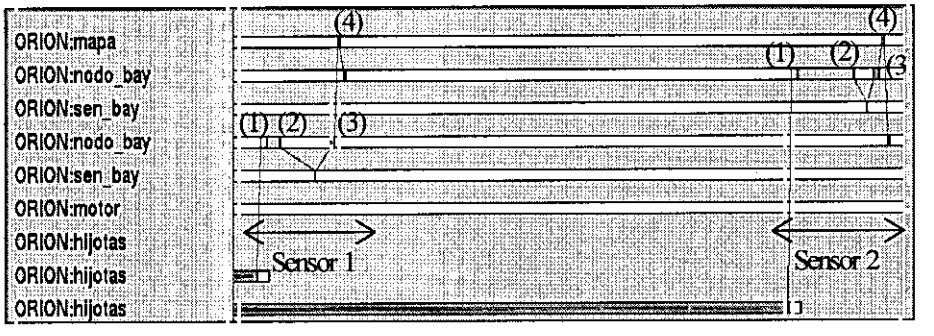
(3) Una vez que se tiene y se almacena en el mapa local propio se envía al proceso de fusión (*mapa*) para que actualice el mapa global.

(4) El nodo de fusión envía la estima correspondiente los otros sensores para que actualicen su creencia del entorno en base a lo que ha afirmado el sensor anterior.

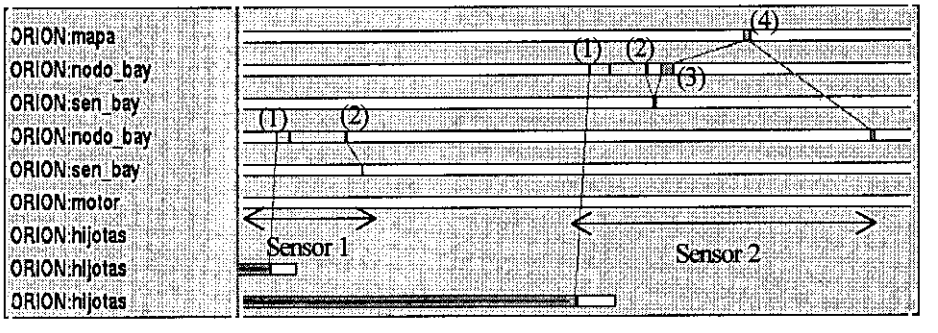
En la figura 5-37c se muestra el caso de que uno de los sensores falle. En este caso como el sensor no contesta, éste no envía nada al nodo de fusión y es igual que si no pasase nada. En el momento que tenga alguna medida disponible o el sensor vuelva a funcionar seguirá trabajando como en el caso anterior sin ningún problema porque mientras ha utilizado las medidas que le proporcionaba el otro sensor.



a) Procesos involucrados en Bayes.



b) Ciclo de Bayes sin fallo del sensor.



c) Ciclo de Bayes con fallo de un sensor.

Figura 5-37. Integración de medidas de entorno

5.4.2. Simulación del sistema de control del robot

Por último se muestra el funcionamiento del robot junto con el sistema de fusión. En las figuras 5-38 y 5-39 se muestran los procesos principales involucrados en el funcionamiento del robot. Se han eliminado los procesos auxiliares y los correspondientes a los sistemas de fusión de entorno y de estado puesto que ya se han descrito en las secciones anteriores. Las tareas mostradas en la figura 5-38 se corresponden a las de la figura 5-9. En concreto los actores mostrados son:

- *runpvm*. Tareas que se encarga de crear los puertos entre procesos, inicializar las variables necesarias (mediante archivos de inicialización y consultas al usuario) para el funcionamiento del robot y crear las demás tareas. Es equivalente al actor *Robot*.
- *Display*. Actor dependiente del sistema utilizado que muestra los resultados en pantalla y es utilizado como interfaz gráfico con el usuario.
- *Navegante*. Es el actor Navegante comentado en la sección 5.3.4.
- *Motores*. Es el actor encargado de realizar el control de los motores, véase la sección 5.3.2.
- *PlanLocal* y *PlanGlobal*. Corresponden al Planificador de la figura 5-9. El comportamiento del planificador se encuentra descrito en la sección 5.3.3 y en la sección 5.3.3.4 se describen los dos actores que realizan la planificación (el planificador local y el planificador global) y su relación.
- *Distri* y *mapa*. Corresponden a los nodos finales de fusión, véase las figuras 5-10 y 5-15, que ofrecen el resultado del sistema de fusión a los actores del robot que lo necesiten.

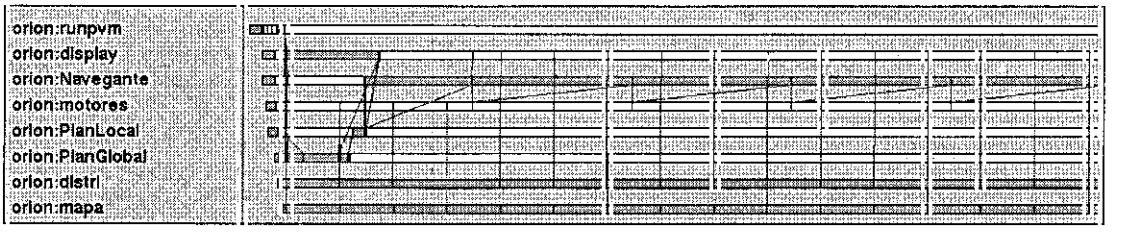


Figura 5-38. Procesos involucrados en el funcionamiento del robot.

En la figura 5-38 se muestra la inicialización del sistema y el funcionamiento cíclico (con una frecuencia de muestreo alta) de las tareas relacionadas con el control (*Navegante*, que vigila el seguimiento de la consigna, y *motores*, que controla el funcionamiento de las ruedas) y la sensorización (*distri* y *mapa* que reciben información fusionada de los sensores lo comunican a los actores interesados).

En la figura 5-39 se muestra el funcionamiento del planificador local (*PlanLocal*) y el planificador global (*PlanGlobal*) cuando el robot se va ha salir de la zona de seguridad y necesita un nuevo mapa local:

(1) El *Navegante* solicita un nuevo camino local al planificador local.

(2) El planificador local solicita al planificador global, que mantiene un mapa actualizado del entorno, un nuevo mapa local.

(3) Una vez el planificador local ha recibido la nueva información de entorno solicita la posición actual al *Navegante* a partir de la cual calcular el camino local, y mientras llega se calcula un quadtree del mapa local.

(4) Se recibe la posición actual y se envía el nuevo camino a seguir.

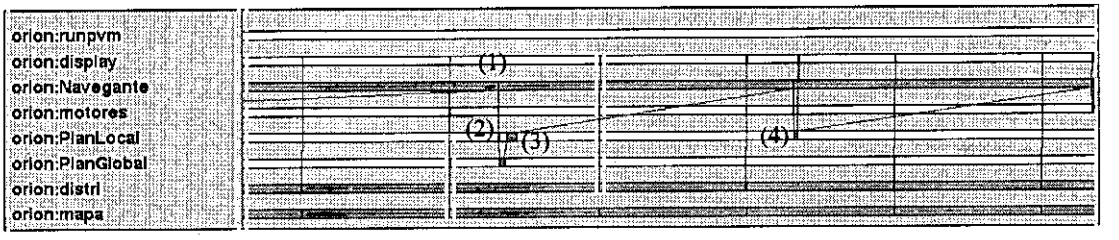


Figura 5-39. Ejemplo de funcionamiento del planificador.

5.5. Resultados del capítulo

En este Capítulo, a partir de las arquitecturas de control más utilizadas en robot móviles terrestres, se ha diseñado una arquitectura centralizada genérica y abierta que puede utilizarse con una amplia variedad robot móviles. Se le ha dotado, también, de la posibilidad de comportamiento reactivo y de un funcionamiento paralelo de las distintas tareas existentes.

Para la especificación de las distintas tareas del robot se ha utilizado una metodología orientada a objetos en tiempo real (*ROOM*); y se ha descrito con claridad su interfaz y funcionamiento. Gracias a la formal especificación mediante ROOM se permite una fácil comprensión del funcionamiento de cada una de las tareas y la fácil modificación, por parte del usuario, de éstas para adaptarlas a sus necesidades concretas.

Por último se ha mostrado un ejemplo de su funcionamiento destacando la cooperación entre tareas y el paso de mensajes realizado.

La implementación de esta arquitectura de control representa un soporte muy adecuado para la utilización del sistema de fusión desarrollado porque su diseño abierto permite adaptarse a distintas aplicaciones y robots de igual forma que el sistema de fusión propuesto puede configurarse para trabajar con diversos sensores. Esto asienta un potente banco de pruebas para verificar la robustez del sistema de fusión y mostrar su facilidad para configurarlo según las necesidades de la aplicación o las posibilidades de los sensores o robots disponibles.

Capítulo 6

Resultados Experimentales

En este Capítulo se muestra el comportamiento de todas las herramientas desarroladas (el sistema de fusión multisensorial y la arquitectura de control) actuando juntas en el manejo de un robot móvil. Las experiencias presentadas se dividen en dos clases: simulaciones y pruebas reales.

Las simulaciones permiten destacar detalles del comportamiento del sistema que no pueden realizarse con facilidad en pruebas reales, pero que mediante simulación son fáciles de provocar y repetir cuantas veces sean necesarias. Con estos experiencias se destaca el comportamiento del robot y del sistema sensorial en distintas situaciones como por ejemplo, reconocimiento de entorno, construcción de mapas, navegación reactiva o detección de objetos en movimiento.

Las pruebas en un robot real ponen de manifiesto la utilidad práctica del sistema de fusión y la arquitectura de control desarrollados. Este caso permite mostrar cómo trabaja el sistema en tiempo real, aunque la velocidad del robot debe ser lenta puesto que la carga en una única computadora es muy grande y no se pueden exigir velocidades más altas. Además, con este ejemplo se muestra cómo se adapta el sistema implementado para trabajar con un robot y sensores reales, destacando su fácil manejo, sencilla adaptación para trabajar con los disponibles y su simple configuración según el comportamiento del robot que se desee.

6.1. Introducción

Para mostrar la eficacia y versatilidad del sistema de fusión desarrollado y su integración en la arquitectura descrita en el Capítulo 5 se utilizará un robot móvil de dos ruedas independientes en entornos interiores. Se realizan dos tipos de experiencias: simulaciones (para comprobar su funcionamiento en distintas situaciones) y pruebas reales (para comprobar su funcionamiento en tiempo real). La arquitectura, tareas del robot y su comunicación con el sistema de fusión multisensorial ya han sido descritas en el Capítulo 5, lo que se pretende señalar en este capítulo es su comportamiento ante distintas situaciones reales, la adaptación de las tareas para llevar a cabo distintos objetivos, su capacidad de sensorización y la facilidad para cambiar de sensores.

Para tener en cuenta, al menos, un sensor representativo de cada uno de los grupos descritos en el Capítulo 2 en un único sistema se utiliza una forma de navegación mixta. Por un lado se fusionan datos de posición obtenidos por sensores de estado, para obtener una estima de la posición mucho mejor que la de cada uno de ellos individualmente. Por otro lado, se fusiona la información referente al entorno del robot para mejorar (en caso de disponer de un mapa) o construir un mapa del entorno por el que se mueve el robot a fin de esquivar los obstáculos que aparezcan en la trayectoria prevista.

El sistema de fusión de estado, para el cálculo de la posición, utiliza los sensores: odómetros en las ruedas, balizas artificiales, dinamo para la obtención de la velocidad y brújula magnética. El modelo de cada sensor será incluido en el sistema en la inicialización a través de un archivo ASCII en un formato preestablecido, de manera que informe de la existencia y funcionamiento de cada sensor. Recordemos que podrían utilizarse otros sensores sin prejuicio para el funcionamiento del sistema.

El sistema de fusión de datos de entorno utiliza estos sensores: un cinturón de ultrasonidos y dos torretas de ultrasonidos (sistema orientable que permite buscar e identificar objetos de forma semejante a visión artificial). El sistema de fusión permite hacer un cálculo cuantitativo de propiedades cualitativas con una cierta incertidumbre, lo cual es propio de los datos ofrecidos por los sensores en cuanto a propiedades del entorno como son tipo de objeto (puerta, pared, etc.). Aunque en estos ejemplos se utiliza únicamente la estima sobre la ocupación o no de una "zona" del entorno del robot. Al igual que en el caso anterior debe ser

incluido en la inicialización del modelo probabilistico de los sensores a través de un archivo ASCII en un formato especificado.

La utilización de un mapa del entorno, en caso de que exista, puede ser tratado de dos formas diferentes según elija el usuario: (1) como un sensor más que indica la ocupación o no de una celda y con una incertidumbre asociada correspondiente a la escala, fiabilidad o precisión, ...; o (2), como una información a priori. En nuestro ejemplo se utiliza el mapa como información a priori para resaltar el funcionamiento de los otros sensores.

Para mostrar el funcionamiento del sistema implementado en el Capítulo 5 ante diversas situaciones y tareas del robot se realizan una serie de simulaciones en la sección 6.2. En esta sección se muestra la facilidad con la que se puede modificar el algoritmo de planificación o el comportamiento del robot para probar distintas técnicas de navegación. Por ejemplo, en la sección 6.2.1 se presentan ejemplos de distintos objetivos de un planificador como son, alcanzar un destino por el camino mínimo, exploración de entornos desconocidos, construcción de mapas, etc.; y, en la sección 6.2.2 se presentan ejemplos de la actuación reactiva del sistema como es la esquiva de obstáculos imprevistos o la detección de objetos en movimiento.

En la sección 6.3 se describe un robot real, sus sensores disponibles y sus restricciones físicas. Con él se muestra cómo el sistema se adapta a un robot real y se realizan dos pruebas en distintos entornos en interiores (un entorno conocido y otro desconocido) para mostrar el correcto funcionamiento del sistema diseñado.

6.2. Simulación del sistema implementado

La simulación del sistema desarrollado permite mostrar cómo actúa el robot y el sistema de fusión ante la información de los sensores y ante las distintas frecuencias de actualización y medidas de cada sensor (para sintonizar el sistema de integración multisensorial), estudiar el comportamiento del robot ante distintos algoritmos como exploración o construcción de mapas u observar su respuesta ante la aparición de objetos imprevistos en la trayectoria (navegación reactiva).

Para simular los ultrasonidos se utiliza un modelo de los sensores semejante al desarrollado por Leonard para sonar [Leonard 90; Leonard 92] . Las torretas de ultrasonidos son simuladas de forma semejante a los cinturones de ultrasonidos, pero se debe simular el giro de las torretas para apuntar al objeto de interés.

Las simulaciones de este capítulo se dividen en dos grandes grupos:

- *Utilización del robot para exploración y construcción de mapas*, muestra cómo la arquitectura desarrollada permite adaptar al robot para su utilización en distintas situaciones. Las modificaciones necesarias están muy localizadas y no afectan al resto de tareas del robot.
- *Navegación reactiva*, muestra cómo el robot reacciona ante eventos imprevistos de manera adecuada. Así, se presentan dos ejemplos la esquiva de un obstáculo imprevisto en la trayectoria y la detección de objetos en movimiento. Este último ejemplo también pone de manifiesto el funcionamiento del factor de olvido en los sensores de entorno.

6.2.1. Utilización del robot para exploración y construcción de mapas

La utilización de robot móviles para estudiar distintas estrategias de exploración y de construcción de mapas está muy extendida, entre las publicaciones recientes se pueden mencionar [Lee 97; Castellanos 98].

Para utilizar el robot como sistema de exploración y de construcción de mapas, únicamente es necesario sustituir el módulo que calcula el camino global y el local mediante Dijistra, por otra función que calcule un camino adecuado para realizar la tareas requerida. Por ejemplo, planificador puede generar caminos en direcciones aleatorias cuando el robot no sepa donde se encuentra o seguir paredes

u objetos dejándolos a un lado determinado del robot (por la derecha o por la izquierda).

Para ello simplemente se sustituye la rutina *calculo_cam_local()* y *calculo_cam_global()* por unas nuevas rutinas que generen caminos adecuadamente según los deseos del usuario.

En la figura 6-1 se muestra cómo se comporta el robot para el caso de construcción de mapas, es decir, no tiene un mapa inicial. El reconocimiento de entorno en este caso ha sido mediante un recorrido siguiendo paredes, pero podría pedirse al robot que vagase libremente o fuese de un sitio a otro determinado.

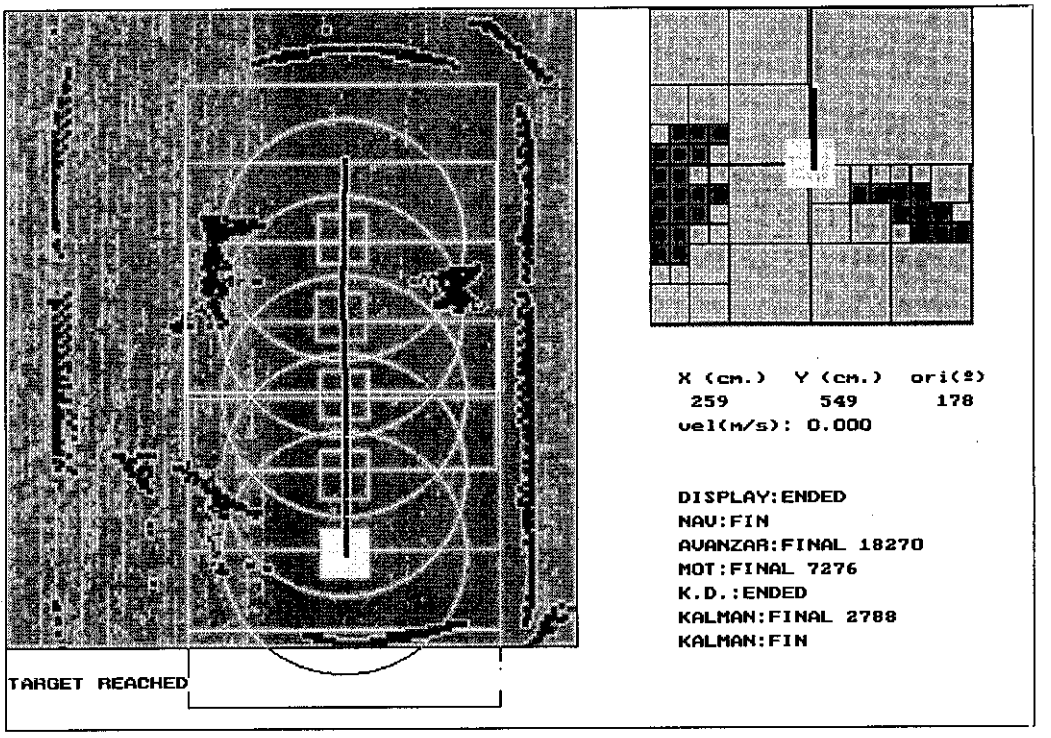


Figura 6-1. Reconocimiento de entorno y construcción de mapas.

La figura 6-1 presenta la pantalla de la computadora que controla el robot (generada por el actor *display*). En ella se muestran una serie de datos referentes a la evolución del robot, se distinguen varias zonas:

A. Mapa Global. Presenta el mapa completo por el que se mueve el robot (si se dispone de él). En éste se representa la información correspondiente a la creencia del estado de cada celda obtenida mediante fusión (creencia global). La probabilidad de ocupación se representa con diferentes colores.

También se distinguen la zona utilizada para el mapa local, el circulo de seguridad, la posición del robot en cada instante y los puntos de consigna dados por el Navegante al Control de motores.

B. Mapa local. Representa una ampliación de la zona próxima al entorno del robot (de tamaño 2.5x2.5 m.). Aquí se representa el quadtree utilizado por el robot y el camino local que sigue.

C. Información sobre estado del robot. Esta información incluye el estado del robot (x, y, ϑ), la velocidad del robot y el estado en el que se encuentran las tareas principales.

D. Información sobre el sistema. Indica la carga de la CPU, el tiempo de simulación y estados del robot (objetivo realizado, errores, etc.).

6.2.2 Navegación reactiva

Aunque la arquitectura de control es centralizada y el Navegante debe enviar los puntos de consigna al Control, se debe dotar al robot de mecanismos de seguridad ante fallos del sistema y de la capacidad de reaccionar ante imprevistos.

Para mantener una cierta seguridad frente a fallos del sistema y en concreto frente a fallos del Navegante se debe asegurar que el robot no quedará a la deriva, esto es, en caso de que falle y deje de funcionar el sistema de Control quedaría a la espera de las consignas del Navegante y continuaría moviéndose indefinidamente con el consiguiente peligro. Para evitar este problema, en caso de falta de consigna el Control detendrá el robot hasta que le lleguen otras nuevas. Esto evita problemas de sincronización o retrasos debidos a grandes necesidades de cálculo, dotando al sistema de un mecanismo de seguridad más. De esta forma si el sistema no es capaz de trabajar a una velocidad suficiente el robot se moverá más despacio o, en el peor de los casos, con un movimiento intermitente.

La aparición de obstáculos imprevistos es otro inconveniente que debe ser tenido en cuenta. Ante estos obstáculos el robot debe reaccionar con la celeridad necesaria para evitar la colisión con ellos. Por supuesto siempre existe una zona donde si aparece un obstáculo no podrá detenerse, pero esto debería evitarse si los sensores funcionan correctamente porque se habría detectado previamente y no se daría tal caso. Además de los obstáculos reales deben detectarse los obstáculos falsos y no tenerlos en cuenta, como son los falsos ecos y las estelas dejadas por

obstáculos en movimiento. Así, tenemos dos problemas principalmente: esquiva de objetos y detección de objetos en movimiento.

Veamos un ejemplo de cada uno de los casos anteriores.

6.2.2.1. Esquiva de objetos

En el caso de que un obstáculo aparezca en la trayectoria trazada, se han incorporado varios sistemas de seguridad para que el robot reaccione ante ese imprevisto de forma adecuada:

1.- Acelerar el tiempo de respuesta. Si el Planificador global detecta un obstáculo en el círculo de seguridad se lo comunica inmediatamente al Navegante, sin pasar por el Planificador local, para que compruebe si ese objeto se encuentra en una posición peligrosa para la seguridad del robot.

2.- Reducción de la velocidad ante la proximidad de objetos. Al enviar primero la información al Navegante se puede actuar inmediatamente sobre el control de motores. Así, si se detecta que el objeto obstruye el camino se indica al control de Motores que disminuya su velocidad hasta parar frente al obstáculo si no se le indica lo contrario.

3.- Cálculo de un nuevo camino alternativo. Mientras el robot se aproxima "con cautela" el Navegante solicita al Planificador Local que calcule una nueva trayectoria para evitar el obstáculo detectado.

El uso de un quadtree para calcular el camino local permite una velocidad de calculo alta y responder rápidamente a la aparición de obstáculos con el cálculo de un nuevo camino para esquivarlo. Además, según la velocidad que se desee se puede aumentar el tamaño de celda, con lo que se consigue un cálculo más rápido pero no será el más óptimo.

Un ejemplo de la detección de un obstáculo y de su esquiva puede verse en el ejemplo mostrado en la figura 6-2. Hay que destacar que el camino global seguido inicialmente, *A*, ha sido calculado cuando todavía no se había detectado el objeto, pero evidentemente no es el seguido ya que hay un obstáculo en su trayectoria. Así, se calculan otros dos caminos globales posteriormente, *B* y *C*, porque el planificador local no es capaz de encontrar un camino local adecuado en su mapa local. También se muestran ejemplos de caminos locales que finalmente no se han seguido como *a* y *c*. Recordemos que la forma de los objetos cambia a medida que nos aproximamos a ellos (debido a la precisión de las medidas) y que

los rodeamos (debido a la perspectiva que los sensores tienen de los objetos) y por lo tanto el camino local se irá modificando de acorde a esa percepción del entorno.

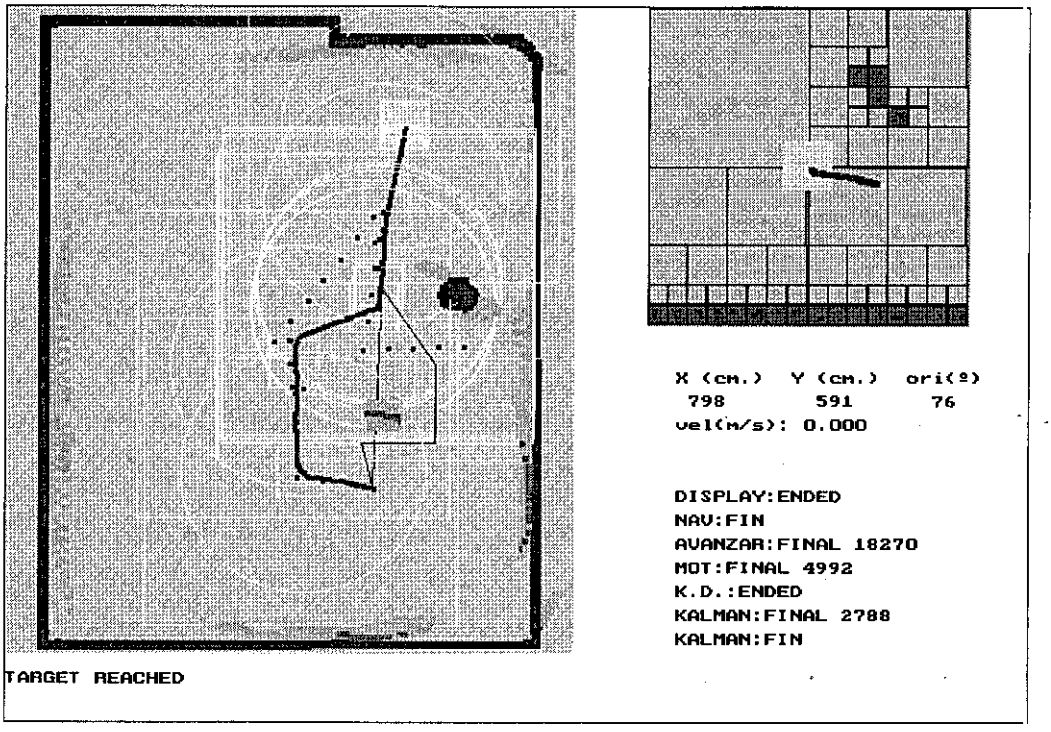


Figura 6-2. Esquiva de un obstáculo imprevisto.

