UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE CIENCIAS FÍSICAS

Departamento de Arquitectura de Computadores y Automática

NAVEGACIÓN AUTÓNOMA DE ROBOTS EN AGRICULTURA: UN MODELO DE AGENTES

MEMORIA PARA OPTAR AL GRADO DE DOCTOR

PRESENTADA POR

Lía García Pérez

Bajo la dirección de la doctora

María C. García – Alegre Sánchez

Madrid, 2005

ISBN: 978-84-669-2553-2

DEPARTAMENTO DE ARQUITECTURA DE COMPUTADORES Y AUTOMÁTICA

FACULTAD DE CIENCIAS FÍSICAS.

UNIVERSIDAD COMPLUTENSE DE MADRID

TESIS DOCTORAL

NAVEGACIÓN AUTÓNOMA DE ROBOTS EN AGRICULTURA: UN MODELO DE AGENTES

Presentada por:

Lía García Pérez

Dirigida por:

Dra. María C. García-Alegre Sánchez

Tutora:

Dra. Matilde Santos Peñas

Madrid, diciembre de 2003

A Juanjo, a mis padres y a mi hermana.

FÁBULA DE LOS TRES HERMANOS

De tres hermanos el más grande se fue
por la vereda a descubrir y fundar
y para nunca equivocarse o errar
iba despierto y bien atento
a cuanto iba a pisar.

De tanto en esta posición caminar
ya nunca el cuello se le enderezó
y anduvo esclavo ya de la precaución
y se hizo viejo queriendo ir lejos
con su corta visión.

Eeeh, eeeh, eh,
eeeh, eeeh, eh
ojos que no miran
mas allá no ayuda al pie.

Uuuh, uuuh, uh,
uuuh, uuuh, uh
óyeme esto y dime
dime lo que piensas tú.

De tres hermanos el del medio se fue
por la vereda a descubrir y a fundar
y para nunca equivocarse o errar
iba despierto y bien atento
al horizonte igual.

Pero este chico listo no podía ver
la piedra, el hoyo que vencía a su pie
y revolcado siempre se la pasó
y se hizo viejo queriendo ir lejos
adonde no llegó.

Eeeh, eeeh, eh,
eeeh, eeeh, eh
ojo que no mira
mas acá tampoco ve.

Uuuh, uuuh, uh,
uuuh, uuuh, uh
óyeme esto y dime
dime lo que piensas tú.

De tres hermanos el pequeño partió
por la vereda a descubrir y a fundar
y para nunca equivocarse o errar
una pupila llevaba arriba
y la otra en el andar.

Y caminó vereda adentro el que más
ojo en camino y ojo en lo porvenir
y cuando vino el tiempo de resumir
ya su mirada estaba extraviada
entre el estar y el ir.

Eeeh, eeeh, eh,
eeeh, eeeh, eh
ojo puesto en todo
ya ni sabe lo que ve.

Uuuh, uuuh, uh,
uuuh, uuuh, uh
óyeme esto y dime
dime lo que piensas tú.

Silvio Rodriguez, “Rabo de nube”, 1979.

Agradecimientos

Es de bien nacido el ser agradecido. Y siguiendo este refrán que mi madre tanto dice, quiero que queden impresos los nombres de muchas personas a las que tengo que agradecer que hoy esté escribiendo estas líneas.

Sin lugar a dudas tengo que agradecer la financiación que me ha permitido mantenerme durante estos últimos cinco años: primero una beca FPI del Ministerio de Ciencia y Tecnología asociada al proyecto CICYT-TAP98-0781 *"Generación de comportamiento complejo para un robot pulverizador de exteriores (AMARA II)"* y luego un contrato a cargo del proyecto MCYT-AGL2002-04468-C03-01 *"Visión artificial y razonamiento espacio-temporal para tratamientos localizados con un tractor autónomo (GEA II)"*. El Instituto de Automática Industrial (IAI) ha sido mi lugar de trabajo durante todo este tiempo y quiero dar las gracias a toda su gente. Al Dr. Salvador Ros, como director del centro le agradezco todos los medios que han estado a mi disposición. Personalmente siempre se ha preocupado de mi tesis y mi trabajo. Sin el buen hacer del personal del taller mecánico y del de electrónica nunca hubiera podido tener a DÉDALO funcionando, gracias a todos. Y centrándome en las personas a las que he tenido más próximas tengo que agradecer a la Dra. María C. García-Alegre, directora de esta tesis, la oportunidad que me dio para comenzar en la investigación. A ella y a los doctores Ángela Ribeiro y Domingo Guinea, tengo que agradecerles también la confianza en mi trabajo y el trato siempre cercano. Al ingeniero Eugenio Villanueva que con su saber ha arreglado tantos problemas, a los ingenieros Pablo Yañez y Rubén Martín que se encargaron de la mecánica y a la Dra. Ana Pozo que me enseñó lo que sé del GPS. Tambiéne recibido mucha ayuda de personas de fuera del IAI. En primer lugar están mis estancias en el extranjero, a las que debo agradecer muchas ideas frescas, nuevas formas de hacer las cosas y un enorme aprendizaje. Gracias al Dr. John Marchant y la gente del Image Lab de Silsoe (UK), al Prof. Zadeh de Berkeley y al Prof. Halme de la University of Technology en Helsinky. En segundo lugar mi tío Tomás ¡qué suerte poder tener un contacto tan directo con la agricultura!; a Miguel Ángel Sotelo que ha leído con mucho cuidado y paciencia el primer borrador de esta tesis y me ha aportado mucho; a Laura Vergara que me proporcionó mucha bibliografía muy interesante de psicología. Y a Matilde Santos,

mi tutora en la Universidad Complutense, que siempre ha estado disponible para echarme una mano; de no haber estado ella no creo que hubiera logrado desenvolverme con el papeleo.

Pero esta tesis ha estado muy presente durante los últimos años en todos los aspectos de mi vida, ¡no ha sido sólo trabajo! Con mucha gente del IAI especialmente con los de la segunda planta del edificio B me resulta difícil separar lo profesional de la amistad; todos han estado siempre pendientes de mi trabajo y nunca nadie ha dejado sin contestar mis preguntas, a todos os debo, al menos, el nombre en esta página: Tere, Carlos, Alberto, José Javier, Eugenio, Luis, Ángela, Domingo y María y todos mis compañeros y amigos con los que he compartido trabajo y fiestas: Pablo, Rubén, Juan Carlos, Ricardo, Bea, Víctor, José María, Rodrigo, Marga, David, Antonio, Leandro, Gonzalo, Tomás, Manuel, Miguel, Dani, Roberto, Óscar, Montse y Fernando. De modo muy especial quiero agradecer a Óscar los buenos consejos y las charlas en los viajes compartidos en el metro; en algunos momentos me hubiera desesperado sin ellas. Marga que siempre ha estado disponible para leer lo que le he dado y corregirlo con entendimiento. Bea ha sido siempre un apoyo entusiasta por todo lo que tenemos en común y su sinceridad a prueba de bombas. Y si alguien me ha animado en mi trabajo, ha estado pendiente de mí y me ha acompañado, con lo difícil que ha sido eso en ocasiones, ése ha sido José María. Sin un "hermano mayor" como él, no habría logrado terminar la tesis. No me cabe en estas líneas todo lo que tengo que agradecerle y lo mucho que he aprendido con él.

Y todos mis amigos de fuera del IAI ¡tantos! Gracias a los que han entendido mis ausencias y me han animado a trabajar en la tesis. Gracias también a los que no lo han entendido tan bien, porque me han forzado a mantenerme en contacto con el mundo fuera del robot. Gracias por la compañía en la distancia, por preguntar incansablemente, por la curiosidad sobre mi trabajo. Todos os merecéis estar aquí: Consuelo, Alberto, Virginia, Fernando, Sergio, Roberto y todos mis compañeros de la Escuela de Adultos San Federico.

Tampoco quiero olvidarme de todos aquellos con los que he compartido la lucha por el reconocimiento de los derechos laborales de los jóvenes investigadores desde Precarios Madrid, gracias a la gente que lucha cada día por lo que es justo.

Y finalmente quiero agradecer a toda mi familia su paciencia al comprobar que cada

domingo me iba corriendo a casa a trabajar, que nunca tenía tiempo durante la semana, gracias a Miguel, Cristina, Camilo y Mari. Especialmente gracias a mis padres, Santiago y Tere y a mi hermana Dorkás por su apoyo, su interés en mi trabajo y por sentirse siempre orgullosos de mí, me han enseñado a ser curiosa y a seguir adelante siempre.

Y a Juanjo por compartir conmigo también la tesis, por sus preguntas, sus ánimos, sus críticas, las charlas las tardes de domingo... Pero sobre todo por ayudarme sin tregua en este empeño mío de seguir adelante con mi vida a pesar de la tesis.

RESUMEN

El objetivo fundamental de esta Tesis Doctoral es el desarrollo de una arquitectura de control para la generación de comportamiento complejo de un robot móvil en un entorno dinámico y de exterior. El alcanzar convenientemente este objetivo ha originado que este trabajo exhiba un carácter multidisciplinar al requerir paradigmas y técnicas de dos grandes disciplinas como son la Robótica y la Inteligencia Artificial. En concreto se propone una arquitectura distribuida de control basada en agentes para la navegación autónoma de vehículos en exteriores y especialmente orientada al laboreo agrícola. Este modelo se ha validado implantándola en un sistema real complejo; un tractor comercial automatizado en el IAI y que constituye uno de los resultados relevantes de esta tesis. De un modo general el trabajo desarrollado y expuesto en esta memoria está organizado alrededor de cinco ejes fundamentales: 1) Una revisión exhaustiva de la robótica en agricultura, para analizar los requisitos y restricciones impuestos por una tarea agrícola. 2) La automatización y sensorización de un tractor comercial con dirección hidráulica, incorporando al mismo las electroválvulas y los cilindros hidráulicos adicionales necesarios para el control automático de la dirección y tracción. Así como la integración en el tractor de un conjunto de sensores: DGPS, brújula digital, láser de barrido, inclinómetros, sensor de choque y odómetros diseñados y desarrollados para el tractor; con la finalidad de poder percibir el estado del sistema/entorno y reaccionar de forma adecuada. La puesta a punto del tractor se completa con la incorporación de un ordenador a bordo que se comunica, vía radio-Ethernet, con cualquier estación de trabajo conectada a la intranet del Instituto de Automática Industrial (IAI-CSIC). 3) El desarrollo del control para el freno y el embrague así como de la implementación de un sistema de control híbrido borroso de la dirección. Este modelo de control constituye una novedad en el campo de la Robótica de vehículos comerciales con sistemas de actuación hidráulica en aplicaciones que requieren una navegación autónoma. 4) El diseño e implementación de un simulador que permite experimentar la bondad del diseño de un proceso y ajustar los parámetros de modulación y estrategias definidos en cada agente. Con este simulador se pretende mitigar la dificultad que tiene la experimentación con un tractor en exteriores, tarea siempre ardua y dependiente de múltiples factores entre los que se encuentran las condiciones

meteorológicas, adversas en muchas ocasiones. 5) Por último la memoria recoge detalladamente el diseño y desarrollo de la arquitectura híbrida propuesta AGRO-AMARA basada en un modelo de agentes organizados en lo que se refiere a la reutilización de habilidades. En su diseño se han seguido los principios básicos de modularidad, reutilización facilidad de escalado y rápida implantación en diferentes plataformas. La arquitectura ha sido planteada con la premisa de conseguir una alta flexibilidad, encapsulando los procesos que conducen a un objetivo de percepción o control, de forma que lo que importa de un agente es su comportamiento y no las técnicas y modelos necesarios para su desarrollo. Por ello la arquitectura propuesta presenta agentes cuyos procesos se han modelado con técnicas de razonamiento aproximado y con aproximaciones analíticas clásicas. Por último decir que los agentes que constituyen la arquitectura AGRO-AMARA han sido validados en su totalidad con el tractor comercial automatizado, en el campo que rodea las instalaciones del IAI-CSIC en Arganda del Rey. Las habilidades del tractor han sido demostradas en misiones de navegación autónoma, tanto en un entorno dinámico abierto como en un campo de cultivo estructurado. Se han realizado con éxito, distintos experimentos de navegación para probar el comportamiento global de la arquitectura y el modo de operación de los distintos agentes.

ABSTRACT

The objective of this thesis is to develop a control architecture to generate complex behavior for a mobile robot in an agricultural environment. To this aim this thesis requires techniques and paradigms of two different topics: Robotics and Artificial Intelligence. The proposed architecture is a distributed agent based control architecture for autonomous navigation of outdoor vehicles specially oriented to agricultural tasks. The model has been implemented on a real complex system: a commercial tractor automated on the IAI workshops. The automated tractor is one of the relevant results of this thesis.

Five topics make up this work: 1) a complete state of art of agricultural robots. 2) The automation and sensor selection of a commercial tractor. Automatic action has been achieved by electro-hydraulic valves and hydraulic cylinders for steering and drive. Sensor system is composed by: a DGPS, a digital compass, a laser scanner, two inclinometers, a collision sensor and two odometers specially designed at the Institute for Industrial Automation (IAI). The automation of the tractor is completed with an on board computer with a wireless Ethernet card. 3) The development of a control system for the brake and clutch and the implementation of a hybrid control system for the steering. 4) The design and implementation of a simulator that allows to experiment different algorithms and to tune modulation parameters and strategies of the different agents. This simulator mitigates the difficulties of the experimentation in outdoors. 5) Finally the detailed design and implementation of an hybrid architecture, named AGRO-AMARA, based on organized agents, to reuse abilities.

The architecture allows high flexibility because processes oriented to a perceptual or action objective are encapsulated separately, so that the agent is typified by its behavior not by the techniques and models used on it. All agents that make up the AGRO-AMARA architecture have been validated on the automated commercial tractor, on the fields surrounding IAI in Arganda del Rey, Madrid. Tractor abilities have been demonstrated on autonomous field navigation missions, both on dynamic open environment and on a structured labor field. Different experiments have been successfully carried out to test global architecture behavior and performance of the several agents.

Índice General

Índice de Tablas

Índice de Figuras

Capítulo 1

Introducción

En los últimos años los avances tecnológicos han permitido la adopción de técnicas innovadoras en el campo de la agricultura, aumentando la rentabilidad económica y reduciendo el impacto medioambiental. Así surge la denominada Agricultura de Precisión (AP), que engloba tecnologías y prácticas encaminadas a minimizar el uso de productos agro-químicos mientras se asegura un control efectivo de plagas, malas hierbas y enfermedades, a la vez que se suministra una cantidad de nutrientes adecuada a los cultivos [Kropff et al., 1997].

A pesar de los beneficios potenciales suministrados por la AP una de las grandes barreras para su adopción como práctica habitual es el coste asociado tanto en complejidad como en tiempo a la realización en el campo de tareas que sigan los principios de la Agricultura de Precisión. Por ejemplo, sin una automatización mínima en el tractor, aplicar herbicida en dosis variables es una tarea que un agricultor no aborda por la dificultad de atender simultáneamente al guiado manual del vehículo y a la apertura selectiva de las secciones de la barra de fumigación

en función de la cantidad de mala hierba percibida. Por ello, una buena práctica agrícola orientada a la AP requiere una gran automatización de las tareas agrícolas. Con esta finalidad, se han desarrollado manipuladores automáticos que permiten al agricultor adoptar el papel de supervisor en tarea de cosechado, poda, o recolección de frutos. Las prestaciones de estos manipuladores son buenas pero deben ser transportados por un vehículo al igual que los operarios.

En esta dirección, existe una tendencia actual en el mercado hacia el desarrollo de sistemas de guiado automático, donde no se prescinde del conductor pero se le proporciona un dispositivo de guiado que le ayuda en la conducción y le permite seguir con mayor precisión trayectorias. Incluso algunos de estos sistemas comerciales toman el control de la máquina durante los tramos rectilíneos de la navegación. Sin embargo, un reto para la automatización completa de la navegación agrícola es la inclusión de mecanismos de seguridad de los operarios que manejan la máquina, del entorno de trabajo y de la integridad del vehículo. Progresivamente y de forma gradual se van dando pasos hacia el desarrollo de sistemas cada vez más autónomos, que permitan trasladar al experto humano desde las tareas de conducción manual a las de teleoperación del tractor hasta las de supervisión de un conjunto de máquinas.

En el aspecto concreto de navegación autónoma de vehículos agrícolas, la Agricultura de Precisión se ve favorecida por los resultados obtenidos en las investigaciones en los últimos años en el área de la Robótica Móvil. Desde la década de los 70 se trabaja en el desarrollo de robots capaces de navegar con un cierto grado de autonomía, investigando sobre arquitecturas de organización del conocimiento y control que doten al robot con mecanismos de decisión y respuestas reactivas adecuadas para afrontar situaciones imprevistas, combinar objetivos diferentes avanzando hacia la generación de sistemas menos dependientes del operario. La mayoría de estos trabajos se han centrado en robots de interiores, normalmente de tamaño pequeño, con gran maniobrabilidad y que deben ejecutar tareas de servicio en entornos de oficina. El desarrollo de una arquitectura de control para robots móviles en agricultura conlleva un análisis en profundidad de las características tanto de los vehículos que se van a utilizar, como del entorno de trabajo y de los objetivos que se persiguen en el ámbito aún inexplorado

de la Agricultura de Precisión.

Esta tesis se enmarca dentro de las lineas de investigación del Instituto de Automática Industrial del Consejo Superior de Investigaciones Científicas sobre mecanismos de percepción y control en arquitecturas para la organización de conocimiento en sistemas complejos con incertidumbre y en la automatización de vehículos para labores agrícolas y de jardinería. El trabajo ha tenido como soporte dos proyectos de investigación del Plan Nacional de I+D: CICYT-TAP98-0781 *"Generación de comportamiento complejo para un robot pulverizador de exteriores (AMARA II)"* y MCYT-AGL2002-04468-C03-01 *"Visión artificial y razonamiento espacio-temporal para tratamientos localizados con un tractor autónomo (GEA II)"*

El objetivo de la tesis es el desarrollo de una arquitectura de control para la navegación autónoma de un robot móvil en un entorno agrícola para la fumigación de olivos. Este objetivo general puede descomponerse en los siguientes sub-objetivos específicos:

- Estudio de las aplicaciones en automatización de tareas agrícolas con especial atención a los robots móviles.
- Estudio de las arquitecturas de control existentes para robots y análisis de las características relevantes de una arquitectura aplicada a la navegación de un tractor agrícola.
- Automatización y sensorización de un tractor comercial. Desarrollo de los controladores para conducción automática y diseño de un sistema de control borroso de la dirección.
- Diseño de una arquitectura de control para un robot agrícola, centrada en la navegación en exteriores. Esta tarea de navegación en exterior, se descompone en una navegación de propósito general y una navegación específica para el tipo de laboreo a realizar. La arquitectura híbrida desarrollada se concibe como un modelo multiagente y requiere la generación de los distintos agentes perceptivos y de actuación necesarios para la consecución de los objetivos planteados.
- Validación de la arquitectura de agentes propuesta en la navegación en exteriores del

tractor comercial AGRIA, denominado DÉDALO.

El trabajo llevado a cabo en esta tesis se describe en la presente memoria en 7 capítulos y dos anexos. La organización es como sigue:

- **Capítulo 1** Presenta la motivación y el marco de desarrollo de la tesis, los objetivos y la organización de la memoria.

- **Capítulo 2.** Muestra el estado del arte en automatización de sistemas para labores agrícolas, clasificando los diferentes tipos de sistemas robóticos existentes; centrándose en los robots móviles en agricultura. Concluye con una valoración de los puntos que se encuentran pendientes de resolución para lograr la navegación autónoma en entornos agrícolas.

- **Capítulo 3.** En este capítulo se analizan los paradigmas de las arquitecturas de control que persiguen la generación de comportamiento autónomo en robots. No pretende ser una revisión exhaustiva sino más bien el análisis de las arquitecturas de control con mayor relevancia a la luz de los requerimientos que plantea la navegación autónoma en entornos dinámicos de exteriores. Al final del capítulo, se formulan los requisitos de partida de una arquitectura de control orientada a la navegación autónoma de robots agrícolas.

- **Capítulo 4.** El capítulo 4 describe la automatización del tractor comercial AGRIA HISPANIA en los talleres del IAI-CSIC en el campus de Arganda del Rey, que dará lugar a una plataforma automatizada, que denominaremos robot DÉDALO, preparada para añadirle las capacidades necesarias para la navegación autónoma. Se presenta: 1) la selección de los actuadores más apropiados y de las variables de control, 2) el controlador borroso de la dirección y los controladores de tracción, 3) las características y función de los sensores de seguridad y reconocimiento del entorno que se han integrado, y finalmente 4) el sistema de proceso y comunicación, así como la arquitectura software implementada.

- **Capítulo 5.** En este capítulo se describe la arquitectura de control AGRO-AMARA, basada en un modelo de agentes, diseñada e implementada para la navegación autónoma del tractor DÉDALO. Esta arquitectura tiene como pilares los principios de reutilización de habilidades y facilidad de escalado. Tanto el diseño de la arquitectura como el de los agentes tiene en cuenta los requisitos y restricciones, extraídas en el análisis realizado en el capítulo 3 sobre robots móviles agrícolas. La tarea de navegación se divide en una navegación global de propósito general, orientada al desplazamiento en campo abierto y una navegación específica orientada al tipo de laboreo a realizar. Se analizan con detalle en cada agente las entradas/salidas, el tipo de representación del sistema/entorno sobre el que trabaja y los algoritmos de proceso para la toma de decisión encapsulados en el agente.

- **Capítulo 6.** Una arquitectura de control diseñada para un robot real tiene que ser validada mediante un conjunto amplio de experimentos. Los experimentos mostrarán si la arquitectura es o no es adecuada a la plataforma robótica, entorno y objetivos. El capítulo 6 describe una serie de experimentos de navegación en el campo con el robot DÉDALO y con el simulador, descrito en el anexo B, orientados a mostrar el funcionamiento de la arquitectura en las tareas de navegación general y de laboreo. Se presentan también otras pruebas que ilustran las prestaciones y capacidad de reacción de los distintos agentes.

- **Capítulo 7.** En este capítulo se resumen las lecciones aprendidas y las principales aportaciones de este trabajo, para finalizar con comentarios relativos a la investigación futura que se corresponden con puntos abiertos en la investigación presentada en esta tesis.

- **Anexo A.** Este anexo recoge el pseudocódigo de los agentes desarrollados en la arquitectura AGRO-AMARA para proporcionar al lector detalles concretos sobre los algoritmos desarrollados.

- **Anexo B.** Debido al gran consumo de tiempo que supone la experimentación con un tractor de tamaño medio, en entornos de exterior con condiciones climáticas variables, se

consideró imprescindible el desarrollo de un simulador, como la herramienta más adecuada, para la depuración y experimentación con los diseños iniciales de agentes. El simulador nunca sustituye las pruebas con el robot real, pero resulta extremadamente útil como banco de pruebas iniciales, permitiendo comprobar el funcionamiento de los agentes en situaciones de difícil experimentación. En este anexo se describe el simulador desarrollado, el modelo cinemático del vehículo y de los sensores y la forma de modelar el mundo.

Capítulo 2

Robots agrícolas

A lo largo de los últimos 50 años hemos asistido a una gran auge de la maquinaria agrícola que ha tenido como finalidad principal la automatización de aquellas tareas que implican para el operador humano un mayor riesgo o fatiga, y en consecuencia una disminución de la precisión. Los productos han evolucionado principalmente en dos direcciones: por un lado en el desarrollo e implantación de manipuladores para la realización de labores agrícolas extensivas, como el cosechado, poda, etc. y por otro hacia el diseño de vehículos con mayor grado de autonomía de forma que cada vez el agricultor adopte el papel de colaborador, gestor o supervisor de las tareas. Esto último tiene como propósito mejorar la concentración del conductor en la propia tarea agrícola disminuyendo la carga de atención que la conducción requiere bajo ciertas restricciones, como por ejemplo asegurar un recorrido completo del campo. Los resultados que se han obtenido con el uso de manipuladores agrícolas automáticos son excelentes, aunque estos deben ser transportados por un vehículo con conducción manual o semiautónoma. No

sucede lo mismo en relación con los sistemas móviles, pues su uso en el campo es aún reducido limitándose en la mayoría de los casos a sistemas de ayuda al guiado manual de alta precisión y sólo en los últimos años han empezado a despuntar los primeros resultados experimentales en exterior fruto de la investigación realizada previamente en laboratorio y en recintos interiores.