6.2.2.2. Detección de objetos en movimiento

Uno de los problemas de la utilización de mapas de ocupación es la detección de objetos en movimiento. La mayor dificultad que existe es evitar que quede un rastro del paso del objeto, es decir, que las celdas que se han visto ocupadas permanezcan ocupadas y el robot tenga una estima del entorno que es falsa.

En nuestro caso, la detección de objetos en movimiento es posible debido a la actualización dinámica del entorno por parte de los sensores y del sistema de fusión. El mecanismo es semejante a la detección de un obstáculo inmóvil, pero se evita la existencia del rastro del objeto (como se indicó en el Capítulo 4) gracias a la existencia del factor de olvido. Se debe tener en cuenta que para un determinado factor de olvido, si el objeto es demasiado rápido no podrá ser detectado.

Para mostrar un ejemplo del funcionamiento del factor de olvido frente a un objeto en movimiento supongamos que el objeto está parado en una posición inicial y antes de comenzar a moverse detecta un objeto que se mueve en la zona de seguridad del robot. Como puede verse en los mapas locales (quadtree) de la figura 6-3, el robot tiene intención de andar en línea recta pero como el objeto aparece por la derecha del robot y se cruza en el camino debe calcular un nuevo camino para esquivarlo. En la figura 6-3b se muestra un instante posterior cuando el objeto se ha desplazado hacia la izquierda y se calcula un nuevo camino, esta vez por el lado derecho. Finalmente, figura 6-3c, el objeto se aparta de la trayectoria que quería seguir el robot y éste calcula de nuevo un camino en línea recta.

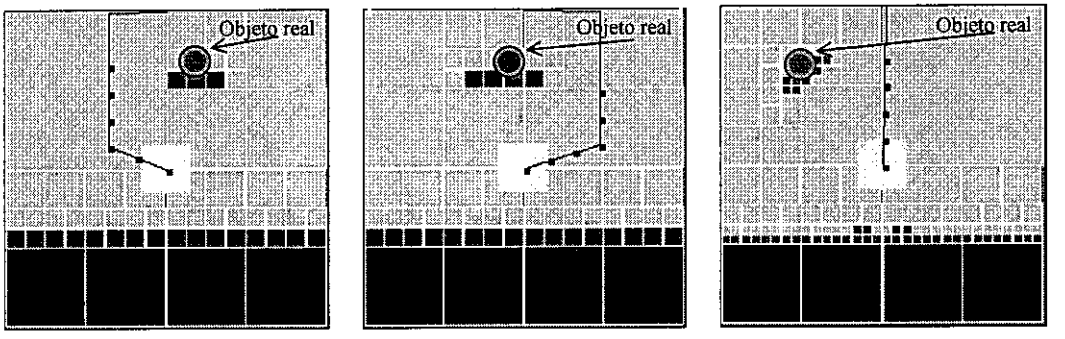


Figura 6-3. Distintos mapas locales en la observación de un objeto en movimiento.

En la figura 6-4 se muestra como ha quedado el entorno del robot en el momento en el que se vuelve a obtener el camino deseado y puede comenzar el robot a trabajar normalmente. Destacar que aunque parece que queda un rastro del objeto. Este rastro no es tal, puesto que no se considera que la zona esté

ocupada (su valor es aproximadamente 0'5) sino que tardará más tiempo en olvidarse completamente (puede verse que el rastro inicial prácticamente ya no está) ya que el valor inicial de esas celdas era de 0'2 debido a la información a priori al trabajar con un mapa inicial.

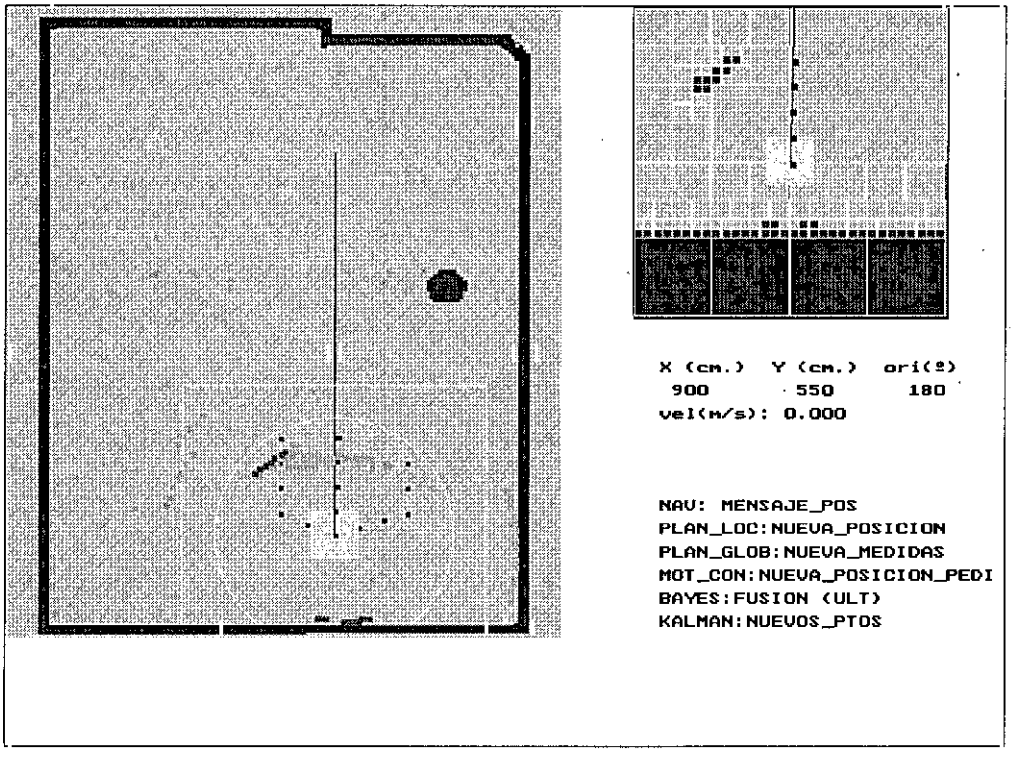


Figura 6-4. Observación de un objeto en movimiento.

6.3. Resultados en pruebas reales

La utilización de un robot real nos enfrenta ha dificultades que no se han previsto en simulación, como por ejemplo la necesidad de respuesta en tiempo real, el ruido e incertidumbre en los sensores que es necesario calibrar, la sintonia del sistema de fusión de entorno para adecuarlo a la velocidad del robot y a la precisión de los sensores y otros muchos pequeños detalles que hacen de cada robot móvil un sistema particular.

Pero a demás de mostrar cómo adaptar el sistema de fusión y la arquitectura de control a un problema real, las pruebas con un robot real permiten corroborar las propiedades y características definidas en los capítulos y mostradas mediante simulaciones. Por ejemplo, su robustez frente a fallos de los sensores, la fusión de medidas para mejorar la estima final o la adaptación del sistema de control para utilizar el mismo robot para distintas aplicaciones.

6.3.1. Adaptación de un sistema real

El robot utilizado para mostrar la efectividad del diseño efectuado ha sido construido en la Universidad de Castilla la Mancha en Ciudad Real. En su construcción han colaborado investigadores de la Escuela Superior de Informática, en concreto del departamento de Ingeniería Eléctrica y Electrónica, del departamento de Informática y del departamento de Física Aplicada.

El robot utiliza distintos sensores, véase la figura 6-5. En el Apéndice C se puede encontrar una descripción técnica de estos sensores, que son utilizados para obtener la siguiente información:

- *Odómetros*, que obtienen el desplazamiento relativo realizado por el robot.
- *Dinamo*, que permite obtener directamente la velocidad absoluta de las ruedas.
- *Sistema de balizas*, que consta de dos receptores de balizas situados en la parte superior del robot y a ambos lados. Los emisores de balizas se colocan en zonas estratégicas del entorno donde de moverá el robot y se le indica al robot su posición exacta por medio de un archivo de datos. Tanto los detectores como los emisores de balizas se han construido y diseñado por el equipo de la Universidad de Castilla la Mancha y han sido

implementados con ultrasonidos. Con ellos se puede obtener su posición y orientación absolutas.

- *Brújula magnética*, que permite obtener la orientación del robot con respecto al norte magnético, con una precisión de una décima de grado.
- *Cinturón de ultrasonidos*, que se encuentra distribuido alrededor del robot de la siguiente forma: 7 en la parte frontal, 2 en cada uno de los laterales y uno en la parte trasera. Y detéctan cualquier objeto alrededor del robot.
- *Torretas de ultrasonidos*, que consta de dos torretas de ultrasonidos orientables en cualquier dirección entre 0 y 360 grados. Una torreta se encuentra en el frontal del robot y la otra en la parte trasera y funcionan independientemente. Aunque están diseñadas para realizar percepción triaural, en estos momentos se utilizan para reforzar la percepción de los ultrasonidos en la localización de obstáculos. Su funcionamiento es equivalente a visión estereoscópica, pero basada en ultrasonidos.

Todos estos sensores se clasifican dentro de uno de los grupos descritos en el Capítulo 2, sensores de estado o sensores de entorno, y se utilizan en los sistemas de fusión correspondientes al tipo de información que proporcionan:

(1) Sensores de estado: odómetros, dinamo, balizas artificiales y brújula magnética.

(2) Sensores de entorno: cinturón de ultrasonidos y torretas de ultrasonidos.

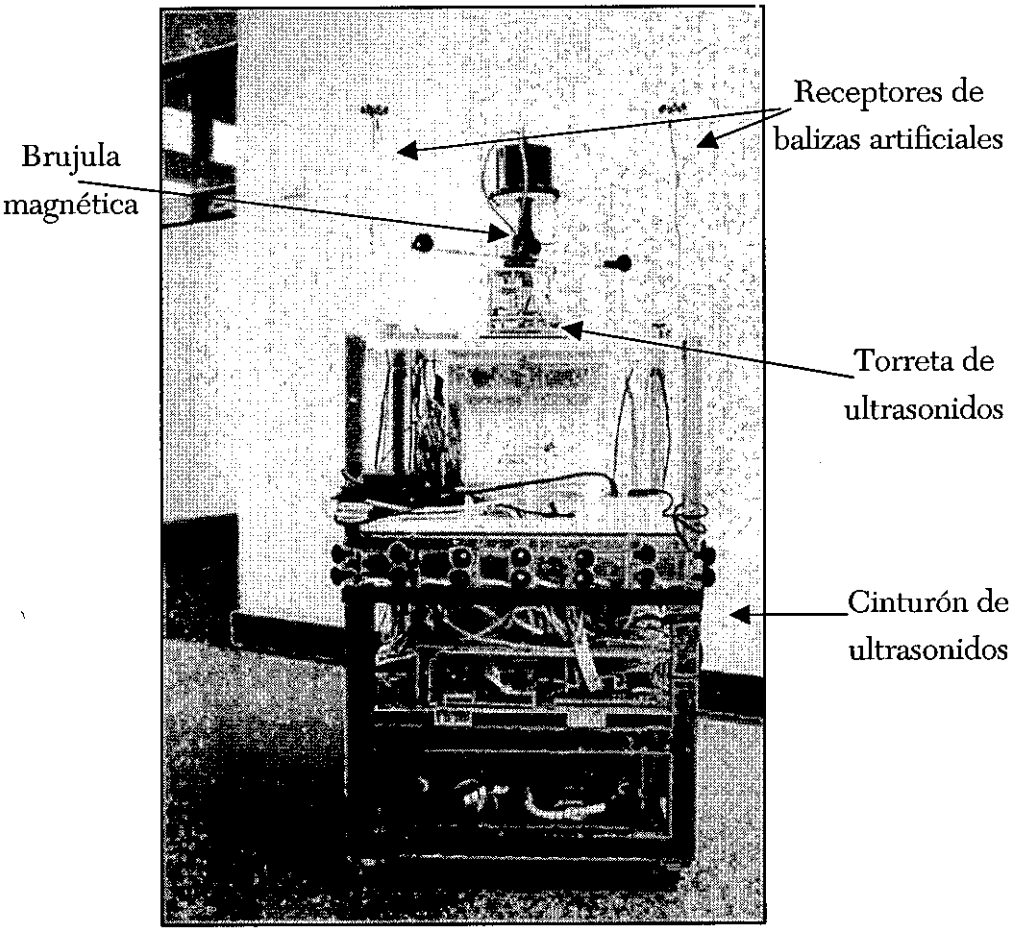


Figura 6-5. Sistema sensorial del robot móvil.

Para trabajar con estos sensores es necesario construir ciertas rutinas que permitan su manejo. Estas rutinas son propias de cada tipo de sensor y es lo que cada usuario debe definir del sistema de fusión y control del robot. Estas funciones son,

- Funciones de alto nivel. Corresponden a la redefinición de la función *Vect_info()* y de la función *EstadoFuncion()* indicadas en el Capítulo 2. La función *Vect_info()* se utiliza para tratar la información procedente del sensor de bajo nivel y construir el vector de estado (o de entorno) que es necesario enviar al sistema de fusión. Sus funciones son sincronismo, conversión de unidades, asociación de datos, etc. La función *EstadoFuncion()* se ocupa del comportamiento de cada sensor: se encarga de recibir órdenes para variar la actuación del sensor de bajo nivel y para indicar el estado de un sensor determinado.

- Funciones de bajo nivel. Corresponden a las funciones próximas al sensor y se encargan de la lectura y control de los sensores propiamente dichos. Estas funciones, corresponden con *leer_dato()* mostrada en el Capítulo 2, son implementadas por quienes conozcan el funcionamiento interno de cada sensor. En este caso han sido diseñadas por investigadores del departamento de Informática y de Ingeniería Eléctrica y Electrónica de la Universidad de Castilla la Mancha en Ciudad Real. El diseño e implementación de todas las rutinas necesarias ha sido en C++ utilizando RTKernel, sistema operativo en Tiempo Real, y de forma que permitan un uso abierto de los sensores para que puedan utilizarse en cualquier configuración que se desee.

La función *vect_info()* de cada sensor es diferente, por ejemplo en el sensor de la baliza artificial se encarga calcular las distancias de cada detector a los emisores y a partir de ellas calcular la posición y orientación del robot, primero en el sistema de coordenadas de las balizas y posteriormente en el sistema de coordenadas global. Otro tipo de tratamiento es el que realiza la función *vect_info()* del cinturón de ultrasonidos, transforma la detección de cada sensor del sistema de coordenadas del robot al sistema global y traduce cada observación en las celdas adecuadas. Además indica una certeza en la observación dependiendo de una función que puede definir el usuario a partir de la información medida como puede ser la separación del eje, amplitud de la onda recibida, etc.

En cuanto a la función *EstadoFunción()*, también será distinta dependiendo de las órdenes que cada nodo de fusión necesita dar al nodo sensor y que éste transmite de forma adecuada al sensor físico. Pueden ser órdenes sencillas como "Medir" (como ocurre en el caso de la brújula magnética o los encoders) o más complejas como "modo de funcionamiento" que se utiliza para cambiar la potencia de emisión del cinturón de ultrasonidos y variar su alcance (en estos ejemplos se utiliza un rango de 40 – 500 cm.) o "modo de observación" que indica qué sensores deben observar el entorno (sólo los delanteros, un muestreo circular, alternado, etc.) dependiendo de qué interese en cada momento. La función *EstadoFunción()* de la torreta, por ejemplo, recibe otro tipo de órdenes como es "resetear" la torreta, para eliminar errores en el giro, o "giro" para apuntar en una dirección determinada.

Finalmente, para configurar el sistema de fusión para que trabaje correctamente con los sensores disponibles, además de implementar las rutinas para la lectura y control de cada uno de los sensores disponibles, es necesario estudiar el ruido de las medidas y crear nuestro modelo de sensor y el modelo de entorno.

Por ejemplo, para los sensores de estado es necesario definir la matriz de ruido del sensor, R, la matriz de estado, Q, en la tabla 6-1.

Ruido del sensor	Brújula magnética			Balizas artificiales			Odómetros		
R	x	y	θ	x	y	θ	x	y	θ
Matriz de Estado	-	-	0.0175	0.02	0.02	0.05	0.02	0.02	0.05
Q	0.1	0.1	0.01	0.15	0.15	1.0	0.15	0.15	1.0

Tabla 6-1. Ruido de estados y de medida para los sensores de posición

Por otro lado, en la tabla 6-2 se indica el periodo de medida de los sensores físicos y el periodo de actualización del estado por parte de cada uno de los sensores lógicos de estado, así como el periodo de medida y de olvido para los sensores de entorno. Todo ello seleccionado por el usuario para obtener un comportamiento concreto.

	Tmedida (seg.)	Tactualización/tolvido (seg.)	
Brujula magnetica	1	1	-
Balizas artificiales	1.5	10	-
Odómetros	0.25	0.5	-
Cinturón de ultrasonidos	1	-	10
Torretas de ultrasonidos	0.6	-	20

Tabla 6-2. Periodos de medida, de actualización y de olvido de los sensores

Conviene señalar, también, que se generan señales de control sobre los motores cada 0.5 segundos y que los sensores de posición envían una estima al

nodo de fusión cada 0.5 segundos de forma que el robot dispone de una medida actualizada de posición cada 0.5 segundo.

Para definir nuestro modelo de entorno es necesario indicar el tamaño de la celda en el mapa de entorno y el tipo de información que ofrecen los sensores. El tamaño de celda elegido es de 10 cm. y la información que se utiliza para fusión es la creencia en la existencia o no de un objeto.

Por último, para comprender globalmente el funcionamiento del sistema se ha construido un diagrama de tiempos, véase la figura xx, donde se muestra el periodo de actuación de cada tarea y un ejemplo de las señales que intercambian. Por claridad no se muestran todas, sino sólo un ejemplo de cada una de las más significativas.

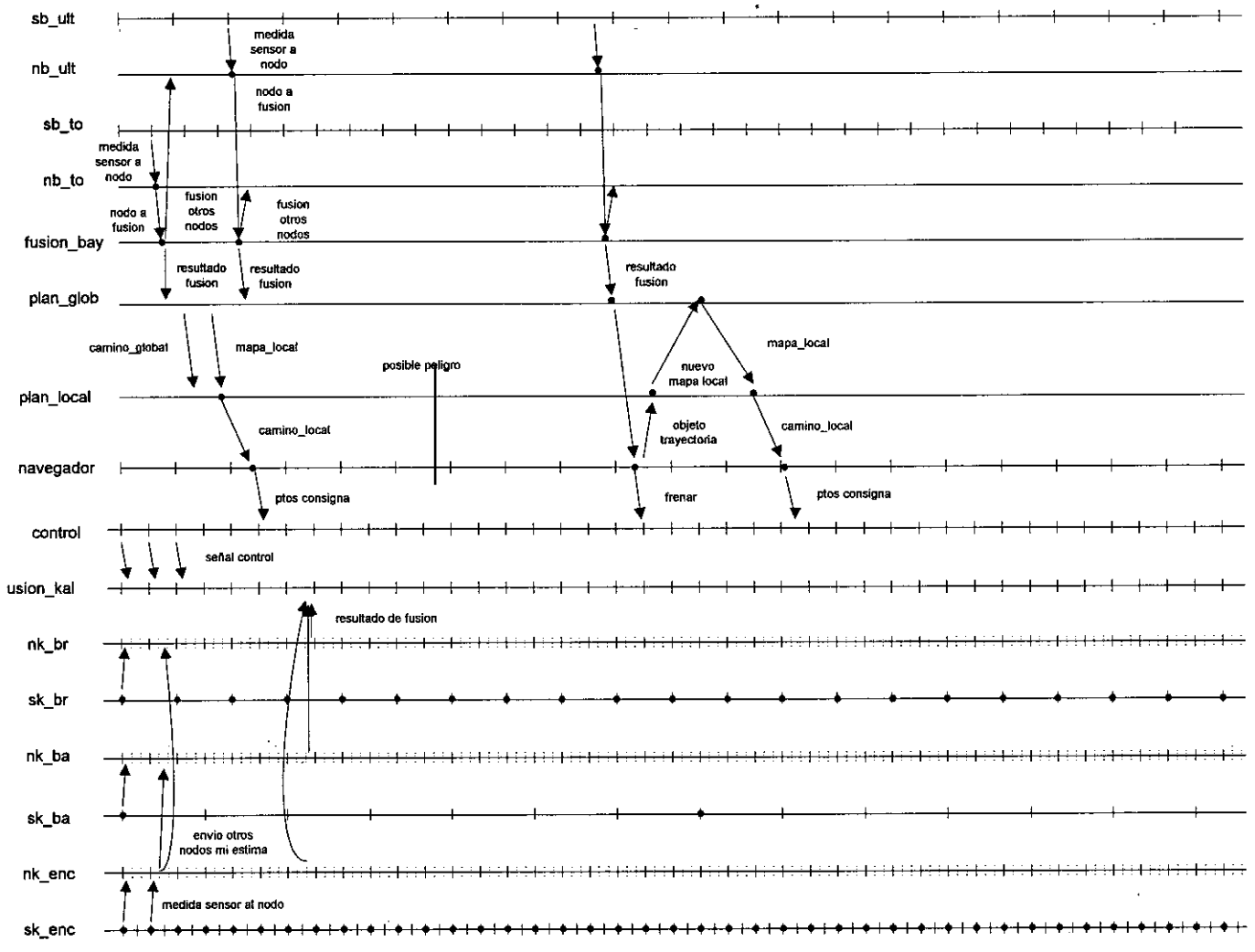


Figura 6-6. Diagrama de las tareas involucradas en el robot

Una vez definido el sistema sensorial, su control y su comportamiento, a continuación se presenta cómo se comporta el robot y su sistema sensorial en diversos experiencias reales.

6.3.2. Movimiento en un entorno conocido

El movimiento del robot se realiza en el entorno mostrado en la figura 6-7. Aquí puede ver la posición de los emisores de balizas artificiales y el alcance que tienen.

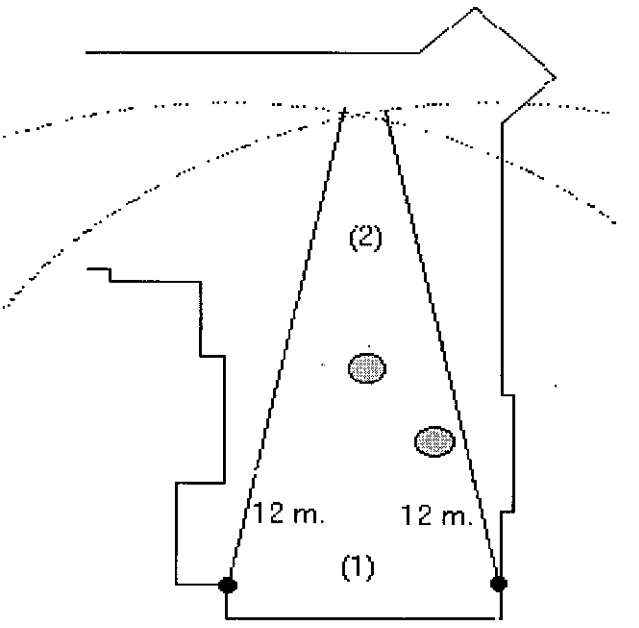


Figura 6-7. Mapa del entorno de pruebas.

En la figura 6-7 se han incluido dos obstáculos que no se encuentran en el mapa disponible por el robot y que utiliza como información a priori en su mapa de ocupación.

En la prueba realizada el robot se mueve desde el punto (1) hasta el (2), pero en su camino se encontrará el obstáculo que no estaba incluido en el mapa, lo esquiva y alcanza su destino marcado. El recorrido realizado puede verse en la figura 6-8, en ella se muestra también el mapa de ocupación cuando ha realizado el recorrido. Aquí se pueden destacar los dos objetos que no aparecían en el mapa a priori del robot.