El relevar al agricultor de la conducción del vehículo permitirá que éste fije su atención en otros aspectos relativos al laboreo, ya sea cosechar, sembrar, o fumigar. Así mismo los sistemas de guiado aumentan la precisión en la conducción de los vehículos agrícolas, logrando una mejor cobertura del terreno. Este punto es fundamental tanto en tareas de recolección como de fumigación. La idea de lograr sistemas de guiado para vehículos agrícolas que posibiliten un cierto grado de autonomía, se halla presente en la comunidad científica desde hace aproximadamente 50 años [Wilson, 2000]. Estos sistemas de guiado se basan en un medida sensorial que permite calcular la diferencia entre la trayectoria deseada y la real. La señal de guiado utilizada en los primeros tiempos y la usada en los dispositivos de los últimos años ha experimentado un cambio sustancial, sin embargo las motivaciones básicas y las ideas subyacentes no han variado [Jahns, 1975, Jahns, 1983, Wilson, 2000].

Aparte del guiado del vehículo existen otras tareas agrícolas susceptibles de automatización por tratarse, por ejemplo, de labores altamente repetitivas, tediosas y arduas [Thuilot et al., 2002], como es el laboreo en grandes extensiones de terreno. En este caso el beneficio de la automatización se plasma en un aumento de la producción [Stentz et al., 2002] ya que la concentración del operador en la tarea repetitiva se desplaza a otras labores. También requieren automatización las actividades peligrosas y en este caso los beneficios se traducirían en un incremento de la seguridad y la salud de los operarios. Por ejemplo, en el caso de cultivos en invernadero, la automatización evita la exposición prolongada de los operarios a las altas temperaturas que alcanzan los recintos de plástico que protegen los cultivos. Así mismo la automatización permite disminuir los costes de operación [Stentz et al., 2002], ya que con técnicas más precisas se reduce el uso de los recursos: energía, trabajo, semillas, fertilizantes y agroquímicos [Scarlett, 2001]. De igual forma una reducción en la cantidad de fertilizantes y agroquímicos mediante una aplicación selectiva [Kassler, 2001] de los distintos productos o

la eliminación mecánica de algunas plagas [Astrand y Baerveldt, 2002] se transforma en un beneficio directo medioambiental al disminuir la contaminación de los suelos y las aguas. Por último, la automatización del proceso de recolección individual de frutos y vegetales [Hagras et al., 2002] basándose en las condiciones de maduración y tamaño [Kassler, 2001] ayudaría al desarrollo de la agricultura orgánica, contribuyendo a una alimentación de mayor calidad. Hay que tener en cuenta además que la recolección de fruta y hortalizas es una operación costosa por el número de operarios necesarios, el gran esfuerzo físico requerido y las condiciones adversas que se han de soportar; alta temperatura en el caso de los invernaderos o muy bajas en recolecciones de invierno, como la aceituna.

Pese a los claros beneficios de la automatización de las tareas agrícolas todavía existen barreras para un su desarrollo [Kassler, 2001]. Algunas de las principales dificultades son: 1) la complejidad de los sistemas mecánicos, que requieren unas adaptaciones precisas para suplir las habilidades de un trabajador especializado y 2) el factor económico, ya que la producción agraria se realiza por temporadas y muchas prácticas agrícolas tradicionales son difíciles de modificar al introducir nuevas tecnologías.

Con respecto a los vehículos autónomos, éstos siguen siendo aún, en la mayoría de los casos sujeto de investigación y experimentación en interiores. De los prototipos ya desarrollados apenas existe oferta comercial y las que aparecen van dirigidas a la incorporación de sistemas cerrados de ayuda al guiado de vehículo. Las dificultades en este campo se resumen en que la navegación de laboreo se desarrolla en un entorno natural al aire libre, parcialmente estructurado y dinámico, donde el terreno es inconsistente, el producto a detectar irregular, las condiciones atmosféricas variables y a veces hostiles (alto o bajo grado de humedad y temperatura) [Edan y Miles, 1994, Henten et al., 2002] y además pueden aparecer animales o personas de forma imprevista [Hagras et al., 1999]. Todas estas condiciones requieren sistemas dotados de dispositivos de seguridad y protección que cumplan con la legislación vigente añadidos a los que se necesitan para dotar al vehículo de un cierto grado de autonomía en la navegación y en el laboreo. Entre las tareas que podrían beneficiarse de la implantación de un cierto grado autonomía en el guiado están las tareas de laboreo extensivo, como la recolección

de cereales; que al igual que segar el césped o retirar la nieve, requieren conducir un vehículo sobre una superficie irregular [Ollis, 1997]. En este tipo de tareas, debido al bajo valor añadido del producto recogido, la incorporación de vehículos autónomos sólo tiene sentido en cultivos de gran extensión. Como solución a caballo entre la conducción manual y la autónoma se encuentran algunos sistemas comerciales enfocados exclusivamente a la ayuda al guiado de tractores, que se describen en detalle en la sección 2.2.2.

Otra tarea agrícola que se vería beneficiada de un guiado autónomo es el tratamiento selectivo de cultivos. Esta nueva forma de agricultura conocida como Agricultura de Precisión está últimamente cobrando especial importancia con el auge de la agricultura orgánica y el deseo de reducir los productos químicos aplicados tanto a los cultivos como al medio ambiente. En este caso las técnicas de automatización son necesarias para lograr una aplicación selectiva y eficaz de fertilizantes y herbicidas. Los sistemas de tratamiento selectivo de cultivos, implican funciones de percepción para el reconocimiento y diferenciación entre cultivo y mala hierba, [García-Pérez et al., 2000b]. Algunos ejemplos relevantes se describen en la sección 2.2.3.

Por último, con el propósito de lograr una presentación clara se han dividido los robots agrícolas en dos tipos atendiendo a la capacidad de desplazamiento autónomo de los mismos. El primer grupo lo forman los robots manipuladores, compuestos básicamente por robots recolectores. La sección 2.1 de este capítulo revisa algunas de las aplicaciones existentes de este tipo de robots en agricultura, horticultura y jardinería. El segundo grupo lo forman los robots o plataformas móviles con cierto grado de autonomía. En la sección 2.2 se analizan las características de los robots móviles agrícolas y se propone una clasificación en dos categorías en función del grado de intervención humana necesaria para su funcionamiento. La primera de estas categorías engloba a los sistemas de guiado de vehículos agrícolas, que se describen en la sección 2.2.2. La segunda, sección 2.2.3, se centra en los sistemas con mayor grado de autonomía. Este capítulo concluye con una breve reflexión sobre el tipo de actividades implementadas con y sin éxito y cuáles pueden ser las futuras líneas de investigación.

2.1 Robots manipuladores

En la recogida selectiva y automática de frutas y hortalizas los robots ofrecen un gran potencial al incrementar la productividad reduciendo el impacto ambiental. En esta sección se describen brevemente una serie de robots recolectores. Sin embargo existen aplicaciones que no son de recolección que también se benefician del uso de robots. Un ejemplo de esto último lo constituye en el campo de la jardinería el robot japonés del laboratorio de ingeniería aplicada a los sistemas agrícolas (LASE) para recogida y preparación de crisantemos [LASE, 2002a]. Se trata de un robot que corta y prepara los crisantemos para su comercialización, figura 2.1. Utiliza un sistema de visión para detección del entorno y entre otras utilidades incluye la de quitar las hojas de la flor.

Figura 2.1 : Robot para la recogida y preparación de crisantemos [LASE, 2002a]

Otro ejemplo de manipulador es el robot transplantador desarrollado en el instituto Brain (Bio-oriented technology research advancement institute), (Japón). Muchas hortalizas se siembran en invernaderos y posteriormente son trasplantadas al campo. El manipulador se diseñó para automatizar el trasplantado automático de hortalizas, figura 2.2. Un tractor transporta la máquina al punto deseado del campo, allí ajusta la distancia entre cada planta

y la profundidad de plantado, adaptándose así a diversos tipos de hortalizas, [BRAIN, 2002]. Otro proyecto de este centro es un vehículo para trasplantar arroz, fertilizar y aplicar productos químicos en arrozales [BRAIN, 2002], figura 2.3. La precisión de este equipo mejora la producción y calidad del arroz.

Figura 2.2 : Máquina transplantadora [BRAIN, 2002]

2.1.1 Robots recolectores

La mayoría de los robots recolectores son manipuladores para invernaderos, que reconocen, seleccionan y recogen diferentes tipos de frutas y hortalizas. Todos ellos disponen de sistemas de reconocimiento y localización de la hortaliza o fruta, así como de un manipulador que permite su recogida. Suelen estar diseñados para ser transportados por un vehículo, que en algunos casos posee cierto grado de autonomía y está especialmente ideado para una determinada aplicación.

A continuación se describen brevemente seis manipuladores para labores de recolección de distintos frutos y hortalizas: cítricos, tomates, pepinos, champiñones, melones y coles.

- **Agribot.** Agribot es un robot recolector de cítricos, figura 2.4, desarrollado en el Instituto de Automática Industrial (IAI-CSIC). Inicialmente fue concebido para seguir una estrategia

Figura 2.3 : Imágenes del vehículo multiuso para arrozales [BRAIN, 2002]

asistida [Jiménez, 1998], de modo que el robot-manipulador, situado sobre una plataforma móvil, estaba dotado de una pinza y un sistema de localización y proceso. El manipulador recoge los frutos seleccionados por el operario gracias a un telémetro láser manejado desde una consola. Posteriormente se sustituyó el proceso de selección manual por uno automático, con la ayuda de un láser de barrido tridimensional [Jiménez et al., 1999]. Con la información proporcionada por este sensor: coordenadas esféricas y atenuación de la señal, se construyen diferentes mapas del entorno, ángulos de azimut y elevación, distancia y atenuación. Usando las dos últimas imágenes junto con un modelo del sensor se obtiene una imagen de reflectancia. Esta imagen, teóricamente, proporciona la medida de la energía absorbida en cada punto, dependiente únicamente de la superficie del objeto. En ella se pueden distinguir diferentes tipos de superficie. El tratamiento en la imagen de reflectancia y de la forma posibilita el

reconocimiento de los frutos lo que unido a la distancia permite determinar la posición final del fruto, necesaria para una recolección selectiva. Los resultados presentados muestran que sólo el $74\,\%$ de los frutos verdes son detectados mientras que los frutos maduros son localizados y recogidos correctamente en el $100\,\%$ de los casos.

Figura 2.4 : AGRIBOT, un robot para la recogida selectiva de cítricos

- **Cosechadora de tomates.** La cosechadora automática, figura 2.5, desarrollada en Japón en el LASE [LASE, 2002d], consta de un brazo manipulador, una mano y un sensor visual instalados en un vehículo comercial de transporte en invernadero que se desplaza automáticamente sobre los carriles del invernadero. El sistema de visión discrimina los tomates, mediante el análisis de la imagen en color y un par de cámaras estéreo localizan las posiciones 3D de los frutos. Finalmente el brazo se mueve hasta el fruto y los efectores finales lo succionan neumáticamente, afinando la localización del fruto mediante un foto-interruptor.

- **Recolector de pepinos (LASE).** El recolector de pepinos, también desarrollado en el LASE-Japón, es un manipulador articulado guiado por una cámara monocroma de TV dotada de un filtro para separar el pepino de las hojas y los tallos por reflectancia espectral, figura 2.6.

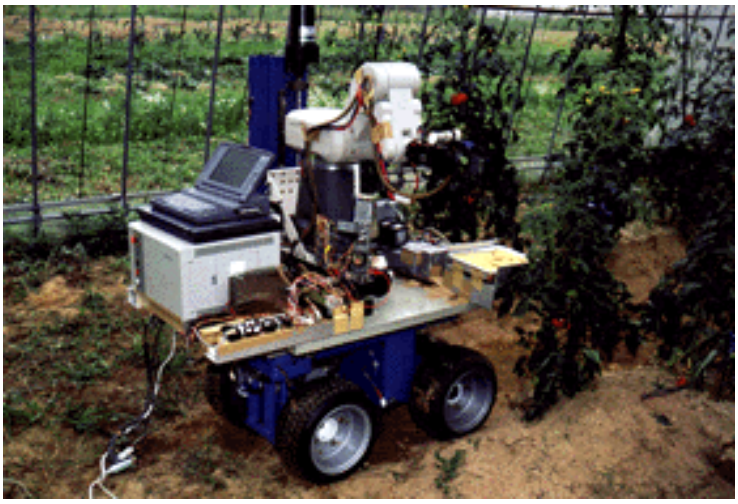


Figura 2.5 : Cosechadora de tomates LASE-Japón [LASE, 2002d]

Con el efector final se detecta el pedúnculo mediante un sensor específico y unos dedos dotados de un elemento cortante lo seccionan recogiendo el pepino [LASE, 2002b]. En esta línea, el laboratorio LASE dispone de un robot recolector de fresas y otro para diferentes labores en viñedos, figura 2.7.

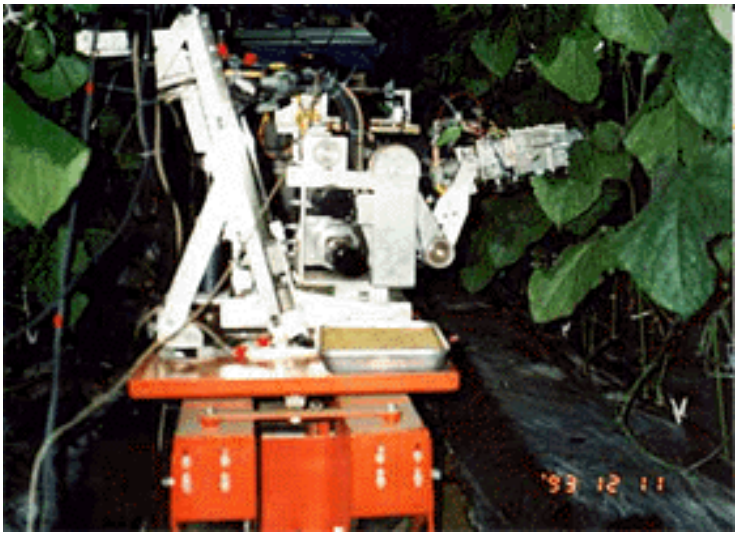


Figura 2.6 : Recolector de pepinos LASE-Japón [LASE, 2002b]

- **Recolector de champiñones.** El recolector de champiñones del Silsoe Research Institute (Reino Unido) está dotado de visión monocroma para localizar y clasificar por tamaños los

Figura 2.7 : Recolector de fresas (izquierda) y robot para labores en viñedos (derecha) [LASE, 2002c]

champiñones, decidiendo cuáles se recogerán primero y el plan de recogida. La recolección se efectúa mediante un mecanismo de succión localizado al final de un robot cartesiano, figura 2.8. Una vez recogido el champiñón se empaqueta al otro lado de la máquina [Institute, 2002].

Figura 2.8 : Robot recolector de champiñones [Institute, 2002].

- **Recolector de melones.** El recolector de melones es un manipulador cartesiano situado sobre una plataforma adaptada para ser remolcada por un tractor, figura 2.9. Detecta los melones sobre imagen en blanco y negro, y calcula su posición para proceder a la recogida en función del grado de madurez, calculado a partir del tamaño y forma del melón, [Edan et al., 2000].

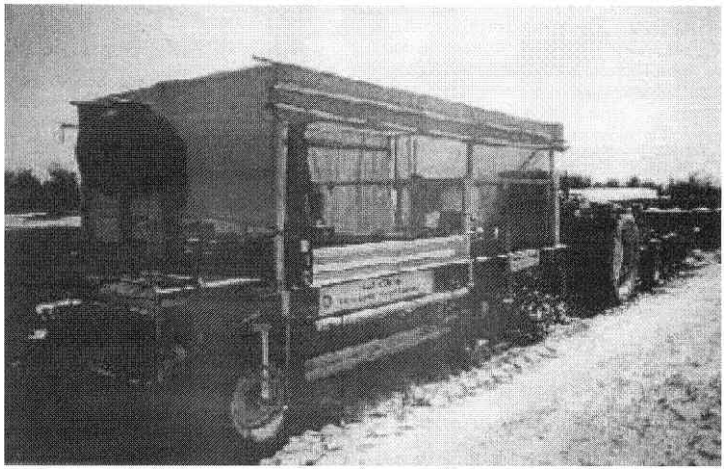


Figura 2.9 : Recolector de melones [Edan et al., 2000]

- **Cosechadora de coles.** El grupo de tecnología agraria del instituto BRAIN ha desarrollado una máquina automática cosechadora de coles, figura 2.10. Esta cosechadora, eleva la col del suelo, corta la cabeza, separándola de las hojas y deposita la hortaliza en un contenedor. El robot no ocasiona daño alguno en la col, e invierte 1 ó 2 segundos en recoger cada pieza, sin ningún error [BRAIN, 2002].

Figura 2.10 : Cosechadora de coles [BRAIN, 2002]

2.2 Robots móviles

La mayoría de los sistemas automáticos descritos anteriormente no disponen de capacidad de desplazamiento y tanto el recolector de champiñones como el de crisantemos o melones se transportan en un tractor. Este apartado se centra en el análisis de los sistemas móviles con una cierta autonomía en aplicaciones de agricultura, horticultura y jardinería, sistemas con capacidad de desplazamiento y decisión sobre el mismo, en entornos sin marcaciones específicas. El concepto de autonomía es gradual, y por ello engloba desde sistemas con habilidad para seguir trayectorias rectilíneas prefijadas hasta sistemas capaces de detectar y reaccionar adecuadamente ante obstáculos imprevistos.

2.2.1 Clasificación y requisitos

Dependiendo del grado de autonomía que se pretende conseguir, varían los requisitos y el planteamiento de implementación de cada sistema. Por ello los trabajos existentes se han dividido en dos grandes grupos:

1. **Sistemas de ayuda al guiado.** Son sistemas de aviso para el seguimiento preciso de trayectorias que reducen el estrés asociado a la realización de tareas que exigen mucha concentración a lo largo de muchas horas, aunque requieren que el operario se encuentre a bordo del vehículo.

2. **Sistemas de navegación autónoma.** En este caso el objetivo es la navegación no tripulada capaz de resolver las situaciones previsibles y gran parte de imprevistos, es en este último aspecto dónde radica el grado de autonomía. En este caso, el operario podría vigilar, desde una estación de control, la navegación y el laboreo de uno o varios tractores [Stentz et al., 2002].

Los requisitos para el funcionamiento seguro y eficaz son mucho más exigentes cuando se trata de navegación sin conductor, que en el caso de sistemas de ayuda al guiado de un vehículo, ya que en este último caso es el operario quien resuelve las situaciones imprevistas críticas.

En cualquier vehículo la seguridad constituye un requisito imprescindible para garantizar la integridad del conductor, vehículo y entorno. De ahí que un vehículo autónomo debe ser capaz de reaccionar ante posibles colisiones, detectando los obstáculos imprevistos y emprendiendo las acciones necesarias para evitarlos mientras sigue un objetivo. También debe ser robusto frente a errores o fallos en la señal de guiado, el sistema de control o el suministro de energía, deteniéndose [Jahns, 1975].

Además de dar una respuesta a los requisitos comunes de seguridad, la autonomía de un robot móvil implica que el sistema debe navegar en ambientes parcialmente conocidos y con incertidumbre sin necesidad de realizar modificaciones en el entorno [Jahns, 1997]. Conviene remarcar que para que pueda ser manejado por un operario que no es experto en robótica, el sistema debe ofrecer toda la información que el usuario requiera [Jahns, 1997] para una supervisión eficiente y segura.

2.2.2 Sistemas de ayuda al guiado

Los sistemas de ayuda al guiado tienen como objetivo el seguimiento de una trayectoria definida desde una posición inicial a otra final, alertando al conductor, mediante dispositivos sonoros o luminosos, cuando el error en la posición supera un umbral.

En el guiado manual el conductor realiza el control comparando la trayectoria deseada con la real, corrigiéndola mediante un giro del volante. En el caso del guiado automático se trata de reemplazar al operador diseñando un sistema de control similar [Jahns, 1983]. Un punto clave lo constituye la señal de realimentación del controlador para comparar la trayectoria definida con la realizada. En este último aspecto es donde se centra prácticamente toda la investigación de guiado automático de vehículos agrícolas.

El desarrollo de los sistemas de ayuda al guiado en vehículos agrícolas comenzó hace más de 70 años [Jahns, 1983]. Desde entonces los sistemas sensoriales han experimentado una gran evolución, existiendo una gran diferencia entre la tecnología que se utilizaba en los inicios y la actual. Hoy en día, prácticamente todos los sistemas automáticos de guiado disponen de

sensores GPS, brújula y algunos de ellos de cámara de vídeo o láser, aunque también se puede encontrar algún sistema comercial con sensores mecánicos [Keicher y Seufert, 2000].

Los sistemas de guiado pueden agruparse en dos categorías. En la primera, **métodos de guiado indirecto**, se encuentran aquellos en los que el operario dirige al tractor remotamente. En la segunda se encuentran los **métodos de guiado directo**; en ellos las señales de guiado proceden de sensores a bordo del vehículo. Estos últimos se dividen a su vez en dos clases, dependiendo del tipo de información, global o local, con la que opera el algoritmo de control. Los sistemas de guiado con información global dirigen al vehículo por una ruta previamente calculada, basada en un mapa del terreno y en la posición del vehículo respecto de un marco absoluto de referencia, calculada mediante un receptor GPS, brújulas o un sistema de balizas. Los sistemas de guiado con información local se basan en la percepción de marcas locales, como pueden ser los patrones de la plantación, los surcos entre cultivos o las plantas individuales. Los últimos trabajos sugieren la conveniencia del uso combinado de información tanto global como local [Stentz et al., 2002, Pilarski et al., 2002].

Métodos de guiado indirecto.

De entre los métodos de guiado indirecto, el control remoto, desechado en los años 80, ha sido recuperado [Fong y Thorpe, 2001] en la actualidad, debido fundamentalmente a los avances en las comunicaciones, mayor ancho de banda, que permiten que el usuario remoto disponga de información completa del entorno. Un vehículo agrícola teleoperado se ha desarrollado en el instituto Brain, [BRAIN, 2002], equipado con dos cámaras para visualizar tanto la zona frontal como la trasera y lateral del vehículo, figura 2.11. Está dotado de un GPS y un sistema de comunicación por radio para enviar datos y recibir las consignas de control de giro, velocidad y freno. El operario visualiza todas las imágenes y señales en un panel de control, mediante el cual puede conducir remotamente el tractor.

En otros trabajos se muestran métodos de guiado remoto por seguimiento directo de un vehículo maestro conducido por un humano, ya sea mediante unión mecánica o sin ella

Figura 2.11 : Tractor controlado remotamente para distintas labores agrícolas [BRAIN, 2002]

[Jahns, 1983]. En [Iida et al., 1998] se describe un vehículo que sigue automáticamente a otro. Utiliza sensores de ultrasonidos situando los emisores en el vehículo maestro y los receptores en el esclavo. A partir de la medida de cuatro distancias diferentes calcula la señal de control que necesita el vehículo esclavo para seguir al maestro. Se utiliza en campos de golf [Torii, 2000].

Métodos de guiado directo

- **Guiado de tractores con sistema de referencia absoluto** En los sistemas de guiado con marco de referencia absoluto la ruta planificada al inicio se ejecuta sin modificación alguna. Es una aproximación adecuada para mundos ideales o totalmente estructurados pero no está preparada para responder ante cambios imprevistos del entorno [Stentz et al., 2002] al disponer únicamente de la posición absoluta.