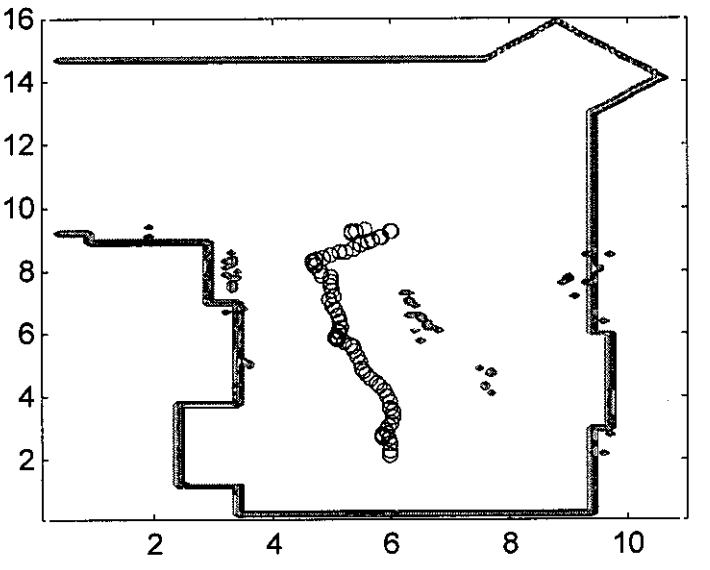
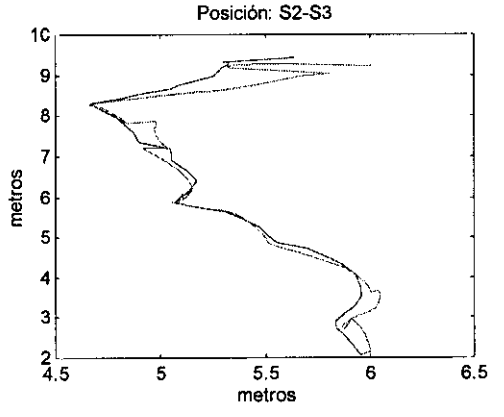


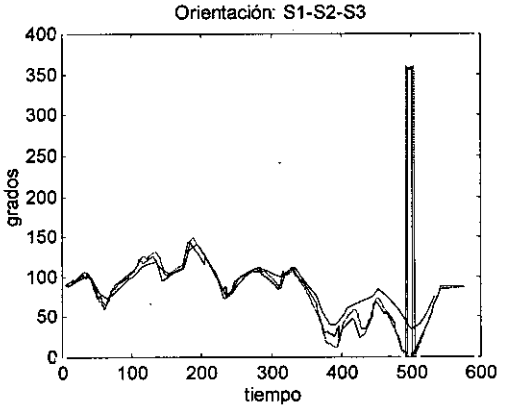
Figura 6-8. Trayectoria seguida por el robot

6.3.2.1 Funcionamiento de los sensores

El recorrido de la figura 6-8 permite mostrar la posición y orientación que cada uno de los sensores lógicos obtiene a partir de las medidas tomadas por sus sensores físicos. En la figura 6-9 puede verse que la posición que obtienen a partir de las observaciones de sus sensores no es siempre el mismo, debido a errores, ruidos, pérdidas de datos, etc. Además, no todos los sensores lógicos pueden calcular las mismas variables de estado, por ejemplo en figura 6-9a sólo se muestra la posición calculada por el sensor S2 (balizas artificiales) y S3 (odómetros), puesto que la brújula magnética no puede obtener la posición a partir de la información de su sensor.



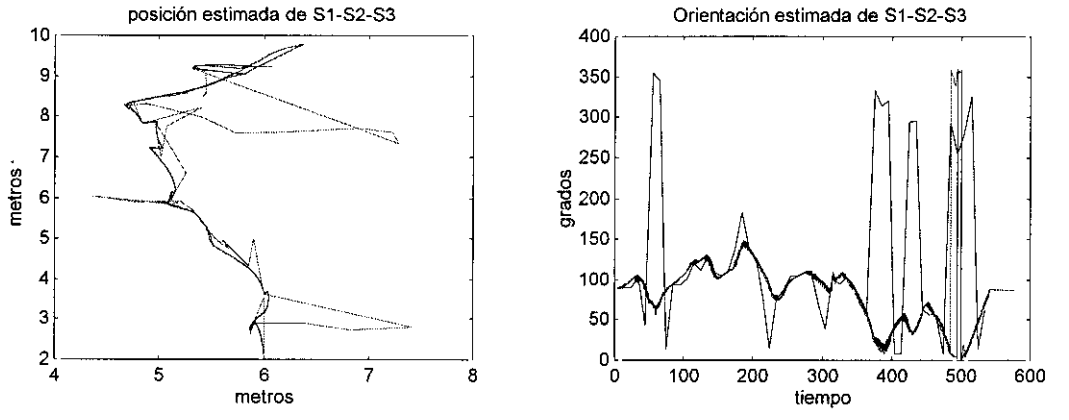
a) Posición



b) Orientación

Figura 6-9. Posición y orientación obtenida de los sensores físicos
(S1, brujula magnética; S2, balizas artificiales; S3, odometros)

En figura 6-10, se muestran las estimas del estado del robot, para cada uno de los sensores de estado. En este caso sí que todos los sensores (nodos de fusión) son capaces de estimar el estado final del robot, que posteriormente enviarán al nodo de fusión para que lo comunique a las demás tareas del robot.



a) Posición b) Orientación

Figura 6-10. Posición y orientación obtenida en el sistema de fusión de estado
(S1, brujula magnética; S2, balizas artificiales; S3, odometros)

En las figuras 6-10a y 6-10b se puede observar como la estima de cada uno de los sensores, debido a que intercambian información entre ellos, es prácticamente la misma. Sólo en algunas ocasiones, debido a fallo de los sensores, a no validaciones de los datos externos o a errores en la comunicación entre nodos, se producen divergencias en las estimas, pero son corregidas en cuanto se normaliza la situación. La causa de las diferencias entre las estimas se describe con más detenimiento en la sección siguiente.

6.3.2.2. Robustez frente a fallos de los sensores

Esta experiencia permite mostrar la robustez del sistema de fusión frente a errores de los sensores. Estos errores producen divergencias en las estimas del estado del robot, como puede verse en la figura 6-10. Esto también puede verse con más claridad en la varianza del nodo de Kalman correspondiente al sensor de las balizas artificiales. Como puede verse en la figura 6-11, su varianza aumenta debido a que asimila información (estimas) de los demás nodos de Kalman (debido a problemas en la comunicación entre procesos) y no puede obtener información

de sus sensores. Esto es debido a que el robot pierde la señal de las balizas debido a que uno de los emisores queda oculto por uno de obstáculos. Sin embargo, a pesar de este inconveniente, se puede observar que el robot sigue funcionando correctamente gracias a la información redundante que ofrecían los encoders y la brújula magnética.

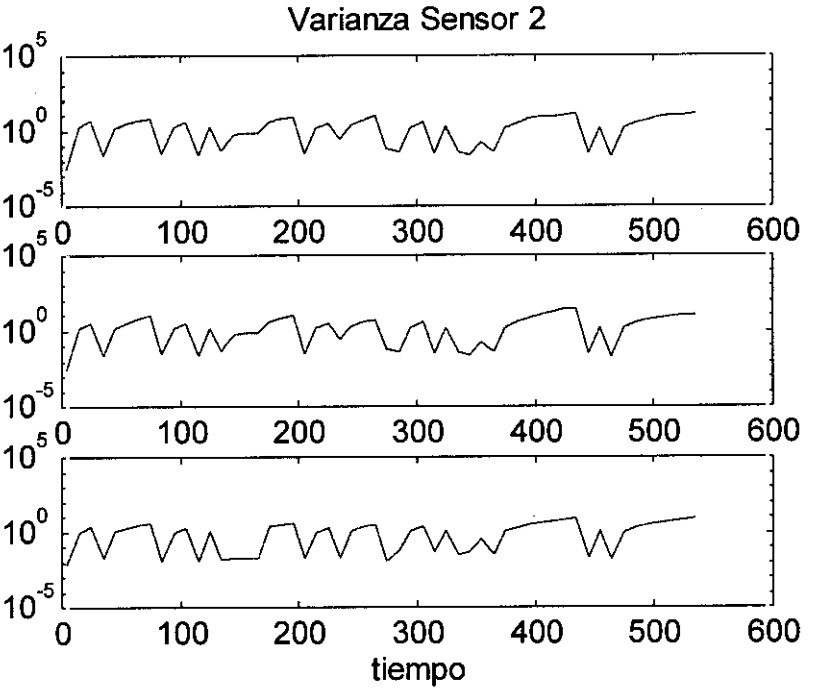


Figura 6-11. Varianza de las balizas artificiales (S2).

6.3.3. Movimiento en un entorno desconocido

El movimiento del robot en un entorno desconocido nos muestra la capacidad del móvil para desenvolverse sin ningún tipo de información a priori. En este caso se realizan dos pruebas una para alcanzar un objetivo determinado y otra para construir un mapa de entorno.

6.3.3.1. Funcionamiento de los sensores de entorno

En la figura 6-12, se puede ver el recorrido del robot para alcanzar un objetivo determinado (del punto (1) al punto (2) señalado en la figura 6-7). En esta figura se han incluido los objetos detectados por los sensores de entorno, con círculos se representan las observaciones del cinturón de ultrasonidos y con asteriscos las correspondientes a ambas torretas. Como puede verse, existen algunas medidas espurias que no afectan al movimiento del robot, puesto que son

eliminadas gracias al factor de olvido. Indicar aquí que las torretas funcionan apuntando al objeto más cercano al robot para reforzar la medida del cinturón de ultrasonidos ya que puede ser un objeto potencialmente peligroso. En este caso al no existir obstáculos, apuntan a las paredes.

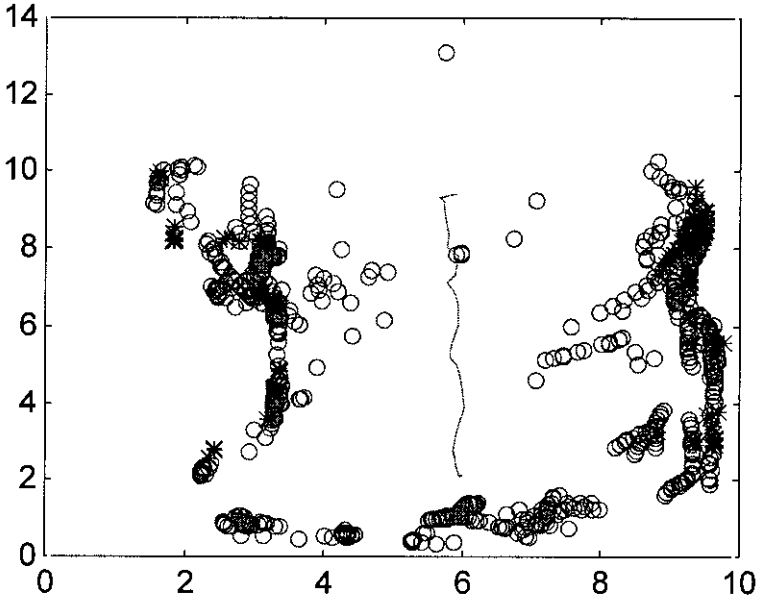
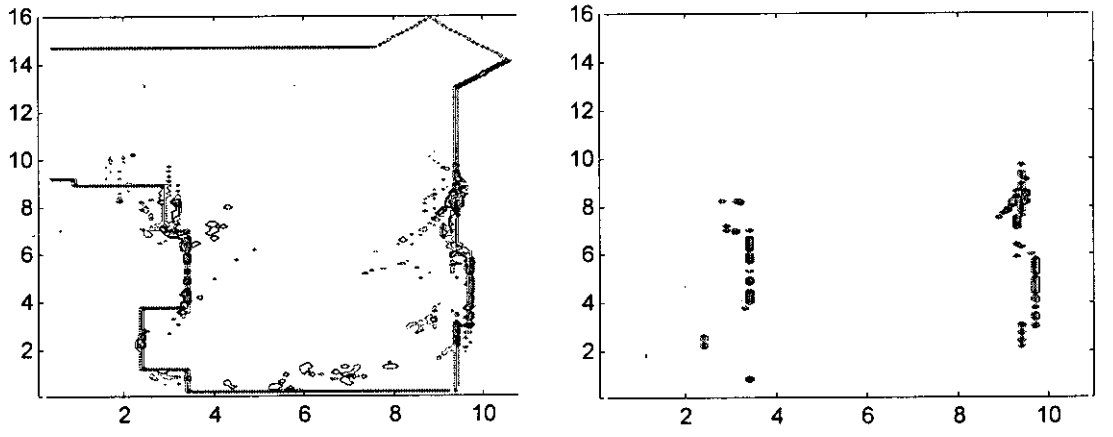


Figura 6-12. Recorrido del robot y observaciones realizadas por los sensores de entorno (o, cinturón de ultrasonidos; * torretas)

La figura 6-12 se muestran todas las medidas realizadas por los sensores de entorno (cinturón de ultrasonidos y torretas) y que son utilizadas para construir el mapa de ocupación a partir de la fusión de estas. El mapa de ocupación resultante puede verse en la figura 6-13 donde se puede destacar la eliminación, mediante la utilización de un factor de olvido, de las medidas que obstruían el camino del robot (debido a ecos erróneos). La figura 6-13a muestra el mapa de ocupación resultante, donde se señalan todas las celdas cuya creencia sea superior a la incertidumbre (0.5) aunque realmente no se consideren ocupadas para el robot. En esta figura se ha dibujado también el mapa real para que se observe la correspondencia entre la estima y el entorno real. En la figura 6-13b se muestran sólo las celdas que son consideradas ocupadas (su valor de creencia supera el 0.64). El valor considerado como ocupado es seleccionado por el usuario y totalmente experimental.



a) Mapa real y mapa de ocupación b) Celdas consideradas como ocupadas

Figura 6-13. Mapa de ocupación obtenido

6.3.3.2. Utilización del robot en construcción de mapas

Para crear un mapa del entorno del robot es necesario modificar el factor de olvido para que olvide más lentamente. Esto es debido fundamentalmente a que el movimiento del robot es muy lento (velocidad máxima 0.1 m/s) y hay muy pocos sensores por la parte trasera del robot, por lo que si olvida demasiado rápido no quedarán celdas consideradas como ocupadas para reconstruir el entorno. Esto no importaba en el caso anterior porque lo más importante es detectar los objetos en su camino (objetos en la parte delantera) y olvidar la información que se ha pasado no es ningún inconveniente.

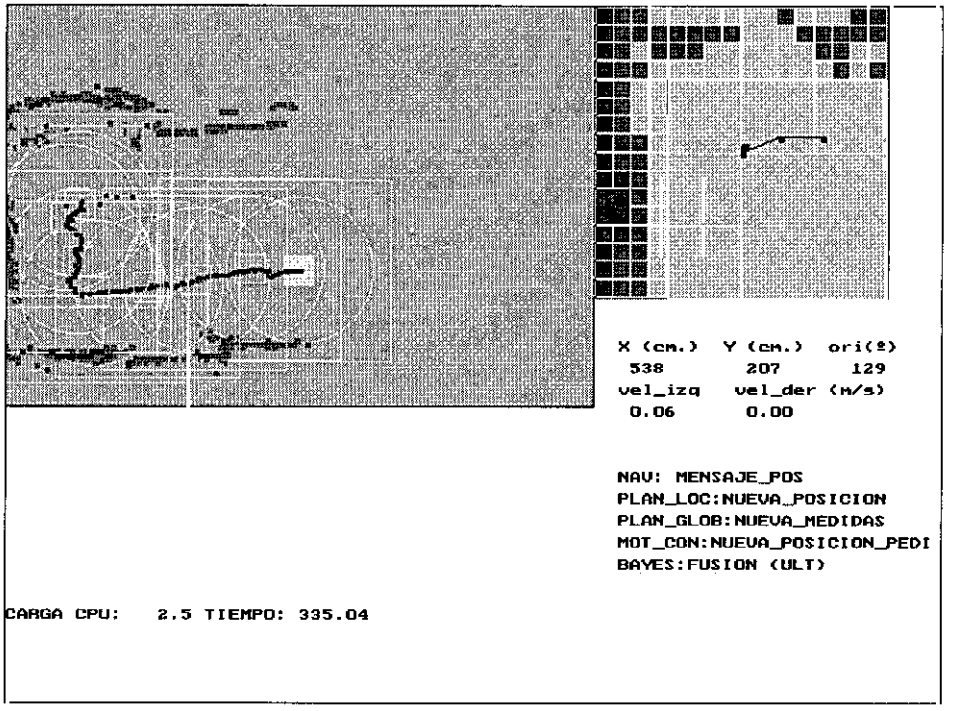


Figura 6-14. Exploración para construcción de mapas.

El algoritmo utilizado por el planificador de caminos es el preparado para exploración y ya ha sido comentado en la sección 6.2.1. En la figura 6-14 se puede observar que el robot se aproxima a la pared detectada y la sigue para construir un mapa del entorno que le rodea. En robot no utiliza ningún mapa a priori.

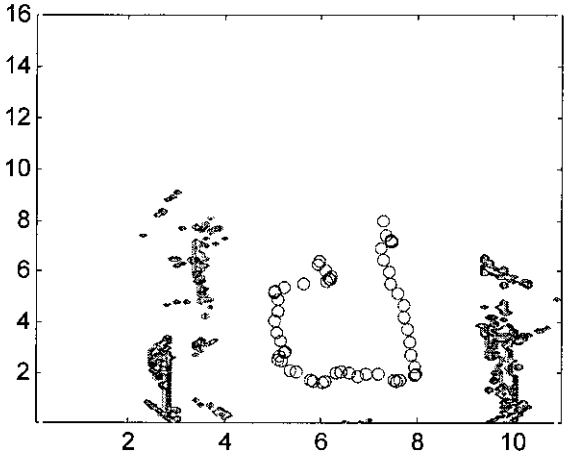


Figura 6-15. Mapa del entorno

El mapa obtenido después del recorrido del robot es el mostrado en la figura 6-15. En este se ha mostrado también el recorrido del robot. Conviene indicar que el robot realiza un recorrido bastante separado de la pared por motivos de seguridad y para evitar la zona muerta del cinturón de ultrasonidos. La zona muerta en este caso es de 0.6 m, debido a que se utiliza un rango de medida de los sensores preparado para detectar objetos a grandes distancias (0.6 – 5 m.).

6.4. Resultados del capítulo

En este Capítulo se han aplicado todos los conocimientos expuestos en esta Tesis, se ha utilizado el sistema de fusión multisensorial, para localizar al robot y para detectar obstáculos, y la arquitectura desarrollada, expresamente para utilizar con facilidad el sistema sensorial en diversos robots móviles.

Como se puede ver a lo largo del Capítulo, el funcionamiento del sistema de fusión es muy robusto y fácil de configurar a gusto del usuario o para adaptarse a un entorno cambiante, para realizar distintas tareas o simplemente cuando las necesidades de la situación lo requieran.

El único problema que surge en este Capítulo es poner de manifiesto todas las características enunciadas en capítulos anteriores, sobre todo en una única experiencia. Esto lleva a mostrar diversas pruebas para destacar los aspectos más importantes y representar gráficamente en cada una de ellas sólo la información que permite poner de manifiesto las características que se pretenden destacar. Sin embargo, no se debe olvidar que en cada una de las experiencias están colaborando para su consecución un enorme número de tareas y que se debe tener en cuenta una gran cantidad de variables, pero sería imposible presentarlas todos los resultados en cada una de las experiencias.

Por último destacar que en las pruebas reales, los algoritmos de control de las ruedas y el planificador de caminos utilizan unos algoritmos muy sencillos puesto que no nos hemos centrado en estos aspectos del problema de un robot móvil, sino en la integración y fusión de múltiples sensores y en la construcción de un sistema fácilmente paralelizable, adaptable a cualquier tipo de sensor e independiente del número de sensores utilizados. Lo que se pone claramente de manifiesto en todas los ejemplos mostrados, tanto en las simulaciones como en las pruebas con el robot real.

Conclusiones

Conclusiones y resultados

La investigación realizada en esta tesis se ha centrado en dos objetivos principales:

1. Desarrollo e implementación de un sistema de integración y fusión multisensorial para vehículos móviles terrestres autónomos. Se ha realizado un diseño abierto que sea independiente del número y del tipo de sensores utilizados (dentro de una amplia variedad de sensores). Además, teniendo en cuenta que puede ser utilizado en sistemas de tiempo real con fuertes restricciones en el tiempo de respuesta, se han escogido aquellas técnicas más convenientes para trabajar de forma distribuida y asíncrona.

2. Se ha diseñado e implementado una arquitectura de control en tiempo real orientada a objetos que permita desarrollar todas las capacidades del sistema de fusión. Siguiendo la misma línea de trabajo mantenida en el d iseño del sistema de fusión, se ha buscado una arquitectura abierta que pueda ser adaptada en diferentes robots móviles autónomos. Con ella se prueban distintos algoritmos y tipos de sensores sin necesidad de utilizar otra arquitectura de control y siempre con el mismo sistema de fusión. Así, se prueban distintas configuraciones de los sensores y se aplican para realizar distintas tareas con sólo configurar el sistema.

Para conseguir cada uno de estos objetivos se han realizado los siguientes trabajos:

1.- Sistema de integración y fusión multisensorial

A. Desarrollo de un sensor lógico mediante la metodología orientada a objetos, que permita la abstracción de los sensores físicos por parte del sistema de fusión multisensorial.

El estudio de la amplia variedad de métodos de integración y fusión multisensorial y su infinidad de aplicaciones, implican que no sea factible la realización de un método universal y eficaz de fusión de sensores. En muchos casos los sistemas sensoriales creados exprofeso para una aplicación determinada son más eficaces que uno general, pero tienen como inconveniente que su utilización es muy restringida. Por este motivo se siguen buscando sistemas de fusión que permitan configurarse para trabajar con un amplio conjunto de problemas, utilizar una amplia variedad de sensores y extender su funcionalidad a otras muchas áreas y aplicaciones.

Puesto que no es factible la creación de un sistema universal de fusión, se debe restringir su aplicación a un conjunto de problemas. El área escogida, debido a su gran variedad de sensores y algoritmos aplicables, es la de robots móviles autónomos. Así, se busca un sistema de integración y fusión multisensorial que sea independiente del tipo y número de sensores utilizados. Para ello, el primer paso es hacer una abstracción de éstos con el fin de separar las técnicas utilizadas de la amplia variedad de dispositivos existentes. Esta abstracción da lugar al sensor lógico.

Cuanto más general y más versátil sea el sensor lógico, más potente es el sistema de fusión. Por ello, después de estudiar los diferentes tipos de sensores disponibles, se le ha dotado de las capacidades necesarias para tratar con los distintos dispositivos sin que el sistema se vea afectado por el cambio de éstos, es decir, de sensores lógicos. Para su descripción y desarrollo se utiliza la metodología orientada a objetos.

B. Utilización de un filtro de Kalman extendido y distribuido para la realizar fusión asíncrona de medidas de estado (vectores de estado y medidas cuantitativas) de distintos sensores, independientemente de su tipo y número.

Las medidas cuantitativas de sensores que permiten conocer el estado del móvil se fusionan mediante un filtro de Kalman, obteniendo así una estima del estado mucho más precisa que con las medidas independientes.

Se escoge un filtro de Kalman extendido debido a que la mayoría de las aplicaciones utilizan un modelo no lineal.

Pero, en esta Tesis se persigue que el sistema de fusión que se obtenga sea robusto, paralelizable, asíncrono e independiente del número de sensores utilizados. Para ello se adaptan las ecuaciones para que se cumplan estas restricciones. Se utiliza un filtro distribuido que es fácilmente paralelizable y capaz de trabajar con sensores distribuidos geográficamente si el sistema lo requiriese. Además, se adaptan sus ecuaciones para permitir un funcionamiento asíncrono de los sensores, de forma que se consigue transformar un comportamiento asíncrono de los sensores en una fusión síncrona que puede ser utilizada en todo sistema. También se ha tenido en cuenta que sea robusto ante fallos y comportamientos anómalos de los sensores mediante distintos test sobre las estimas obtenidas. Todo ello ha sido probado en diversas situaciones, observando que funciona correctamente tanto asíncronamente como ante el fallo o supresión de sensores.

C. Utilización de las redes bayesianas como técnica de fusión de medidas de entorno (vectores de características y medidas cualitativas). Diseño de un método de fusión mediante redes bayesianas y utilización de redes bayesianas dinámicas para dotar al sistema de un comportamiento temporal.

La fusión de las medidas de entorno tiene los mismos requerimientos que las de estado, pero debido a su naturaleza cualitativa, su tipo incertidumbre y la dificultad de disponer de modelos físicos que las describan, hace que un filtro de Kalman no sea un método factible. Por ello se buscan otros métodos de fusión para las medidas cualitativas basados en métodos probabilísticos, heurísticos o en métodos basados en inteligencia artificial.