Las dos ventajas principales de este tipo de sistemas son: 1) no es necesario modificar el entorno instalando balizas y 2) el cálculo de la posición es inmediato. Sin embargo son sistemas de elevado coste, proporcionan las medidas a una frecuencia baja, sufren pérdidas de la señal por causas muy diversas y en algunas aplicaciones necesitan un mapa georreferenciado del entorno

de trabajo. En estos casos se navega casi exclusivamente con el sistema de posicionamiento por satélite, el GPS. Al ser sistemas de ayuda al guiado no contemplan la aparición imprevistos ya que el operario se encarga de resolverlos.

Algunos sistemas comerciales utilizan el GPS, ya sea RTK-GPS o DGPS (en función de la precisión requerida [Trimble, 2002]). El sistema AutoPilot de Trimble [Trimble, 2002], figura 2.12, consta de un DGPS (o RTK-GPS) y un sistema de control que actúa sobre la dirección hidráulica del tractor. El agricultor selecciona en la consola dos puntos que definen el camino rectilíneo por el cual AutoPilot guía al tractor, basándose en la posición obtenida mediante el GPS. Al llegar al segundo punto, AutoPilot alerta para que se retorne al modo manual a fin de realizar el giro. AutoPilot registra en memoria el recorrido efectuado por el tractor, para evitar repeticiones sobre zonas que ya se han recorrido.

Figura 2.12 : Sistema comercial AutoPilot de Trimble

El sistema de IntegriNautics, AutoFarm [Integrinautics, 2002], figura 2.13, consta de cuatro receptores GPS, tres en el tractor y otro más en la estación base para el cálculo de correcciones que incrementen la precisión hasta un valor inferior a la pulgada. Con los tres receptores GPS a bordo del tractor se obtiene su posición e inclinación lateral y frontal. En el caso de que el accionamiento de la dirección estuviese automatizado, se podría abordar un control automático guiado por la localización GPS. Aunque se trata de un sistema caro, los fabricantes argumentan

que reduce los costes de la operación optimizando el uso de productos químicos y además permite emplear la máquina tanto de día como de noche.

Figura 2.13 : Sistema comercial AutoFarm de IntegriNautics

Beeline [Beeline, 2002] utiliza también un DGPS para obtener la posición del vehículo, y controlar el giro del tractor en el seguimiento preciso, error de 2 (cm.), de una trayectoria rectilínea, figura 2.14.

Figura 2.14 : Sistema comercial de control de trayectoria rectilínea con GPS Beeline [Beeline, 2002]

A diferencia de los sistemas comerciales anteriormente expuestos el sistema de guiado de Thuilot et al. [Thuilot et al., 2002] permite la conducción automática de un vehículo en trayectorias curvas además de rectilíneas, figura 2.15. Integra los resultados obtenidos con

un CP-DGPS (Carrier Phase DGPS) de precisión centimétrica en el modelo cinemático del tractor. Mediante un filtro de Kalman se genera el algoritmo de control para cualquier tipo de trayectoria. Los resultados son muy precisos en los segmentos rectilíneos de una trayectoria típica en agricultura (dos rectas unidas por una curva). Sin embargo la precisión disminuye en el tramo curvo, ya que al tratarse de giros muy cerrados en algunas ocasiones se llega al valor de saturación de los actuadores. No obstante no es necesaria tanta precisión en estos tramos curvos que constituyen trayectorias de transición entre las zonas rectilíneas.

Figura 2.15 : Tractor con sistema de guiado basado en CP-DGPS [Thuilot et al., 2002]

Muy similar al trabajo anterior pero con vehículo de diseño propio, es el del National Agricultural Research Center (NARC) en Japón. El sistema de guiado utiliza en este caso un filtro de Kalman para obtener la posición instantánea del tractor a partir de las medidas DGPS y de un giróscopo de fibra óptica. El error en las trayectorias rectilíneas es inferior a $0{,}10$ (m.) y en los giros en U es de $0{,}12$ (m.), [Torii, 2000].

- **Guiado con información local** Los sistemas de guiado basados en información local aprovechan la detección de estructuras y características del entorno, a fin de localizar de forma relativa el vehículo y permitir su guiado. Normalmente se basan en la detección de patrones de

plantación, surcos, o en diferencias entre zonas afectadas o no por una operación de laboreo, para corregir la trayectoria del robot.

Debido a la reducción del coste de los sistemas sensoriales, los métodos de guiado que inicialmente utilizaban dispositivos mecánicos para detectar las hileras de plantas, por ejemplo maíz, han sido sustituidos por otros sensores. Tecnologías que hace 20 años eran inasequibles por su precio y reducidas prestaciones [Jahns, 1983] como los sistemas de visión, los ultrasonidos o el láser comienzan a emplearse en la actualidad. La integración de estos sensores, permite hoy en día ampliar el campo de aplicación de los sistemas de guiado a cultivos sin necesidad de imponer un contacto físico sensor-cultivo. Entre los sensores que no requieren contacto físico se encuentran las cámaras CCD, el láser, los sensores de ultrasonidos y los telémetros; siendo las cámaras de visión las más difundidas. Las ventajas de las cámaras frente al de los sistemas de guiado con información global radican en su portabilidad, coste inicial del equipo y posibilidades de extracción de información compleja. Aún existen muchos problemas por resolver antes de que se extienda el uso de las cámaras en los sistemas comerciales, ya que el análisis de imágenes en tiempo real y en exteriores es una tarea compleja.

- **Guiado de tractores con cámara de visión.** A pesar del elevado coste de desarrollo de las aplicaciones basadas en imagen visual y los problemas asociados a los cambios de luminosidad y polvo ambiental, las posibilidades de extracción de conocimiento de las cámaras hacen que éstas sean el sistema sensorial más utilizado para el control del guiado de un tractor. Por otro lado, el posicionamiento mediante visión artificial permite aprovechar la estructura en surcos de las plantaciones y utilizarla para controlar el robot. En esta línea se enmarcan los trabajos [Billingsley y Schoenfish, 1995, Caladín et al., 2001, Tillet et al., 1993, Slaughter et al., 2000]. El tractor guiado por visión de la universidad de Tokio [Torii, 2000] usa la representación del color HSI (saturación, intensidad y tono) para lograr un sistema más robusto frente a los cambios de iluminación; el de la universidad de Hokkaido, también guiado por visión, ha sido probado con éxito en campos de espinacas [Torii, 2000]. Steeroid [Billingsley y Schoenfish, 1995] presenta un sistema de guiado automático que provoca un giro

con ángulo proporcional a la distancia observada entre el tractor y los surcos. Sin embargo tanto el polvo como los cambios de iluminación constituyen todavía un problema en la aplicación de sistemas de visión en exteriores.

- **Guiado de tractores con láser.** El dispositivo láser, contrariamente a la cámara CCD, es independiente de los cambios de luminosidad. Sin embargo no es tan versátil como una cámara y sólo puede aplicarse para la generación de mapas 2D de profundidad de las estructuras presentes en el entorno que reflejan el haz de luz. Sí en la operación de segar, la estructura o patrón de guiado es la diferencia de alturas entre zona de cultivo cortado y sin cortar. A partir de la detección de este cambio, es posible alinear el tractor [Chateau et al., 2000], figura 2.16. Por la filosofía de ajuste que se sigue únicamente lo utilizan cosechadoras y segadoras. Es un sistema poco versátil aunque carece de los inconvenientes de la visión. El sistema comercial Laser Pilot de la empresa Claas [Claas, 2002], figura 2.17, utiliza también un láser para detectar zona cosechada y no cosechada, a fin de alinear el tractor y optimizar el solapamiento entre zonas.

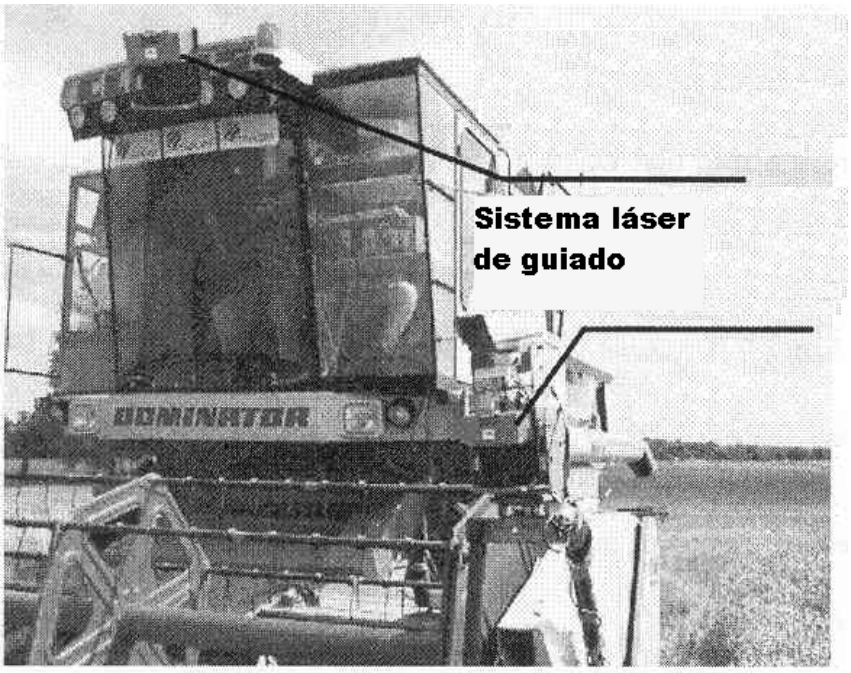


Figura 2.16 : Tractor con sistema de guiado láser [Chateau et al., 2000]

Figura 2.17 : Sistema comercial de guiado: AutoPilot de Claas

El sensor láser se utiliza también con un sistema de balizas reflectoras, para establecer la posición del robot mediante triangulación, conociendo la posición de los reflectores [Keicher y Seufert, 2000]. La precisión de este sistema es alta, pero está limitado por: 1) el número de reflectores que aumenta con las dimensiones del campo de cultivo y 2) la necesidad de determinar la localización de las balizas con precisión.

2.2.3 Sistemas de navegación autónoma

La autonomía es un concepto gradual y aunque el objetivo final de la robótica aplicada a la agricultura el desarrollo de sistemas autónomos para la realización de las tareas agrícolas, aún estamos lejos de una plataforma comercial autónoma. La dificulta para reproducir los mecanismos de razonamiento y percepción humanos provocan que hasta el momento, se han desarrollado únicamente plataformas semiautónomas económicamente viables sólo en aquellos casos en los que el valor de producto obtenido deja un margen de beneficio [Stentz et al., 2002]. Por otro lado existe cierta resistencia a la introducción de nuevas prácticas en cultivos fundamentalmente por el aprendizaje que requiere la utilización de nuevas tecnologías. Las máquinas semiautónomas implementadas permiten al operario intervenir sólo en ocasiones excepcionales mientras supervisa la navegación de uno o varios sistemas. En definitiva,

aumentar la autonomía del robot consiste en reducir el número de situaciones en las que es necesaria la intervención humana; y para ello se requiere que los sistemas sean robustos y seguros.

El alto grado de repetitividad que muestran la mayoría de las labores agrícolas y la existencia de un supervisor humano, favorecen la automatización de vehículos agrícolas con un diseño más conservador, ya que es posible delegar en el operario la resolución de situaciones imprevistas complejas.

De los nueve vehículos agrícolas que se describen a continuación únicamente 3 son plataformas comerciales: el tractor fumigador de naranjos y la cosechadora DEMETER del Robotics Institute (RI), Carnegie Mellon University y el recolector de pepinos de Wageningen, Países Bajos. El resto son vehículos prototipos de laboratorio: un fumigador de coliflores en Inglaterra; un vehículo para su utilización en agricultura orgánica, arrancando malas hierbas en campos de remolacha; un prototipo para la investigación sobre auto organización mediante aprendizaje “online”, aplicable a agricultura sostenible; el robot español AURORA para invernaderos y dos robots agrícolas japoneses, uno de propósito general y otro dedicado a la plantación de arroz.

Vehículo agrícola para plantaciones de hortalizas del SRI

El objetivo del robot desarrollado en Silsoe Research Institute (SRI) del Reino Unido, figura 2.18, es la navegación siguiendo líneas de cultivo para realizar una aplicación selectiva de productos químicos, tras la detección y segmentación de las malas hierbas frente a las hortalizas. La precisión requerida en el guiado, para fumigación selectiva de cultivos en huerta, sólo se puede alcanzar con un vehículo automatizado [Hague et al., 1999a].

El sistema de navegación genera las consignas de guiado a partir de la ubicación de éste con respecto a las líneas de cultivo, que actúan como los cables guía en vehículos filoguiados industriales. Las imágenes se capturan con un filtro infrarrojo y se segmentan, separando plantas, malas hierbas y suelo. De la secuencia de imágenes segmentadas se recuperan los

parámetros de movimiento del vehículo, siguiendo ciertas características de las plantas a lo largo de las diferentes imágenes. A partir de la intersección entre las imágenes y de la información de calibrado de la cámara, se reconstruye el mapa del cultivo. Del mapa se extrae la información de la orientación relativa del vehículo con respecto de la línea de cultivo, [Sanchiz et al., 1998]. Igualmente se calcula la posición de las plantas, de las malas hierbas y del instante de apertura de las válvulas para aplicar el pesticida selectivamente.

En trabajos posteriores [Hague et al., 1999a, Hague et al., 1999b, Lindgren et al., 2002] se han añadido al robot otro sensores como la odometría y la brújula. La información de estos sensores se integra en el modelo cinemático del vehículo, junto con los datos proporcionados por el sistema de visión, y mediante un filtro de Kalman extendido se deducen los parámetros de control. El robot navega de modo autónomo con precisión entre líneas de cultivo detectando el final de surco y girando. La ausencia de plantas y el conocimiento aproximado de la longitud de un surco proporcionan la información adicional necesaria para detectar que el final del surco. El tratamiento de las imágenes recogidas permite distinguir entre planta de mala hierba y suelo para aplicar el herbicida selectivamente.

Figura 2.18 : Robot para tratamientos de hortalizas: Silsoe Research Institute [Hague et al., 1999b]

Vehículo autónomo para arrancar malas hierbas de la universidad de Halmstad

Se ha desarrollado en la universidad de Halmstad (Suecia) un robot múvil para plantaciones orgánicas de remolacha, donde no es posible el uso de herbicidas. El robot [Astrand y Baerveldt, 2002], figura 2.19, consta de dos sistemas de visión, uno frontal para guiado del vehículo por reconocimiento de las líneas de remolachas y otro enfocado hacia el suelo para distinguir las remolachas frente a las malas hierbas y calcular su posición. La arquitectura de control, de tipo pizarra [Hayes-Roth, 1985], está organizada en una capa de control y otra de aplicación. Esto permite aislar el control de giro y arranque de hierbas de la capa de aplicación, encargada de tareas de más alto nivel. La posición calculada sobre la imagen, es enviada al control de la cuchilla para su elevación, evitando que corte la remolacha. El proceso de discriminación en la imagen se basa en tres propiedades: intensidad del plano de color verde, grado de compactación y elongación de la planta. A pesar de tratarse de un vehículo autónomo, no incluye ningún mecanismo de detección de obstáculos [Astrand y Baerveldt, 2002], ni se describen el control del giro.

Figura 2.19 : Robot para arrancar malas hierbas en campos de remolachas orgánicas [Astrand y Baerveldt, 2002]

Vehículo agrícola inteligente IFV5, de la universidad de Manchester

El vehículo descrito en [Hagras et al., 2002] ha sido desarrollado en un proyecto europeo enfocado a la agricultura orgánica (IFV de Intelligent Farm Vehicle), figura 2.20. El objetivo del proyecto es dotar al robot IFV de la capacidad de autoorganizarse mediante aprendizaje "online", con un interfaz sencillo para un usuario inexperto. En el robot se han instalado un sensor GPS, una brújula y una cámara de visión para visualización de sus comportamientos: seguir línea a la izquierda y a la derecha, evitar obstáculo y perseguir objetivo. El aprendizaje parte de un comportamiento inicial definido mediante conjuntos borrosos con funciones de pertenencia fijas, para ir aprendiendo las reglas o modelo cualitativo, a partir de las relaciones entre las entradas sensoriales y las salidas de los actuadores. Una vez aprendido el mejor conjunto de reglas, se pasa a la segunda tarea de aprendizaje dedicada al ajuste de los parámetros de las funciones de pertenencia. En una última etapa se persigue el aprendizaje de los parámetros de coordinación entre los tres comportamientos ya depurados. Se demuestra experimentalmente cómo, tras un periodo de aprendizaje, mejora el comportamiento al seguir perfiles irregulares de cultivo .

Robot de invernaderos AURORA de la Universidad de Málaga

Se trata de un robot autónomo diseñado específicamente para invernaderos en la Universidad de Málaga [Mandow et al., 1996]. En los invernaderos el trabajo es especialmente duro debido a las altas temperaturas, los elevados niveles de humedad y la escasa ventilación. Estos factores hacen que la fumigación, necesaria para combatir las plagas que se desarrollan al amparo de las condiciones favorables del invernadero, sea particularmente peligrosa para el operario. AURORA consta de una plataforma octogonal móvil no holonómica cuya fuente de energía es un generador AC alimentado con gasolina, figura 2.21. Su sistema sensorial está compuesto por diferentes tipos de sensores de ultrasonidos: digitales de rango corto y medio y analógicos de rango medio. Dispone de codificadores de posición en las ruedas y cámara de vídeo para facilitar la supervisión humana. La arquitectura de control consta de cinco niveles, usuario, supervisor,

Figura 2.20 : Intelligent Farm Vehicle, IFV [Hagras et al., 2002]

generador de referencias, ejecutivo y *servo*. El nivel usuario gestiona las comunicaciones con el usuario local. El nivel supervisor es un controlador supervisor de secuencia que coordina el comportamiento global del sistema mediante eventos de comienzo, de espera y temporizadores. El generador de referencias se compone de un conjunto de comportamientos básicos, donde cada uno de ellos produce un esquema de movimiento del robot: seguir pared, seguir pasillo, girar, abrir boquilla, seguridad y avanzar. El ejecutivo controla los sensores internos y los actuadores, e incluye un módulo para que el usuario pueda conducir manualmente el robot. Finalmente el nivel *servo* controla la mecatrónica del vehículo. Los comportamientos de navegación se ejecutan en secuencia, pero de modo concurrente con el comportamiento de seguridad. Su navegación se ha probado en diferentes invernaderos y ha mostrado un buen funcionamiento en tareas de navegación y aplicación de tratamientos.

Figura 2.21 : Robot móvil en invernaderos AURORA [Mandow et al., 1996]

Tractor autónomo y plantadora de arroz en Japón

En Japón existen trabajos relativos a la automatización de labores agrícolas, muchos de ellos vienen recogidos en [Torii, 2000]. Sin embargo existen pocas publicaciones accesibles sobre estas investigaciones por lo que la información disponible es escasa.

En el instituto Brain de Japón se ha desarrollado un tractor autónomo de cultivo, figura 2.22, que integra dispositivos de seguridad, funciones de autodiagnóstico y alarmas, además de realizar un seguimiento automático de objetos en movimiento [BRAIN, 2002, Torii, 2000].

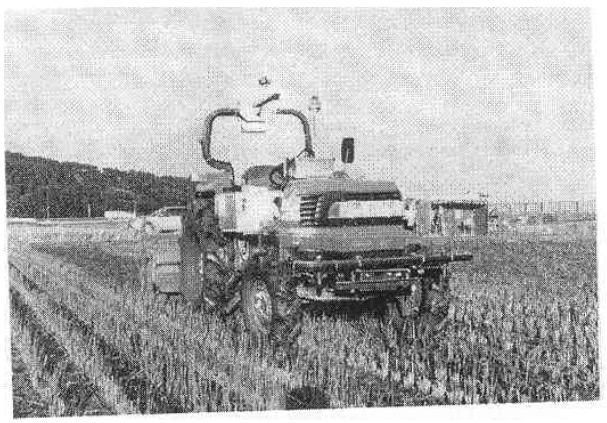


Figura 2.22 : AP-L1, tractor [Torii, 2000]

El Instituto NARC de Japón, referencia un vehículo autónomo para plantar arroz, donde

la localización se obtiene mediante un receptor RTK-GPS y un giróscopo de fibra óptica [Torii, 2000].

Es posible encontrar una referencia al tractor autónomo para producción de forraje desarrollado por el Instituto Nacional de Investigación de Praderas (NGRI) en colaboración con la Universidad de Utsunomiya, Japón. La localización del tractor se obtiene a partir de las medidas de un giróscopo de fibra óptica y un sensor ultrasónico de efecto Doppler [Torii, 2000].

Tractor autónomo para campos de naranjas del RI

Uno de los últimos y más relevantes trabajos en tractores semiautónomos ha sido el realizado por Stentz et al. [Stentz et al., 2002], del RI (Robotics Institute, Carnegie Mellon University), validado en campos de naranjas en Florida con recorridos de hasta 7 (km.). El vehículo es un tractor comercial (John Deere) que opera en dos modos, entrenamiento y semiautónomo. En en primer modo el operario conduce el tractor y graba datos del recorrido mediante una consola auxiliar. En operación semiautónoma el tractor (o la flota de tractores) sigue uno de los caminos previamente almacenados, visualizado en la consola remota del operador. Ante un evento inesperado, activa un mensaje de alarma en la pantalla remota con la información pertinente; en el caso de un obstáculo esta información es una región de la imagen centrada en el obstáculo. Cuando se habla de la localización del vehículo los posibles fallos en posicionamiento en el caso de los sistemas de ayuda al guiado, sección 2.2.2, son resueltos por el conductor que va a bordo del vehículo. En este caso los autores argumentan la conveniencia de emplear múltiples sensores para evitar la parada del tractor y la consiguiente intervención del humano ante un fallo del único sensor disponible. En concreto cuando la localización se realiza a través de un receptor GPS conviene disponer de sensores redundantes ya que este tipo de dispositivo pierde en ocasiones la señal debido en la mayoría de los casos a oclusiones o errores multicamino. En este tractor se ha optado por incorporar un sistema de posicionamiento basado en un RTK-GPS, un giróscopo de fibra óptica, odómetros en las cuatro ruedas y un radar de efecto Doppler, figura 2.23. La información de estos sensores se combina mediante un filtro de Kalman extendido. El seguimiento de un camino se realiza basándose en la estimación de la posición en el ciclo

anterior y en un algoritmo de persecución pura.

Figura 2.23 : Tractor semi-autónomo para plantaciones de cítricos [Stentz et al., 2002]

Finalmente, puesto que se trata de un vehículo sin conductor es imprescindible dotarlo con un módulo de detección de obstáculos. Para ello dispone de dos cámaras CCD y un sensor láser de barrido. El sistema de percepción de obstáculos se basa en una red neuronal, entrenada con imágenes del campo clasificadas por un operario. La red neuronal localiza en tiempo real elementos extraños en la imagen tridimensional reconociendo texturas y colores. Sin embargo la estrategia ante un obstáculo que bloquea el camino del tractor es únicamente el aviso al operario supervisor, mostrándole la zona donde se ha localizado el problema.

Cosechadora autónoma DEMETER del RI

Esta cosechadora [Pilarski et al., 2002], también del RI (Carnegie Mellon University), planifica la cosecha de un campo completo y ejecuta su plan simultáneamente a la detección de obstáculos inesperados. Al igual que el robot para campos de naranjas, [Stentz et al., 2002], integra varios dispositivos en el sistema de posicionamiento: receptor GPS, codificadores de posición de la rueda y giróscopo. Además la cosechadora está dotada de un sistema de visión

con tres módulos: un seguidor de líneas de cultivo (detección de cereal cortado frente al no cortado), un detector de final de campo y otro de obstáculos. El funcionamiento de este vehículo ha sido probado intensivamente en el campo, figura 2.24. Comparando los resultados obtenidos en la localización con GPS con los de la localización con el sistema de visión se comprueba que existe mayor precisión en la localización basada en GPS.

Figura 2.24 : Cosechadora DEMETER [Pilarski et al., 2002]

Robot autónomo recolector de pepinos en invernaderos de Wageningen

El robot desarrollado en Wageningen, Paises Bajos, figura 2.25, es un vehículo autónomo dotado de un manipulador con efector final, dos sistemas de visión y un controlador de trayectorias sin colisión para el manipulador [Henten et al., 2002].