En esta Tesis se han utilizado redes bayesianas porque tienen una serie de propiedades que las hacen muy adecuadas para su utilización en un sistema versátil y general: (1) su configuración sencilla y directa, que abre posibilidad de la utilización de métodos gráficos para su descripción y configuración; (2) el uso de sensores distribuidos y la estructuración del sistema en capas; (3) la natural adaptación del sistema para su implementación paralela; y (4) la utilización de sensores de forma

asíncrona, ya que la información se fusiona en el momento que esté disponible.

Pero, como en todo método de fusión, se deben tener en cuenta aspectos temporales que tengan en cuenta la evolución tanto del entorno como del propio sistema. Por ello se ha diseñado el sistema de fusión de medidas de entorno con redes bayesianas dinámicas que dotan al método de fusión de un comportamiento dinámico.

Como ejemplo de su utilización, versatilidad y fácil configuración se han escogido los mapas de creencia, donde se ha destacado su funcionamiento como sistema de fusión y su comportamiento temporal.

2.- Arquitectura de control en tiempo real

A. Diseño en tiempo real de una arquitectura de control para un robot móvil autónomo terrestre utilizando la metodología orientada a objetos.

Se ha propuesto una arquitectura de control abierta, que permita mostrar el comportamiento del sistema de integración y fusión sensorial diseñado. Siguiendo con la misma filosofía del sistema sensorial, se persigue su uso en diversos robot móviles autónomos de una forma inmediata o con sólo unas modificaciones sencillas en las tareas necesarias (incluso pueden insertarse otras nuevas).

Se ha realizado una especificación y diseño muy claro y sencillo, aunque formal. Estas propiedades se obtienen directamente si se utiliza una metodología orientada objetos. Por ello se utiliza el lenguaje de especificación orientado a objetos en tiempo real (*ROOM*). La especificación mediante ROOM permite que cualquier usuario pueda adaptar el sistema a las necesidades concretas de su problema, utilizar sus propios algoritmos o probar nuevas técnicas de planificación, control, etc. De igual forma se puede utilizar el mismo sistema de fusión, incorporado en esta arquitectura, para probar distintos sensores o para aplicarlo en diversos robots móviles sin la necesidad de cambiar de sistema de control.

B. Implementación de la arquitectura de control del robot y del sistema de fusión en un sistema operativo de bajo coste.

En esta Tesis se ha procurado que, además de la sencillez en la utilización del sistema de fusión, la implementación de los algoritmos

desarrollados sea llevada a cabo en sistemas de bajo coste. Para ello se ha utilizado un PC como plataforma de trabajo y el sistema operativo DOS. Pero como se necesita un sistema operativo de tiempo real, se ha utilizado RTKernel que dota de funcionalidad en tiempo real al sistema MS-DOS. El lenguaje utilizado para la implementación de todo el sistema es en C y C++, lo que permite una portabilidad total a cualquier otra plataforma con sólo cambiar la parte que utiliza las rutinas de tiempo real y las utilizadas para la representación gráfica (única parte que tiene en cuenta el sistema físico utilizado). Esto permite la adaptación del sistema sensorial a cualquier robot de laboratorio con muy poco esfuerzo y coste, dando a su vez una alternativa a los sistemas que implementados en Linux.

C. Demostración del funcionamiento del sistema de integración y fusión multisensorial en un robot real.

El comportamiento del sistema de fusión multisensorial y de la arquitectura de control propuestos ha sido probado mediante simulaciones y pruebas reales. Estas experiencias pretenden poner de manifiesto todas las propiedades buscadas en esta tesis: (1) la utilización de distintos sensores por un robot, sin modificar el sistema implementado; (2) el funcionamiento distribuido y en paralelo de todas las tareas y sensores involucrados en el sistema; (3) la versatilidad y sencillez del sistema de fusión multisensorial para adaptarlo a las propiedades y cualidades de cada sensor; (4) el funcionamiento asíncrono del sistema y de los sensores; y (5), la facilidad con la que se puede modificar el comportamiento de las tareas, e incluso de insertar nuevas tareas y funcionalidades, para que el robot se adapte a un problema particular.

Además, las pruebas en un robot real muestran los pasos necesarios para adaptar el sistema a un caso concreto y permiten verificar el funcionamiento del sistema y comprobar su eficacia ante problemas reales de ruido, errores, restricciones de tiempo duras, etc.

Este trabajo ha contado con la ayuda de diversos investigadores y con el apoyo de los siguientes Proyectos de Investigación:

1.- Proyecto ROB89-0103-C02-01 del Plan Nacional (1989-1992).

"Modelización, simulación, calibración sensorial y navegación inercial en sistemas de automatización avanzada"

Director: Prof. D. Sebastian Dormido Becomo.

Este proyecto, en lo referente a esta tesis, ha permitido estudiar sistemas de posicionamiento de vehículos autónomos terrestres en entornos exteriores. Estos vehículos están dotados de sistemas de localización inerciales (giróscopos y acelerómetros) y de ayuda externa mediante sistemas de localización, como por ejemplo posicionamiento mediante satélites (GPS). Esto a permitido investigar sobre sistemas inerciales, filtros de Kalman y sensores de posicionamiento.

2.- Proyecto TAP94-0832-C02-01 de la CICYT (1994-1997).

"Generador automático de código de control en tiempo real e integración multisensorial: aplicación a procesos mecánicos y químicos"

Director: Prof. D. Jesús Manuel de la Cruz García.

Este proyecto a permitido profundizar en los sistemas de integración y fusión multisensorial, centrándose en un robot móvil en entornos interiores (hacia donde a derivado esta tesis) y en un sistema general de fusión e integración sensorial. En el se ha diseñado un generador de código en tiempo real cuya filosofía y conocimientos se han utilizado en el desarrollo de la arquitectura de control del robot presentada en esta tesis.

Valoración de los resultados y posibles trabajos futuros

En este trabajo se presenta un diseño sencillo de un sistema de integración y fusión multisensorial para robot móviles autónomos. En concreto se ha utilizado para robot en interiores, pero puede utilizarse en otras aplicaciones sin ninguna dificultad.

El sistema se ha implementado en un sistema operativo en tiempo real de bajo coste, sobre PC y en C y C++. Esto permite su utilización en múltiples aplicaciones robóticas de laboratorio que utilicen diversos sensores, así como la prueba y la evaluación de la eficacia de distintas configuraciones de sensores.

La experimentación sobre un robot real ha mostrado la sencillez de su utilización y la facilidad de su adaptación a cualquier problema, obteniéndose resultados muy satisfactorios ya que se ha comprobado:

1.- La sencillez con la que se adapta y configura el sistema para la utilización de diversos sensores.

2.- La eficacia del sistema para trabajar con procesos distribuidos y con sensores de forma independiente.

3.- La ventaja que supone disponer de un sistema que permite obtener buenas precisiones con sensores de bajo coste debido a la posibilidad de fusionar sus medidas.

4.- La ventaja de integrar distintos sensores para obtener aspectos que con un único sensor sería imposible.

Por último, con este trabajo se dispone de una plataforma para futuras investigaciones en robótica e integración multisensorial. El sistema de fusión y la arquitectura de control implementadas asientan la base sobre la que se pueden diseñar y probar distintos algoritmos de planificación, control y fusión entre otras, e investigar en técnicas de visión artificial, redes neuronales artificiales, control mediante lógica difusa, etc.

Esta Tesis engloba muchas áreas de investigación y es imposible abarcarlas a todas. Así, la construcción de los circuitos que manejan el sistema sensorial y el robot, el software para su manejo, las comunicaciones entre procesos, el uso de sistemas distribuidos o el control del robot entre otros, implican la colaboración de distintos grupos de trabajo. En este caso han participado, además del grupo del área de Ingeniería de Sistemas y Automática de esta Universidad, investigadores del

departamento de Ingeniería Eléctrica y Electrónica, del departamento de Informática y del departamento de Física Aplicada de la Universidad de Castilla la Mancha e investigadores del departamento de Informática y Automática de la Universidad Nacional de Educación a distancia.

Además de las posibles investigaciones que pueden realizarse sobre esta plataforma, esta Tesis deja abierta la posibilidad a futuros trabajos de máxima actualidad, como:

- Utilización del sistema de fusión e integración multisensorial con un robot móvil en entornos exteriores, en el que las necesidades sensoriales son muy importantes y la complejidad de su integración y fusión aumenta enormemente.
- Probar el sistema de fusión en vehículos inerciales que trabajan con sistemas inerciales y GPS puesto que sus requerimientos de tiempo y precisiones son completamente distintos.
- Inclusión de la teoría de decisión y manejo de sensores sobre el sistema de fusión implementado para determinar qué tipo de sensores son más convenientes fusionar en cada caso. Aprovechando en ese caso las redes bayesianas para utilizarlas como redes de decisión.
- Utilización de otros sensores de entorno distintos a los basados en ultrasonidos, como por ejemplo cámaras para visión estereoscópica o radar laser.
- En esta tesis se ha apuntado la gran potencia de las redes bayesianas dinámicas en el campo de la fusión multisensorial. A partir de éstas se puede llegar a aunar las dos técnicas de fusión utilizadas en una sola.
- Diseñar con un editor gráfico, basado en ROOM, la arquitectura de control del robot propuesta para que sea modificable de forma gráfica. Pero aun llegar más lejos y extender esta cualidad a la incorporación o supresión de distintos sensores, de forma que se pueda configurar de forma gráfica el sistema sensorial disponible y los sensores que se utilizan en cada caso.
- Utilizar una distribución de las tareas de control descritas en múltiples procesadores, lo que permitirá algoritmos más complejos y velocidades del robot más altas.

Todos los trabajos señalados representan campos relacionados con el tema de esta Tesis en los cuales se investiga y se siguen desarrollando sistemas, teorías y aplicando nuevas técnicas actualmente, y marcan muchas de las líneas de trabajo de nuestro grupo de investigación.

Publicaciones realizadas

Con motivo del trabajo desarrollado se han publicado los siguientes trabajos:

- G. Pajares, J.M. de la Cruz y J. A. López-Orozco. *Stereo Matching Using Herbbian Learning*. IEEE Trans. on System, Man, and Cybernetics. Part B. Aceptada para publicación.
- G. Pajares, J.M. de la Cruz y J. A. López-Orozco. *Improving Stereovision Matching Through Supervised Learning*. Pattern Analysis and Applications. Ed. Springer-Verlag. Vol 1, No. 1. Pág. 105-120. Londres, 1998.
- O. Rodríguez Polo, J.M. de la Cruz, J.A. López-Orozco, E. Besada. *Diseño de prácticas de control digital utilizando multi-tarea en tiempo real y modelado orientado a objetos*. XIX Jornadas de Automática. Pag. 39-43. Madrid. Septiembre, 1998.
- J.A. López-Orozco, J.M. de la Cruz, E. Domínguez, E. Besada, O. R. Polo. *An Open Sensing Architecture to Autonomous Mobile Robots*. Proc. of the IEEE Int. Symposium on Computational Intelligence in Robotics and Automation (CIRA). ISCI/CIRA/ISAS Joint Conference. Pág. 610-615. Gaithersburg, MD. Septiembre, 1998.
- J.A. López-Orozco, J.M. de la Cruz, J. Sanz y J. Flores. *Multisensor Fusion of Enviroment Measures Using Bayesian Networks*. Proc. of the Int. Conf. on Multisource-Multisensor Information Fusion. Pag. 487-493. Las Vegas, USA. Julio, 1998.
- J. A. López-Orozco, J. M. de la Cruz, P. Ruiperez, J. Aranda. *Generador de trayectorias para la simulación de sistemas de navegación inercial*. II Congreso de Usuarios de MATLAB (MATLAB'96). Madrid. Septiembre, 1996.

- J. A. López Orozco, P. Ruiperez, J. Aranda, J. M. de la Cruz, J. M. Pérez, J. Minguez: *Simulación de un sistema GPS en Matlab.* I Congreso de Usuarios de MATLAB (MATLAB'95). Madrid. Septiembre, 1995.
- J.A. López Orozco, P. Ruipérez, J.M. de la Cruz, J. Aranda. *Modeling and simulation of navigation systems: an INS simulation Matlab toolbox.* EUROSIM'95. Pags. 607-612. Viena. Septiembre, 1995.
- J. Aranda, J.M. de la Cruz, P. Ruipérez, J.A. López Orozco and S. Dormido. *Simulation study of techniques for the self-alignment of a strapdown inertial navegation system.* Proceedings of the Conference on Modelling and Simulation. European Simulation Multiconference. Pag. 611-615. Barcelona. Junio, 1994.
- J. A. López Orozco. *Algoritmos para un sistema de navegación strapdown.* Trabajo de doctorado. Mayo, 1993.

Bibliografía

Abidi 92 — M.A. Abidi y R.C. Gonzalez *The Use of Multisensor Data for Robotic Applications*. IEEE Transactions on Robotic and Automation, Vol. 6, No 2. 1992.

Agapito 93 — de Agapito J.A., de Agapito L., Schneider M., García Rosa R. Y de Pedro T. *Fuzzy logic applied to gas sensors*. Sensors and Actuators B, 15-16, pp. 105-109, 1993.

Agent 91 — Agent, A. *The Advantages of Absolute Encoders for Motion Control*. Sensors, pp. 19-24. Abril 1991.

Alami 98 — Alami R., Chatila R., Fleury S., Ghallab M. y Ingrand F. *An Architecture for Autonomy*. International Journal of Robotics Research, Vol. 17, No. 4. Abril, 1998.

Allen 87 — Allen P.K *A Framework for Implementing Multisensor Robotic Task*. . Proc. ASME Int. Comput. Engr. Conf. And Exhibition. New York, 1987.

Almond WWW — Russell Almond. Página Wed sobre Software de Redes de Creencia. *http://bayes.stat.washintong.edu/almond/belief.html.*

Andersen 89 — Andersen K., Olesen K.G., Jensen F.V. y Jensen F. *HUGIN – a shell for building Bayesian belief universes for expert systems*. Procs. Of the Eleventh Int. Joint Conference on Artificial Intelligence. Detroit, 1989.

Anderson 79 — Anderson B.D.O., Moore J.B. *Optimal Filtering*. Ed. Prentice-Hall, New Jersey, 1979.

Antony, 97 — Antony, R. T. *Database Support to Data Fusion Automation*. Proceedings of the IEEE. Vol 85, No. 1. Enero 1997.

Aranda 89 — Aranda J. *Modelización y calibración de giróscopos laser y alineamiento de*

sistemas ligados (strapdown). Tesis doctoral. Dpto. Informática y Automática. UNED, 1989.

Arkin 87 — Arkin, R.C. *Motor Schema Based Navigation For a Mobile Robot: A Approach to Programming by Behavior.* Proc. of the IEEE International Conference on Robotics and Automation. pp. 264-271. 1987

Assi 98 — Assi Abdel-Azim. *Data Fusion for Medical Applications.* Proc. of the International Conference on Multisource-Multisensor Information Fusion. FUSION'98. Las Vegas, USA. Julio, 1998.

Astrom 80 — Astrom K.J. *Maximum Likelihood and Prediction Error Methods.* Automática. Vol. 16, 1980.

Aurenhammer 91 — Aurenhammer F. *Voronoi diagrams: a survey of a fundamental geometric data structure.* ACM Comput. Surv., 23. 1991.

Avolio 93 — Avolio, G. *Principles of Rotary Optical Encoders.* Sensors. pags. 10-18. Abril 1993.

Awad 96 — Awad M., Kuusela J. y Ziegler J. *Object-Oriented Technology for Real-Time systems: A practical Approach using OMT and Fusion.* Prentice Hall. Upper Sadder River, N.Y., 1996.

Azouaoui 98 — Azouaoui O. y Chohra A. *Evolution, Behaviour, and Intelligence of Autonomous Robotic Systems (ARS).* 3rd IFAC Symposium on Intelligent Autonomous Vehicles. pp. 139-145. Marzo, 1998. España.

Bares 89 — Bares J., Herbert M., Kanade T., Krotkov E. Michell T., Simmons R., Whittaker W. *Ambler - An Autonomous Rover for Planetary Exploration.* IEEE Computer Magazine, June 1989

Barnett 81 — Barnett, J.A. *Computacional methods for a mathematical theory of evidence.* Proc. 7th Int. Joint Conf. Artif. Intell. Vancouver, Canadá, pags. 868-875, 1981.

Bar-Shalom 90 — Bar-Shalom, Y. *Multitarget-multisensor tracking: advanced applications.* Artech House, Norwood, 1990.

Bar-Shalom 93 — Bar-Shalom Y. y Li X.R. *Estimation and Tracking. Principles, Techniques and Software.* Artech House, 1993.

Barshan 95 — Barshan B. Y Durrant-Whyte H.F. *Inertial Navigation Systems for Mobile Robots.* IEEE Trans. on Robotics and Automation, Vol. 11, No 3. Junio, 1995.

Baumgartner 94 — Eric T. Baumgartner and S. B. Skaar. *An Autonomous Vision-Based Mobile*

Robot. IEEE Transactions on Automatic Control, Vol. 39, n° 3, pp. 493-502. Marzo 1994

Beer 91 — Beer R.D., Chiel H.J. y Sterling L.S. *A biological perspective on autonomous agent design.* In Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back. Edited by P. Maes. MIT Press, 1991.

Bell 94 — Bell D.A., Levine S.P., Koren Y., Jaros L.A., Borenstein J. *Design Criteria for Obstacle Avoidance in a Shared-Control System.* RESNA'94 Annual Conference, June 1994.

Bendat 71 — Bendat J. Y Piersol A.G. *Random Data: Analysis and Measurement Procedures.* John Wiley & Son, Inc. 1971.

Berg 83 — Berg, R. F. *Estimation and Prediction for Maneuvering Target Trajectories.* IEEE Transactions on Automatic Control. Marzo, 1983

Bishop 95 — Bishop C.M. *Neural Networks for Pattern Recognition.* Oxford University Press. Oxford, 1995.

Bishop 97 — Bishop R. *The Potential for Vehicle-Higway Automation in the United States.* Proc. IEE International Conference on Int. Rob. and Sys. pp. 1215-1216. 1997.

Bissé 95 — Bissé E. *Optimal Path Generation for a Simulated Autonomous Mobile Robot.* Autonomous Robots, pp. 1-18. 1995.

Bogler 87 — Bogler P.L. *Shafer-Dempstre Reasoning with Applications to Multisensor Target Identification Systems.* IEEE Trans. on Systems, Man, and Cybernetics. Vol. 17, No. 6. Noviembre, 1987.

Borenstein 89 — Borenstein J. y Koren Y. *Real-time obstacle avoidance for fast mobile robots.* IEEE Trans. on Systems, Man, and Cybernetics. Vol. 19, No. 5. Septiembre, 1989.

Borenstein 91a — Borenstein J. y Koren Y. *The vector field histogram – fast obstacle avoidance for mobile robots.* IEEE Journal of Robotics and Automation, Vol. 7, No. 3. Junio, 1991.

Borenstein 91b — Borenstein J. y Koren Y. *Histogramic in-motion mapping for mobile robot obstacle avoidance.* IEEE Journal of Robotics and Automation, Vol. 7, No. 4, 1991.

Borenstein 95a — Borenstein J., Everett H.R., y Feng L. *Where Am I? - Systems and Methods for Mobile.* Edition of a University of Michigan internal report, Septiembre 1995.

Borenstein 95b — Borenstein J., Wehe D., Feng L. Y Koren Y. *Mobile Robot Navigation in Narrow Aisles with Ulrasonic Sensors*. ANS 6th Topical Meeting on Robotics and Remote Systems. Monterey, CA. Febrero, 1995.

Borenstein 97 — Borenstein J., Everett H.R., Feng, L. y Wehe D. *Mobile Robot Positioning C. Sensors and Techniques*. Journal of Robotic Systems. Monográfico, Mobile Robots. Vol. 14, No. 4, Abril 1997.

Borenstein 97a — Borenstein J. *The GuideCane - A Computerized Travel Aid for the Active Guidance of Blind Pedestrians*. Proc. Of the IEEE Int. Conf. on Robotics and Automation, Albuquerque. Abril, 1997.

Borenstein 97b — Borenstein J. y Evans J. *The OmniMate Mobile Robot- Design, Implementation, and Experimental Results*. IEEE International Conf. On Robotics and Automation. Alburqueque, 1997.

Britting 71 — Britting, K.R. *Inertial Navigation System Analysis*. Wiley, New York, 1971.

Brooks 86 — R. A. Brooks. *A Robust Layered Control System For A Mobile Robot*. IEEE Journal of Robotics and Automation, vol. RA-2, nº 1. Marzo, 1986

Brooks 91 — Brooks R.A. *Elephants don't play chess*. In Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back. Edited by P. Maes. MIT Press, 1991.

Bruin 97 — Bruin L. Et al. *Navstar GPS User Equipment and Applications*. NATO team. US Air Force Space Division, NAVSTART GPS Joint Program Office. Los Angeles. Junio, 1987.

Buchanan 94 — Buchanan, B.G. y Shortliffe, E.H. *RuleBased Expert Systems : The MYCIN Experiments of the Stanford Heuristic Programming Project*. Reading. Ed. Addison-Wesley, 1984.

Budenske 97 — Budenske J y Gini M. *Sensor Explication: Knowledge-based Robotic Plan Execution through Logical Objects*. IEEE Trans. On Systems, Man, and cybernetics. Vol 27, No 4. Agosto, 1997.

Buede 97 — Buede D.M. y Girardi P. *A Target Identification Comparison of Bayesian and Dempster-Shafer Multisensor Fusion*.

Campion 96 — Campion G., Bastin G., y D'Andréa-Novel B.. *Structural Propierties and Classification of Kinematic and Dynamic Models of Wheeled Mobile Robots*. IEEE Trans. on Robotics and Automation. Vol. 12, No. 1, 1996.

Campos 98 — Campos D. y Llinas J. *Possibilities for Improved Formal Processing Methods*

for Situation Assessment. Proc. of the International Conference on Multisource-Multisensor Information Fusion. FUSION'98. Las Vegas, USA. Julio, 1998.

Castellanos 98 — Castellanos J.A:, Martínez J.M., Neira J. y Tardós J.D. *Experiments in multisensor mobile robot localization and map building.* 3rd IFAC Symposium on Intelligent Autonomous Vehicles. Madrid. Marzo, 1998.

Chang 96 — Chang C.C. y Song K. *Ultrasonic Sensor Data Integration and Its Application to Environment Perception.* Journal of Robotics Systems, Vol. 13, No. 10, 1996.

Chaudhuri 85 — Chaudhuri B.B. *Applications of Quadtree, Octree, and Binary tree Descomposition Techniques to Sharpe Analysis and Pattern Recognition.* IEEE Trans. on Pattern Analysis and Machine Intelligence. Vol. 7, No. 6. Noviembre, 1985.

Chee 96 — Chee B.Y., Sherman Y., Lang T. Y Tse P.W.T. *Fuzzy Mobile Robot Navigation and Sensor Integration.* Proceedings of the Fifth IEEE Int. Conf. on Fuzzy Systems (FUZZ-IEEE 96). Septiembre, 1996.

Chellappa 97 — Chellappa, R. et al. *On the Positioning of Multisensor Imagery for Exploitation and Target Recognition.* Proceedings of the IEEE. Vol 85, No. 1. Enero 1997.

Conners 97 — Conners R.W., Kline D.E. y Drayer T.H. *Machine Vision Technology for the Forest Products Industry.* IEEE Computer. Julio, 1997.

Cooper 90 — Cooper G.F. *The computational complexity of probabilistic inference using Bayesian belief networks.* Artificial Intelligence. Vol. 42. 1990.

Cox 89 — Cox, I.J. *Blance: Position Estimation for an Autonomous Robot Vehicle.* Proc. of the IEEE/RSJ International Workshop on Robots and Systems Japan, 1989.

Cuena 86 — Cuena, J. y col. *Inteligencia Artificial : Sistemas expertos.* Ed. Alianza-Informática, 1986.

Dagum 93 — Dagum P. y Luby M. *Approximating probabilistic inference in Bayesian networks is NP-hard.* Artificial Intelligence. Vol. 60. 1993.

Dam 96 — Van Dam J.W.M., Kröse J.A. y Groen F.C.A. *Adaptative Sensor Models.* IEEE/SICE/RSI Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems, Washington D.C. Diciembre, 1996.

Danchick 90 — Danchick, R. J. *Navy Navigation satellite system status.* En Navigation.

Land, Sea, Air, and Space. Edited by M. Kayton.. IEEE Aerospace and Electronic Systems Society: 207-210. 1990

Daniel 97 — Daniel M.M., Willsky A.S. *A Multiresolution Methodology for Signal-Level Fusion and Data Assimilation with Applications to Remote Sensing.* Proceedings of the IEEE, Vol 85, No 1, January, 1997.

Davis 91 — Davis L. *Handbook of Genetic Algorithms*. Ed. Van Nostrand Reinhold. New York, 1991.

Dean 89 — Dean T. y Kanazawa K. *A model for reasoning about persistence and causation*. Computational Intelligence. Vol. 5. 1989.

Delingette 95 — Delingette H. *Trajectory Generation with Curvature Constraint based on Energy Minimization.* IEEE Int. Workshop on Intelligent Robots & Systems. Noviembre, 1995.

Dempster 67 — Dempster, A.P. *Upper and lower probabilities induced by a multivalued mapping*. Annals of Mathematical Statistics, No 38, pags. 325-339, 1967.

Dempster 68 — Dempster, A.P. *A Generalization of Bayesian Inference.* J. Royal Statistical Soc., vol. 30, 1968, pags. 205-247.

Díez 94 — Diez J. *Sistema experto bayesiano para ecocardiografía*. Tesis doctoral. UNED, 1994.

Díez 96 — Díez, F.J. *Local conditioning in Bayesian networks*. Artificial Intelligence. Vol. 87, 1996.

Doyle 95 — Doyle R.S. y Harris C.J. *Multi-Sensor Data Fusion for Helicopter Guidance using neuro-fuzzy estimation algorithms*. Int. Conf. on System, Man, and Cybernetics. Octubre, 1995.

Dubois 92 — Dubois D. y Prade H. *Combination of Fuzzy Information in The Framework of Possibility Theory.* En Data Fusion in Robotics and Machine Intelligence. Eds. Abidi M.A. y Gonzalez R.C. Academic Press, 1992.

Dubov 95 — Bubov D. *Mars Pathfinder*. JPL Technical Report 95-0892. Editado en Aerospace America, Julio, 1995.

Duda 73 — Duda, R.O. y Hart, P. *Pattern Classification and scene analysis*. J. Wiley & Sons. New York, 1973.

Duda 79 — Duda, R.O., Hart, P., Konolige, K. y Reboh, R. *A Computer-Based Consultant for Mineral Exploration*. Tech. report, SRI International.

Duda 81 — Duda, R., Gaschinig, J. y Hart, P. *Model design in the prospector consultant*

system for mineral exploration. Expert Systems in the Microelectronics Age. Ed. Edinburgh Univ. Press, pags. 153-167, 1981.

Durrant-Whyte 88 Durrant-Whyte, H.F. *Integration, Coordination and Control of Multi-sensor Robot Systems*. Kluwer, Boston, 1988.

Durrant-Whyte 90 Durrant-Whyte H.F., Rao B.Y.S. y Hu H. *Toward a Fully Decentralized*. Proceedings IEEE Int. Conf. Robot. Automat., 1990.

Durrant-Whyte 96 Durrant-Whyte 96, H.F. *Guided Vehicle for Cargo Handing Applications*. International Journal Robotics Research. Pp. 407-440. Octubre, 1996.

Edelsbrunner 87 Edelsbrunner H. *Algorithms in Combinatorial Geometry*. Springer-Verlag, Heidelberg, 1987.

Elfes 86 A. Elfes. *A distributed control architecture for an autonomous mobile robot*. Artificial Intelligent, Vol. 1, nº 2, 1986.

Elfes 87 A Sonar-Based Real-World Mapping and Navigation. IEEE Journal of Robotics and Automation. Vol. 3, No 3, 1987.

Elfes 90 Elfes, A. *Occupancy Grids: A Stochastic Spatial Representation for Active Robot Perception*. Proc. of the Sixth Conference on Uncertainty in AI. Julio 1990.

Elfes 92 Elfes A. *Multi-source Spatial Fusion Using Bayesian Reasoning*. En Data Fusion in Robotics and Machine Intelligence. Eds. Abidi M.A. y Gonzalez R.C. Academic Press, 1992.

Ellowitz 92 Ellowitz H.I. *The Global Positioning System*. J. Microwave. Special Report. Abril, 1992.

Espinosa 98 Espinosa F., García R., Mazo M., López E. y Mateos R. *Modelling and simulation of the Kinematic and Dunamic Behavior of a Fork-Lift-Turck*. 3rd IFAC Symposium on Intelligent Autonomous Vehicles. pp. 75-79. Marzo, 1998. España.

Evans 92 J. Evans, B. Krishnamurthy, B. Barrows, T. Skewis, ans V. Lumelsky. *Handling Real-World Motion Planning: A Hospital Transport Robot*. IEEE Control Systems. pp. 15-19. Febrero 1992.

Everett 95 Everett, H. R. *Sensors for mobile Robots. Theory and Application*. A. K. Peters. Wellesley, Massachusetts, 1995.

Fayyad 96 Fayyad U.M., Piatetsky-Shapiro G. Y Smyth P. *From Data Mining to Knowledge Discovery: An Overview*. En Advances in Kowlegde Discovery and Data Mining, pags. 1-36. AAAI PressMenlo Park, CA 1996.

Fleury 94 — Fleury S., Herrb M. y Chatila R. *Design of a modular architecture for autonomous robot.* Proc. IEEE Int. Conf. on Robotics and Automation. San Diego, California, 1994.

Friess 88 — Friess P. *Empirical Accuracy of Positions Computed from Airborne GPS Data.* En High Precision Navigation. Editado por K. Linkwitz y U. Hangleiter. Pags. 163-175. Springer-Verlang. Berlin, 1988.

Fukunaga 90 — Fukunaga K. *Introduction to Statistical Pattern Recognition.* Ed. Academic Press. New York, 1990.

Fukunaga 93 — Fukunaga K. *Statistical Pattern Recognition.* En Handbook of Pattern Recognition and Computer Vision. Eds. Chen C.H., Pau L.F. y Wang P.S.P. World Scientific Publishing Company, 1993.

Fung 83 — Fung, P. T. y Grimble, M. J. *Dymanic Ship Positioning Using a Self-Tuning Kalman Filter.* IEEE Transactions on Automatic Control. Marzo, 1983.

Gat 90 — Gat, E., Slack, M. G., Miller, D. P., Firby, R.J. *Path Planning and Execution Monitoring for a Planetary Rover.* Proc. of IEEE International Conference on Robotics and Automation. 1990, pag. 20-25

Gat 90 — Gat E., Slack M.G., Miller D.P., Firby R.J. *Path Planning and Execution Monitoring for a Planetary Rover.* IEEE International Conference on Robotics and Automation, 1990.

Geist 92 — Geist G.A. y Sunderam V.S. *Network-based concurrent computing on the PVM system.* Concurrency: Practice & Experience, Vol. 4, No. 4, Junio, 1990

Geist 94 — Geist A., Beguelin A., Dongarra J. Manchek R., Jiang W. y Sunderam V. *PVM: A User's Guide and Tutorial for Networked Parallel Computing.* MIT Press, 1994.

Geist 96 — Geist G.A., Kohl J.A. y Papadopoulos P.M. *PVM and MPI: a Comparision of Features.* Calculateurs Parallels. Vol. 8, No. 2, 1996.

Gelb 74 — Gelb A., *Applied Optimal Estimation.* The MIT Press, Cambridge, 1974.

Gertler 92 — Gertler J.J. y Anderson K.C. *An evidential reasoning extension to quantitative model-based failure diagnosis.* IEEE Transactions on Systems, Man, and Cybernetics, Vol 22, No ¿??. Marzo, 1992.

Gill 81 — Gill E.P., Murray W. y Wrigth M.H. *Practical Optimization.* Academic Press. San Diego, 1987.

Giralt 84 — Giralt, G. Chatila R., Vaisset M. *An Integrated Navigation and Motion Control System for Autonomous Multisensory Mobile Robots.* International

Symposium on Robotics Research, 1984.

Goldberg 89 — Goldberg D.E. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, Reprinted with corrections, 1989

Golub 89 — Golub, G.H. and Van Loan, C.F. *Matrix Computations.* The Johns Hopkins University Press, Baltimore 1989.

Gomma 93 — Gomma H. *Software Design Methods for Concurrent and Real-Time Systems.* Reading, MA. Addison-Wesley, 1993.

Gomma 95 — Gomma H. *Reusable Software Requirements and Architectures for Families of Systems.* The Journal of Systems and Software. Vol. 28, No 3. Marzo, 1995.

Gonzalez 96 — Gonzalez J., Ollero A. *Estimación de la posición de un Robot Móvil.* Revista de Informática y Automática de la AEIA, Vol 29, No 4. Diciembre, 1996.

Goodwin 77 — Goodwin G.C. y Payne R.L. *Dynamic System Identification. Experiment design and data analysis.* Ed. Academic Press. New York, 1977.

Grewal 93 — Grewal M.S. y Andrews A. P. *Kalman Filtering. Theory and Practice.* Prentice-Hall, Inc. 1993.

Groover 89 — Groover M.P., Weiis M., Nagel R.N. y Odrey N.G. *Robotica industrial. Tecnología, programación y aplicaciones.* Mc. Graw-Hill, 1989.

Gullekson 97 — Gullekson G. *ATM Protocol Stack Design with the ROOM Methodology.* Communications Systems Design. Vol. 3, No 12. Diciembre, 1997.

Hager 90 — Hager G.D. *Task-Directed Sensor Fusion and Planning. A computational approach.* Kluwer, Boston, 1990.

Hall 92 — Hall, D.L. *Mathematical Techniques in Multisensor Data Fusion.* Artech House, Boston, 1992.

Harmon 86 — Harmon, S.Y. *Autonomous Vehicles.* Encyclopedia of Artificial Intelligence. pp. 39-45. De. Wiley, New York.

Harmon 87 — Harmon S.Y. *The Ground Surveillance Robot (GSR): An Autonomous Vehicle Designed to Transit Unkown Terrain.* IEEE Journal of Robotics and Automation, Vol 3, No 3, June 1987.

Hashemipour 88 — Hashemipour, S.Y., Roy, S. y Laub, A.J. *Decentralized Structures for Parallel Kalman Filtering.* IEEE Trans. Auto. Control, 33(1), pags 88-94. 1988.

Haykin 94 — Haykin S. *Neural Networks. A comprehensive Foundation.* IEEE Press. Toronto, 1994.

Heckerman 86 — Heckerman, D. *Probabilistic interpretations for MYCIN's certainty factors*. Uncertainty in Artificial Intelligence. Ed. L.N. Kanal y J.F. Lemmer, pags. 167-196, 1986.

Heckerman 95 — Heckerman D., Breese J. y Rommelse K. *Decision-Theoric Trobleshooting*. Communications of the ACM, Vol. 38, No. 3. Marzo, 1995.

Henderson 84 — Henderson T.C. y Shilcrat E. *Logical Sensor Systems*. Journal of Robotics Systems. Vol. 1, No 2. 1990.

Henderson 87 — Henderson T.C y Weitz E. *Multisensor Integration in a Multiprocessor Enviroment*. Proc. ASME Int. Comput. Engr. Conf. And Exhibition. New York, 1987.

Henkel 87 — Henkel, S. *Optical Encoders : A Review*. Sensors, pags. 9-12. Septiembre 1987.

Herbert 88 — Herbert M., Kanade T., Kweon I. *3-D Vision Techniques for Autonomous Vehicles*. NSF Range Image Understanding Workshop, 1988.

Hilera 95 — Hilera J.R: y Martínez V.J. *Redes Neuronales Artificiales. Fundamentos, modelos y aplicaciones*. Ed. RA-MA, 1995.

Holle 90 — Holle, S. *Incremental Encoders Basics*. Sensors, pags. 22-30. Abril 1990.

Hopfield 82 — Hopfield J. *Neural Networks and physical systems with emergent colective computational abilities*. Proc. of the National Academy of Sciences, Vol. 79. 1982 Reimpreso en Neurocomputing (Anderson y Rosenfeld Eds.) MIT Press, 1988.

Hurteau 94 — Hurteau R., DeSantis S., Begin E. y Gagner M. *Laparascopic Surgey Assisted by a Robotic Cameraman*. Proc. Interantional IEEE Conf. On Robotics. and Automation. Vol 3. Pp. 2286-2289.1994.

Hutchinson 92 — Hutchinson S.A. y Kak A.C. *Multisensor Strategies Using Dempster-Shafer Belief Accumulation*. En Data Fusion in Robotics and Machine Intelligence. Eds. Abidi M.A. y Gonzalez R.C. Academic Press, 1992.

Jackins 83 — Jackins C.L. y Tanimoto S.L. *Quad-Trees, Oct-Trees, and K-Trees: A Generalized Approach to Recursive Decomposition of Euclidean Space*. IEEE Trans. on Pattern Analysis and Machine Intelligence. Vol. 5, No. 5. Noviembre, 1983.

Jain 95 — Jain Ze-Sun y Chau Y.A. *Optimum Multisensor Data Fusion for Image Change Detection*. IEEE Trans. on Systems, Man, and Cybernetics. Vol. 25, No. 9, Septiembre, 1995.

Jefferys 92 — Jefferys W.H. y Berger J.O. *Ockham's Razor and Bnayesian Analysis.* American Scientist, Vol. 80. Enero, 1992.

Jensen 96 — Jensen F.V. *An introduction to Bayesian networks.* Ed. Springer Verlag. New York, 1996.

Kafrissen 84 — Kafrissen E. Y Stephans M. *Industrial Robots and Robotics.* Reston, Reston, 1984.

Kalata 84 — Kalata, P. R. *The Tracking Index: A Generalized Parameter for α-β and α-β-γ Target Trackers.* IEEE Transactions on Aerospace and Electronic Systems. Vol 20, No. 2, pags. 174-182. Marzo 1984.

Kam 97 — Kam, M., Zhu, X. Y Katala, P. *Sensor Fusion for Mobile Robot Navigation.* Proceedings of the IEEE. Vol 85, No. 1. Enero 1997.

Kanayama 97 — Kanayama Y.J. y Hartman B.I. *Smooth Local-Path Planning for Autonomous Vehicles.* International Journal of Robotics Research. Vol. 16, No. 3, Junio, 1997.

Kelly 94 — Kelly A. *Essential Kinematics for Autonomous Vehicles.* Informe técnico CMU-RI-TR-94-14. Carnegie Mellon University, Pittsburgh. USA, 1994.

Kessler 92 — Kessler et al. *Functional Description of the Data Fusion Process.* Tech. Rep. Office of Naval Technol. Naval Air Development Center. Enero 1992.

Khabit 86 — Khabit O. *Real-Time Obstacle Avoidance for Manipulators and Mobile Robots.* International Journal of Robotics Research, Vol. 5, No. 1. 1986.

Kim 83 — Kim J.H y Pearl J. *A computational model for combined causal and diagnostic reasoning in inference systems.* Proceedings of the Eighth Int. Joint Conference on Artificial Intelligence. Alemania . Karlsruhe, 1983.

Kjaerulff 92 — Kjaerulff U. *A computational scheme for reasoning in dynamic probabilistic networks.* Proc. of the Eigth Conference on Uncertainty in Artificial Intelligence. 1992.

Kohl 96 — Kohl J.A. y Geist G.A. *XPVM 1.0 User's Guide.* Computer Sicence and Mathematics Division. Oak Ridge National Laboratory. ORNL/TM-12981. Noviembre, 1996.

Kriegman 87 — Kriegman D.J., Triendl E., Binford T.O. *A Mobile Robot: Sensing, Planning, and Locomotion.* IEEE International Conference on Robotics and Automation, 1987.

Kriegman 89 — Kriegman D.J., Triendl E., Binford T.O. *Stereo Vision and Navigation in Buildings for Mobile Robots.* IEEE Transsctions on Robotics and

Automation, Vol 5, No 6, December 1989.

Krotkov 96 — Krotkov E. y Simmons R. *Perception, Planning, and Control for Autonomous Walking With the Ambler Planetary Rover*. International Journal of Robotics Research. pp. 155-180. Abril, 1996.

Lacey 93 — Lacey G., Waldron R., Dinten J. y Lilley F. *Flexible Multi-Sensor Inspection System For Solder-Joint Analysis*. En Machine Vision Applications, Architectures, and Systems Integration II. SPIE Conference. Septiembre, 1993.

Lallement 98 — Lallement A., Dufaut M. y Husson R. *Sensor Cooperation and World Perception of the Autonomous Mobile Robot Romane*. 3rd IFAC Symposium on Intelligent Autonomous Vehicles. Madrid. Marzo, 1998.

Langsam 97 — Langsam, Y., Augenstein, M.J. y Tenenbaum, A.M. *Estructuras de datos con C y C++*. 2º Edición. Ed. Prentice Hall, 1997.

Larson 76 — Larson, R.E. y Casti, J.L. *Principles of Dynamic Programming*. Marcel Dekker Inc. New York, 1976.

Latombe 91 — Latombe, J.C. *Robot Motion Planning*. Kluwer Academic Publishers. London.

Lauritzen 88 — Lauitzen S.L. y Spiegelhalter D.J. *Local computations with probabilities on graphical structures and their application to expert systems*. J. Royal Statist. Soc. Ser. Vol. 50. 1988.

Lee 97 — Lee D. y Recce M. *Quantitative Evaluation of the Exploration Strategies of a Mobile Robot*. The Int. Journal of Robotics Research, Vol. 16, No. 4. Agosto, 1997.

Leick 90 — Leick, A. *GPS Satellite Surveying*. John Wiley & Sons, 1990.

Leonard 90 — Leonard J. *Directed Sonar Sensing for Mobile Robot Navigation*. Tesis doctoral. Robotic Research Group. Department of Engineering Science, University of Oxford, 1990.

Leonard 92 — Leonard J. y Durrant-Whyte H.F. *Directed Sonar Sensing for Mobile Robot Navigation*. Kluwer Academic Publisher. Boston, 1992.

Lewis 86 — Lewis F.L. *Optimal Estimation with an introduction to Stochastic Control Theory*. John Wiley & Sons, Inc. 1986.

Libby 96 — Libby, E.W. y Maybeck, P.S. *Sequence Comparison Techniques for Multisensor Data Fusion and Target Recognition*. IEEE Trans. On Aerospace and Electronic systems, vol 32, No 1, enero 1996.

López-Cachero 88 López Cahero, M. *fundamentos y métodos de estadística.* Ediciones Pirámide S.A. Ed. actualizada, 1988.

Lozano-Pérez 79 Lozano-Pérez T. y Wesley M.A. *An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles.* Communications of the ACM, Vol. 22, No. 10, 1979.

Luo 89 Luo, R. C. y Kay M.G. *Multisensor Integration and Fusion in Intelligent Systems.* IEEE Trans. on Systems, Man, and Cybernetics. Vol 19, No 5 Sep. 1989.

Luo 92 Luo R.D: y Kay M.G. *Data Fusion and Sensor Integration: State-of-art 1990s.* En Data Fusion in Robotics and Machine Intelligence. Eds. Abidi M.A. y Gonzalez R.C. Academic Press, 1992.

Maes 91 Maes P. *Situated agents can have goals.* In Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back. Edited by P. Maes. MIT Press, 1991.

Manyika 94 Manyika J. y Durrant-Whyte H. F. *Data Fusion and Sensor Management. A descentralized information-theoretic approach.* Ellis Horwood series, New York, 1994.

Maravall 93 Maravall D. *Reconocimiento de formas y Visión Artificial.* Ed. RA-MA. Madrid, 1993.

Martin 86 Martin, G. J. *Gyroscopes May Cease Spinning.* IEEE Spectrum, pags. 48-53. Febrero, 1986.

Matthies 88 Matthies L y Elfes A. *Integration of Sonar and Stereo Range Data Using a Grid-Based Representation.* Proc. of the IEEE Int. Conf. on Robotics and Automation, pp 727-733, 1988.

Maybeck 79 Maybeck, P. S. 1979. "Stochastic Models, Estimation, and Control". Vol 1. Academic Press. London.

McMillan 88 McMillan J.C. *An integrated Land Navigations System.* IEEE Position Location and Navigation Symposium, 1988.

Menezes 95 Menezes P, Dias J, Araújo H. y Almeida A. *Low Cost Sensor Based Obstacle Detection and Description. Experiments with Mobile Robots using Grid Representation.* Proc. Of the Fouth Int. Symposium on Experimental Robotics. ISER'95. Agosto, 1995.

Meng 93 Meng, M. y Kak, A.C. *Mobile Robot Navigation Using Neural Networks and Nonmetrical Environment Models.* IEEE Control Systems Octubre 1993.

Pags. 30-39

Minsky 69 — Minsky M. y Papert S. *Perceptrons*. MIT Press, 1969.

Mitchell 88 — Mitchell, J.S. *An algorithmic Approach to Some Problems in Terrain Navigation*. Artificial Intelligence, 37. 1988, pags 171-201

Mitchell 88 — Mitchell J.S.B. *an Algorithmic Approach to Some Problems in Terrain Navigation*. Artificial Intelligence, 37, 1988.

Moravec 88 — Moravec H.P. *Sensor fusion in certainty grids for mobile robots*. AI Magazine. Pp 61-74, 1988.

Morawski 89a — Morawski P. *Understanding Bayesian Networks*. AI Expert, pp. 44-48. Mayo, 1989.

Morawski 89b — Morawski P. *Programming Bayesian Belief Networks*. AI Expert, pp. 74-89. Agosto, 1989.

MPI94 — MPI Forum. *MPI: A message passing interface standard*. International Journal of Supercomputer Applications. Vol 8, No. 3/4, 1994.

Müller 96 — Müller J.P. *The Design of Intelligent Agents. A Layared Approach*. Lecture Notes in Artificial Intelligence. Springer, 1996.

Murphy 98 — Murphy R.R. *Dempster-Shafer Theory for Sensor Fusion in Autonomous Mobile Robots*. IEEE Trans. on Systems, Man, and Cybernetics. Vol. 14, No. 2. Abril, 1998

Nandhakumar 97 — Nandahkumar N. y Aggarwal J.K. *Physics-Based Integration of Multiple Sensing Modalities for Scene Interpretation*. Proceedings of the IEEE, vol. 85, No. 1, Enero. 1997

Neapolitan 90 — Neapolitan, R. E. *Probabilistic Reasoning in Expert Systems. Theory and Algorithms*. John Wiley & Sons, inc. 1990.

Nelson 88 — Nelsson W.L., Cox, I.J. *Local Path Control for an Autonomous Vehicle*. Proc. of the IEEE International Conference on Robotics and Automation, Philadelphia 1988.

Nelson 88 — Nelson W. L. y Cox I.J. *Local Path for an Autonomous Vehicle*. Proceedings of the IEEE Int. Conf. on Robototics and Automation. Abril, 1988.

Nicholson 92 — Nicholson A.E. y Brady J.M. *The data asociation problem when monitoring robot vehicles using dynamic belief networks*. Proc. of 10th European Conference on Artificial Intelligence. Viena. Austria, 1992.

Nicholson 94 — Nicholson A.E. y Brady J.M. *Dynamic Belief Networks for Discrete*

Monitoring. IEEE Trans. on System, Man, and Cybernetics. Vol. 24, No. 11. Noviembre, 1994.

Nickson 85 — Nickson, P. *Solid-State Tachometry*. Sensors, pags. 23-26. Abril 1985.

Nilsson 69 — Nilsson, N.J. *A Mobile Automaton: An Application of Artificial Intelligence Techniques*. Procceedings of 1st Int. Joint Conference on Artificial Intelligence. Washington D.C., 1969.

Olin 91 — Olin K.E., Tseng D.Y. *Autonomous Cross-Country Navigation. An Integrated Perception and Planning System*. IEEE Expert (DARPA/VISION & NAVIGATION). pp. 16-30. Agosto 1991.

Pao 93 — Pao, Y. *Neural net computing for pattern recognition*. En Handbook of Pattern Recognition and Computer Vision. Eds. Chen C.H., Pau L.F. y Wang P.S.P. World Scientific Publishing Company, 1993.

Papoulis 90 — Papoulis A. *Probability and Statistics*. Prentice-Hall Inc. Englewood Cliffs, 1990.

Parkinson 83 — Parkinson B.W. y Gilbert S.W. *NAVSTAR: Global Positioning System – Ten Years Later*. Proceedings of the IEEE, Vol. 71, No 10. Octubre 1983.

Payton 90 — Payton D.W., Rosenblantt J.K., Keirsey D.M. *Plan Guided Reaction*. IEEE Transactions on Systems, Man, and Cybernetics, Vol 20, No 6, Nov. 1990.

Payton 91 — D. w. Payton, T. E. Bihari. *Intelligent Real-Time control of Robotic Vehicles*. Comunications of the ACM. Vol. 34, N° 8. pp. 48-63. Agosto 1991

Payton 91 — Payton D.W., Bihari T.E., *Intelligent Real-Time Control of Robotics Vehicles*. Communications of the ACM. Vol 34, No 8, 1991.

Payton 91 — Payton, D.W. *Internalized Plans: A Representation for Action Resources*. In Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back. Edited by P. Maes. MIT Press, 1991.

Pearl 82 — Pearl J. *Reverend Bayes on inference engines: A distributed hierarchical approach*. Proceedings of the Conf. on Artificial Intelligence. Pittsburgh, Pennsylvania, 1982.

Pearl 88 — Pearl, J. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Ed. Morgan Kaufmann, Palo Alto, 1988.

Pham 92 — Pham T.M. *Kalman Filter Mechamization for INS Airstart*. IEEE AES System Magazine, January, 1992.

Pires 98 — Pires G, Numes U. y de Almeida A.T. *Robchair- A Semi-Autonomous Wheelchair for Disabled People*. 3rd IFAC Symposium on Intelligent

Autonomous Vehicles. pp. 648-652. Marzo, 1998. España.