El robot autónomo utiliza las tuberías de calefacción del invernadero como raíles, para guiado y soporte. El vehículo se alimenta con un motor de continua y un *servo* controlador combinado con un codificador. El manipulador, de 7 grados de libertad, posiciona el efector final para el agarre y succiona el pepino a la vez que corta el tallo que une el fruto a la planta. La detección y localización precisa del fruto se realiza mediante el sistema de visión que consta de

dos cámaras. La primera, situada sobre el vehículo, se encarga de detectar el fruto, analizando las propiedades espectrales de la imagen. Basándose en el volumen del pepino, calculado en la imagen, se determina su grado de madurez y de calidad. La segunda cámara, situada en el efector final, permite guiar al efector en la aproximación final al pepino.

La generación de trayectorias libres de colisión se calcula a partir de la información 3D disponible y el conocimiento del entorno y de la estructura del robot. Se ha experimentado con el robot en un invernadero con una eficacia del $80\,\%$ en recolección de pepinos y velocidad de 45 segundos por fruto.

Figura 2.25 : Robot recolector de pepinos en invernaderos [Henten et al., 2002]

2.3 Navegación autónoma en agricultura

En los últimos 50 años se han automatizado un gran numero de tareas agrícolas, fundamentalmente con el uso de manipuladores, como se muestra en la sección 2.1. Ahora bien, son sistemas automáticos que deben ser transportadas, bien sea por un vehículo autónomo o bien sea por uno conducido manualmente para la realización de una tarea. También se han

descrito, en la sección 2.2.2, los sistemas de guiado automático para vehículos agrícolas, útiles cuando se trata de labores en grandes extensiones de terreno, como el cosechado de cereales o el segado de grandes extensiones de césped. Sin duda este tipo de aplicaciones se han beneficiado enormemente del desarrollo de las técnicas de posicionamiento mediante satélite y en concreto de las elevadas precisiones de los sistemas RTK-GPS y DGPS. Sin embargo hasta la fecha son técnicas muy costosas, poco apropiadas para extensiones de tipo medio.

En la sección 2.2.3 se han presentado los ejemplos más significativos de vehículos agrícolas con un grado de autonomía más allá del simple guiado. Pese a que la agricultura es un entorno propicio para la utilización de robots móviles, por el tipo de tareas, tediosas, repetitivas y en algunos casos perjudiciales para la salud del operario, son muy escasos los trabajos relativos a robots móviles autónomos en este campo. Sin embargo, el traspaso de las investigaciones desarrolladas en el campo de la robótica a vehículos agrícolas puede contribuir, hoy en día, al desarrollo de robots agrícolas viables en la realización de un conjunto de tareas tanto en campos de cultivo como en invernaderos.

La seguridad es un requisito fundamental en estos sistemas. La navegación segura supone la capacidad de reacción ante posibles colisiones, frente a errores o fallos en la señal de guiado, en el sistema de control y en el suministro de energía. Además un robot móvil autónomo en agricultura debe poder navegar en ambientes parcialmente conocidos y con incertidumbre sin necesidad de realizar modificaciones en el entorno, con un interfaz con el operario que ofreciendo todo la información necesaria sea de fácil manejo.

Ninguno de los nueve tractores descritos en la sección 2.2.3, que se consideran representativos del estado del arte en robots móviles en agricultura cumplen los requisitos mencionados en el párrafo anterior: seguridad frente a colisiones, errores y suministro de energía, navegación sin modificar el entorno, interfaz sencillo con el operario. Algunos tienen en cuenta la necesidad de evitar colisiones: [Mandow et al., 1996, Stentz et al., 2002, Pilarski et al., 2002, Hagras et al., 2002] e incluso describen e implementan mecanismos para la detección de obstáculos; un segundo grupo sin embargo se conforman con avisar al operario [Stentz et al., 2002], y un tercer grupo no describe cómo implementan la operación de evitar

obstáculo, [Henten et al., 2002].

Ninguno de los trabajos contempla explícitamente cómo gestionar fallos en los sistemas sensoriales, de actuación, de energía o de comunicaciones. La mayoría de los sistemas están muy adaptados a la tarea que tienen que desarrollar, lo que permite, aun en contra de la versatilidad siempre deseable, una implementación más sencilla, completa y eficaz. Integrar los criterios de eficacia y seguridad en los robots agrícolas conduce a una reflexión sobre cómo organizar sus sistemas de control, percepción y representación. En concreto casi todos los sistemas autónomos y semiautónomos analizados en la sección 2.2.3 carecen de una investigación conducente a una organización adecuada de los procesos perceptivos, de control y de comunicación.

Los robots autónomos, ya estén destinados a tareas agrícolas o a cualquier otro tipo de tareas, son sistemas complejos y cuanto más complejo es un sistema más relevancia adquiere el término organización. Al aumentar la complejidad desaparecen las fronteras entre el resultado y el mecanismo para conseguirlo. Ahora bien, la forma de llegar a un resultado tiene una enorme influencia en el funcionamiento del sistema, tanto en la calidad del resultado como en el tiempo necesario para obtenerlo. Además y a fin de conseguir una comercialización real de estos robots agrícolas, es necesario que sean sistemas escalables y que puedan construirse en serie. Por ello la organización interna ha de ser clara, accesible y fácil de modificar. De ahí que un aumento de la autonomía de los robots agrícolas conlleve una organización de los procesos internos del robot apropiada al vehículo, tarea y entorno de trabajo. La investigación en robótica móvil lleva ocupándose desde los años 70 de la organización del control del robot para conseguir comportamiento autónomo. En robótica el modelo de organización de los procesos internos se define como Arquitectura de Control.

En este sentido el campo de la agricultura, y en concreto el de la agricultura de precisión, puede verse muy favorecido con la aplicación de las investigaciones desarrolladas en los últimos años en el campo de la robótica [Baerveldt, 2002], adaptándolas a los requisitos particulares de los vehículos agrícolas, entornos de exteriores y labores del campo. En concreto, una idea central de esta tesis es que las investigaciones en el campo de las arquitecturas de control para

robots móviles pueden ser de gran utilidad al implantarse en vehículos agrícolas, siempre que se adecúen convenientemente a un entorno dinámico con mecanismos que permitan afrontar situaciones imprevistas, combinar objetivos diferentes y a veces contrapuestos, para avanzar hacia sistemas con un grado cada vez mayor de autonomía.

En el capítulo 3 se revisan conceptos básicos relativos a robots y autonomía así como los diferentes aspectos de los paradigmas existentes en arquitecturas de control, con el objeto de extraer de los enfoques analizados las pautas que deben guiar el desarrollo de una arquitectura de control para robots agrícolas, uno de los objetivos centrales de esta tesis.

Capítulo 3

Arquitecturas de control para la generación de comportamiento autónomo

Una de las características de los animales más evolucionados es su capacidad para desarrollar y utilizar herramientas. El ser humano hace uso de ellas desde sus comienzos, y éstas han ido evolucionando con el soporte de los avances tecnológicos, hasta convertirse en sistemas complejos y sofisticados. Incluso careciendo de la tecnología necesaria para construirlos, el hombre siempre ha imaginado artefactos mecánicos que pudiesen realizar tareas de modo autónomo, es decir, sin necesidad de una intervención humana. Este tipo de máquinas, con muy distinto grado de complejidad, es lo que denominamos **robots**.

Las herramientas se han utilizado en agricultura desde que el ser humano empezó a cultivar la tierra en los albores de la civilización. En los últimos años las labores agrícolas se han beneficiado de la automatización de la maquinaria, como se ha mostrado en el capítulo 2. La

integración de robots móviles en agricultura puede contribuir a mejorar la salud de los operariso y el impacto medioambiental, con un incremento en los beneficios y la calidad de los productos. Para que estos robots lleven a cabo las tareas asignadas con la precisión y seguridad requeridas se requiere, no sólo la integración de múltiples sensores y procesadores, sino la organización de los procesos de control que llevan a cabo en interacción con el medio. En definitiva, nos encontramos con unas máquinas muy complejas en las que hay que combinar las habilidades de percepción, razonamiento y control en tiempo real para la consecución con éxito de una misión. Esta combinación exige una reflexión sobre el modelo de organización de dispositivos sensoriales, actuadores, procesadores y procesos, de acuerdo con las expectativas de operación en un dominio concreto y de fácil crecimiento en el futuro para abordar diferentes dominios de trabajo y objetivos. Este paradigma de la robótica cognitiva se conoce bajo la denominación de **arquitectura de control**.

En este capítulo se presentan los conceptos relacionados con los robots móviles agrícolas, autonomía y arquitectura de control, secciones 3.1 y 3.2, y se revisan las diferentes aproximaciones en arquitecturas de control para la generación de comportamiento autónomo. En primer lugar se analizan tres arquitecturas deliberativas clásicas, sección 3.3: la arquitectura propuesta para el robot Shakey, basada en STRIPS [Fikes y Nilsson, 1971], por ser este el primer robot móvil diseñado con técnicas de Inteligencia Artificial; la arquitectura NHCA [Meystel, 1987] y RCS [Albus, 1991], por constituir dos evoluciones del paradigma deliberativo y proponer una organización jerárquica de la descomposición funcional percibir-planificar-actuar en diferentes niveles de abstracción. Ambas cuentan con el respaldo de una implementación en robots reales. Seguidamente se presentan las propuestas mas representativas de arquitecturas reactivas, sección 3.4: Brooks [Brooks, 1986] y Patti Maes [Maes, 1990b]. Se finaliza con le exposición de algunas arquitecturas híbridas que integran características deliberativas y reactivas, sección 3.5. Las arquitecturas híbridas se pueden dividir en dos grupos: 1) basadas en comportamientos, describiéndose la arquitectura de Mataric [Mataric, 1991], la arquitectura AuRA [Arkin y Balch, 1997] y la arquitectura DAMN [Rosenblatt, 1995]; y 2) las arquitecturas denominadas de niveles, donde se ha incluido la arquitectura RAP

y 3T [Gat, 1992, Firby, 1987], TCA [Simmons, 1994], Saphira [Konolige et al., 1997] y la arquitectura AMARA [García-Alegre y Guinea, 1992]. El capítulo termina con una reflexión, sección 3.6, sobre las propiedades más relevantes de las distintas arquitecturas, con vistas a su aplicación en navegación autónoma de vehículos en agricultura.

3.1 Robots, autonomía y arquitectura de control

Según la RIA (*Robotics Industry Association*) un robot es un manipulador programable y multifuncional, diseñado para mover materiales, piezas, herramientas o dispositivos especializados mediante movimientos programados y variables con objeto de realizar diversas tareas. Esta definición, aunque se ajusta perfectamente a los robots industriales no lo hace tanto a los denominados robots móviles autónomos. Una definición mucho más adecuada a este caso es la propuesta por Arkin [Arkin, 1998]: **un robot inteligente es una máquina capaz de extraer información del entorno y usar el conocimiento que posee sobre el mundo para moverse de forma segura y con un propósito**. Esta definición incluye las capacidades requeridas para un robot autónomo. Por un lado la capacidad de **percibir**, para lo cual ha de disponer de sensores que le permitan detectar su estado y el del entorno y extraer información relevante de los mismos; además esta definición añade que la información debe transformarse en conocimiento útil para la acción, lo que significa que la **percepción está orientada a la actuación**. Finalmente, el robot se mueve con un **objetivo** o propósito y debe hacerlo de forma **segura**.

Por todo ello esta definición de las propiedades de un robot entronca directamente con el concepto de “inteligencia situada”, según la cual un robot inteligente es aquel que muestra un comportamiento inteligente, ya que lo único que se puede medir a fin de determinar su grado de inteligencia, es el resultado de sus acciones sobre el mundo [MacFarland y Bosser, 1993]. Este modo de entender la inteligencia implica que: 1) al ser las acciones las que determinan el grado de inteligencia de un robot, es necesario disponer de un sistema físico, con una dotación de sensores y actuadores para interactuar con el entorno y materializar el comportamiento, 2)

los patrones de comportamiento apropiado emergen de la interacción del agente físico con el entorno [Beer, 1990] y no sólo como consecuencia de las características del mismo.

Si entendemos *autonomía*, según el DRAE, como **condición de quien, para ciertas cosas, no depende de nadie**, es evidente que el concepto expuesto de comportamiento inteligente está estrechamente relacionado con el de comportamiento autónomo. De esta forma, podemos considerar que un robot es autónomo cuando es capaz de alcanzar un tipo de objetivos sin la intervención del operador humano, es decir, sin depender de nadie; de ahí que un robot autónomo debe disponer de capacidad para interaccionar con un entorno, que puede ser dinámico, para alcanzar por sí mismo los objetivos propuestos.

Siguiendo la definición de Arkin, los robots móviles agrícolas son vehículos adaptados a un medio rural, como son los tractores, y cuyo objetivo es realizar una tarea agrícola o bien facilitar su realización, mediante una navegación con un cierto grado de autonomía, -con o sin un conductor-, que centra su atención en el control del laboreo. Una definición ampliada, podría ser: **un robot móvil agrícola es un vehículo capaz de extraer información del entorno, combinarla con el conocimiento inyectado inicialmente sobre sus posibilidades y limitaciones y las del entorno de trabajo, para navegar de forma segura y autónoma por el campo con el objetivo de realizar o de facilitar la realización de alguna tarea agrícola**.

El objetivo principal de diseño de un robot es conseguir la resolución adecuada de un un determinado tipo de tareas (de navegación, recolección, escardado) con la menor intervención humana. El robot dispone de unos **objetivos**, en este caso relacionados con el entorno agrícola, y su consecución va a depender de las características propias del robot así como de los resultados de su interacción con el entorno, en algunos casos dinámico.

El diseño y la realización del robot se complican a medida que el entorno de trabajo deja de estar totalmente estructurado y las tareas a realizar son heterogéneas. De ahí que cada vez sea más necesaria la incorporación de múltiples dispositivos sensoriales, de proceso, de control y comunicación, que permitan la percepción, razonamiento y toma de decisión apropiadas ante la definición de un objetivo. Los sensores posibilitan la extracción de información del entorno a

fin de propiciar una interacción adecuada y la toma de decisiones para resolver imprevistos. Los actuadores permiten al robot influir sobre su entorno de modo activo, modificando su relación con el mundo, ya sea manipulando objetos o mediante cambios de posición. Los dispositivos de proceso permiten establecer relaciones entre los modelos y las variables instantáneas medidas para razonar sobre la conveniencia de realizar una determinada acción, que puede dar lugar tanto a un cambio físico observable como a un cambio de la representación del sistema/entorno. Finalmente, el sistema de comunicación establece el canales de comunicación con otros procesadores entre los que se incluye al operador humano.

Estos cuatro subsistemas conforman el *hardware* y *software* del robot y son los que van a permitir la sustitución del conductor experto en la navegación y control para una tarea de laboreo agrícola. Con este fin el sistema adquiere información relativa a su estado y al estado del entorno, que unido a las representaciones iniciales del mundo que posee el robot, permiten decidir las acciones instantáneas a realizar y cuya secuencia temporal produce un comportamiento observable del robot. Un robot puede, por lo tanto, ser considerado como un sistema complejo dotado de distinto tipo de habilidades: perceptivas, motoras, de procesamiento y de comunicación con el exterior. En el campo de la Robótica e Inteligencia Artificial, la organización de estos conocimientos y habilidades en la consecución de un objetivo, se conoce como **arquitectura de control** del robot. Ahora bien, esta definición aparece matizada, dependiendo de los distintos grupos de investigación; [Kortenkamp et al., 1998b] la define como la organización del software de control del robot.

Teniendo en cuenta los puntos claves a los que debe responder una arquitectura de control, -actuación, percepción, proceso y comunicación-, definimos como arquitectura de control de un robot: **la estructura de organización de conocimientos, representaciones, procesos y comunicaciones, que conducen a alcanzar un conjunto de objetivos en interacción con un entorno**. Esta definición de arquitectura divide el problema en dos partes: 1) extraer la información relevante del sistema/entorno que será función del objetivo y 2) razonar sobre la información disponible, incierta e incompleta en sistemas reales, para tomar las decisiones adecuadas a la consecución de los objetivos propuestos.

La importancia de las investigaciones en arquitecturas de control, en el campo de la Robótica, queda patente al comprobar que la mayoría de los autores [Dudek y Jenkin, 2000, Murphy, 2000] dividen la corta historia de la Robótica en periodos que coinciden con cambios en los paradigmas de las arquitecturas de control y toma de decisión para robots. Los inicios históricos de la Robótica se remontan a principios de los años 40 [Dudek y Jenkin, 2000], cuando Wiener desarrolla un dispositivo de control de armas antiaéreas, integrando en la misma plataforma información sensorial, procesamiento de señales y decisión sobre la actuación. El desarrollo de las teorías de control discurre en paralelo con la Robótica, pues el control realimentado refleja la idea de que ciertas medidas del estado interno o externo de la máquina influyen en su comportamiento. Sin embargo, a finales de los años 70, con el desarrollo de la Inteligencia Artificial, se produce el primer gran salto en la Robótica, al construirse un robot autónomo utilizando técnicas de Inteligencia Artificial, que le permiten razonar sobre un modelo del entorno. El robot Shakey, desarrollado en el *Stanford Research Institute,* es el primer robot móvil que posee estas características.

Sin embargo, estos robots diseñados a partir de los paradigmas de la Inteligencia Artificial, se dedicaban esencialmente a la tarea de construcción y mantenimiento de modelos ideales, con un alto coste computacional y sin la inclusión de ningún grado de incertidumbre, siendo incapaces de responder ante un mundo real como lo haría un operador humano. En este contexto Brooks, [Brooks, 1986] propone una arquitectura reactiva, postulando que no es necesario ni el razonamiento simbólico, ni disponer de un modelo del mundo para poder actuar de forma reactiva y en tiempo real, es más, que tanto el modelo como el razonamiento simbólico merman la capacidad de reacción del robot ante imprevistos. Este planteamiento supuso una verdadera "revolución", al postular la interacción con el medio como el mejor modelo para la generación de comportamientos reactivos inteligentes. Desde entonces, las denominadas arquitecturas híbridas han integrado las características más valiosas tanto de los modelos deliberativos como de los reactivos en un intento de armonizar y complementar el diseño de las arquitecturas de control para robots móviles. Esta combinación tiene como fin la obtención de comportamiento autónomo en la realización de un conjunto de objetivos o misiones en mundos reales con

conocimiento incompleto e incertidumbre.

3.2 El problema de la arquitectura de control

Un robot móvil agrícola debe navegar en entornos de exterior dinámicos, por lo tanto tiene que decidir en cada ciclo de control la mejor acción como respuesta a su interacción con el entorno. Este punto es conocido en Robótica como el problema de la **selección de acción**, que viene condicionada por el tipo de entorno y la configuración de estructuras en el mismo. Por ello una acción, que puede ser adecuada ante una determinada situación no lo es en otras que pueden parecer muy semejantes. Para resolver este conflicto el robot debe extraer y elaborar la información relevante del entorno, esto es, **percibir** a través de los sensores disponibles. Estos datos del entorno constituyen la información instantánea adicional, que hay que integrar en los modelos de partida para decidir qué acción realizar en cada ciclo de control. Los datos pueden almacenarse de múltiples formas y con diversas resoluciones, en distintos momentos y con períodos variable de muestreo de la señal, y constituyen el marco o ventana espacio-temporal de **representación** del conocimiento. Habida cuenta de que una buena representación de un problema supone, al menos, la mitad de su resolución, su importancia es crucial.

En esta sección se van a describir en detalle los cuatro aspectos en que se ha dividido una arquitectura se control: percepción, representación, comunicación y selección de acción, como paradigmas básicos sobre los que van a discurrir todos los procesos en el robot para la consecución de un conjunto de objetivos.

3.2.1 Percepción

Debido a la complejidad de los procesos de percepción artificial en un mundo real y dinámico, los estudios realizados hasta la fecha han ido abordando aspectos locales o imponiendo fuertes restricciones a fin de convertirlo en un problema abordable. A medida que la velocidad de los procesadores ha ido en aumento y los costes de muchos de los

dispositivos sensoriales se han visto reducidos, son cada vez más numerosos los grupos de investigadores que pueden hacer frente al reto de elaborar los datos sensoriales crudos. En esta misma dirección, crece la investigación en mecanismos de fusión de percepciones sensoriales encaminadas a la confirmación de propiedades de estructuras naturales por muy diversos métodos, sensores de ultrasonidos y cámaras CCD en entornos de interior y láser y sistemas de visión en color en exteriores [Rodríguez et al., 1994, Franklin y Firby, 1997, Murrieta-Cid et al., 2002].

Además hay que tener en cuenta que las decisiones de actuación de un robot móvil en un entorno dinámico dependen de su capacidad de percepción del entorno, y que los dispositivos sensoriales que la soportan proporcionan información limitada y con un grado de incertidumbre. Cualquier sensor lleva inherente un error que es inversamente proporcional a su resolución, por lo tanto la información que proporciona es incierta; y limitada pues cada sensor sólo es capaz de percibir unas propiedades de los objetos presentes en el entorno y en una región acotada del espacio, de acuerdo con su máximo alcance.

No todas las propiedades que se pueden extraer de los datos sensoriales poseen la misma relevancia para un robot ante un objetivo concreto, es más, la mayor parte de la información es irrelevante para la tarea. La información relevante que actúa como desencadenante de una reacción funcional en el robot se denomina **estímulo**. Los estímulos independientemente de su tipo y complejidad, constituyen el núcleo central de la tarea de reconocimiento. Así, aunque los sensores estén midiendo en continuo el estado del entorno, la percepción debe ir dirigida a la búsqueda de estos estímulos y en este sentido actuar como un filtro. Este mecanismo se define **atención** indicando que el procesamiento perceptivo se centra sólo en la detección de los estímulos relevantes para una misión, con la consiguiente reducción del coste computacional. Los mecanismos de atención deben ser dinámicos, y dependientes de la tarea. En el caso de un tractor en tareas de fumigación de campos de olivos el estímulo “olivo”, es crucial cuando el robot está ejecutando la tarea de fumigación, pero se convierte en un obstáculo más a evitar ante una tarea de navegación global desde el hangar hasta una posición cualquiera en el campo.

En muchas ocasiones se requieren estímulos que no están contenidos directamente en las

lecturas sensoriales instantáneas, por lo que es necesario un proceso complejo de elaboración y fusión de datos crudos sobre una memoria espacio temporal de alcance medio o largo [Prassler et al., 2000]. Hay que tener en cuenta que un estímulo dinámico, únicamente puede ser elaborado si se dispone de una memoria para poder percibir variaciones en el tiempo. Algunas características, como color y distancia a un árbol sólo pueden obtenerse si se fusiona la información procedente de distintos sensores, como la información de profundidad de un objeto proporcionada por un láser de barrido con la información de color proveniente de una cámara de vídeo. En esta línea, estudios recientes en psicología de la percepción visual [Monserrat, 1998] corroboran este punto, pues a nivel inconsciente preatentivo ya existe un procesamiento que parece ir encaminado a resaltar el orden y las estructuras presentes en la imagen. Posteriormente, y ya sólo en el foco de atención se realizan los procesos dirigidos a organizar la imagen, siguiendo las leyes de Gestalt [Monserrat, 1998].

En sintonía con la afirmación de Meystel [Meystel, 1987], relativa a que una vez resuelto el problema de la percepción, la complejidad reside en la selección de la acción, pensamos que gran parte de la complejidad que entraña la generación de comportamiento inteligente reside en la capacidad de estructurar apropiadamente la información disponible, de extraer los estímulos relevantes a partir de los datos sensoriales crudos y de simultanear esa tarea con un razonamiento encaminado a la toma de decisiones sobre la acción en los actuadores o sobre el modelo del mundo.

3.2.2 Representación

Una vez que los procesos perceptivos extraen de la información sensorial los estímulos que se consideran relevantes para la tarea en curso, hay que decidir cómo organizarlos. Este problema constituye lo que en esta tesis se define como el problema de la **representación**, que abarca desde la estructura en la que se expresan los estímulos, la forma en que se almacenan, cuándo se buscan, cuánto tiempo se mantienen en la memoria, hasta el análisis de los procesos que acceden a la misma.