Pradhan 94 — Pradhan M., Middleton G.M. y Henrion M. *Knoledge engineering for large belief networks*. Proc. of Uncertainty in Artificial Intelligence. Seattle, 1994.

Press 94 — Press W.H, Teukolsky S.A., Vetterling W.T., Flannery B.P. *Numerical Recepies in C. The Art of Scientific Computing*. Ed. Cambrige University Press. 2ª edición, 1994.

Prieto 95 — Prieto A. M. Fundamentos Geométricos del Diseño en Ingeniería. Edición de la Escuela Superior de Ingenieros Aeronauticos. Madrid, 1995.

Rao 93 — Rao, B.S.Y., Durrant-Whyte, H.F. y Sheen, J.A. *A Fully Decentralized Multi-Sensor system For Tracking and Surveillance*. The Int. Journal of Robotics Research, vol. 12 (1), Feb. 1993.

Rembold 88 — Rembold U. *The Kalrsruhe Autonomous Mobile Assembly Robot*. Proc. of the IEEE International Conference on Robotics and Automation, 1988.

Rich 94 — Rich, E. y Knight, K. *Artificial Intelligence*. Ed. Mc Graw-Hill, 1994.

Richardson 88 — Richardson, J.M : Marsh K.A. *Fusion of Multisensor Data*. Int. Journal of Robotics Research Vol 7, No 6, 1988.

Rintanen 91 — Rintanen K. *Smooth Low Trajectory Control for Autonomous Mobile Robot*. PANORAMA, Technical Report. Mayo, 1991.

Ruipérez 94 — Ruipérez P. *Filtro de Kalman en el alineamiento de sistemas de navegación ligados*. Tesis doctoral. Dpto. Informática y Automática. UNED, 1989.

Russell 95 — Russell S.J. y Norvig P. *Artificial Intelligence. A modern Approach*. Prentice-Hall. New Jersey, 1995.

Saffiotti 91 — Saffiotti A. y Umkehrer E. *PULCINELLA. A General Tool for Propagating Uncertainty in Valuation Networks*. Procs. Of 7th Conference on Uncertainty in Artificial Intelligence. Los Angeles, California, 1991.

Saffiotti 94 — Saffiotti A. y Umkehrer E. *Automatic Construction of Valuation systems from General Clauses*. Trans. on Systems, Man, and Cybernetics. Vol. 24, No. 11. Noviembre, 1994.

Salcudean 94 — Salcudean S.E. y Yan J. *Towards a Force-Reflecting Motion-Scaling System for Microsurgey*. Proc. Interantional IEEE Conf. On Robotics. and Automation. Vol 3. Pp. 2286-2289.1994.

Salgian 98 — Salgian G. y Ballard D. *Using Visual Routines to Drive in a Virtual Enviroment*. 3rd IFAC Symposium on Intelligent Autonomous Vehicles. pp.

565-570. Marzo, 1998. España.

Santos WWW — Eugene Santos. Página Wed sobre herramientas de Redes de bayesianas. *http://www.afit.af.mil/Schools/EN/ENG/LABS/AI/BayesianNetworks/tools3.html.*

Schneider 96 — Schneider M. *Planificación basada en Percepción Activa para la Navegación de un Robot Móvil. Instituto de Automatica Industrial (CSIC). Tesis Doctoral, 1996.*

Selic 95 — Selic B., Gullekson G. Y Ward P.T. *Real-Time Object-Oriented Modeling.* John Wiley & Sons, Inc. New York, 1995.

Shachter 86 — Shachter R.D. *Evaluating influence diagrams.* Operations research, 34. pp. 871-882. 1986.

Shafer 76 — Shafer, G. A *Mathematical Theory of Evidence.* Princeton Univ. Press, 1976.

Shetty 96 — Shetty S., Alouani A.T. *A multisensor Tracking System With an Image-Based Maneuver Detector.* IEEE Trans. On Aerospace and Electronic systems, vol 32, No 1, enero 1996.

Shimoga 96. — Shimoga, K.B. *Robot Grasp Synthesis Algoritms: A Survey.* Pp. 230-266. Junio, 1996.

Shortliffe 75 — Shortliffe, E.H. y Buchanan, B.G. *A model of inexact reasoning in medicine.* Mathematical Biosciences, pags. 351-379, 1975.

Simmons 94 — Simmons R.G. *Structured Control for autonomous robots.* IEEE Trans. on Robotics and Automation. Vol. 10, No. 1. Febrero, 1994.

Snir 95 — Snir M., Otto S., Huss-Lederman S., Walker D. y Dongarra J. *MPI: The Complete Reference.* MIT Press, 1995.

Sorenson 70 — Sorenson, H.W. *Least-Squares Estimation: from Gauss to Kalman.* IEEE Spectrum. July 1970.

Stieler 82 — Stieler B. y Winter H. *Gyroscopic instruments and their application to flight testing.* AGARD Flight Test Intrumentation Series. NATO. Septiembre, 1982.

Sunderam 90 — Sunderam V.S. *PVM: A framework for parallel distributed computing.* Concurrency: Practice & Experience, Vol. 2, No. 4, Diciembre, 1990.

Synder 92 — Snyder, S., Schipper B., Vallot L., Parker N. y Spitzer C. *Differential GPS/Inertial Navigation Approach/Landing Flight Test Results.* IEEE AES Magazine., May, 1992.

Thomopoulos 90 Thomopoulos, S.C.A. *Sensor Integration and Data Finish*. J. Robotic Syst. Vol. 7, No 3, Junio 1990, pags. 337-372.

Thompson 77 Thompson, A.M. *The Navigation System of the JPL Robot*. Proceedings of the International Joint Conference on Artificial Intelligence, 1977. pp. 749-757.

Thorpe 88 Thorpe C, Herbert M.H., Kanade T., Shafer S.A. *Vision and Navigation for the Carnegie-Mellon Navlab*. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 10, No 3, May 1988.

Thorpe 91a C. Thorpe, M. Herbert, T. Kanade and S. Shafer. *Toward Autonomous Driving: The CMU Navlab. Part I - Perception*. IEEE Expert (DARPA/VISION & NAVIGATION). pp. 31-42. Agosto 1991.

Thorpe 91b C. Thorpe, M. Herbert, T. Kanade and S. Shafer. *Toward Autonomous Driving: The CMU Navlab. Part II - Architecture and Systems*. IEEE Expert (DARPA/VISION & NAVIGATION). pp. 44-52. Agosto 1991.

Thorpe 97 Thorpe C. *Mixed Traffic and Automated Higways*. Proc. IEE International Conference on Int. Rob. and Sys. pp. 1011-1017. 1997.

Turk 88 Truk M.A., Morgenthaler D.G., Gremban K.D., Marra M. *VITS - A Vision System for Autonomous Land Vehicle Navigation*. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 10, No 3, 1988.

Valavanis 97 Valavanis K.P., Gracanin D., Matijasevic M., Kolluru R. y Demetriou G.A. *Control Architectures for Autonomous Underwater Vehicles*. IEEE Control Systems. Diciembre, 1997.

Varshney 96 Varshney P.K. *Distributed Detection and Data Fusion*. Springer-Verlag. New York, 1996.

Viswanathan 97 Viswanathan R. And Varshney P. K. *Distributed Detection With Multiple Sensors: Part I- Fundamentals*. Proceedings of the IEEE, 85 (1), pags. 54-63. Enero, 1997.

Waltz 86 Waltz, E. L. *Data Fusion for C3I: A Tutorial:* Command, Control, Communications Intelligence (C3I) Handbook. EW Communications, Inc. Palo Alto, California, 1986, pags. 217-226.

Waltz 90 Waltz, E.L. y Llinas, J. *Multisensor Data Fusion*. Artech House, Boston, 1990.

Weisbin 89 Weisbin C.R., Saussure G., Einstein J.R., Pin F.G. *Autonomous Mobile Robot Navigation and Learning*. IEEE Computer Magazine, June 1989.

White 88 White, F. et al. *A Model for Data Fusion.* SPIE Conference on Sensor Fusion. Orlando. Abril 1988.

White 91 White, F. *Data Fusion Lexicon.* Data Fusion Subpanel for C3. Code 4202. California, 1991.

Wong 90 Wong S.K.M., Yao Y.Y., Bollmann P. Y Burger H.C. *Axiomatization of qualitative belief structure.* IEEE Transactions on Systems, Man, and Cybernetics, Vol 21, No ¿??. Enero, 1990.

Yager 94 Yager R.R., Fedrizzi M. Y Kacprzyk J. Eds. *Advances in the Dempster-Shafer Theory of Evidence.* New York. Wiley, 1994.

Yanco 98 Yanco H.A. *Integrating Robotic Research:A Survey of Robotic Wheelchair Development.* AAAI Spring Symposium on Integrating Robotic Research, California 1998.

Yang 94a Yang J. Y Singh M.G. *An evidential reasoning approach for multiple-attribute decision making with uncertainty.* IEEE Transactions on Systems, Man, and Cybernetics, Vol 24, No ¿??. Enero, 1994.

Yang 94b Yang J. Y Sen P. *A general multilevel evaluation process for hybrid MADM with uncertainty.* IEEE Transactions on Systems, Man, and Cybernetics, Vol 24, No ¿??. Octubre, 1994.

Yeddanapudi 97 Yeddanapudi M., Bar-Shalom Y. and Pattipati K.R. *IMM Estimation for Multitarget-Multisensor Air Traffic Surveillance.* Proceedings of the IEEE, vol. 85, No. 1, Enero. 1997

Yen 90 Yen J. *Generalizing the Dempster-Shafer Theory to Fuzzy Sets.* IEEE Trans. on Systems, Man, and Cybernetics. Vol. 20, No. 3. Mayo, 1990.

Zadeh 75 Zadeh, L.A. *Fuzzy Logic and approximate reasoning.* Synthese, No 30, pags. 407-428, 1975.

Zadeh 78 Zadeh, L.A. *Fuzzy Sets as a basis for a theory of possibility.* Fuzzy Sets and Systems, vol. 1 No 1, pags. 3-28, 1978.

Zadeh 79 Zadeh, L.A. *On the validity of Dempster's rule of combination od evidence.* Univ. California, Berkeley, Memoria UCB/ERL M79/24, 1979.

Zheng 89 Zheng Y.F. *Integration of Multiple Sensors into a Robotic System and its Performance Evaluation.* IEEE Trans. on Robotics and Automation, Vol 5 No 5 , Oct. 1989.

Fernández 98 Fernández J. *Técnicas avanzadas de procesamiento morfológico y textural. Aplicación a la inspección de baldosas cerámicas.* Tesis doctoral. E.T.S.I.

Industriales. Universidad Politécnica de Madrid, 1998.

Hedenborn 95 Hedenborn P. y Bolmsjö G. *Robotics in automated inspection*. Int. Journal of Production Economics. No. 41, 1995.

Hedenborn 94 Hedenborn P. y Agren B. *Enhanced use of sensors in welding applications*. Proc. of the 10th ISPE/IFAC Int. Conference on CAD/CAM, Robotics and Factories of the Future CARs & FOF. Ottawa, Canada. 1994.

Balaguer 98 Balaguer C., Pastor J.M., Giménez A., Padrón V.M. y Abderrahim M. *ROMA: Multifuncional autonomous self-supported climbing robot for inspection applications*. 3rd IFAC Symposium on Intelligent Autonomous Vehicles. pp. 139-145. Marzo, 1998. España

Zong 98 Zong G.H., Tso S.K., Liu R. y Wang W. *An autonomous mobile robot dor glass-roof cleaning*. 3rd IFAC Symposium on Intelligent Autonomous Vehicles. pp. 139-145. Marzo, 1998. España

Escalera 95 De la Escalera A. *Percepción del entorno y localización de robots móviles autónomos*. Tesis doctoral. E.T.S.I. Industriales. Universidad Politécnica de Madrid, 1995.

Castellanos 98 Castellanos J.A., Martínez J.M., Neira J. y Tardós J.D. *Experiments in Multisensor Mobile Robot Localization and Map Building*. 3rd IFAC Symposium on Intelligent Autonomous Vehicles. pp. 139-145. Marzo, 1998. España

Tardos 91 Tardós J.D. *Integración Multisensorial para Reconocimiento y Localización de Objetos en Robótica*. Tesis doctoral. Dpto. Ingeniería Eléctrica e Informática, Universidad de Zaragoza. Febrero, 1991.

Armingol 98 Armingol J.M., Moreno L.E., Garrido S., de la Escalera A. y Salichs M.A. *Mobile Robot Localization Using a Non-linear Evolutionary Filter*. 3rd IFAC Symposium on Intelligent Autonomous Vehicles. pp. 139-145. Marzo, 1998. España

Torres 93 V. Torres, J.A.G. Fortes, A. Ollero, J. Gonzalez, A.J. Reina. *Descripción del sistema sensorial del robot móvil RAM-1*. 3º Congreso Nacional de la Asociación Española de Robótica. Zaragoza. Noviembre, 1993.

González 98 González J.R. y Fernández J.A. *A Flexible Software System for Implementing Robots Control Architectures*. 3rd IFAC Symposium on Intelligent Autonomous Vehicles. pp. 139-145. Marzo, 1998. España

López-Orozco 95b López Orozco J.A., Ruipérez P., de la Cruz J.M. y Aranda J.. *Modeling and simulation of navigation systems: an INS simulation Matlab toolbox.*

EUROSIM'95. Viena, 1995.

López-Orozco 95a López Orozco J. A., Ruiperez P., Aranda J., de la Cruz J. M, Pérez J. M. y Minguez J.: *"Simulación de un sistema GPS en Matlab"*. I Congreso de Usuarios de MATLAB (MATLAB'95). Madrid, 1995.

Jochem 93 Jochem T.M, Pomerleau D.A. y Thorpe C.E. *MANIAC: A Next Generation Neurally Based Autonomous Road Follower*. Proc. of the Int. Conf. on Intelligent Autonomous Systems.Pittsburgh, Pennsylvania, USA. 1993.

Pomerleau 93 Pomerleau D.A. *Neural nertwork for mobile robot guidance*. Kluwer Academic. Boston, 1993.

Booch 94 Booch G. *Object-oriented analysis and design with applications*. Addison-Wesley. Object Technology Series. 2ª edición. 1994.

Rumbaugh 91 Rumbaugh J., Blaha M., Premerlani W., Eddy F. y Lorensen W. *Object-Oriented Modeling and Design*. Prentice Hall, 1991.

Douglass 98 Real-Time Uml: Developing Efficient Objects for Embedded Systems. Addison-Wesley. Object Technology Series. 1998.

Chong 90 Chong C.Y., Mori S. y Chang K.C. *Distributed multitarget multisensor tracking*. En Multitarget Multisensor Tracking: Advanced Applications. Ed. Artech House. Norwood, 1990.

Gasós 99 Gasós J. *Integration Linguistic Descriptions and Sensor Observations for the Navigation of Autonomous Robots*. Para aparecer en Fuzzy logic techniques for autonomous vehicle navigation. A. Saffioti y D. Driankov, Editores. Lecture Notes in Computer Sciences. Springer-Verlag, 1999.

Powel 98 Powel B.D. *Real-Time Uml: Developing Efficient Object for Embedded Systems*. Object Technology Series. Addison-Wesley. Enero, 1998.

Selic 98 Selic B. y Rumbaugh J. *Using UML for Modeling complex Real-Time Systems*. Technical Paper. The Object Management Group. ObjectTime Co. Marzo, 1998.

Samet 89 Samet H. *Applications of spatial data structures: computer graphics, image processing, and GIS*. Series in Computer Science. Addison-Wesley. 1989.

Xiao 97 Xiao J., Michalewicz Z., Zhang L. y Trojanowski K. *Adaptative Evolutionay Planner/Navigator for Mobile Robots*. IEEE Trans. on Evolutionary Computation, Vol. 1, No. 1. Abril, 1997.

Fletcher 87 Fletcher, R. *Practical Methods of Optimization*. John Wiley & Sons. 2º Edición. 1987.

Laplante 93 Laplante, P. *Real-Time Systems Design and Analysis. An Engineer's Handbook*. Ed. IEEE Press. New York, 1993.

Benett 94 Benett, S. *Real Time Computer Control. An Introduction*. Ed. Prentice Hall. New York, 1994.

Apéndice A

Cálculo de la credibilidad en redes bayesianas

Definiciones

Para calcular la credibilidad de un nodo X en una red bayesiana es necesario introducir varios conceptos que nos permitirán comprender mejor la obtención de las ecuaciones siguientes. Entre otros conceptos destacamos la *independencia condicional* y la *separación direccional*.

Hipótesis de independencia condicional

En la figura A-1 podemos ver que Y_1 y Y_2 son dependientes siempre que el valor de X sea desconocido, pero si se conoce el valor de X, por ejemplo +x, entonces el valor de Y_1 es independiente de Y_2, es decir,

$$P(y_2 \mid +x, y_1) = P(y_2 \mid +x) \tag{A.1}$$

ó

$$P(y_2, y_1 \mid +x) = P(y_2 \mid +x) \cdot P(y_1 \mid +x) \tag{A.2}$$

El carácter ascendente y descendente de la propagación de la evidencia en la red permite realizar inferencias abductivas (qué estado de un objeto explica mejor lo medido por los sensores) e inferencias predictivas (cual es la probabilidad de medir algo en concreto por un sensor).

Debe tenerse en cuenta que en las redes bayesianas no sólo los arcos aportan información sobre las dependencias entre variables, sino que la ausencia de un arco es una forma implícita de aportar información. Por ejemplo, en la figura A-1, X y U son independientes y por lo tanto $P(x \mid u) = P(x)$. Sin embargo, si se conoce el valor de $Y_2 = +y_2$, se vuelven dependientes:

$$P(x \mid u,+y_2) \neq P(x \mid +y_2) \tag{A.3}$$

Esta asimetría, reflejo de la causalidad implícita en la red, es la característica esencial de las redes bayesianas y se conoce como *separación direccional*.

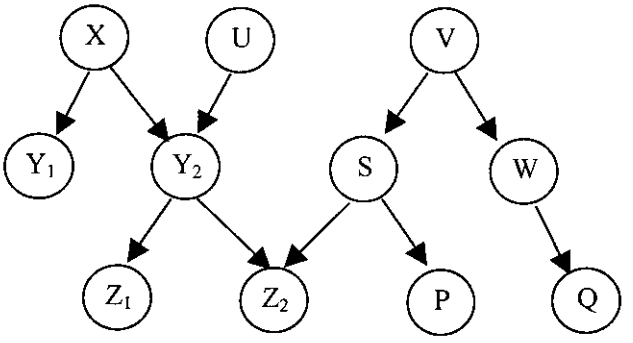


Figura A-1. Red bayesiana de ejemplo

Separación direccional

Llamada *d-separación* por Pearl [Pearl 88], se dice que existe separación direccional en un grafo dirigido acíclico y conexo si, dada una distribución de probabilidad sobre sus variables, en un nodo cualquiera X el conjunto de sus padres, $pa(X)$, separa condicionalmente este nodo de otro subconjunto $\{Y\}$ en el que no haya descendientes de X. Esto es,

$$P(x \mid pa(x), y) = P(x \mid pa(x)) \tag{A.4}$$

Se indica conexo aunque no es necesario por dos motivos, primero porque la mayoría de los algoritmos y propiedades de las redes bayesianas sólo son correctos para grafos conexos; y segundo, porque en caso de tener, por ejemplo, dos partes inconexas se podrían considerar como dos modelos independientes y trabajar con ellos como dos redes separadas.

Así, a partir de las definiciones anteriores se puede definir una red bayesiana como: un grafo acíclico dirigido y conexo mas una distribución de probabilidad sobre sus variables, que cumple la propiedad de separación direccional.

Por último, destaquemos algunas propiedades de las redes bayesianas que se deducen de las anteriores:

1.- Si un nodo no tiene padres, por ejemplo el nodo V de la figura A-1, entonces la ecuación (A.4) se traduce en $P(v \mid pa(v)) = P(v \mid \emptyset) = P(v)$ que es la probabilidad a priori del nodo V. Además, aplicando la misma condición tenemos que $P(s \mid u) = P(s)$, puesto que el nodo U no pertenece a los padres de S. Por lo tanto se puede concluir que dos nodos cualesquiera que no tengan un antepasado en común son independientes a priori.

2.- Si V es antepasado de S y éste es a su vez antepasado de P y no existe ningún otro camino que permita llegar desde V a P, entonces V y P quedan condicionalmente separados por S.

$$P(p \mid s,v) = P(p \mid s) \tag{A.5}$$

3.- Dos nodos que son (a priori o condicionalmente) independientes continuarán siéndolo mientras no se conozca el valor de alguno de sus descendientes. Por ejemplo, X y W en la figura A-1 son independientes aunque se conozca el valor de P (no es un descendiente común).

$$P(x \mid w,p) = P(x \mid p) \tag{A.6}$$

4.- Dada una red bayesiana, su distribución de probabilidad puede expresarse mediante el producto de las distribuciones condicionales de cada uno de los nodos de sus padres, una sencilla demostración puede verse en [Diez 94].

$$P(x_1,x_2,...,x_n) = \prod_i P(x_i \mid pa(x_i)) \tag{A.7}$$

Esta propiedad es muy importante, puesto que permite describir una red en términos de la probabilidad condicionada en cada nodo (o la probabilidad a priori en nodos sin padres), en lugar de dar una probabilidad conjunta en función de los nodos existentes, lo que necesitaría un número exponencial de parámetros. Sin embargo, el número de parámetros necesarios para dar la probabilidad condicional es proporcional al número de nodos (siempre que el número de padres y valores de cada variable sean finitos).

Algoritmo para propagación de la evidencia

Para mostrar cómo se propaga y calcula la evidencia en una red bayesiana utilizaremos el algoritmo basado en paso de mensajes desarrollado en [Kim 83] y posteriormente ampliado por Pearl para árboles [Pearl 88].

Estudiemos primero el caso más simple posible (silogismo), véase la figura A-2, para introducir la notación utilizada y comprender el mecanismo de propagación de la evidencia.

Consideremos un cierto atributo x, se define la credibilidad de x como la probabilidad de ese atributo teniendo en cuenta la existencia de la evidencia e:

$$Cr(x) = P(x \mid e) \tag{A.8}$$

Aplicando el teorema de Bayes pero condicionado a la evidencia existente sobre las causas del nodo X, e^+, y la evidencia sobre los efectos de X, e^-; obtendremos,

$$Cr(x) = P(x \mid e^+, e^-) = \frac{P(e^- \mid x, e^+) \cdot P(x \mid e^+)}{P(e^- \mid e^+)} \tag{A.9}$$

Pero como sabemos que ningún arco puede ir de la zona e^- a la zona de e^+ y aplicando la separación direccional, se pueden considerar independientes puesto que se conoce x. Así, podemos tomar $P(e^- \mid x, e^+) = P(e^- \mid x)$ y obtendremos

$$Cr(x) = \frac{P(e^- \mid x) \cdot P(x \mid e^+)}{P(e^- \mid e^+)} \tag{A.10}$$

Esta ecuación puede interpretarse como:

$P(e^- \mid e^+) = \alpha$, puede considerarse como un *factor de normalización.*

$$P(e^- \mid e^+) = \sum P(e^- \mid x) \cdot P(x \mid e^+) \tag{A.11}$$

$P(x \mid e^+) = \pi(x)$ ≡ *Credibilidad debida a las causas*. Probabilidad aportada a X debida a sus causas (normalmente se incluyen factores a priori) y mide las relaciones de x con sus causas,

$P(e^- \mid x) = \lambda(x)$ ≡ *Probabilidad de los efectos de X* supuesto X, es la probabilidad teórica que mide las relaciones de

X con sus efectos (incluyen factores a posteriori),

Con esta interpretación obtenemos una nueva expresión para la credibilidad,

$$Cr(x) = \alpha \cdot \lambda(x) \cdot \pi(x) \tag{A.12}$$

Para calcular π(x) podemos aplicar la descomposición condicional a todos los valores de la causa *U*:

$$\pi(x) = P(x \mid e^+) = \sum_u P(x \mid u, e^+) \cdot P(u \mid e^+) = \sum_{u_j} P(x \mid u_j) \cdot \pi(u_j) \tag{A.13}$$

Donde *P*(x/*u*) corresponde a la matriz de relaciones entre *U* y *X*; el sumatorio representa la suma de cada uno de los valores que puede tomar u_j para el valor del nodo *X* (*x*) en estudio por el valor de la credibilidad de u_j ($\pi(u_j)$). Es decir, es la suma de la credibilidad de los valores u_j ponderados por la influencia que tiene cada u_j en la credibilidad de *x*.

Para calcular λ(x), procedemos de la misma forma, pero teniendo en cuenta el efecto de *X* y aplicando la propiedad 2, ecuación (0.5), de separación condicional a $P(e^- \mid x, y)$, se obtiene:

$$\lambda(x) = P(e^- \mid x) = \sum_y P(e^- \mid x, y) \cdot P(y \mid x) = \sum_{y_j} \lambda(y_j) \cdot P(y_j \mid x) \tag{A.14}$$

Donde *P*(*y*|*x*) corresponde a la matriz de relaciones entre los nodos *X* e *Y*; el sumatorio representa la suma de cada una de las probabilidades de que ocurra un efecto y_j conociendo la existencia de *x* por la probabilidad de los efectos de y_j ($\lambda(y_j)$). Es decir, es la suma de la probabilidad de los efectos de los distintos valores que puede tomar *y* (y_j) ponderados por el efecto que tiene cada valor y_j sobre el valor de *X* considerado.

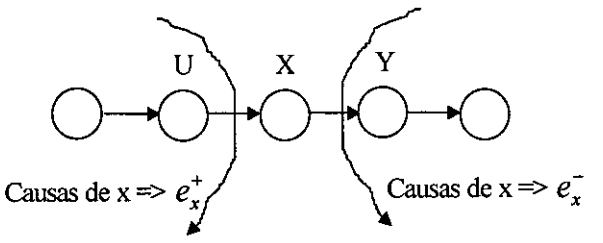


Figura A-2. Estructura en cadena.

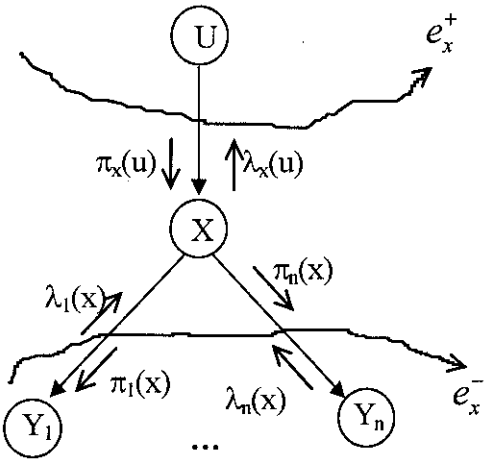


Figura A-3 . Estructura en árbol.

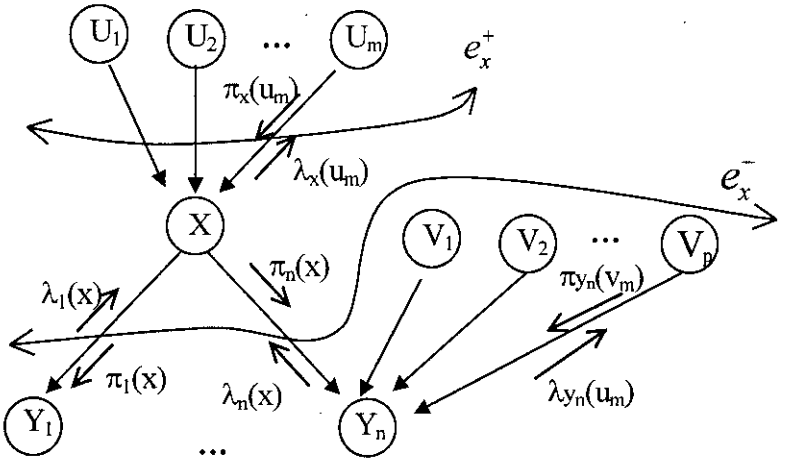


Figura A-4 . Red simplemente conexa (poliárbol).

Nodo *X*	
Emite mensajes	Recibe mensajes
m mensajes $\lambda_x(u)$	m mensajes $\pi_x(u)$
n mensajes $\pi_{y_i}(x)$	n mensajes $\lambda_{y_i}(x)$

Tabla A.1. Mensajes emitidos y recibidos por cada nodo.

En el caso de los árboles y poliárboles (una red simplemente conexa) las ecuaciones son más complejas. Teniendo en cuenta los mensajes que un nodo puede emitir o recibir, tabla A.1, partiendo de la ecuación (A.10) y teniendo en cuenta la figura A-4, las ecuaciones para propagar la evidencia pueden calcularse como sigue.

Para calcular $\pi(x)$ podemos aplicar la descomposición condicional a todas las causas U de X:

$$\begin{aligned}\pi(x) &= P(x \mid e^+) = \sum_{\bar{u}} P(x \mid \bar{u}, e_X^+) \cdot P(\bar{u}, e_X^+) = \\ &= \sum_{\bar{u}} P(x \mid \bar{u}) \cdot P(u_1, e^+_{U_1 X}, \ldots, u_n, e^+_{U_n X}) = \sum_{\bar{u}} P(x \mid \bar{u}) \cdot \prod_{i=1}^{m} P(u_i, e^+_{U_i X}) = \\ &= \sum_{\bar{u}} P(x \mid \bar{u}) \cdot \prod_{i=1}^{m} \pi_X(u_i)\end{aligned} \tag{A.15}$$

donde,

$$\pi_{Y_j}(x) = P(x, e^+_{XY_j}) = P(x, e_X^+, e^-_{XY_k} k \neq j) = P(x \mid e_X^+) \cdot \prod_{k \neq j}^{n} P(e^-_{XY_k} \mid x) \tag{A.16}$$

$$\pi_{Y_j}(x) = \pi(x) \cdot \prod_{k \neq j}^{n} \lambda_{Y_k}(x) \tag{A.17}$$

Para calcular $\lambda_{Y_k}(x)$, recordemos que los descendientes de X pueden tener más padres además de X, véase figura A-4,

$$\begin{aligned}\lambda_{Y_j}(x) &= P(e^-_{XY_j} \mid x) = \sum_{y_j} \sum_{\bar{v}} P(e^-_{Y_j}, y_j, e^+_{XY_j}, v \mid x) = \\ &= \sum_{y_j} \sum_{\bar{v}} P(e^-_{Y_j} \mid y_j) P(y_j \mid x, v) P(e^+_{XY_j}, v \mid x)\end{aligned} \tag{A.18}$$

$$\lambda_{Y_j}(x) = \sum_{y_j} \left[\lambda(y_j) \sum_{\bar{v}} P(y_j \mid x, \bar{v}) \prod_{l=1}^{p} \pi_{Y_j}(v_l) \right] \tag{A.19}$$

Y finalmente $\lambda(x)$ se calcula como anteriormente:

$$\lambda(x) = P(e^-_{XY_1}, \ldots, e^-_{XY_m} \mid x) = \prod_{j=1}^{m} \lambda_{Y_j}(x) \tag{A.20}$$

Estas fórmulas son recursivas, por lo tanto necesitamos una condición de terminación para que el algoritmo esté completo. Para ello recordemos que si un nodo no tiene padres $\pi(x) = P(x)$ que es su probabilidad a priori y que es uno de

los parámetros que define la red. De esta forma tenemos resulto el problema de la condición de terminación y además vemos cómo puede incluirse la información a priori en la red.

Pero también es necesario conocer para todo nodo hoja Z (nodo sin hijos) $\lambda(z)$. En caso de que no exista ninguna información sobre él se introduce el mismo valor a cada una de las variables de ese nodo, por ejemplo $\lambda(z) = \vec{1}$. Puesto que un vector constante no modifica el valor de la creencia de Z y por tanto no influye en el resultado final. Sin embargo, si se obtiene cierta evidencia sobre ese nodo debe incluirse en $\lambda(z)$. Si el nodo sobre el cual obtenemos una cierta evidencia no es un nodo hoja, por ejemplo S, el valor obtenido no debe sustituir a $\lambda(s)$ sino modificarla de la forma mostrada en la ecuación (A.21) [Morawski 89b]. Con esto tenemos el mecanismo para introducir evidencia externa en cualquier nodo de la red. Esta evidencia externa corresponderá con las observaciones que se hayan realizado con los sensores disponibles, los cuales no tienen porqué encontrarse en nodos hoja, como se muestra en el ejemplo [Morawski 89a].

$$\lambda(s) = \lambda_{anterior}(s) \cdot \lambda_{externa}(s) \qquad \text{(A.21)}$$

Implementación distribuida

El algoritmo anterior es puramente recursivo, pero podría programarse como una algoritmo iterativo si se necesitase menos gasto de memoria. En una implementación distribuida cada variable de la red corresponde con un proceso o procesador (si se dispone de un procesador para cada variable). En este caso además de almacenar las probabilidades a priori y condicionales correspondientes a ese nodo debe almacenar la estructura local de la red, es decir, conocer a qué nodos está conectados y cuales son sus causas y sus efectos.

La implementación distribuida mostrada a continuación está tomada de [Diez 94]. Las ecuaciones anteriores se encuentran desacopladas y pueden calcularse en paralelo de modo que serán las utilizadas en cada nodo, un exquema de estas ecuaciones y su relación se puede ver en la figura 0-5. Sólo queda describir qué mensajes y en qué momento puede enviarlos a cada nodo: un nodo X puede enviar un mensaje a otro Z si y sólo si ha recibido ya todos los mensajes procedentes de los demás vecinos. Un nodo X con m causas y n efectos, como el de la figura A-4, que haya recibido q mensajes se encontrará en uno de los siguientes estados:

1.- $q \leq n+m-2$, el nodo X se encuentra esperando por lo menos dos mensajes más y por tanto no puede enviar ningún mensaje.

2.- $q = n+m-1$, el nodo X ha recibido todos los mensajes que esperaba menos uno (el del nodo Z, por ejemplo) por lo que puede calcular y enviar el mensaje correspondiente a ese nodo (nodo Z).

3.- $q = n+m$, el nodo X ya ha recibido todos los mensajes que esperaba y puede enviar los mensajes que le faltaban por enviar.

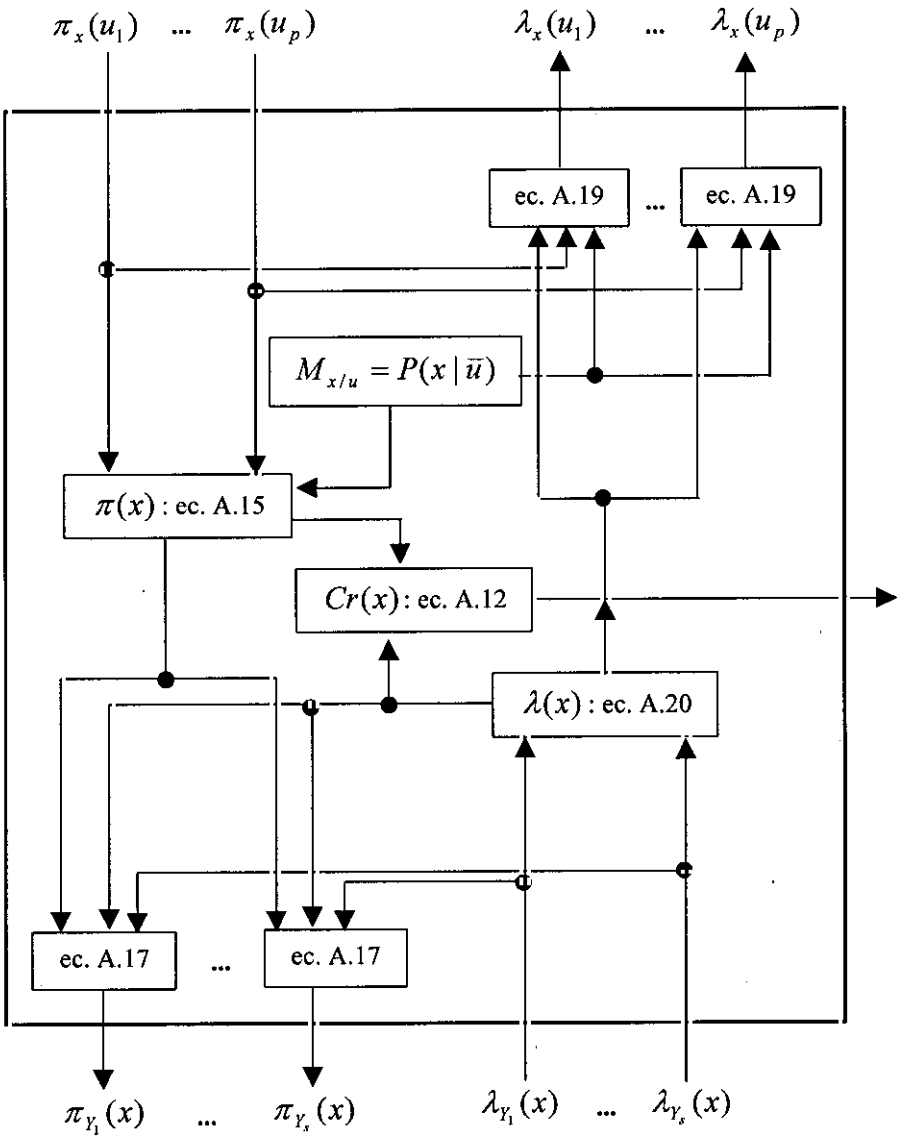


Figura 0-5. Cálculos en el nodo distribuido X.

No es necesario tener un mecanismo global de control puesto que es posible mostrar que siempre hay algún nodo dispuesto a enviar algún mensaje y que el proceso no se interrumpe hasta que el algoritmo se haya completado. Esto es muy interesante puesto que permite construir la red distribuida como una red asíncrona donde el número de mensajes recibidos determina qué mensajes pueden ser calculados y enviados a cada nodo.

Herramientas software

Existen en el mercado multitud de programas y herramientas gráficas para el diseño, evaluación o aprendizaje de las redes bayesianas. En [Almond WWW] y en [Santos WWW] se describen distintas herramientas comerciales o de libre distribución e indican donde encontrarlas.

Por ejemplo, entre los programas de libre distribución se puede disponer en la Universidad Carnegie Mellon (CMU) de una implementación en Pascal del algoritmo de Pearl para redes en forma de árbol [Morawski 89]; *MSBN*, desarrollado por el grupo de investigación en teoría de decisión de Microsoft soporta Windows 95 y NT y permite crear redes de creencia de forma gráfica, evaluar dinámicamente las redes, crear diagramas de influencia y tomar decisiones para la resolución de problemas [Heckerman 95]; *Pucinella*, desarrollado en el IRIDIA, Universidad libre de Bruselas, está escrito en CommonLisp y consta de una librería de funciones para crear, modificar y evaluar distintos métodos de tratamiento de incertidumbre como la teoría de la probabilidad, Dempster-Shafer o la teoría de la posibilidad [Saffiotti 91; Saffiotti 94]; y muchos otros.

Existen también gran número de herramientas comerciales que utilicen redes bayesianas, por ejemplo *Graphical-Belief 2.0*, sucesora de *Belief 2.1* desarrollada por Almond en la CMU, es una implementación en CommonLisp del algoritmo de Dempster y Kong y el de Lauritzen y Spiegelhalter para modelos gráficos probabilísticos, utiliza redes bayesianas y diagramas de influencia y su característica principal es que permite el uso de funciones de creencia y distribuciones de probabildad; *Dxpress*, comercializado por la compañía Knowledge Industries (KI) dirigida al desarrollo de sistemas expertos, es una herramienta gráfica que permite representar el conocimiento de un problema mediante redes bayesianas, orientado a problemas de diagnóstico, control de procesos y diagnóstico médico; *HUGIN*, desarrollada en la universidad de Aalborg

y compatible con sistemas Unix y Windows 95, es una herramienta gráfica para desarrollar redes bayesianas, permitiendo introducir incertidumbre en sus parámetros [Andersen 89; Jensen 96]; etc.

Apéndice B

Metodología ROOM

ROOM (*Real-Time Object-Orient Modeling*) [Selic 95] es un lenguaje de especificación gráfica, orientado a objetos, para modelar sistemas de tiempo real. Sus principales características son:

- *Orientado a eventos*, lo que permite que el modelo se adapte perfectamente a los sistemas reactivos.
- *Concurrencia*, característica propia de la definición de los elementos básicos de ROOM. Es básica en cualquier sistema empotrado de tiempo real, y en sistemas robóticos es imprescindible debido a la propia naturaleza del problema.
- *Distribución*, la abstracción de los objetos como máquinas lógicas y la creación de una máquina virtual en ROOM permiten la distribución de las tareas entre distintas máquinas. Esta facilidad es aconsejable en robots móviles debido a la gran necesidad de cálculo que requieren sus funciones y que obliga a distribuir su procesamiento en distintas máquinas para poder cumplir las restricciones de tiempo necesarias.
- *Recursividad*, permite especificar un modelo con varios niveles de detalle. Puede darse a dos niveles: a nivel de objetos, objetos complejos pueden definirse a partir de otros objetos; y a nivel de estados, el comportamiento de un objeto se modela como una

máquina de estados jerárquica en la que los estados se descomponen en subestados hasta el nivel de profundidad necesario.

- *Modelado incremental*, permite la expresión y verificación de las propiedades del modelo del sistema de modo incremental.
- *Reutilización*, puesto que soporta el paradigma de la programación orientada a objetos. Esto permite la reutilización de objetos y la herencia como un mecanismo potente en la captura de la abstracción.

En este trabajo se aprovechará la parte más básica del lenguaje de modelado ROOM, fundamentalmente su especificación gráfica y su filosofía de mensajes. Esto es suficiente para modelar las tareas y sistemas de control de nuestro robot de forma clara, concisa y coherente.

Por ello, este apéndice no pretende se un resumen completo de las características de ROOM sino sólo de aquellas que han sido utilizadas en la presente tesis o aquellas que permiten aclarar conceptos sobre ROOM o sobre su utilización. Así, a continuación se presentan algunas definiciones y características propias de ROOM.

Actores

Un actor es una máquina lógica que representa una unidad funcional independiente que puede ejecutarse concurrentemente con otros actores. Cada actor desempeña una tarea dentro del sistema diseñado y puede comunicarse con el resto de actores a través de las conexiones marcadas. Como podemos ver, un actor representa un objeto activo que tiene un propósito claramente definido. Activo se refiere a que puede ejecutarse concurrentemente con otros objetos activos (actores). Un actor es un elemento semejante a lo que se conoce como agente.

Los datos sobre los que trabajan los actores no son compartidos entre ellos, sino que se comunican mediante paso de mensajes. Un actor abstrae la información que contiene y no deja que sea manipulada por el exterior.

Interfaz entre actores

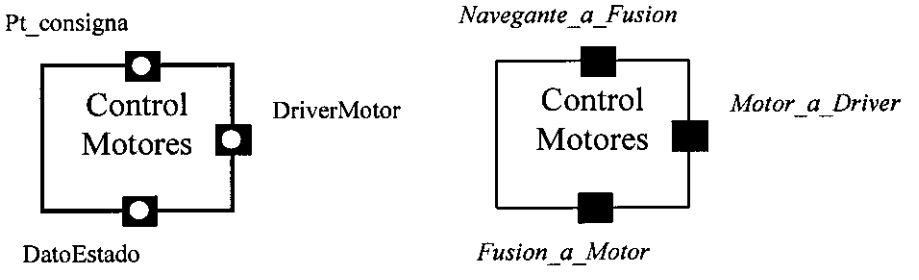
ROOM define tres formas diferentes de interfaz: Puertos, Puntos de acceso a servicios (SAP, *Service Access Points*) y los Puntos de provisión de servicio (SPP, *Service Provision Points*).

De forma somera, puede decirse que los Puertos son puntos de comunicación entre actores o elementos de la misma capa y los SAP y los SPP son servicios que proporciona un sistema de abstracción de la máquina donde se ejecuta el actor denominado máquina virtual.

Notación gráfica

Una referencia de una clase actor viene representada por un rectángulo con un nombre único. Los puertos se representan por pequeños cuadrados situados en su perímetro, véase la figura B-1a). Si son negros son tal como se explican en su definición, si son blancos representan el conjugado del puerto indicado (los mensajes de entrada son de salida y viceversa).

Por supuesto, los nombres de los puertos deben ser los utilizados en la definición del actor. En la definición de la clase de la que se instancian diferentes actores, la representación de los puertos cambia para indicar que es una clase y no una instancia concreta de dicho actor. En este caso su perímetro será con trazo grueso y sus puertos se representarán por cuadrados con un círculo inscrito (cuadrado negro y círculo blanco, o a la inversa para el conjugado) como se muestra en la figura B-1b).



a) Clase *ControlMotores* b) Actor instancia de la clase *ControlMotores*

Figura B-1. Representación de una Clase Actor y su instancia.

Mensajes y protocolos

La comunicación entre actores, como ya se indicó anteriormente, se establece exclusivamente mediante paso de mensajes. La información a transmitir se pasará a un *Objeto mensaje* que se enviará al actor correspondiente mediante un Servicio de comunicaciones.

Se puede utilizar tanto comunicación Síncrona como Asíncrona.

Mensajes: Es un tipo especial de objeto de datos que contiene los siguientes campos: una señal, una prioridad y un tipo de datos (opcional). Si no se envían datos en un mensaje se especifica por NULL.

Mensaje: {señal, prioridad, tipoDato}

Protocolos: Representa el patrón según el cual se realizará el intercambio de mensajes entre los actores. En el se representan las señales y tipos de datos que se utilizarán como entrada y como salida.

El protocolo conjugado (P') es un protocolo igual al P pero cuyos mensajes de entrada son los de salida de P y los de salida son los de entrada de P. Por ejemplo, el protocolo correspondiente a los puertos del actor de la figura B-1 son:

```
Protocol class Navegante_a_ControlMotor
 In: { {nueva_consigna, info_consigna}, {F_emergencia, null} }
 Out: {  }

Protocol class Fusion_a_Motor
 In: { {nuevo_est, tipo_estado } }
 Our: { {mensaje_fin, null}, {mensaje_u, u} }

Protocol class Motor_a_Driver
 In: { {lectura_señal, tipo_lectura } }
 Our: { {señal_control, tipo_control} }
```

Composición de los actores

Los tipos de relaciones estructurales posibles son: relaciones de comunicación y relaciones de contención.

Las *relaciones de comunicación* se representan mediante uniones físicas entre dos puertos, que deben ser compatibles. La unión entre dos actores viene dada por un puerto y su conjugado. Por supuesto, la secuencia de mensajes deben ser compatibles. Un puerto que no esté conectado no podrá recibir mensajes y lo que se envíe por él no lo recibirá nadie.

Las *relaciones de contención*, permiten construir actores a partir de otros actores encapsulando propiedades y funciones de distintos actores. Esto permite una mayor abstracción de la funcionalidad del actor y de su implementación, permitiendo a su vez una mejor depuración.

En los actores compuestos el comportamiento está en los actores que lo componen y en un comportamiento propio denominado *componente comportamiento* o simplemente *comportamiento*. Estos actores presentan componentes de interfaz con actores externos y con los actores contenidos en él y pueden referirse a estos últimos directamente.

Puertos

Un puerto es el elemento que representa el conjunto de mensajes definidos por el protocolo de una clase, representa el interfaz de los actores con una clase en particular. Un puerto se encarga de enviar, recibir y colocar en una cola los mensajes que llegan a él.

Tiene dos partes una de entrada y otra de salida. El protocolo *P* asociado con el puerto, siempre se define con respecto a la parte de salida. Esto es, mensajes que le lleguen a la entrada.

Clasificación de los puertos

Puertos de transmisión o *repetidor* (*relay ports*): se conectan con un componente actor de igual nivel. Los mensajes que le llegan no son vistos por la componente comportamiento. Aparecen en el perímetro de un actor. Se representan mediante un rectángulo (negro o blanco) en los actores internos, puertos *e1*, *e2*, y *e4* que representa puertos externos al actor; y mediante un rectángulo blanco inscrito en uno negro (o viceversa) para los puertos de transmisión de conexión al exterior, por ejemplo el puerto *p3* de la figura B-2, que conecta el actor *A2* con el exterior.

Puertos finales: (los puertos *p1*, *p2*, *e3*) pueden ser acreditados o referenciados por el comportamiento. Son el enlace entre la estructura y el comportamiento puesto que son visibles en ambas dimensiones. Pueden ser de dos clases:

- Puertos finales externos: puertos parte del interfaz del actor conectados directamente con el comportamiento, por ejemplo los puertos *p1* y *p2* de la figura B-2.

- Puertos finales internos: conectan el componente comportamiento con un actor interno, por ejemplo el puerto *e3*.

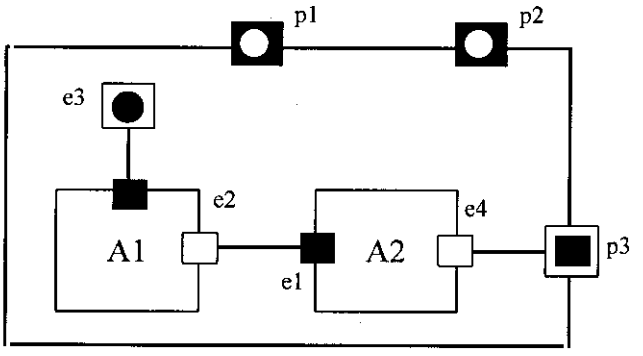


Figura B-2. Tipos de puertos.

Réplicas o Reproducciones (Replicatives)

ROOM proporciona réplicas de actores y réplicas de puertos: se utilizan para modelar referencias múltiples a una clase actor o a una clase protocolo. La réplica es un atributo de la referencia y no de la clase.

Una *réplica de un actor* representa múltiples referencias a actores iguales. Se denomina factor de réplica a un entero que indica el número máximo de replicas de un actor que pueden utilizarse simultáneamente en tiempo de ejecución. Su representación es mediante borde doble y un rectángulo en su esquina superior derecha donde se indica el factor máximo de réplica como puede verse en la figura B-3a).