En una arquitectura de control hay que evaluar su habilidad para representar todas las clases de conocimiento que se necesitan para resolver la tarea seleccionada en el entorno concreto [Tecuci, 1998], es decir, lo adecuada que es la representación (o representaciones) al problema planteado. La representación va a constituir el conocimiento compartido entre los diferentes procesos de acción de la arquitectura. Y la manera de intercambiar este conocimiento en sus diferentes formas de representación, bien sea mediante paso de mensajes entre agentes o bien sea mediante una memoria compartida de tipo pizarra [Ribeiro, 1991], va a variar de unas arquitecturas a otras.

Habitualmente la representación de la información y en especial de la información perceptiva, ha sido obviada en la mayoría de las arquitecturas y sin embargo es un problema no trivial. Las opciones van desde diferentes modelos completos del entorno con distintas resoluciones, modelo multiresolucional, utilizados por los diferentes procesos de generación de comportamiento [Albus, 1991, Meystel, 2003], a la opción de una representación subsimbólica, propugnando la unión directa entre medidas sensoriales instantáneas con acciones sobre los actuadores, como en el caso de la arquitectura de subsunción de Brooks [Brooks, 1986].

3.2.3 Selección de acción

El mecanismo de selección de acción [Maes, 1990a], aborda el problema de la selección de la acción más adecuada en cada instante, a partir de la verificación de los siguientes requisitos: 1) los objetivos y acciones instantáneas relevantes para el agente, 2) las acciones que contribuyen al objetivo en curso y 3) capacidad de planificación, para finalmente generar un comportamiento observable robusto y reactivo. Esta decisión puede incidir o no sobre los dispositivos físicos o actuadores del sistema. Existen diferentes opciones para modelar la toma de decisión sobre la acción más adecuada; desde la teoría de control clásica, a las aproximaciones de control inteligente, pasando por las tablas de situación-acción y los planificadores.

La teoría de control se enfrenta al problema del cálculo de la acción de control que debe aplicarse a una planta o sistema con el objetivo de corregir o limitar la desviación de

la medida de salida con respecto al valor deseado [Ogata, 1990]. Tanto la idea de bucle de control- conseguir un comportamiento como iteración infinita de **ciclos de control**- como la de realimentación proceden de la teoría clásica de control y ambas están consensuadas en robótica móvil.

El método tradicional de diseño de un sistema de control comienza con el desarrollo de un modelo de la planta, de los sensores y de los actuadores. En un sistema de control clásico el modelo es analítico, en los sistemas de control inteligente basados en técnicas de razonamiento aproximado, el modelo se expresa mediante un conjunto de reglas definidas mediante variables numéricas o lingüísticas. En el caso de un robot móvil, la planta la constituye el sistema ($robot + entorno$) y este conjunto es el que resulta muy difícil de modelar en todos sus aspectos. En un robot de exteriores la dificultad aumenta, pues existen variables que cambian a lo largo del día: el tipo de terreno, la luminosidad, las condiciones medioambientales (viento, polvo, humedad, lluvia). Un sistema de control se diseña para unos objetivos concretos bien definidos, como pueden ser el tiempo de respuesta máximo o la mayor sobreelongación permitida. Sin embargo para un robot móvil estos objetivos deben modificarse a lo largo de los periodos de operación, por lo que es necesario arbitrar unos mecanismos que regulen las transiciones entre los distintos modos de operación.

Existen pocos ejemplos de robots guiados únicamente por un modelo clásico de control para alcanzar objetivos de alto nivel. Esto se debe en gran parte al salto en complejidad que existe entre los lazos de control de bajo nivel (motores eléctricos, por ejemplo), al control de los objetivos de alto nivel que interactúan de forma compleja [Kortenkamp et al., 1998b]. Este cambio en el nivel de complejidad se resume en los siguientes aspectos: 1) dificultad para modelar el robot y el entorno dinámico en el que se desenvuelve y 2) múltiples objetivos variables en el tiempo.

Los objetivos de los robots móviles pueden ser lo suficientemente complejos como para que resulte conveniente dividirlos en subobjetivos más sencillos; por ejemplo, si un robot fumigador detecta un nivel bajo del pesticida, debe regresar al hangar para rellenar los tanques en vez de continuar con su objetivo de fumigación de todo el campo. Esta división de

una tarea compleja o misión en una secuencia optimizada de subtareas de menor alcance, constituye lo que denominamos **planificación** [Simmons, 1994]. Los planificadores se han empleado tradicionalmente en la generación de comportamiento. En el caso de la navegación el planificador elabora una secuencia de movimiento basada en el modelo del entorno que posee. La gran dificultad en el empleo de planificadores como mecanismo de selección de acción reside en que en un entorno dinámico el modelo del entorno con el que trabaja el planificador puede no ser válido en el momento de ejecutar la acción. Además de mostrar un comportamiento adecuado, el robot ha de disponer de capacidad de reacción ante objetos imprevistos, esto es, debe ser capaz de reaccionar a tiempo [Simmons, 1994]. Es imprescindible una reacción rápida ante obstáculos imprevistos en el camino, como pueden ser cajas, operarios o tractores que se cruzan o bien ante un cambio abrupto en la pendiente del terreno, a fin de garantizar siempre la seguridad y supervivencia del sistema y de los elementos presentes en el entorno.

En las arquitecturas que encapsulan cada una de las diferentes habilidades del robot en comportamientos (ya sea con este nombre o bajo la denominación de esquema o agente), el mecanismo de selección de acción puede implementarse fusionando las preferencias de los comportamientos. Esta fusión puede realizarse, en el caso de la navegación, expresando la preferencia de cada comportamiento como un campo de potencial [Arkin, 1987]. También puede articularse mediante árbitros, que calculan la acción deseada en base a la preferencia de cada comportamiento pesada con un factor proporcional al nivel de relevancia de dicho comportamiento respecto de los objetivos y el entorno [Rosenblatt, 1997].

Otra opción es que la acción final proceda únicamente de uno de los comportamientos, aquel más relevante en cuanto a los objetivos y condiciones del entorno, como en [Maes, 1989, García-Alegre y Recio, 1998, Brooks, 1986].

3.2.4 Comunicación

Por otro lado, un aspecto cada vez más relevante en la investigación en robótica se refiere a la especificación de objetivos, en la dirección de favorecer una comunicación amigable y

sencilla entre el operario humano -con su lenguaje y habilidades concretas- y el robot, dotado usualmente de un lenguaje de comandos [Arkin et al., 2003, Fong y Thorpe, 2001]. El interés en este tema es creciente debido fundamentalmente a la conveniencia de que usuarios no expertos en robótica, informática y control, puedan manejar los robots en tareas de campo, en este caso en un entorno agrícola [Ribeiro et al., 2003].

Todos los robot móviles autónomos disponen de una interfaz de comunicación con el humano. En algunos casos está comunicación se favorece con organizaciones de tipo cliente-servidor que permiten el empleo de internet para la supervisión de los robots [Trevelyan, 1997]. El uso del *World Wide Web* como interfaz de comunicación es muy atractivo ya que no requiere entrenamiento de los usuarios, sin embargo es susceptible de problemas causados por el ancho de banda variable [Fong y Thorpe, 2001].

3.2.5 Generación de comportamiento

Conseguir que el robot muestre en cada instante un comportamiento apropiado al objetivo perseguido resulta extremadamente complejo debido fundamentalmente a: 1) la admisión de entornos dinámicos, 2) la complejidad en la detección de estructuras con incertidumbre en las medidas y en la interpretación de las relaciones espacio-temporales que existen entre ellas, y 3) las limitaciones del sistema, tanto en lo que se refiere a sus recursos físicos como a las capacidades de percepción, deliberación y reacción.

- **Entornos dinámicos.** Los entornos de exterior no controlados son dinámicos y por lo tanto están sujetos a la aparición de elementos imprevistos. De ahí que sea imposible generar comportamiento a partir de un modelo o tabla que relacione las condiciones sensoriales con una salida de control, pues es imposible definir las infinitas combinaciones de los datos proporcionados por múltiples sensores. Por ello, una arquitectura de control debe hacer frente al abanico de imprevistos tentativos de aparición y desconocidos que puedan presentarse en el escenario de trabajo. Su respuesta debe ser apropiada en el sentido de que sea lo suficientemente

rápida para evitar por ejemplo una colisión o vuelco del vehículo, sin perder de vista los objetivos de la misión del robot. Así, ante un obstáculo imprevisto en su trayectoria, el robot debe evitarlo con una desviación mínima respecto de la dirección que llevaba hacia el objetivo.

- **Información compleja, incompleta, incierta y dispersa.** Al tratarse de sistemas y entornos complejos, una parte de la información de interés no se percibe de forma directa. Por un lado, existen estructuras que sólo se pueden percibir con fiabilidad mediante la integración de información procedente de varios sensores, bien sea por carecer de suficiente información con sólo un sensor, por ejemplo la distancia a una hilera de árboles puede determinarse empleando una cámara de visión y un láser; o bien sea porque la información de la que se dispone es altamente incierta. De ahí que en muchos casos sea conveniente elaborar los estímulos con información redundante, procedente de dos o más dispositivos sensoriales complementarios. Por ejemplo, para estimar la posición de un robot en exteriores es conveniente combinar la información de diferentes tipos de sensores: GPS, odometría, brújulas y giróscopos. Por otro lado en ciertas situaciones es necesario mantener una memoria temporal de la señal, como es el caso de distinguir entre objetos móviles y estáticos. Finalmente, existen situaciones en las que el flujo de información sensorial puede desbordar el sistema de proceso, como sucede con una cámara de visión de resolución alta. Sin embargo, en la mayoría de los casos, sólo una parte reducida de esta información es objeto de interés, por lo que habrá que extraer convenientemente la estructura del fondo en un tiempo limitado, si se persigue una respuesta del sistema en tiempo real. El tratamiento de la complejidad e incertidumbre inherente a un sistema móvil real en un entorno de exteriores debe organizarse desde una arquitectura de control para la consecución de los objetivos propuestos. Además la arquitectura tiene que abordar el proceso de extracción de los estímulos necesarios para poder calcular la acción más apropiada.

- **Recursos limitados.** Cualquier sistema real posee unas limitaciones en todas sus habilidades, desde la percepción hasta la representación y la actuación. Estas restricciones están reflejadas en el modelo de control encargado de asignar recursos y prioridades a las diferentes

tareas. La asignación de prioridades puede no estar explícita pero siempre ha de existir para que el robot reaccione a tiempo. Puesto que los recursos computacionales son limitados, conviene organizar la información de forma que sólo se opere con aquella que es estrictamente imprescindible en cada momento. Por otra parte, es posible modular la acción a realizar de acuerdo con las restricciones motoras del sistema físico. En especial, con aquellos vehículos que, como los tractores, poseen una configuración no holonómica que reduce su capacidad de maniobra en trayectorias muy cerradas o con cambios muy pronunciados de pendiente. Un robot no-holonómico posee un ángulo de giro máximo y una velocidad de trabajo que hacen que desde una posición inicial no sea siempre posible alcanzar cualquier posición objetivo.

3.2.6 Criterios de evaluación de una arquitectura de control

Las principales características del comportamiento generado por una arquitectura se resumen en los siguientes párrafos:

1. **Orientación a los objetivos**. La arquitectura debe considerar en todas sus decisiones el mantenimiento de los objetivos finales del sistema, de forma que la interacción con el entorno no desvíe considerablemente su comportamiento local de la misión global asignada. Son los objetivos finales los que guían en mayor grado la automatización, la selección de los sensores y el diseño de la arquitectura de control para un robot.

2. **Orientación al entorno y capacidad de reacción**. La capacidad de reacción permite al robot responder a tiempo. Ser reactivo implica por un lado, detectar a tiempo el estímulo ante el cual es necesario reaccionar, y por otro, capacidad de ejecución inmediata de la acción más adecuada. La capacidad de reacción garantiza de algún modo, la seguridad o supervivencia del sistema y es una característica imprescindible para los robots agrícolas autónomos, capítulo 2. También permite al robot “aprovecharse” de las circunstancias de una situación concreta [Tyrrel, 1993] ocurrida durante el recorrido del robot, por ejemplo detectar un nuevo camino desbloqueado más corto.

3. **Robustez**. Cualquier tipo de arquitectura tiene que ser robusta frente a fallos de las partes que componen el sistema, los imprevistos que puedan surgir en el entorno, incertidumbre e información incompleta. En el caso de los robots móviles agrícolas la robustez adquiere una mayor importancia, al tratarse de un entorno sin marcas artificiales, con cambios bruscos en las condiciones medioambientales, donde además, el fallo puede producirse cuando el robot está alejado del operador humano. Robustez en estos casos implica disponer siempre de una respuesta que evite el deterioro o destrucción del sistema y del entorno.

4. **Facilidad de escalado y organización modular**. Una característica importante del modelo es la facilidad que ofrece para añadir nuevas habilidades, sin tener que perturbar considerablemente el diseño, lo que implica un diseño modular y una posibilidad de crecimiento gradual de competencias. Esta propiedad permite añadir con facilidad nuevos sensores y en consecuencia procesos perceptivos y motores para realizar un conjunto más amplio de tareas con un grado superior de autonomía.

5. **Versatilidad**. Es la capacidad de adaptarse con facilidad y rapidez a entornos y objetivos variables. Es muy deseable que la arquitectura de un robot agrícola pueda aplicarse a otros entornos con características similares, sin necesidad de realizar una gran adaptación a los nuevos entornos.

A continuación se analizan diferentes arquitecturas en función de su capacidad para resolver las funciones de percepción, representación y selección de acción. Cualquier arquitectura de control tiene que organizar las habilidades de un robot a fin de dar una respuesta coherente y robusta a la consecución de un conjunto de objetivos ante un tipo de entornos. Sin embargo coincidimos con muchos autores, [Murphy, 2000, Arkin, 1995] en que no existe la arquitectura de control ideal, sino diferentes diseños o aproximaciones a la solución de un determinado tipo de problemas: de navegación para recorrido completo de campo laboreo, de percepción de mala

hierba para fumigación selectiva, de detección de fruto para corte, de detección y localización precisa de hortalizas para abonado y riego selectivo, etc.

Una arquitectura de control posee unas líneas generales de diseño y organización, que posteriormente se inician y modulan para cada sistema específico robot-entorno-objetivos. Esto conlleva una serie de restricciones que hay que considerar antes de abordar una arquitectura para la generación de comportamiento autónomo. En el caso particular abordado en esta tesis se trata de un robot móvil para navegación autónoma en un entorno agrícola dinámico. Por ello, en las siguientes secciones se presentan los paradigmas que agrupan las propuestas de arquitecturas de control para robots móviles.

3.3 Arquitecturas deliberativas

Las arquitecturas tradicionalmente consideradas como **deliberativas** son las herederas de las investigaciones en Inteligencia Artificial (AI) clásica y fueron las primeras en surgir en el campo de la robótica [Brooks, 1991]. Enfocan el problema de la generación de comportamiento en un robot como una tarea secuencial que exige una representación explícita del mundo para tomar decisiones basándose en esta representación y siguiendo las directrices marcadas por un planificador. Plantean el problema como una secuencia de estrategias de: percepción del entorno, construcción de un modelo del mundo, planificación basada en ese modelo y actuación. Los orígenes epistemológicos de este enfoque están en el fondo en la filosofía cartesiana que considera el alma como el ente que decide el comportamiento [MacFarland y Bosser, 1993]. Implícitamente asumen que la solución a un problema complejo consiste en la subdivisión de éste en tareas más sencillas y fáciles de abordar.

En IA Simbólica el problema de “selección de acción” se entiende como un proceso de búsqueda de la mejor acción partiendo de unas condiciones iniciales, en una determinada situación y con un objetivo concreto. Para ello se dispone de un abanico de acciones posibles y mecanismos para anticipar el resultado de cada acción. Así, el planificador Strips utilizado con el robot Shakey busca la secuencia de operadores-acciones más adecuados para pasar de

un estado de mundo inicial al estado de mundo final. Para ello dispone de una tabla donde cada operador tiene asociadas las precondiciones necesarias para que pueda ejecutarse, así como las situaciones que serán ciertas cuando este operador se ejecute y las que serán falsas. En estas arquitecturas el control es monolítico y centralizado. El cálculo de la mejor acción la realiza el planificador, que tiene en cuenta toda la información necesaria en el modelo del mundo elaborado un módulo perceptivo.

La percepción del entorno se basa en considerar fiables las informaciones proporcionadas por los sensores abordando el problema de la percepción de una forma en general, como la observación del entorno dirigida a construir un modelo completo del mundo. La representación es por lo tanto explícita y centralizada en dicho modelo, que es objetivo, pues en él están representados los objetos con propiedades intrínsecas, independientemente del observador y de sus intereses. Inicialmente los modelos del entorno los construía el diseñador y se incorporaban manualmente al robot. Posteriormente se planteó la conveniencia de que el propio robot fuera capaz de construir los modelos del mundo para incrementar su autonomía, sin llegar a una experimentación al efecto. Estos modelos requieren un entorno estable y conocimiento completo del mismo para tomar decisiones, que se asume que nunca pueden fallar.

La descomposición funcional teórica del flujo de información desde los sensores hasta los actuadores describe el funcionamiento de las arquitecturas deliberativas. Hay una primera etapa de *percepción* en la cual la información procedente de los sensores se funde en un único modelo completo del mundo. Sobre ese modelo se calcula la mejor acción teniendo en cuenta todos los aspectos del problema. En el problema típico de navegación de robots, el modelo del mundo suele ser un mapa del entorno y el "*razonamiento*" consiste en planificar una trayectoria desde la ubicación actual del robot hasta el punto destino, sorteando todos los obstáculos contenidos en el mapa. Este camino se expresa como una secuencia de posiciones tentativas intermedias que conducen al destino. Finalmente, una etapa de *ejecución* se encarga de llevar a cabo el plan calculado, accediendo directamente sobre los actuadores del robot para que siga la trayectoria que conecta todas las posiciones. Un ejemplo de navegación de robots desde este modelo es la propuesta de [Crowley, 1985] para navegaciónde un robot móvil inteligente IMR (*Intelligent*

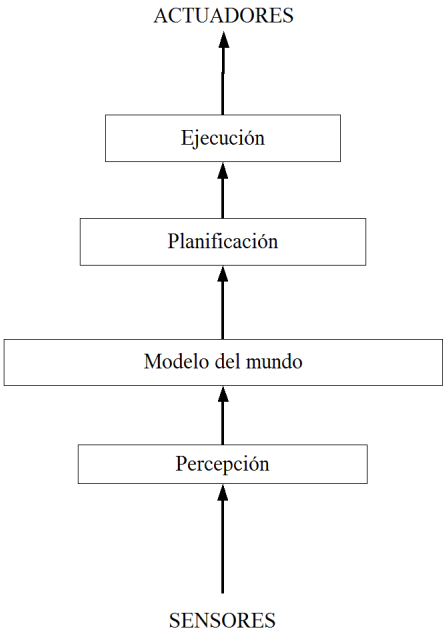


Figura 3.1 : Arquitecturas deliberativas: modelo de descomposición funcional

Mobile Robot). La navegación se concibe como un proceso que comienza con un modelo del entorno al que se añade la información precedente de los sensores de ultrasonidos del robot. En base a este mapa se elabora un plan de navegación general expresado como una secuencia de caminos. El robot replanifica cada segmento del camino para detectar obstáculos imprevistos y finalmente ejecuta el plan.

El principal escollo del paradigma deliberativo estriba en considerar que la percepción es completa y objetiva. El buen funcionamiento del sistema depende de que el modelo del mundo sea correcto, actualizado y coherente, ya que la actuación se calcula razonando sobre dicho modelo, pero mantener un modelo del mundo con estas propiedades no es un problema trivial. Por otro lado, el modo de razonamiento persigue que el resultado final sea correcto y no que se ejecute a tiempo, careciendo por tanto de la capacidad de reacción necesaria para enfrentarse a un mundo real dinámico. Sin embargo en muchas ocasiones puede resultar más conveniente actuar a tiempo, aunque la acción no sea óptima.

Otro aspecto a tener en cuenta es la escalabilidad, ya que si el sistema deliberativo no dispone de conocimiento completo no es capaz de ejecutar acción alguna. Este punto constituye un serio problema a la hora de desarrollar un sistema, pues no se puede probar un comportamiento hasta que todos los módulos que lo componen están totalmente desarrollados.

Las arquitecturas basadas en el paradigma deliberativo son adecuadas para robots teleoperados [Murphy, 2000] ya que la percepción y el modelo del mundo, cuello de botella de este enfoque, los aporta el operador humano, y el robot sólo toma decisiones de bajo nivel. En resumen, las arquitecturas deliberativas calculan la acción basándose en la descomposición de tareas realizada por un planificador. En este sentido priman las capacidades de planificación frente a las de reacción. El cambio de objetivos a lo largo de una tarea dependerá de la descomposición que haga el planificador y de su interpretación del modelo del mundo, de modo que la responsabilidad reside de nuevo en el planificador. En esta propuesta todos los aspectos disponibles del mundo interesan de la misma manera, y no existen mecanismos de atención discriminatorios, pues se trata de elaborar un modelo completo del entorno. Estas arquitecturas propugnan un modelo del mundo que pretende ser homomórfico con la realidad [Albus, 1991] y que puede en algunos casos distribuirse en niveles jerárquicos, [Albus, 1991, Meystel, 1987].

3.3.1 Arquitectura del robot Shakey

El robot Shakey, desarrollado en el *Stanford Research Institute* a principios de los setenta es considerado como el primer robot construido en base a técnicas procedentes del campo de la Inteligencia Artificial [Murphy, 2000]. Su comportamiento esta diseñado mediante una variante del método GPS (*General Problem Solver*), y recibe la denominación de STRIPS. Persigue la composición de operadores capaz de transformar un mundo inicial en uno final que satisfaga el objetivo [Fikes y Nilsson, 1971].

La arquitectura STRIPS intenta reducir las diferencias, aplicando operadores tanto al estado inicial como al final (análisis de *medios-fines* [Fikes y Nilsson, 1971]). Cada uno de los operadores requiere la satisfacción de un conjunto de precondiciones para ejecutarse. Además

dispone de una lista con los hechos que se cumplirán al ejecutarse el operador *add_list* y los que dejarán de ser ciertos *delete_list*. Por ejemplo: el operador $ir(a, b)$ requiere que el robot se sitúe en la posición *a*, $r(a)$ será su precondición. Su *add_list* sería $r(b)$ y la *delete_list* $r(a)$. STRIPS trabaja recursivamente, de forma que si no logra el objetivo en el primer paso, identifica la precondición que falla y busca un operador que la contenga en su *add_list* y así sucesivamente.

En el robot Shakey se asume que la percepción del mundo es completa, sin incertidumbre (modelo estático del mundo) y cerrada en el sentido de que todas las posibilidades se han contemplado. Además se asume que las acciones del robot Shakey siempre tienen las consecuencias que inicialmente se planificaron. Por todo ello Shakey constituye un ejemplo clave del paradigma deliberativo en su etapa inicial.

3.3.2 Arquitectura jerárquica y anidada NHCA

La arquitectura NHCA desarrollada por A. Meystel utiliza una estructura jerárquica de controladores como método para abordar la complejidad de un sistema [Meystel, 1987]. Se incluye en el paradigma deliberativo ya que, aunque no tiene la estructura monolítica característica de estos sistemas, tiene como base la repetición del esquema de descomposición funcional, percibir-planificar-actuar, figura 3.1, en cada uno de los niveles en los que se descompone el modelo de control. Su definición de autonomía lleva implícito el paradigma deliberativo: *autónomo presupone la existencia de etapas de planificación y toma de decisiones sin intervención humana* [Meystel, 1987].

La arquitectura NHCA propone un modelo jerárquico de niveles con múltiple resolución para la toma de decisiones [Meystel, 2003], *el alcance de la toma de decisiones y el grano del nivel aumenta cuando la precisión disminuye* [Meystel, 1987]. Esta misma visión se plantea en los controladores inteligentes de Saridis [Saridis, 1985] en lo que se denomina el principio de decremento de precisión con incremento de inteligencia (*decreasing precision with increasing intelligence*). Esta jerarquía se aplica tanto al ámbito de la toma de decisiones, correspondiente con la selección de acción, como al de la percepción y la organización del conocimiento. La

división en niveles jerárquicos obedece al diferente alcance que poseen las decisiones que hay que tomar. Cada nivel opera sobre una representación diferente del mundo caracterizada por una determinada resolución. Todos los niveles poseen una representación o mapa del mismo mundo pero a medida que se asciende en la jerarquía la resolución que posee el mapa disminuye, aumentando el grado de abstracción. En NHCA cada nivel tiene un ritmo de actualización específico, de manera que la frecuencia de actualización está unida al grado de abstracción, por ello la frecuencia es menor para los niveles superiores. Resumiendo, cada nivel va asociado a un valor de resolución, precisión y frecuencia de muestreo, que será más alto cuanto más próximo a los dispositivos físicos se encuentre el nivel. Cada nivel dispone de tres entradas, el objetivo, la representación del mundo y la información sensorial y una salida, el plan de acciones. Este plan constituye la entrada objetivo para el controlador de nivel adyacente inferior y así sucesivamente en una organización jerárquica de controladores enlazados.

Cada uno de los controladores así definidos implementa la siguiente secuencia, recepción de la información sensorial y del plan de actuación procedente del nivel adyacente superior, actualización de la representación del mundo a ese nivel, y generación de un plan de acciones para el nivel adyacente inferior. Así, en la jerarquía los planes fluyen desde los niveles más altos a los más bajos, y la información para la actualización del mapa de abajo hacia arriba.

En trabajos posteriores [Meystel, 2003] se incluye dentro de NHCA el aprendizaje por niveles, respetando siempre la estructura jerárquica.

3.3.3 Arquitectura RCS

Otra arquitectura relevante dentro de la escuela deliberativa es la arquitectura RCS *(Real-time Control System)* propuesta por Albus, [Albus, 1991].Uno de los mayores obstáculos para aplicar las técnicas de Inteligencia Artificial a los robots industriales es la ausencia de un estándar común de diseño y de este hecho surge la propuesta de la arquitectura RCS. Considera que son cuatro los elementos que componen la inteligencia: (1) los procesos sensoriales, encargados de elaborar la información sensorial, (2) el modelo del mundo o sistema de

representación elegido, (3) la generación de comportamiento guiada por los algoritmos que definen el modo de operación del sistema y (4) el juicio de valores que constituiría lo que denominamos, el sistema de selección de acción.

RCS aboga por un sistema jerárquico, donde cada nivel consta de diferentes nodos, uno por cada subsistema de actuación (manipulador, plataforma de movimiento, etc.) De manera similar a lo que propone NHCA, los diferentes nodos de un nivel comparten una representación común del conocimiento, un único modelo del mundo y un canal común de comunicación con los dos niveles adyacentes, el superior y el inferior. Los módulos de generación de comportamiento transmiten hacia los niveles superiores informes de estado, y hacia el nivel inferior acciones. Los resultados del procesamiento sensorial y la información de estado fluyen hacia arriba en la arquitectura jerárquica y las acciones generadas fluyen hacia abajo hasta alcanzar los dispositivos físicos de actuación. Cada una de las capas ejecuta una versión de la descomposición funcional de tareas: percibir-planificar-actuar, refinada puesto que RCS incluye en su planificación predicciones sobre el modelo del mundo mediante la simulación de acciones.

RCS es una arquitectura muy ambiciosa, tal vez demasiado detallada según las quejas de algunos investigadores que se vieron demasiado restringidos a la hora de implementarla [Murphy, 2000]. Sin embargo, se utiliza con éxito en diferentes plataformas robóticas (MAUV-robot submarino, TMAP y NASREM).

3.3.4 Ventajas e inconvenientes de las arquitecturas deliberativas

La robustez, la seguridad y la escalabilidad, que son características fundamentales para un robot móvil, constituyen, sin embargo, los puntos débiles de las arquitecturas deliberativas. Deliberar permite optimizar las acciones a tomar pero esto sólo se puede realizar cuando se dispone de información completa (modelo completo del entorno) y además sólo es eficaz si se puede ejecutar a tiempo, de ahí que cuando esto no sucede el modelo no es suficientemente robusto. En un robot agrícola no es posible disponer de un modelo completo del mundo, ya que se trata de un entorno no controlado donde pueden aparecer de modo imprevisto otros

vehículos, personas, etc, y pueden ocurrir cambios: una irregularidad nueva en el terreno, un árbol caído... Tampoco ofrece seguridad frente a imprevistos, puesto que estas arquitecturas priman la correcta elaboración de cada acción frente a la velocidad de cálculo, en detrimento de la capacidad de reacción del robot; y en ocasiones es fundamental reaccionar a tiempo frente a optimizar una respuesta. En el caso de un robot agrícola la capacidad de reacción es fundamental puesto que ante situaciones de peligro, como una pendiente muy pronunciada tiene que tomar la acción adecuada inmediatamente antes de volcar. Otro inconveniente de los sistemas deliberativos radica en la necesidad de completar todo su diseño, es decir todas las etapas del esquema funcional percibir-modelar-planificar-actuar, para ser operativo. Esto constituye un problema en el diseño de arquitecturas con un crecimiento gradual, como sucede en los robots móviles agrícolas, al ir aumentando el repertorio de objetivos, la dotación sensorial, y consecuentemente los algoritmos de coordinación de competencias y flujo de información del sistema, para dotarle de una mayor autonomía.

Por otro lado, la capacidad para planificar es fundamental cuando se dispone de un sistema complejo que pretende optimizar las tareas a realizar. Esto conlleva la descomposición de objetivos en subobjetivos más sencillos para abordar la complejidad algorítmica inherente a la consecución de las misiones asignadas. Las arquitecturas deliberativas sin embargo son versátiles puesto que la estrategia de descomposición de tareas es independiente del dominio o entorno sobre el que opera, lo cual es muy ventajoso en un entorno agrícola para el laboreo de múltiples parcelas con distinta configuración. Entre las desventajas de los modelos basados en el paradigma deliberativo se encuentra la de no ser apropiados para operar en entornos ricos en información. Más aún cuando se dispone de una dotación sensorial compleja, pues la ejecución del esquema percibir-modelar-planificar-actuar conlleva un elevado coste computacional, y la respuesta del sistema se ralentiza en extremo.

El principal escollo de las arquitecturas fieles al paradigma deliberativo se refiere, por lo tanto, a la percepción y la representación, al considerar la percepción objetiva y orientada a elaborar un modelo del mundo que represente la realidad. Esta concepción de la percepción la convierte en un problema inabordable por múltiples razones, entre ellas porque los sensores dan

una información incompleta e incierta de la realidad. Sin embargo esta arquitectura es adecuada para los sistemas semi-autónomos, pues es el operador el encargado de percibir el mundo y elaborar el modelo, dejando al robot la planificación de acciones y su ejecución.

3.4 Arquitecturas reactivas

A fin de dar una respuesta a los puntos débiles de las arquitecturas deliberativas, surgieron las investigaciones en lo que actualmente se denominan arquitecturas **reactivas**, [Brooks, 1986]. Los sistemas completamente reactivos, en su etapa inicial, estaban compuestos por un conjunto de comportamientos básicos que acoplan de forma directa percepción y actuación. El comportamiento global emergía de las interacciones de los módulos básicos entre sí y con el entorno. Inicialmente estos módulos no implementaban funciones complejas de percepción y planificación, y sólo activaban una reacción ante un estimulo específico obtenido a partir de los datos sensoriales crudos que constituían la representación instantánea del mundo.

Este enfoque incorpora la noción del robot como un agente físico situado, donde el entorno constituye una parte activa en la generación de su comportamiento observable. Las arquitecturas reactivas se han aplicado con éxito a múltiples robots para operación en tiempo real. Las diferentes propuestas muestran un diseño modular parecido, donde las diferencias aparecen en el modo en el que interactúan los diferentes comportamientos o módulos y en el método elegido para combinar las respuestas simultáneas de varios módulos. El cálculo de la acción dependerá, por tanto, del algoritmo de combinación de acciones que se haya diseñado y que opera únicamente sobre la representación instantánea del mundo.

Con respecto a los mecanismos de percepción, el paradigma reactivo propone una percepción orientada a la acción directa, donde cada estímulo está íntimamente unido con una salida de actuación. Por lo tanto, no existe más modelo del mundo que el que proporcionan los datos en un instante dado, sin estructuras de representación para mantener el estado del sistema/entorno, “el mundo es su mejor modelo”.

Las arquitecturas reactivas generan comportamientos reflejos seguros y eficaces ante

cambios rápidos en el entorno. Al no contemplar mecanismos de planificación no asumen ningún conocimiento previo o modelo del mundo y por lo tanto son robustas ante la incertidumbre, la falta de información y los cambios impredecibles del entorno.

A continuación se analizan brevemente dos arquitecturas reactivas, que podrían llamarse “puras”, es decir se trata de modelos constituidos por conjuntos de pares [condición-acción] que unen directamente entradas sensoriales con salidas a dispositivos de actuación sin disponer de memoria alguna [Mataric, 1992a]. La separación de estas arquitecturas reactivas puras de las basadas en comportamientos se fundamenta precisamente en este hecho, considerando que los modelos reactivos puros prohíben el almacenamiento de representaciones del entorno y del estado interno.

La primera de las arquitecturas reactivas analizadas es la arquitectura de subsunción propuesta por Brooks, origen de este paradigma e implementada posteriormente en múltiples robots móviles. La segunda es la ASM *(Action Selection Mechanism)* de Pattie Maes que introduce el concepto de propagación de la activación.

3.4.1 Arquitectura de subsunción

La arquitectura de subsunción planteada por Brooks descompone el sistema de control en comportamientos que se distinguen unos de otros por su competencia, es decir por la tarea que realizan (evitar obstáculos, bordear paredes o vagabundear sin rumbo fijo) y no por la funcionalidad inherente (planificación, modelado del entorno o decisión sobre posibles acciones), al contrario que en el paradigma deliberativo, figura 3.2. Cada uno de estos comportamientos es un máquina finita de estados que une ciertas entradas sensoriales con las salidas de actuación. Las dos interacciones posibles entre los diferentes comportamientos son la subsunción (suplantación de entradas) o la inhibición (supresión de salidas) [Brooks, 1986, Brooks, 1987], figura 3.3. Mediante la subsunción un módulo de nivel de competencia superior “suplanta” las entradas de uno inferior, modulando así su salida. Mediante la inhibición de una salida un módulo puede desactivar la salida de otro módulo.

Los objetivos del robot móvil están en la mente del diseñador que genera los diferentes módulos dando lugar a un comportamiento observable que emerge de la interacción de los comportamientos entre sí y de éstos con el entorno. Los objetivos están, por tanto, implícitos en las conexiones que unen los diferentes comportamientos.

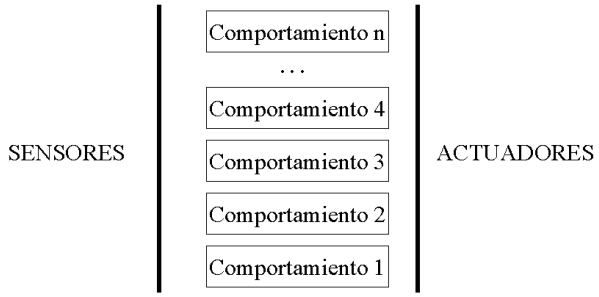


Figura 3.2 : Arquitecturas reactivas: modelo de descomposición por competencia

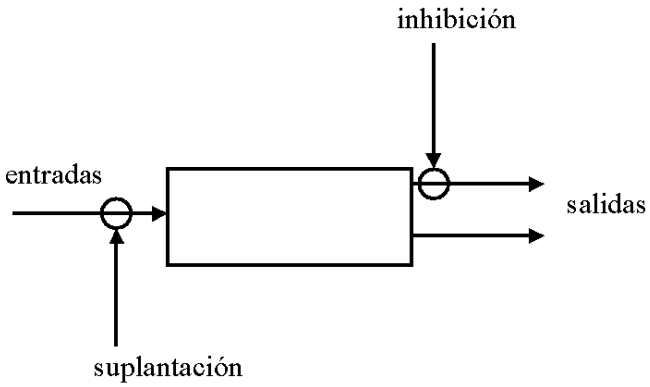


Figura 3.3 : Entradas y salidas en comportamientos de la arquitectura de subsunción

En el caso descrito en [Brooks, 1986] los comportamientos se organizan en tres niveles de competencia, el primero de ellos implementa la competencia de evitar colisiones, el segundo amplía esta competencia a vagabundear sin chocar y el tercero incluye el dirigirse hacia los lugares que se consideran interesantes. La actuación, por lo tanto, se materializa mediante un conjunto fijo de prioridades, que determinan la ejecución del comportamiento más urgente. Los comportamientos de alto nivel pueden anular las prioridades de los de bajo nivel mediante los mecanismos de suplantación e inhibición.

No existe mezcla alguna de comportamientos, ni fusión de varias tendencias al elegir la actuación más ventajosa en cada momento; sólo están presentes las pautas indicadas por el diseñador, combinando adecuadamente comportamientos, para que la acción resultante sea la apropiada.

En este paradigma, la percepción está completamente orientada a la acción, y las señales sensoriales instantáneas son las entradas a los comportamientos, sin que se plantee una representación del sistema/mundo; es más, Brooks explicita [Brooks, 1986] que no existe memoria compartida alguna ni representaciones del sistema/entorno.

Esta arquitectura supuso una ruptura con respecto a los paradigmas deliberativos de la Inteligencia Artificial, llamando la atención sobre el hecho de que no es necesario disponer de un modelo completo del mundo para decidir el tipo de acción a realizar y que el control se puede distribuir en varios niveles de competencia. Esto es, el control no reside en un único centro de decisión sino que el sistema se compone de varios niveles de competencia que se ejecutan en paralelo y la acción a ejecutar en cada instante surge de la interacción entre competencias y prioridades. La distribución del control ha sido una constante en casi todas las arquitecturas desarrolladas con posterioridad, tanto en forma de niveles de competencia, como de agentes, esquemas o comportamientos. Esta distribución ha planteado, desde entonces, el problema de la organización, coordinación, combinación o arbitraje de los distintos comportamientos unitarios de un sistema para la generación de comportamientos cada vez más complejos.

3.4.2 Arquitectura de Selección Dinámica de Acción ASM

En esta arquitectura, Pattie Maes plantea el problema de la selección de acción como una dinámica de energías que se propagan a través de la red de comportamientos y que se concentra en el más adecuado en relación al estado sistema/entorno y a los objetivos [Maes, 1989, Maes, 1990a, Maes, 1990b].

El modelo propuesto por Maes consta de un conjunto de módulos con una determinada competencia que interactúan para generar un comportamiento emergente adecuado a los

objetivos perseguidos y a la situación del entorno. Los módulos se describen mediante los siguientes parámetros: 1) *precondiciones* o condiciones requeridas por el módulo para su ejecución, 2) *poscondiciones* o consecuencias en el mundo de la acción ejecutada , y 3) *nivel de activación* del módulo.

Para que un módulo se ejecute en el tiempo t es condición necesaria que todas sus precondiciones sean ciertas. En base a esta descripción de los módulos se establecen relaciones de sucesión, antecedencia y conflicto entre ellos. El tipo de relación entre los módulos condiciona la transmisión de la *energía de activación* en la red. El valor que toma la energía determina la ejecución de un modulo. Tres fuentes diferentes inyectan energía a la red: el estado del mundo, los objetivos a conseguir y los objetivos alcanzados.

En [Maes, 1990a] se presentan los resultados, en simulación, obtenidos por un robot que puede realizar dos tareas: 1) pintar un tablón y 2) pintarse a sí mismo. En esta arquitectura no se contempla la percepción, y las precondiciones toman únicamente valores binarios, lo que constituye una simplificación excesiva sobre la realidad del mundo [Tyrrel, 1993]. Además no se abordan los mecanismos para percibir dichas precondiciones. La representación por lo tanto se reduce a los valores instantáneos que se emparejan a las precondiciones, de acuerdo con el paradigma de los sistemas reactivos.

La arquitectura de Maes es robusta frente a imprevistos y fallos en módulos siempre y cuando la tarea pueda ser realizada por el resto de los módulos. El mecanismo de propagación de la energía de activación permite al sistema adaptarse a nuevos objetivos. Igualmente ocurre con cambios repentinos en el entorno, que generan variaciones en la energía inyectada en la red, modificando el comportamiento observable. Se demuestra, [Maes, 1990a] siempre en simulación, que el sistema responde de un modo suficientemente rápido y que la velocidad de respuesta se ve influida por el proceso de sintonía de los parámetros de la red. La relación entre la percepción y la acción es compleja, ya que influyen todos los parámetros de la red que condicionan la transmisión de la energía entre los diferentes módulos a partir de un estado del entorno. Este proceso de sintonía permite sesgar la selección de acción hacia un carácter más finalista -orientado a objetivos- o más reactivo -orientado al entorno-. A pesar de carecer

de módulos explícitamente deliberativos, la dinámica del sistema conlleva una planificación implícita.

Tyrrell [Tyrrel, 1993] aplica este mecanismo de selección dinámica de acción a un entorno simulado para el análisis de comportamiento animal. En su experimentación surgen problemas en la transmisión de energía entre módulos, ya que resulta imposible sintonizar los parámetros para que la energía procedente de los módulos apetitivos, -aquellos cuyo objetivo es lograr situaciones para que los objetivos puedan conseguirse- sea correcta (pues sólo contribuyen indirectamente a la consecución de la energía). Se resalta la imposibilidad de acciones de compromiso, pues una vez elegido el comportamiento a ejecutar éste toma el control completo del sistema.

3.4.3 Ventajas e inconvenientes de las arquitecturas reactivas

Las arquitecturas reactivas muestran propiedades opuestas, en muchos aspectos, a las de las arquitecturas deliberativas, sección 3.3, sobre todo en relación con las características requeridas para un robot móvil agrícola, capítulo 2 sección 2.2. Los robots controlados mediante una arquitectura reactiva tienen como características principales la robustez y la capacidad de reacción, ambos aspectos fundamentales para un vehículo agrícola en un entorno dinámico. El enfoque reactivo se adapta bien cuando: 1) la triada robot-entorno-objetivo permite anticipar en el diseño los pares acción-reacción más apropiados, 2) los sensores proporcionan toda la información necesaria y 3) la tarea a realizar está bien definida y no hay cambio de objetivos [Mataric, 2001].

En el caso de los robots agrícolas, el entorno de exteriores hace que no sea posible anticipar los pares acción-reacción en todas las situaciones, puesto que se pretende trasladar la misma arquitectura a entornos y tareas ligeramente distintos. Por esta misma razón interesa que la arquitectura asuma la posibilidad de realizar cambios en los objetivos, por ejemplo que el operador especifique una posición objetivo diferente en cada sesión de trabajo.

La percepción en exteriores complica aún más el diseño de un sistema totalmente reactivo

puesto que los estímulos necesarios pueden no estar contenidos en un único sensor o pueden requerir mayor grado de elaboración que la que se obtiene a partir de una representación instantánea.

Sin embargo a medida que crece el número de comportamientos básicos, la gestión de las relaciones entre ellos y el diseño de un sistema de arbitraje se hace cada vez más complicado. El inconveniente más grave que estas arquitecturas, al carecer de memoria a medio o largo plazo es su incapacidad para acometer comportamientos complejos que requieran una percepción más elaborada, una posibilidad de predicción hacia el futuro y una optimización del comportamiento observable.

3.5 Arquitecturas híbridas

Las arquitecturas deliberativas y las reactivas representan dos extremos opuestos del problema de una arquitectura para la organización del conocimiento y control. Las deliberativas ponen su énfasis en la parte racional de la resolución del problema, insistiendo en una representación completa y por lo tanto difícil de manejar y actualizar; en los métodos generales de resolución de problemas y en la planificación para tomar la decisión más correcta. Por el contrario, los enfoques reactivos se centran en la capacidad de reacción frente a cambios en el entorno, insistiendo en que el robot está inmerso en un mundo concreto, que sus respuestas deben ser rápidas, sencillas y que es la interacción del sistema con el entorno la que guía el comportamiento observable, sin necesidad ni de memoria ni de representación ni planificación.

Entre ambos extremos se sitúan las arquitecturas híbridas que pretenden incorporar, con diferentes grados, tanto capacidades de deliberación como de reacción a los robots móviles, a fin de que puedan desenvolverse de modo autónomo en un mundo dinámico y además que incorporen mecanismos de planificación y representación a medio o largo plazo. Dentro de esta sección se han agrupado dos enfoques:

1. Las arquitecturas híbridas que parten de ampliar las arquitecturas reactivas mediante la inclusión de capacidades deliberativas y de representación [Mataric, 1992b]. Estas

arquitecturas se analizan en la sección 3.5.1; siguiendo la nomenclatura propuesta en [Mataric, 2001] las denominamos como arquitecturas basadas en comportamientos. A diferencia de las arquitecturas reactivas puras, sección 3.4, estos sistemas permiten almacenar el estado del sistema/entorno en una o varias representaciones, posibilitando funciones deliberativas y de aprendizaje.

2. Las arquitecturas híbridas de tres capas, sección 3.5.2, que parten de un planteamiento opuesto, dotar de capacidades reactivas a modelos más próximos a las arquitecturas deliberativas. Estas arquitecturas están formadas por capas; habitualmente tienen una capa deliberativa que agrupa los procesos más racionales; una capa de bajo nivel con comportamientos reflejos, que potencia la capacidad de reacción ante imprevistos; y una capa intermedia de interacción y coordinación entre ellas.

3.5.1 Arquitecturas híbridas basadas en comportamientos

Las arquitecturas basadas en comportamientos se encuentran influidas por modelos de comportamiento animal. Han sido defendidas en los trabajos de Maja Mataric [Mataric, 2001] por las mejoras que presentan frente a las arquitecturas puramente reactivas. La unidad básica de estas arquitecturas es el comportamiento y el funcionamiento global del robot surge de la composición de estos comportamientos y de su interacción con el entorno al igual que en los modelos reactivos. Pero a diferencia de éstos los comportamientos pueden disponer tanto de una representación del sistema/entorno a medio o largo plazo, como de capacidades deliberativas.

Estos comportamientos pueden organizarse en niveles, sin restricción alguna sobre la escala temporal y de representación de cada nivel. La construcción de una arquitectura para robots basados en comportamientos comienza por el diseño del nivel inferior, que implementa los comportamientos básicos de supervivencia y sigue con la integración gradual de comportamientos con capacidades cada vez más complejas, acordes con los objetivos perseguidos.

El mecanismo de selección de acción en estas arquitecturas es dependiente del contexto. Las

salidas de los distintos comportamientos pueden combinarse o anularse unas a otras.

En cuanto a la percepción y a la representación no existe restricción alguna, no existe una percepción centralizada ni un modelo del mundo común y tanto los procesos como la memoria se hallan distribuidos. Las arquitecturas basadas en comportamientos han encontrado un nicho claro de aplicación en el desarrollo de sistemas con múltiples robots, en el diseño de comportamiento colectivo y aprendizaje [Parker, 1998, Mataric, 2001]. En esta sección se presentan tres arquitecturas de comportamientos, la arquitectura desarrollada por Mataric [Mataric, 1991, Mataric, 1992b] fundadora de esta aproximación, la arquitectura AuRA propuesta por Arkin [Arkin y Balch, 1997] y la arquitectura DAMN de Rosenblatt.

Arquitectura de comportamientos M.Mataric

Esta arquitectura surge de la necesidad de ampliar la arquitectura de subsunción con la finalidad de ofrecer soluciones más allá de una navegación al azar segura. Por ello incorpora a la arquitectura de subsunción representaciones del objetivo, de la posición del robot y del entorno [Mataric, 1992b] extendiendo el modelo de subsunción a dominios que requieren representaciones internas del mundo. Esta representación interna del espacio se materializa en un mapa cognitivo inspirado en un modelo neurobiológico del cerebro de una rata, a su vez basado en los resultados de la experimentación sobre la información espacial que manejan estos animales [Mataric, 1991]. La innovación radica en que este mapa se almacena como una red de comportamientos dentro de una arquitectura de subsunción con tres capas.