Las réplicas pueden ser de un actor fijo (el factor es fijo), de un actor opcional (se puede crear dinámicamente hasta el máximo indicado en el factor de réplica) o de un actor importado (igual que en los actores opcionales).

Réplica de puertos: colección de puertos en un actor que tienen el mismo protocolo y nombre. Lo normal es que sirvan para conectarse a actores replicados. Los mensajes enviados/recibidos a un puerto replicado son difundidos a todas las copias de puertos de ese tipo. Se representan, como puede verse en la figura B-3b, por un doble borde.

a) Replica de un actor

b) Replica de puertos (Externos, de Comunicación y Finales)

Figura B-3. Réplica de una actor y de los puertos

Estructuras dinámicas

Actores dinámicos

Estos actores, denominados *actor opcional*, no son creados cuando es creado el actor que lo contiene sino posteriormente. Incluso, una vez creado puede ser destruido. La notación gráfica para un actor opcional puede verse en la figura 0-4.

La opcionabilidad de un actor es un atributo del objeto actor en lugar de un atributo de la clase, lo que permite una mayor reusabilidad. Cuando un actor es creado dinámicamente todos sus componentes son creados automáticamente con él y los puertos que poseen son activados de modo que los canales de comunicación se encuentran preparados para aceptar mensajes inmediatamente. La creación y destrucción de actores opcionales es realizado por el componente comportamiento del actor continente. Por esta razón es imposible destruir el componente comportamiento.

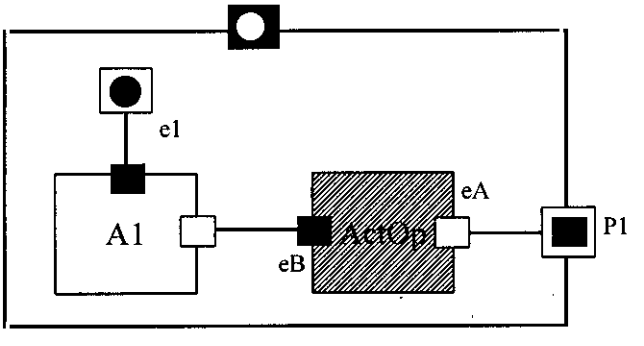


Figura B-4. Actor opcional

Relaciones dinámicas de actores

En este caso, la necesidad de que un actor ya existente establezca una relación dinámica con otro es lo que da lugar a este tipo de capacidad, y al actor se le denomina *actor referencia importado*, su notación gráfica puede verse en la figura B-5. Este actor es semejante a una plantilla en la que es posible insertar dinámicamente un actor ya existente de otro marco. Como analogía con los lenguajes de programación, podría ser comparado a una declaración de tipo puntero.

Un actor particular puede ser colocado en el hueco de un actor importado por el comportamiento del actor que lo contiene. Esta operación se conoce como *importar*, la operación inversa se denomina *deportar*.

Un actor puede ser insertado en una referencia importada sólo si es equivalente al actor de referencia importado. Esto es, sólo componentes que han sido explícitamente diseñados pueden ser importados en un hueco dado.

Las restricciones impuestas a través de equivalencias permiten un control más refinado sobre la sustitución de parámetros que los lenguajes polimórficos tradicionales. Esto se debe a que las equivalencias no sólo limitan qué clases pueden ser sustituidas, sino también qué referencias particulares pueden ser sustituidas. Esto es esencial si la integridad de la arquitectura debe ser preservada, ya que las relaciones arquitectónicas se definen entre referencias mejor que entre clases.

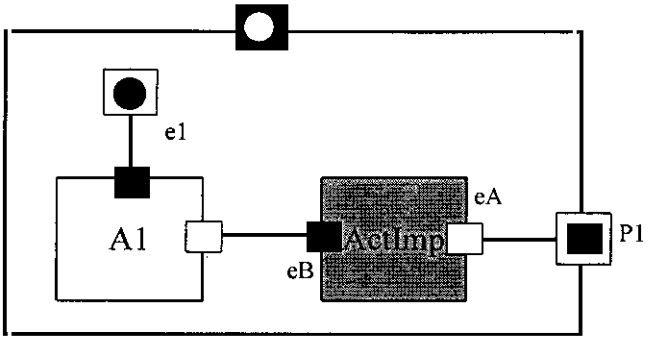


Figura B-5. Actor de referencia importado

Descomposición en capas

Un actor es una estructura jerárquica que puede contener otros actores. Se define por una especificación de clase que incluye entre sus atributos, la especificación de la estructura contenida. El espacio encerrado dentro de un actor se denomina Marco (*decomposition frame*) y los actores que aparecen en él se denominan actores iguales o pares (*peer*).

Una capa en ROOM es sencillamente un actor con algún interfaz especializado. Está basada en el modelo OSI (*Open System Interconnections*) y es una relación jerárquica.

La capa inferior se puede ver como una "máquina programable virtual" abstracta. La capa superior es el "programa" que dirige las operaciones de la maquina. Se denominan *servicios* a las facilidades que proporciona la máquina virtual.

Una de las premisas principales de la estratificación es que una capa sólo depende del intérprete inmediatamente debajo de él y no de otras capas bajas. Todas las interacciones entre capas adyacentes en ROOM tienen lugar en puntos discretos de la frontera entre capas denominados "*Service Access Points*" o SAPs. Hacia abajo se hacen peticiones de servicio invocando primitivas u operaciones asociadas con el servicio por debajo del SAP. Por ejemplo, mandar un mensaje a un par remoto o contestar a éste. Hacia arriba, notifica la llegada de un mensaje originado por un par remoto o los resultados de un procesamiento de una petición de servicio anterior (confirmaciones). Las definiciones de un mensaje y los protocolos para la comunicación entre capas es idéntica a la especificada anteriormente.

SAP = (*Service Access Point*) asociado a la capa superior.

SPP = (*Service Provision Point*) asociado a la capa inferior.

Siempre el protocolo de un SPP es el conjugado de un SAP. Se representa con un arco dirigido, cuya flecha apunta del usuario del servicio al servicio proporcionado. La conexión explícita entre un SAP y un SPP se hace mediante la conexión entre capas, y se comporta en todo lo demás como la comunicación vía puertos. El actor que contiene un SPP se define como una capa de servicios. El actor que posee el SAP se considera como la capa que usa el servicio, figura B-6.

La comunicación entre capas permite el paso de mensajes entre actores sin la necesidad de un enlace explícito, permite desarrollar de forma independiente como "capa" actores de alto nivel que representan servicios, a nivel de implementación, centralizados y ampliamente compartidos. Su integración se hace en tiempo de ejecución.

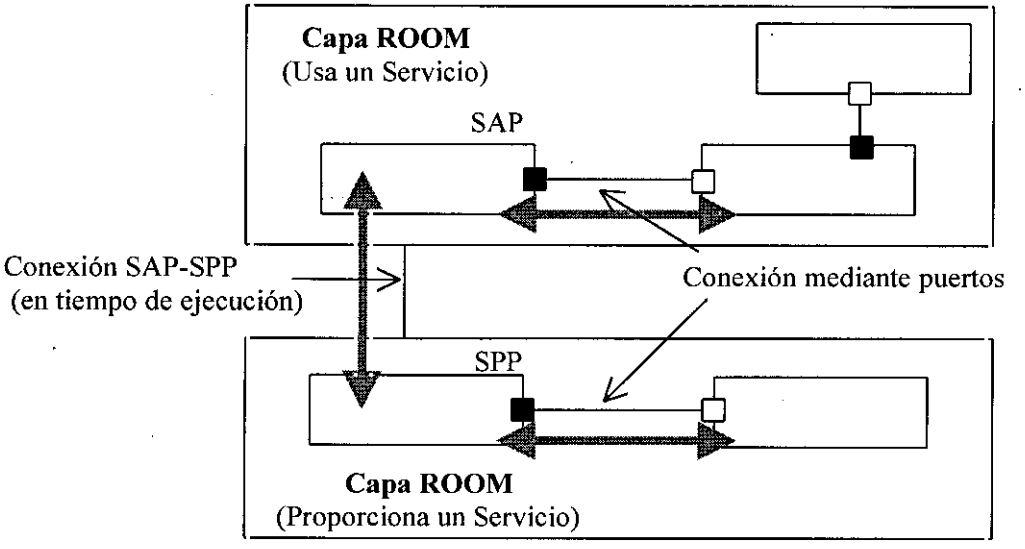


Figura B-6. Servicios SAP y SPP.

Diagramas de ROOM

Los diagramas de ROOM representan estados y transiciones. En un estado siempre se espera que ocurra un evento para producir una transición. En todo momento se examina la llegada de cada mensaje para ver si satisface las especificaciones de disparo de cualquiera de las transiciones ligadas al estado actual. Si no es así no se hace nada. Una transición lleva de un estado a otro. Cada transición puede tener una *acción asociada*. La acción consiste en un conjunto de pasos que se limitan a operar sobre los siguientes objetos:

. Variables de estado extendidas.

. Variables temporales para resultados intermedios durante la ejecución de la acción.

. Objetos de interfaz (puertos internos, SAPs, SPPs) para comunicarse con otros objetos.

El mensaje que causó el suceso se guarda automáticamente en una variable de estado extendida predefinida. *msg* es el puntero al mensaje.

Cuando se crea un actor se realiza una *transición inicial*. La transición inicial saca al actor del estado *Inicio* para que comience su funcionamiento normal (característica que distingue un objeto activo de uno pasivo). Esta transición puede tener una acción asociada que se puede usar para iniciar las variables de estado extendidas y para enviar mensajes de sincronización con otros actores.

Se denomina *punto inicial* al punto de arranque fuente de la transición inicial. No es un estado, sólo puede haber una transición desde (I). Cuando el actor se crea dinámicamente puede recibir un mensaje inicial para provocar la transición inicial.

Transiciones

Toda transición ocurre por la llegada de un mensaje por uno de los componentes de interfaz. Una especificación de disparo consta de los siguientes tres elementos :

t: {señal, puerto, [condición]}

Señal, especifica una señal que debe ser válida según la especificación del protocolo asociado con el componente de interfaz. *Puerto*, es el nombre del *componente de interfaz* (puerto, SAP o SPP). *Condición*, es una expresión booleana (condición de guarda) que se evalúa automáticamente cuando se gestiona la señal recibida y si es TRUE se realizará la transición correspondiente. Si no se especifica guarda, por defecto es TRUE.

Otros tipos de disparo son:

- *Disparos compuestos*: una transición puede ocurrir por más de un suceso. Se especifica con *"or"*.
- *Disparo universal*: es causado por cualquier señal válida (si se verifica la guarda). Por ejemplo, {*,*,guarda}

A las transiciones se le asignan nombres que tienen que ser únicos para cada estado. En otros estados se pueden usar el mismo nombre. En el ejemplo de la figura B-7 se han denominado los distintos eventos con *evN* además de mostrar la transición y la acción que lleva asociada.

La especificación de transiciones con cláusulas del tipo *"and"* se pueden hacer utilizando estados intermedios, de modo que cuando se recibe se pasa a otro estado donde se espera otra señal y así sucesivamente, lo que lleva a crear un árbol de estados con todas las combinaciones posibles de las señales.

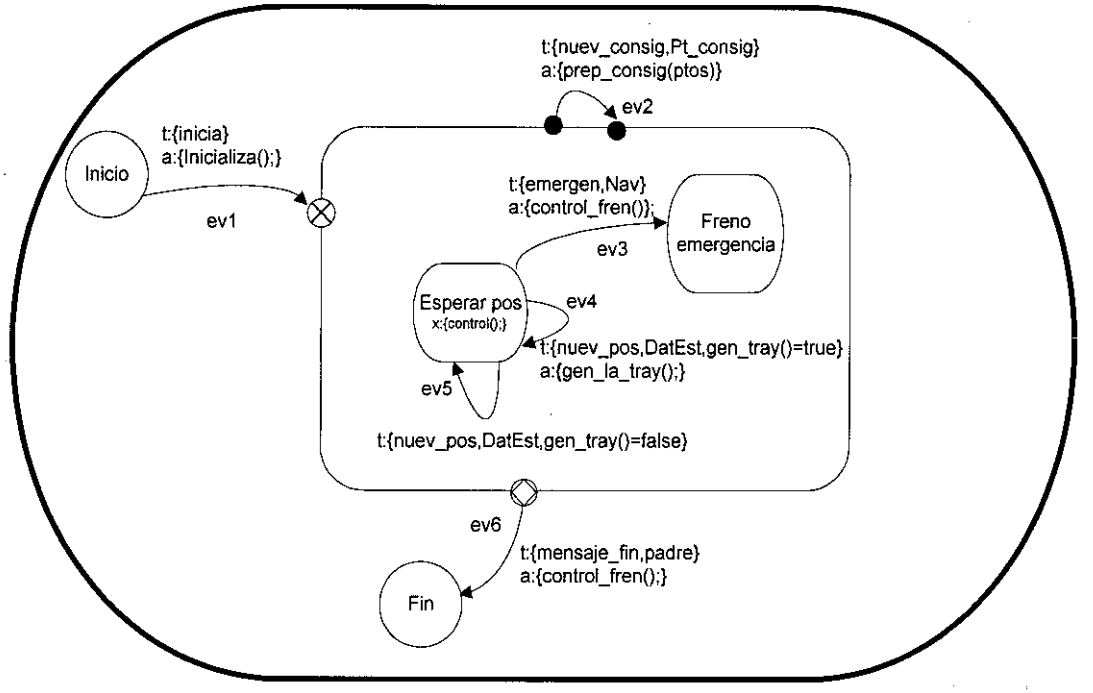


Figura B-7. Ejemplo de un diagrama de estado

Acciones de entrada y acciones de salida del estado

Son opcionales (por ejemplo el evento ev5 no tiene asociada ninguna acción en la acción). Las acciones a realizar pueden realizarse en la transición de un estado a otro (por ejemplo el evento ev3) o en ese estado (a la entrada del estado *Esperar_pos*). Cuando es una autotransición (ev4) se realiza de la misma forma. Siempre se ejecuta la acción de la transición que se realizó en ese estado.

Apéndice C

Sistema sensorial del robot

El robot tiene unas dimensiones de 65x65 cm. Y una altura máxima, debido al captador de balizas, de 1.75 m. Su peso es de unos 60 kg. aproximadamente.

Dispone de dos potentes motores, uno para cada rueda que se controlan individualmente, de corriente continua de 150 w. Con dinamo incorporada que permite obtener la velocidad de las ruedas con precisión sin necesidad de ningún tipo de integración de la velocidad. La controladora de cada motor dispone de un circuito con estabilización de la velocidad.

Para la alimentación de los motores dispone de dos baterías de 47 A. que permiten al robot una autonomía de aproximadamente 30 minutos.

En cuanto a los sensores incorporados al robot, además de la dinamo, disponemos de:

Sensores de estado:

- *Encoders*. En cargados de medir el giro de cada una de las ruedas, permiten obtener el desplazamiento y giro del robot. Los encoder utilizados son un modelo de HP de la serie HEDS-5500 con una precisión de 2000 pasos por vuelta. El circuito de adaptación que se encarga de la cuenta de los pasos de encoder es el HCTL-2020 que dispone de un contador de 16 bits.

- *Brújula magnética*. Permite conocer la orientación del robot con respecto al norte magnético. Tiene una precisión de hasta una décima, de grado aunque su error en las décimas de grado es grande. La brújula digital es una Azimuth 314AC con una unidad display de 25'. Se comunica con la computadora por el puerto serie con una frecuencia de 1 mensaje por segundo codificados según el estándar NMEA 0183 de la marina de los EEUU.
- *Balizas activas*. Permiten conocer la posición del robot con una precisión de centímetros. Se han construido mediante ultrasonidos de 40 Kciclos. Su distribución es: dos receptores en alto a los lados del robot y dos emisores de baliza en puntos fijos del entorno. Ambos constan de una roseta de receptores y de emisores de ultrasonidos respectivamente para permitir un funcionamiento omnidireccional.

Sensores de entorno:

- *Cinturón de ultrasonidos*. El cinturón de ultrasonidos alrededor del robot permite localizar obstáculos en su entorno. Consta de 12 emisores de 40 Kciclos y de un rango de medida ajustable, que en nuestro caso es de 30 cm. hasta 5 m. Su disposición es la mostrada en la figura.

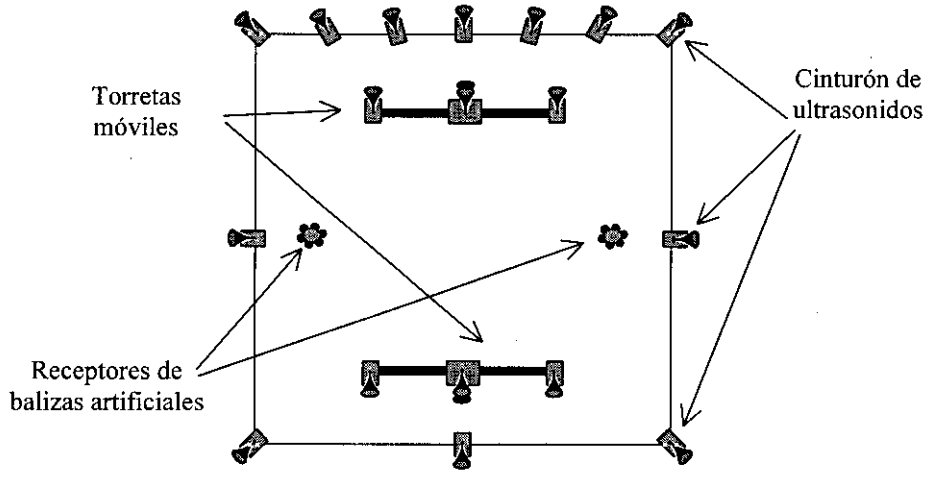


Figura C-8. Sensores de entorno

- *Torretas móviles.* Permiten localizar obstáculos y realizar un análisis de la forma u orientación de los objetos detectados. Consta de un emisor de 40 Kciclos y de tres receptores, lo que permite percepción triaural con posibilidad de análisis de la envolvente de los ecos recibidos. El robot dispone de dos torretas, una en la parte delantera y otra en la trasera ambas móviles con un motor paso a paso de precisión 3 grados.

Apéndice D

Matrices del modelo del sistema y de los sensores

El modelo de movimiento del móvil y de los sensores ya ha sido descrito en la sección 3.5.1. Aquí se mostrarán las matrices utilizadas en el filtro de Kalman para el caso centralizado y el caso distribuido.

Filtro de Kalman centralizado y distribuido

En el caso centralizado se dispone de un vector de estado grande debido al aumento de estados necesario para la brújula y los encoders. El estado final será,

$$\vec{x}_i(t_i) = (x(t_i), y(t_i), \vartheta(t_i), \vartheta_c(t_i), x_e(t_i), y_e(t_i))$$

Estos seis estados serán utilizados en un filtro de Kalman centralizado y pueden utilizarse igualmente en el filtro de Kalman distribuido, obteniendo las siguientes matrices:

- *Matrices del sistema y de ruido:*

La matriz del sistema, modelo de primer orden, utilizada en el filtro centralizado o en el filtro de cada sensor en caso distribuido será,

$$F_i^1(\delta t_i) = \begin{bmatrix} 1 & 0 & -\delta t_i \cdot V & 0 & 0 & 0 \\ 0 & 1 & -\delta t_i \cdot V \cdot \vartheta(t_i) & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1/\tau & -1/\tau & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (0.22)$$

Y la estima inicial del estado del móvil y su covarianza son,

$$\vec{x}_i(0 \mid 0) = (0,0,0,0,0,0)^T \tag{0.23}$$

$$P_i(0 \mid 0) = \begin{bmatrix} 1000 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1000 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1000 \end{bmatrix} \tag{0.24}$$

- *Matrices del modelo de los sensores*

La covarianza del ruido Gaussiano de los estados (Q) su matriz de ganancia (G) y la covarianza de la precisión en la observación (R) correspondiente a cada uno de los sensores son:

a) Brújula magnética:

$$Q_c(t_i) = 20 \tag{0.25}$$

$$G_c(t_i) = \begin{bmatrix} 0 & 0 & \delta t_i & 0 & 0 & 0 \end{bmatrix}^T \tag{0.26}$$

$$R_c(t_i) = \pi/180 \tag{0.27}$$

b) Balizas artificiales:

$$Q_b(t_i) = \begin{bmatrix} 14 & 0 & 0 \\ 0 & 14 & 0 \\ 0 & 0 & 4 \end{bmatrix} \tag{0.28}$$

$$G_b(t_i) = \begin{bmatrix} \delta t_i & 0 & 0 \\ 0 & \delta t_i & 0 \\ 0 & 0 & \delta t_i \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{0.29}$$

$$R_b(t_i) = \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0.01 \end{bmatrix} \tag{0.30}$$

c) Odómetros:

$$Q_e(t_i) = \begin{bmatrix} 8 & 0 & 0 \\ 0 & 8 & 0 \\ 0 & 0 & 2 \end{bmatrix} \tag{0.31}$$

$$G_e(t_i) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \delta t_i \\ 0 & 0 & 0 \\ 0 & \delta t_i & 0 \\ 0 & 0 & \delta t_i \end{bmatrix} \tag{0.32}$$

$$R_e(t_i) = \begin{bmatrix} 0.2 & 0 & 0 \\ 0 & 0.2 & 0 \\ 0 & 0 & 0.2 \end{bmatrix} \tag{0.33}$$

Estas matrices pueden utilizarse en los sensores distribuidos de la forma descrita en el Capítulo 3. El uso de este tipo de matrices permite la comparación directa con el filtro centralizado, pero provoca un aumento en el tamaño de las matrices utilizadas.

Filtro de Kalman completamente distribuido

Un alternativa a las matrices propuestas anteriormente es la utilización de modelos separados para cada sensor. De esta forma se pueden añadir o eliminar sensores sin que su uso afecte a los demás, al tener que modificar el vector de estados utilizado. En este caso el vector de estados que se comunica entre los sensores es exclusivamente los estados de interés del móvil y los estados internos de cada sensor se utilizan de forma independiente por parte de cada sensor sin ser comunicados a los demás.

La matriz lineal del modelo de transición de estados será distinta para cada sensor, por ejemplo:

Modelo utilizado por la brújula:

$$F_i^1(\delta t_i) = \begin{bmatrix} 1 & 0 & -\delta t_i \cdot V & 0 \\ 0 & 1 & -\delta t_i \cdot V \cdot \vartheta(t_i) & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/\tau & -1/\tau \end{bmatrix} \tag{0.34}$$

Y la estima inicial del estado del móvil y su covarianza son,

$$\vec{x}_i(0|0) = (0,0,0,0)^T \tag{0.35}$$

$$P_i(0|0) = \begin{bmatrix} 1000 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 \\ 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 1000 \end{bmatrix} \tag{0.36}$$

Modelos para la baliza y los encoders:

$$F_b^1(\delta t_i) = \begin{bmatrix} 1 & 0 & -\delta t_i \cdot V \\ 0 & 1 & -\delta t_i \cdot V \cdot \vartheta(t_i) \\ 0 & 0 & 1 \end{bmatrix} \tag{0.37}$$

$$F_e^1(\delta t_i) = \begin{bmatrix} 1 & 0 & -\delta t_i \cdot V & 0 & 0 \\ 0 & 1 & -\delta t_i \cdot V \cdot \vartheta(t_i) & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{0.38}$$

Y las matrices de covarianzas son semejantes a la de la brújula.

La covarianza del ruido Gaussiano de los estados (Q) su matriz de ganancia (G) y la covarianza de la precisión en la observación (R) correspondiente a cada uno de los sensores son:

a) Brújula magnética:

$$Q_c(t_i) = 20 \tag{0.39}$$

$$G_c(t_i) = \begin{bmatrix} 0 & 0 & \delta t_i & 0 \end{bmatrix}^T \tag{0.40}$$

$$R_c(t_i) = \pi/180 \tag{0.41}$$

b) Balizas artificiales:

$$Q_b(t_i) = \begin{bmatrix} 14 & 0 & 0 \\ 0 & 14 & 0 \\ 0 & 0 & 4 \end{bmatrix} \tag{0.42}$$

$$G_b(t_i) = \begin{bmatrix} \delta t_i & 0 & 0 \\ 0 & \delta t_i & 0 \\ 0 & 0 & \delta t_i \\ 0 & 0 & 0 \end{bmatrix} \tag{0.43}$$

$$R_b(t_i) = \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0.01 \end{bmatrix} \tag{0.44}$$

c) Odómetros:

$$Q_e(t_i) = \begin{bmatrix} 8 & 0 & 0 \\ 0 & 8 & 0 \\ 0 & 0 & 2 \end{bmatrix} \tag{0.45}$$

$$G_e(t_i) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \delta t_i \\ 0 & \delta t_i & 0 \\ \delta t_i & 0 & 0 \end{bmatrix} \tag{0.46}$$

$$R_e(t_i) = \begin{bmatrix} 0.2 & 0 & 0 \\ 0 & 0.2 & 0 \\ 0 & 0 & 0.2 \end{bmatrix} \tag{0.47}$$