La primera capa implementa un conjunto de comportamientos reflejos básicos, -volver, evitar, alinear y corregir- que conforman el comportamiento de movimiento real del robot siguiendo contornos y evitando colisiones sin conocer la posición del robot.

La segunda capa se dedica a la detección de marcas en el entorno, donde cada marca se corresponde con un conjunto de parámetros de movimiento del robot y de lecturas de los sensores de ultrasonidos. Cada marca detectada constituye un nodo del mapa topológico, almacenado como un comportamiento, y que está formado por entradas, salidas y reglas.

Esta capa incorpora la diferencia fundamental con las arquitecturas reactivas ya que incluye representación espacial y memoria en forma de comportamientos. El robot a lo largo de su recorrido va almacenando nodos. Para navegar, desde el nodo objetivo se expande la activación hasta la posición actual encontrando el camino óptimo sobre el mapa topológico. La tercera capa es la encargada de realizar los cálculos sobre el mapa topológico para la planificación de caminos.

La selección de acción se efectúa mediante el mecanismo de subsunción, es decir mediante una red de prioridades establecida en la fase de diseño que se materializa en el modo de conexión de entradas y salidas de los comportamientos.

La percepción está inequívocamente orientada y unida a la acción y la representación del entorno se construye a través de los comportamientos que representan las marcas espaciales almacenadas en un mapa topológico del entorno.

La arquitectura se ha demostrado con experimentación en tiempo real realizada con el robot Toto.

Arquitectura para Robots Autonomos AuRA

La arquitectura AuRA (*Autonomous Robot Architecture*) aparece a mediados de los años 80, y se articula mediante el concepto de *esquema*. Esta arquitectura se ha implementado en robots con características muy diferentes [Arkin, 1987, Arkin, 1989, Arkin y Balch, 1997, Arkin et al., 2000, Sttoytchev y Arkin, 2001, Arkin et al., 2003], evolucionando a partir del modelo inicial. El concepto esquema proviene del campo de la psicología y neuro-biología [Arbib, 1989, Arbib y Cobas, 1990, Arbib y Liaw, 1995], y ha sido integrado en el desarrollo de sistemas artificiales por R.Arkin.

Un *esquema* se define como la especificación genérica de un agente o proceso computacional [Arkin, 1987]. Se contemplan dos tipos de esquemas: 1) los esquemas motores dedicados al cálculo de la acción del robot y 2) los esquemas perceptivos que elaboran las entradas sensoriales o estímulos requeridos por los esquemas motores. Los esquemas se

organizan en una red dinámica cuya configuración tiene como soporte la percepción en un instante dado y los objetivos que se persiguen. Cada esquema motor proporciona su salida en forma de campo de potencial, integrándose los vectores resultantes de los esquemas relevantes para de obtener una única acción de control.

Posteriormente [Arkin y Balch, 1997] la arquitectura AuRA, se ha completado con la integración de procesos deliberativos, aproximándose más a las arquitecturas híbridas de niveles. Los aspectos deliberativos incluyen un *Planificador de Misiones* para optimizar la secuencia de objetivos, un *Razonador Espacial* para elaborar el plan de movimientos y un *Secuenciador del Plan* que, junto al *Controlador de Esquemas*, se encarga de activar las instancias de los esquemas apropiados. Además, AuRA incorpora en el proceso de selección de acción el conocimiento sobre el estado interno del robot, que se basa en el concepto biológico de homeostasis [Arkin, 1992] y se materializa en un nuevo tipo de esquema: el esquema señal. Los esquemas señal son de dos tipos: 1) transmisores que recogen las señales sensoriales del estado interno del robot y 2) receptores, que modifican el esquema motor al que van asociados modulándolo con la variable homeostática a la que son sensibles. Este estado interno contribuye, junto con el estado del entorno elaborado por los esquemas perceptivos, al mecanismo de selección de acción. En trabajos más recientes el estado interno se ha incorporado a un subsistema de motivaciones [Sttoytchev y Arkin, 2001], con el mismo propósito, que el estado interno del robot influya en el proceso de selección de acción junto con el estado del entorno [Arkin et al., 2003].

El mecanismo de selección de acción en AuRA se realiza mediante la composición de las salidas de los comportamientos motores. La percepción se plantea, igual que la representación del entorno, orientada a la acción, y los esquemas perceptivos están asociados al esquema motor que requiere su percepción [Arkin, 1987].

Arquitectura distribuida para navegación de vehículos DAMN

Al igual que las dos arquitecturas anteriores, DAMN (*Distributed Architecture for Mobile Navigation*) es el resultado de la búsqueda de mejoras al modelo de subsunción. Se centra en la fusión de comandos, dado que en la arquitectura de subsunción únicamente se atiende a la salida del comportamiento de mayor nivel sin considerar las recomendaciones de los menos prioritarios [Rosenblatt y Payton, 1989]. Los comportamientos en DAMN tienen unas entradas con un peso asociado, y poseen un nivel de activación y una salida. Cada comportamiento aporta votos a las distintas alternativas de salida y su peso, que puede ser modificado en tiempo de ejecución por el *Gestor de Modos*. Los comportamientos, que se ejecutan de modo asíncrono, envían sus votos a los *árbitros*, que deciden las acciones de los controladores del vehículo [Rosenblatt, 1997]. En DAMN el mecanismo de selección fusiona las preferencias de los comportamientos con un peso proporcional al nivel de activación, considerando los votos de todos los comportamientos. El trabajo de Yen et al. se propone un mecanismo basado en conjuntos borrosos para la selección de acción en el modelo DAMN [Yen y Pfluger, 1995].

El enfoque de la arquitectura DAMN permite incorporar los planificadores como un módulo más que envía sus votos al árbitro, de forma que los planes se transforman en recursos y contribuyen con una salida ponderada del mismo modo que el resto de los comportamientos [Payton et al., 1990a]; constituyen lo que se denomina planes interiorizados [Payton et al., 1990b]. En trabajos más recientes [Rosenblatt, 2000], DAMN ha evolucionado hacia la fusión de utilidades en lugar de comandos. Estas utilidades expresarían el “deseo” de cada comportamiento de estar en un estado particular respecto al mundo, valorándose las posibles consecuencias de cada acción.

Tanto la percepción como la representación del sistema/entorno están orientadas a la acción y distribuidas siguiendo las pautas de una arquitectura basada en comportamientos.

Esta arquitectura se ha demostrado en un vehículo para navegación en campo a través sin mapa [Rosenblatt, 1997].

3.5.2 Arquitecturas híbridas de niveles

Las arquitecturas híbridas de niveles pretenden combinar la respuesta a tiempo de los enfoques reactivos con la selección de una acción orientada a objetivo propia de los enfoques deliberativos [Mataric, 2001, Murphy, 2000]. A diferencia de las arquitecturas basadas en comportamientos, sección 3.5.1, las arquitecturas híbridas tienen dos componentes claramente diferenciados, por un lado los deliberativos y por otro los reactivos. Los componentes reactivos proporcionan respuestas rápidas del sistema ante necesidades inmediatas, como evitar un obstáculo imprevisto, mientras que los deliberativos planifican las tareas utilizando representaciones abstractas y simbólicas y ventanas temporales amplias. El problema se traslada ahora a cómo combinar ambas capas, la reactiva y la deliberativa, para obtener un comportamiento observable apropiado a los objetivos. Normalmente esta combinación se lleva a cabo mediante una capa intermedia que constituye realmente el gran reto en el diseño de estas arquitecturas.

A continuación se describen las cuatro arquitecturas seleccionadas como representativas de este paradigma: la arquitectura 3T y los paquetes reactivos RAP, la arquitectura TCA y la arquitectura Saphira. Estas tres arquitecturas están implementadas en diferentes plataformas robóticas y están ampliamente difundidas en la comunidad robótica. Por último se describe la arquitectura AMARA, desarrollada en el IAI-CSIC y predecesora de la arquitectura propuesta en esta tesis.

Arquitectura de Paquetes Reactivos RAP y Arquitectura 3T

La principal motivación de esta propuesta es conseguir una planificación en tiempo de ejecución que permita abordar entornos dinámicos. En estos entornos las acciones no pueden ser anticipadas pues el estado del sistema/entorno en el momento en el que se realiza la planificación no siempre coincide con el que del momento de la acción [Firby, 1987]. Este tipo de planificación se conoce como “Planificación reactiva”, y el mecanismo que se propone para su implementación son los *paquetes de acción reactiva*, RAPs. Los RAPs se encargan

de expandir los planes esquemáticos que pueda proporcionar un planificador, en instrucciones detalladas en tiempo real [Firby, 1992].

Cada RAP dispone de un conjunto de métodos para ejecutar su cometido así como de una colección de procesos de percepción distintos, que se aplicarán dependiendo del contexto sistema/entorno. En lugar de métodos, un RAP puede contener a su vez un conjunto de RAPs, generando instrucciones para habilitar, deshabilitar o reconfigurar las habilidades reactivas. Un requisito de los RAPs es que tiene que integrar un inicio y un fin de actuación y la capacidad para comunicar el éxito o fracaso de su tarea.

La arquitectura 3T (de *3-tiered*) [Bonasso et al., 1997] es una arquitectura de tres capas en la que el nivel inferior está compuesto por una colección de comportamientos reactivos, o "habilidades" según la notación de [Bonasso et al., 1997]. El nivel superior consiste en un planificador de planes esquemáticos y la capa intermedia es el ejecutor de RAP.

El planificador realiza la descomposición inicial de tareas o plan esquemático, apoyándose en la colección de RAPs disponibles de modo que cada sub-tarea corresponde a uno o más RAPs [Bonasso et al., 1997]. El intérprete RAP selecciona una de las tareas para su ejecución, y si representa una acción básica la ejecuta directamente; en caso contrario añade a la cola de ejecución las RAPs internas. El siguiente paso consiste en comprobar si el objetivo que persigue la RAP se ha cumplido: en caso afirmativo pasa a la siguiente tarea; si no, busca entre los métodos de la RAP aquellos que son aplicables en el contexto actual y se añaden a la cola de ejecución sustituyendo a la tarea seleccionada [Firby et al., 1995].

El mecanismo de selección de acción de la arquitectura 3T hace uso del contexto, que es quien decide en último lugar cuál es el método más adecuado para ejecutar una acción, entre todos los posibles.

La percepción está totalmente orientada a la acción, pues las habilidades perceptivas se incluyen como métodos en los RAPs. Sin embargo, existe un modelo del mundo común que se actualiza por medio del intérprete de RAP y de los sensores [Firby, 1987]. El mantenimiento de este modelo del mundo junto con los mecanismos de coordinación entre RAPs son los puntos más débiles de esta arquitectura.

La arquitectura 3T y los RAPs han dado soporte, con éxito, al comportamiento de varios robots reales, entre ellos el robot K2A cuya misión es el reconocimiento de personas, el robot Chip recolector de basura [Kortenkamp et al., 1998a], el robot MITRE navegador en entornos de oficina y diversos brazos robóticos [Bonasso et al., 1997].

Arquitectura de Control de Tareas TCA

La arquitectura de Control de Tareas TCA, está compuesta por módulos que se comunican mediante paso de mensajes a través de un módulo central, el *módulo de control central (central control module)*. Este módulo es el responsable de dirigir y procesar los mensajes adecuadamente y de mantener la información de control de las diferentes tareas. Este intercambio de mensajes entre módulos va construyendo un árbol de tareas, donde se incluye: 1) cómo descomponer una tarea en subtareas, 2) cuándo planificar y cuándo ejecutar, 3) cómo monitorizar la ejecución y 4) cómo reaccionar ante imprevistos [Simmons, 1994].

Los módulos se comunican mediante seis tipos de mensajes: información, pregunta, objetivo, comando, monitor y excepción. Los mensajes de información y pregunta sirven para intercambiar información. Los mensajes de excepción gestionan los posibles errores en la ejecución de tareas. Cuando llega al control central un mensaje de tipo comando, objetivo o monitor, TCA construye un árbol de tareas, donde los nodos pueden ser los objetivos, que a su vez se vuelven a expandir en ramas, o los comandos que son directamente ejecutables. Los árboles de tareas son dinámicos, puesto que su expansión depende del contexto, y en circunstancias diferentes una misma tarea dará lugar a árboles distintos [Simmons y Apfelbaum, 1998]. Los nodos pueden estar en cuatro estados: deshabilitado, habilitado, activo y completo y llevan asociadas restricciones temporales que permiten controlar la expansión de los nodos y cuándo ejecutarlos, es decir permiten intercalar planificación y ejecución.

El diseño de TCA proporciona un marco donde se pueden integrar tareas tanto deliberativas como reactivas. Gran parte de las habilidades reactivas de TCA se deben

a los *monitores*, o nodos en el árbol de tareas que pueden ser invocados repetidamente [Simmons y Apfelbaum, 1998] proporcionando un sistema de “alerta” ante contingencias. TCA contempla también la gestión de excepciones asociándolas a un nodo. Si algo falla, se especifica la causa y se busca un nodo capaz de gestionarla. TCA es una arquitectura muy completa, que permite añadir con facilidad comportamientos, monitores y mensajes. Se ha demostrado en la ultima década en diferentes robots: Xavier y HERO [Simmons et al., 1997] en interior y AMBLER [Simmons, 1994] en exterior. AMBLER es un robot de seis patas de tamaño grande para el desplazamiento por terrenos accidentados. En este robot las capacidades deliberativas incorporadas en la arquitectura TCA son especialmente importantes para la planificación de movimientos seguros optimizando energía.

Arquitectura Saphira

Saphira es una de las arquitecturas más extendidas en la comunidad de robótica. Los orígenes de Saphira se remontan a comienzos de los noventa en el *Centro de Inteligencia Artificial* del SRI. La difusión de Saphira se ha favorecido al estar fundamentada en el paradigma de comunicaciones cliente-servidor, donde el servidor es un programa que se encarga: 1) del control del bajo nivel en comunicación directa con los actuadores, 2) de la adquisición de datos sensoriales, 3) del procesamiento básico de algunas señales sensoriales, como las de los codificadores de posición y 4) del envío y recepción de datos de los clientes. Saphira se implementa como un proceso cliente, por tanto independiente de cualquier plataforma robótica. Algunos de los robots que realizan tareas basados en una arquitectura tipo Saphira son: Flakey, Erratic, Pioneer, Kephera [Konolige et al., 1997].

El núcleo central de Saphira es el espacio perceptivo local, LPS (*Local Perceptual Space*). Se trata de una representación geométrica centralizada en la que coexisten representaciones del entorno a diferentes “niveles”: rejilla de ocupación, superficies y los denominados “artefactos” o representaciones de estructuras abstractas que son utilizadas por los comportamientos orientados a un objetivo (un pasillo, por ejemplo). Mantener representaciones a distintos niveles es una tarea compleja y costosa desde el punto de vista computacional, pero tiene

la gran ventaja de ofrecer a cada comportamiento la representación más adecuada. Así, los comportamientos reactivos más rápidos trabajan directamente con rejillas de ocupación mientras que los orientados a objetivo utilizan una representación de artefactos.

Diversos procesos perceptivos mantienen actualizado el LPS, ya sea añadiendo información nueva, borrando información obsoleta o procesando información del propio LPS para extraer estructuras de alto nivel, por ejemplo para extraer segmentos de paredes a partir de datos sensoriales crudos. Estas características se combinan en hipótesis de objetos, por ejemplo dos segmentos lineales paralelos son la hipótesis de un nuevo concepto, el pasillo.

Los comportamientos en Saphira están implementados con conjuntos borrosos y constan de dos procesos diferentes: 1) una actualización de las variables borrosas que necesita el comportamiento en base a la información del LPS y 2) un conjunto de reglas borrosas con variables lingüísticas en los antecedentes -que se instancian en cada ciclo de control-, y cuyos consecuentes son las posibles acciones del robot [Saffiotti et al., 1993]. Las salidas lingüísticas de todos los comportamientos se combinan en lo que los autores denominan “Combinación dependiente del contexto”. Este método de fusión de comandos permite combinar las preferencias de los diferentes comportamientos teniendo en cuenta el contexto, de manera que éste puede modificar el peso de cada comportamiento en el cálculo de la acción final según la situación. Así comportamientos relevantes ante una situación determinada influirán más en la salida que otros que no lo son tanto. La combinación de salidas mediante el algoritmo de “Combinación dependiente del contexto”, permite fusionar las preferencias de comportamientos reactivos con las de comportamientos orientados a objetivos. Además, en la parte deliberativa, Saphira lleva integrado un planificador de tareas PRS-lite encargado de descomponer el objetivo de alto nivel en objetivos parciales que pueden ser secuenciados de diferentes maneras, añadiendo así flexibilidad a la etapa de planificación.

Arquitectura AMARA

La arquitectura AMARA (Arquitectura MultiAgente para Robots Autónomos) desarrollada en el IAI-CSIC [Garcia-Alegre et al., 1995], [García-Alegre y Guinea, 1992], [García-Alegre y Recio, 1998] se basa en el concepto de agente definido como un proceso con capacidad de percepción, cómputo y acción para conseguir o mantener un objetivo y se considera equivalente al concepto tradicional de comportamiento [García-Alegre y Recio, 1998]. Los agentes en AMARA están organizados jerárquicamente en niveles que gravitan en torno a una representación específica del sistema y entorno. Los agentes poseen habilidades deliberativas y reactivas sin restricción alguna en su complejidad. Los agentes de un nivel se comunican con los de los niveles adyacentes por medio de un lenguaje común, formado por términos relacionados con la percepción, representación del entorno y actuación. La organización jerárquica permite abordar la complejidad de un problema al descomponer la realización de una tarea en subobjetivos que son resueltos por las competencias de los diferentes niveles que componen la arquitectura.

El proceso de selección de acción se ejecuta mediante arbitraje, de manera que en cada nivel, en cada instante de tiempo, sólo se ejecuta un agente. Esto se debe a que el espacio perceptivo de los agentes se ha configurado de forma que sea mutuamente excluyente. El mecanismo de arbitraje depende de los objetivos del sistema y de la situación del robot, de forma que los agentes de un nivel son arbitrados por los de niveles superiores adyacentes.

Cada agente integra un conjunto de procesos perceptivos y deliberativos dirigidos a la consecución del objetivo que le define. La arquitectura explícitamente muestra dos flujos de información: 1) perceptiva se propaga desde los sensores a las capas superiores y 2) de actuación que se propaga en sentido contrario desde el usuario que desencadena la misión hasta los actuadores localizados en el nivel inferior. En cada nivel de la jerarquía existe una representación del sistema/entorno soportada por una memoria de profundidad variable. La arquitectura ha sido demostrada en navegación en interiores de oficina en tareas de reconocimiento visual de estructuras con el robot Boss [Garcia-Alegre et al., 1995] y

en la tarea concreta de seguimiento de personas guiada por visión con el robot Hermes [Cañas y García-Alegre, 1999]. La arquitectura AMARA también ha sido propuesta en [García-Alegre y Guinea, 1992] como modelo para la navegación de un robot agrícola.

3.5.3 Ventajas e inconvenientes de las arquitecturas híbridas

Un robot móvil agrícola necesita incorporar habilidades tanto reactivas como deliberativas, si quiere desenvolverse con seguridad optimizando la realización de sus tareas en un entorno dinámico con incertidumbre. Las arquitecturas híbridas son robustas, fuertemente distribuidas, especialmente en el caso de las arquitecturas basadas en comportamientos. Las arquitecturas de niveles están dotadas de mecanismos de detección de fallos en la ejecución y de capacidad para intentar resolverlos ya sea mediante un mecanismo de tratamiento de excepciones como en TCA o mediante la implementación de opciones diversas para resolver una misma tarea, como en la arquitectura RAP.

Todas las arquitecturas presentadas tienen a su favor la facilidad para añadir nuevas funcionalidades sin tener que modificar excesivamente el código, y muchas de ellas están implementadas sobre plataformas robóticas diversas, realizando tareas diferentes en entornos dispares [Rosenblatt, 2000, Arkin, 1987, Bonasso et al., 1997, Simmons, 1994, Konolige et al., 1997]. Algunas de ellas se han implementado en robots que realizan tareas en exteriores como AuRA, DAMN o TCA pero no existe ninguna propuesta en el ámbito de la agricultura si exceptuamos la arquitectura AMARA [García-Alegre y Guinea, 1992].

La cuestión clave, tanto en las arquitecturas basadas en comportamientos como en las híbridas de niveles, está en cómo combinar un modelo orientado primordialmente a la consecución de unos objetivos con otro orientado a la interacción rápida con el entorno, esto es, en cómo combinar capacidades reactivas con deliberativas [Arkin, 1995]. Después de un análisis en profundidad de las diferentes propuestas, se comprueba que cada arquitectura resuelve de un modo diferente el compromiso entre deliberación y acción. Arquitecturas como ALLIANCE [Parker, 1998] o AuRA añaden un tipo de variables denominadas motivaciones

que sesgan el comportamiento del robot hacia la satisfacción de unos objetivos internos.

Otro de los retos de estas arquitecturas consiste en cómo coordinar los diferentes comportamientos y cómo realizar un arbitraje en situaciones de conflicto. Las opciones que surgen son múltiples, desde la fusión de comandos en AuRA por medio de la combinación de campos de potencial, la votación en DAMN, o la combinación con pesos de variables borrosas en Saphira, hasta la selección de un comportamiento "ganador" que determina la acción a realizar en la arquitectura de Mataric, TCA y RAP.

Con respecto a la forma de integrar la representación en la arquitectura, las arquitecturas híbridas tienden hacia modelos centralizados aunque admiten distinto tipo de representaciones, como sucede con la arquitectura Saphira. Las arquitecturas basadas en comportamientos proponen representaciones distribuidas y fuertemente orientadas a la resolución de una tarea concreta.

Un problema adicional de las arquitecturas híbridas de niveles, ausente en las basadas en comportamientos, se discute en [Volpe et al., 2001] y gira en torno a la decisión sobre la localización de las fronteras entre cada uno de los niveles. La apuesta de estas arquitecturas es heredada del paradigma deliberativo: requisitos de alto nivel de inteligencia, múltiple granularidad y la frecuencia de percepción y de toma de decisiones. La implementación de comportamientos que manejan información con alto grado de abstracción plantea problemas en la realización de tareas en tiempo real critico.

3.6 Hacia una arquitectura para robots agrícolas autónomos

En las secciones 3.2, 3.3, 3.4 y 3.5 se han definido los conceptos básicos de una arquitectura y se han descrito los paradigmas más establecidos en el diseño de las arquitecturas de control. Se ha analizado los diferentes aspectos de cada arquitectura y cómo se adecuan éstos a las aplicaciones en el campo de la agricultura. En esta sección final del capítulo se van a analizar las consecuencias sobre la arquitectura de control que imponen los requisitos definidos por el tipo de aplicación y vehículos agrícolas. El tipo de objetivos, robots y entorno van a ser los

condicionantes de la arquitectura de control más adecuada.

La primera de las características de una arquitectura para un robot móvil agrícola es que debe ir dirigida hacia la consecución de un **objetivo explícito** bien definido. Un robot móvil agrícola se construye para la realización de una misión concreta que debe cumplir, no se trata de tareas de vagabundear y observar. Si la tarea del robot es la fumigación de un determinado campo de frutales, la arquitectura ha de asegurarse del cumplimiento de la tarea y de que el robot no "de vueltas" por el campo. Para ello son necesarias las habilidades deliberativas y la planificación. Por lo tanto, una arquitectura de las denominadas reactivas puras, sección 3.4, queda descartada.

Además de conseguir un objetivo debe hacerlo de la manera más **eficaz** posible. Un robot agrícola es una herramienta de trabajo y por lo tanto tiene que ser eficaz, la arquitectura tiene que dotarle de los mecanismos necesarios para que cumpla su misión en el menor tiempo posible y de la manera más adecuada. La reducida capacidad de maniobra de los robots agrícolas impone una buena planificación de la ruta, que evite rodeos y retrocesos innecesarios por el coste de tiempo que conllevan este tipo de maniobras. Sin una buena planificación el robot puede encontrarse en callejones sin salida de los que le resulte imposible salir de modo autónomo y haya que requerir la presencia del operario. La arquitectura debe dotar al robot de habilidad para hacer frente a una tarea con ciertas restricciones de tiempo y espacio. No obstante las velocidades relativamente lentas de los vehículos agrícolas en tareas de laboreo, van a facilitar el desarrollo y la ejecución a tiempo de los procesos tanto perceptivos como motores.

Sin embargo, el entorno agrícola es dinámico y no completamente controlado, por lo cual debe ser capaz de enfrentarse con éxito a imprevistos, y, en consecuencia, poseer una alta **capacidad de reacción** y robustez ante fallos del sistema. Además los vehículos agrícolas poseen un gran peso que les confiere una fuerte inercia. Hay que garantizar su seguridad y la del entorno, máxime cuando va a estar a grandes distancias del operario, de ahí que necesite una capacidad de reacción alta frente a objetos imprevistos. Y dado que las distancias en el campo son grandes, la arquitectura debe responder también ante fallos en las partes del sistema a fin de garantizar su seguridad. Por ejemplo, ante pérdidas de la señal de alguno de los sensores

de posicionamiento, como ocurre en ocasiones con el GPS, se debe asegurar una respuesta correcta del sistema. Una arquitectura para la navegación de un robot agrícola tiene que tener componentes reactivos que garanticen su reacción correcta ante imprevistos.

Las características del entorno agrícola contribuyen a que las actuaciones del robot no siempre tienen el efecto deseado: las ruedas patinan con facilidad, hay piedras, terrones, surcos, badenes ... Por ello se hace imprescindible la **monitorización de los resultados de las acciones del robot** en el entorno. Esta monitorización implica que el robot tiene que estar constantemente percibiendo su entorno, en línea con las arquitecturas con componentes reactivos, y actualizando sus creencias sobre el estado del mundo.

Los entornos de exteriores son, además de dinámicos, particularmente complejos, ya que a la dificultad del reconocimiento de las estructuras presentes se une una alta variabilidad de las condiciones del entorno, tales como, la iluminación, el terreno, la visibilidad o el polvo. Por ello en exteriores la percepción orientada a la reconstrucción del mundo es un problema inabordable. Cabe pensar entonces en una **percepción orientada a la tarea**, y si no totalmente distribuida, como en la arquitectura de Maes, sí por lo menos heterogénea como sucede en la arquitectura Saphira. La incertidumbre en entornos de exteriores es elevada, debido a las condiciones ambientales de iluminación, los perfiles poco regulares de los objetos naturales, etc. Las técnicas basadas en la lógica borrosa para el diseño de los diferentes comportamientos y de las estrategias perceptivas son un buen método para gestionar esta incertidumbre. Bastantes arquitecturas emplean técnicas borrosas, como Saphira, AMARA y la variante de DAMN propuesta en [Yen y Pfluger, 1995].

Sin embargo, aprovechando que el entorno no es totalmente desconocido -si el robot tiene que fumigar naranjos es posible disponer de un patrón para su reconocimiento- es posible **incorporar a priori toda la información disponible** sobre el sistema, el entorno, y las peculiaridades de las tareas a fin de facilitar tanto la percepción como la organización del conocimiento.

Otra característica deseable de la arquitectura para un robot agrícola es la **versatilidad**, esto es, capacidad para abordar, con cambios mínimos, tareas en entornos ligeramente distintos y con

objetivos diferentes. Esta característica está relacionada con las ventajas de las arquitecturas con componentes deliberativos, que permiten el cambio de objetivos al estar éstos explícitos.

Una arquitectura para la navegación de robots agrícolas se desarrolla habitualmente de modo gradual, añadiendo progresivamente nuevas capacidades que aumenten el grado de autonomía del robot. Es habitual diseñar en primer lugar un sistema de teleoperación e ir añadiendo sucesivamente habilidades para navegar dirigiéndose a un objetivo, evitando colisiones, con una planificación de la trayectoria, etc... Por eso es importante que la arquitectura esté diseñada para **facilitar la incorporación de nuevos comportamientos**. Esta característica está directamente relacionada con la **escalabilidad**, permitiendo al robot manejar problemas cada vez más complejos que demanden una cantidad mayor de conocimiento sin cambios directos en los mecanismos internos de la arquitectura. Como el entorno es diverso y las funciones son especializadas, no todas las habilidades se deberían utilizar en una tarea especifica de la arquitectura, por ejemplo en una navegación genérica puede no ser necesario forzar trayectorias rectilíneas que sin embargo son características de la navegación para el laboreo. La descomposición del comportamiento global en módulos expertos en distintas tareas, es decir la **modularidad**, favorece esta reutilización de habilidades. Las arquitecturas basadas en comportamientos contemplan estas propiedades. En el caso de la arquitectura AuRa incluso los mecanismos perceptivos se encapsulan en módulos (*esquemas* en esta arquitectura).

En resumen una arquitectura para la navegación autónoma de un robot agrícola debería tener las siguientes características: orientación a objetivos, planificación para la actuación eficaz, capacidad de reacción, capacidad para monitorizar de los resultados de las acciones del robot, percepción orientada a la tarea y adecuada a entornos con incertidumbre, posibilidad de incorporación a priori de toda la información disponible, facilidad para añadir nuevos comportamientos, escalabilidad y modularidad.

Por lo tanto debería enmarcarse en las arquitecturas híbridas, siguiendo alguna de las pautas ya formuladas en AMARA: modularidad, facilidad de escalado, comportamientos borrosos y distribuidos, orientación a objetivos y capacidad de reacción. La arquitectura ha

de dedicar además, una especial atención al problema de la percepción, favoreciendo tanto la representación como la percepción distribuidas y orientadas a los objetivos, capaz de integrar y utilizar toda la información disponible del entorno, de la tarea y del robot.

Capítulo 4

Automatización de un tractor agrícola

En los últimos años se está trabajando en la automatización progresiva de vehículos para navegación en exteriores, tanto por tierra como por mar y aire. Normalmente se comienza por el desarrollo de sistemas de ayuda a la navegación en modo teleoperado para continuar hacia un modo de operación con cierto grado de autonomía. Las áreas que han experimentado mayor auge son aquellas que están relacionadas con el transporte, ya sea terrestre [Dickmanns, 1993], marítimo o aéreo [Sugeno y Nguyen, 1994, Montgomery et al., 1995]. El problema del transvase de los conocimientos adquiridos en estas áreas a otros campos es el encarecimiento del producto a desarrollar. En la actualidad empiezan a aparecer sistemas autónomos en la realización de tareas y servicios que implican riesgo, dureza o en tareas muy repetitivas, gracias a la reducción de coste de los sistemas sensoriales y de proceso y al apogeo de las tecnologías de la información y las comunicaciones. En la actualidad, estos sistemas autónomos se podrían incorporar con gran beneficio, en el campo de la minería, tanto a cielo abierto

como subterránea, en vigilancia forestal y medioambiental, para exploraciones en regiones heladas, desérticas o planetarias [Whittaker et al., 1998], [Stocker, 1998], en labores agrícolas, en horticultura [Ollis, 1997] o en jardinería [Mandow et al., 1996], [Noguchi y Terano, 1997], [Tillet et al., 1993].

La investigación en telerrobótica pretende reducir esfuerzos y riesgos haciendo que el operario sólo actúe en aquellas situaciones en las que es realmente imprescindible. Se trata de extender las capacidades de manipulación, percepción y actuación humana a lugares remotos, de difícil acceso o peligrosos, dotando al robot de autonomía a fin de reducir al mínimo la intervención humana [Trevelyan, 1997]. De hecho en muchas aplicaciones la teleoperación constituye la solución más apropiada y la labor de exploración, supervisión y reconocimiento seguirán en muchos ámbitos bajo la supervisión humana, incluso en los casos de guiado y control directo [Fong y Thorpe, 2001]. Sin embargo en la investigación dirigida al desarrollo de robots autónomos, la teleoperación constituye una etapa inicial, siendo posteriormente una herramienta de gran utilidad en las etapas de adquisición e interpretación de datos sensoriales. En cualquier caso, el desarrollo de un sistema de navegación tanto teleoperado como autónomo a partir de una plataforma comercial convencional requiere siempre una larga etapa de diseño e integración de los sistemas de actuación, percepción, proceso y comunicación. El gran volumen de información que hay que interpretar en tiempo real para ejecutar la actuación adecuada es el cuello de botella de estos sistemas, ya sean autónomos o teleoperados.

Con el objetivo de desarrollar una arquitectura de organización del conocimiento y control para sistemas móviles en tiempo real, de posible aplicación en laboreo agrícola, se ha procedido a la automatización en los talleres del Instituto de Automática Industrial del CSIC, de un tractor agrícola diseñado y comercializado por la empresa AGRIA HISPANIA, que denominaremos DÉDALO. La primera sección de este capítulo ofrece una descripción de la plataforma comercial y de las pautas a seguir en su automatización. La automatización de un vehículo implica en primer lugar la elección e instalación de unos **actuadores** que sustituyan a los mecanismos originales de control manual de la conducción del vehículo, que fueron diseñados para ser accionados por los brazos y piernas de un operario. La selección del tipo de actuadores

a integrar constituye la primera etapa en el proceso de automatización de cualquier vehículo y va a depender fundamentalmente de : 1) los objetivos propuestos, 2) las condiciones del entorno y 3) las características del vehículo. De este proceso de selección, diseño e instalación del sistema de actuación trata la sección 4.2 de este capítulo.

La siguiente consideración para incrementar la autonomía del sistema es la selección de una dotación sensorial que le permita conocer su estado interno y el estado del entorno, con el fin de cerrar los lazos de control de la conducción. En la elección de los sensores se tienen en cuenta los mismos parámetros que en la elección de los actuadores, si bien aquí habría que considerar la complejidad de interpretación de la información suministrada por los sensores. La descripción de los sensores instalados a bordo del tractor ocupa la sección 4.3. El siguiente paso consiste en la elección de un sistema de proceso y almacenamiento de la información, así como de un sistema de comunicación con el operario, soportado por un buen interfaz hombre-máquina. En la sección 4.4 se describe el sistema de proceso y comunicación del robot. La última sección, sección 4.5, detalla el sistema de control borroso propuesto para el guiado automático del vehículo y la experimentación realizada con varios controladores en lazo cerrado, ante la inadecuación de un control en lazo abierto.

4.1 Tractor DÉDALO

La automatización se ha llevado a cabo en su totalidad en las instalaciones del IAI-CSIC, en el campus de Arganda del Rey. El vehículo es un tractor agrícola comercial Modelo 9900, diseñado y comercializado por la empresa española AGRIA-HISPANIA S.A. (Amorebieta-Vizcaya), figura 4.1. Es un tractor articulado de tamaño pequeño con capacidad para desempeñar labores en campos de cultivo mediante el arrastre de aperos, transporte de remolques y transmisión del movimiento a máquinas auxiliares como pueden ser trilladoras, desbrozadoras, empaquetadoras o bombas de riego [Arias-Paz, 2000].

El tractor pesa 1200 (kg.), lleva un motor diésel de 3 cilindros, dirección hidráulica integral y 12 velocidades, 8 hacia adelante y 4 hacia atrás, en el intervalo de $1{,}3$ (km/h) a 25 (km/h). Las

Figura 4.1 : Tractor AGRIA HISPANIA modelo 9900: DÉDALO

dimensiones se muestran en la figura 4.2 y se detallan en la tabla 4.1.

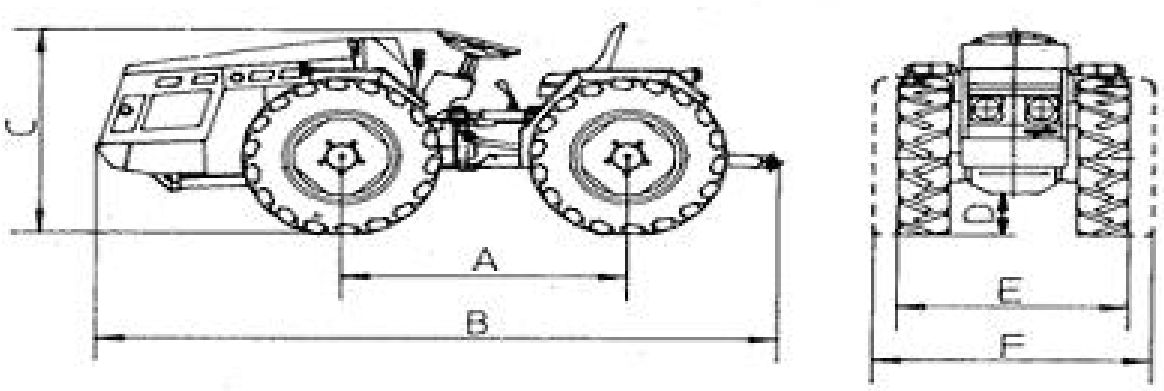


Figura 4.2 : Tractor DÉDALO

Este modelo de tractor posee una dirección asistida integral, y la presión del fluido que circula en el circuito hidráulico de dirección produce el movimiento de la articulación de giro. Este mecanismo de articulación central es sencillo y ocupa un espacio reducido, por ello el

Característica	Medida (mm)
A	1130
B	3168
C	1110
D	250
E	918
F(max.)	1020

Tabla 4.1 : Dimensiones del tractor DÉDALO

tractor es más estrecho y gira con mayor facilidad entre líneas de cultivo estrechas.

La automatización se ha focalizado en la dirección y en los los pedales de freno y embrague. Así pues, una vez que se ha seleccionado manualmente una marcha, el tractor DÉDALO se mueve a velocidad constante, en el caso de que ambos pedales estén sueltos, o bien permanece parado, si están presionados.

4.2 Sistema de actuación automático

Los sistemas de actuación ponen en movimiento el vehículo a partir de las acciones del conductor o de las órdenes del sistema de control. Los sistemas de actuación automática más difundidos en robótica utilizan fundamentalmente tres tipos de energía: neumática, hidráulica o eléctrica. El comportamiento de los actuadores es crítico en lo que se refiere a su velocidad de movimiento y potencia, pues condicionan el funcionamiento observable del robot. Por ello es conveniente analizar las características de los tres tipos de actuadores ante un determinado sistema y aplicación. Entre las características más relevantes se encuentran: relación coste/potencia y peso/volumen, velocidad, precisión, posibilidad de control continuo y finalmente facilidad de mantenimiento.

Las diferentes características de los sistemas de actuación condicionan el funcionamiento de cada uno de ellos y su ámbito de aplicación. Los **motores eléctricos** son muy utilizados por su facilidad de control, alimentación, limpieza, instalación y nivel de ruido [Barrientos et al., 1997]; sin embargo su principal inconveniente radica en disponer de una potencia muy limitada para un coste medio. De ahí que se utilicen mayormente para

aplicaciones de interiores en robótica móvil, donde los tamaños y pesos de los robots son reducidos.

En el otro extremo, en lo que a potencia se refiere, se sitúan los **actuadores hidráulicos**, con una excelente relación potencia-peso y una gran capacidad de carga, que los hacen adecuados para su integración en grandes máquinas como son tractores, cosechadoras o excavadoras. Estos actuadores funcionan con aceites minerales a gran presión (entre 50 y varios cientos de bares), y, debido a su baja compresibilidad, la precisión que pueden alcanzar es alta y por lo tanto resulta relativamente sencillo realizar un control continuo de los mismos. Además, las elevadas presiones de trabajo permiten desarrollar grandes fuerzas y soportar cargas sin ningún aporte extra de energía. Ahora bien, estos sistemas requieren unos conocimientos muy específicos tanto para su instalación como para su mantenimiento, además de equipos especiales para: filtrado de partículas, eliminación de aire, refrigeración y control de distribución. Queda añadir, que el mantenimiento de los actuadores hidráulicos exige mayor dedicación pues la alta presión genera con facilidad fugas en las uniones de los circuitos [Barrientos et al., 1997].

Entre ambas alternativas se sitúan los **actuadores neumáticos**, de funcionamiento similar a los hidráulicos pero que, en lugar de aceites minerales, funcionan con aire a presión (entre 5 y 10 bares). Debido a la compresibilidad del aire los actuadores neumáticos no consiguen una buena precisión de posicionamiento, sin embargo su sencillez y robustez los hacen especialmente adecuados para sistemas donde únicamente se necesita posicionar el eje en dos localizaciones como en manipuladores sencillos para la apertura y cierre de pinzas [Barrientos et al., 1997]. Al igual que los actuadores hidráulicos, también necesitan de una instalación y mantenimiento específico para disponer del aire comprimido, pero las fugas de aire no ensucian el entorno. Su coste es asequible, poseen una precisión y fuerza moderadas [Rorabaugh, 1995], pero introducen una gran contaminación acústica. Su velocidad de respuesta, en comparación con los actuadores hidráulicos, es mucho mayor, sin embargo su principal inconveniente, junto con la poca precisión en el posicionamiento, está en la dificultad de realizar un control continuo de ellos [García-Pérez et al., 2000a].

4.2.1 Actuadores electrohidráulicos

La elección de la actuación hidráulica en la automatización del tractor DÉDALO es evidente porque pueden reutilizarse los recursos de fábrica del tractor, en concreto la bomba hidráulica activada por el motor de combustión. La existencia de esta bomba facilita la integración de nuevos circuitos dotados de cilindros y electroválvulas. Acoplados al pedal del freno y a la varilla del embrague se han añadido dos cilindros hidráulicos con el objetivo de lograr un posicionamiento en dos estados: presionado y suelto. En el caso de la dirección no ha sido necesario instalar un nuevo cilindro hidráulico, ya que la dirección asistida es hidráulica de fábrica, y para automatizarla se ha instalado únicamente un circuito hidráulico adicional que por medio de una llave pasa de control manual de la dirección a control automático, figura 4.3. En el caso del embrague y del freno la llave inhabilita el modo automático, cerrando el paso del aceite a los circuitos adicionales, pero el modo manual siempre está activo, permitiendo al conductor presionar el pedal de embrague y freno en todo momento.

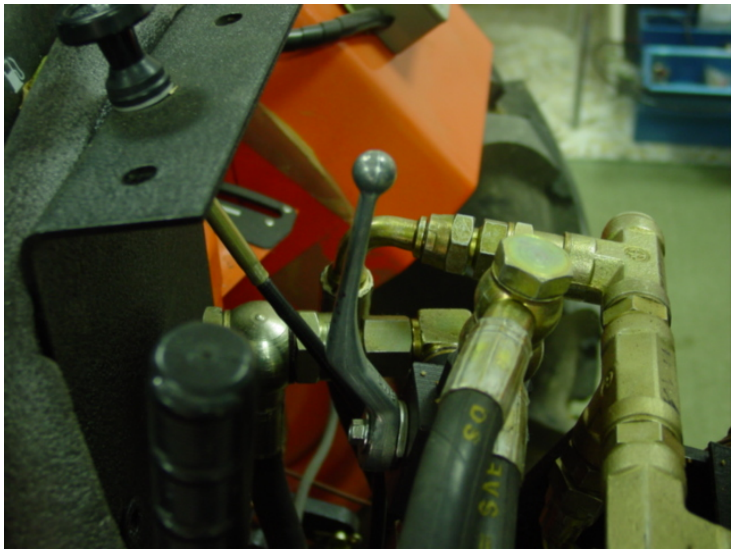


Figura 4.3 : Llave de paso de control manual a automático de dirección

El esquema del circuito hidráulico de actuación sobre la dirección, se muestra en la figura 4.4

El vástago entra o sale del cilindro produciendo un giro en la dirección hacia la derecha

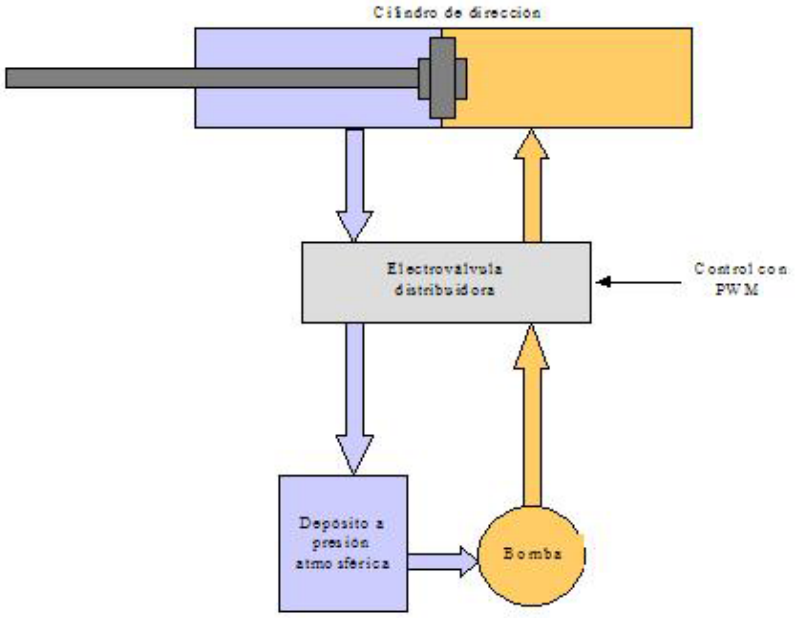


Figura 4.4 : Esquema del circuito hidráulico de dirección

o izquierda respectivamente, en función de qué cámara se encuentre a presión atmosférica del depósito. Para determinar el sentido de circulación del fluido se ha integrado una electroválvula hidráulica distribuidora de tres posiciones y cinco vías, figura 4.5. Este dispositivo se controla mediante la excitación de dos bobinas que generan el campo magnético necesario para establecer la configuración de apertura/cierre en válvula que fija el sentido de circulación del aceite.

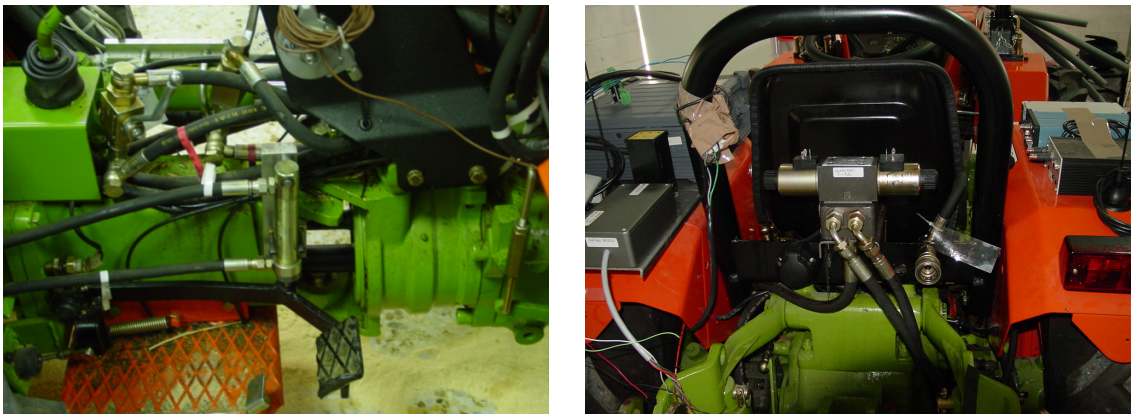


Figura 4.5 : Cilindro hidráulico acoplado al pedal de freno (izquierda) y electroválvula (derecha)

Electroválvulas hidráulicas

La bomba impulsa el fluido del depósito de aceite y, dependiendo de la configuración de la electroválvula establecida con el circuito de excitación de las bobinas el aceite, éste entra en una de las dos cámaras del cilindro. El fluido entrante presiona el vástago y lo desplaza, produciendo un giro en la dirección, saliendo a la vez por la otra cámara, hacia el tanque, aceite a presión atmosférica. Dependiendo del estado de excitación de cada una de las dos bobinas existen tres modos de funcionamiento del circuito hidráulico: giro hacia la izquierda, giro hacia la derecha y ausencia de giro.

- **Giro a la izquierda.** El giro de la dirección hacia la izquierda se consigue excitando una de las bobinas, B1 y dejando en reposo la otra B2, figura 4.6. De esta manera el aceite entra a presión en la cámara C1 del cilindro y se produce el retorno de aceite a baja presión desde la cámara C2 al depósito. Consecuentemente el vástago entra en el cilindro y la dirección se mueve hacia la izquierda.

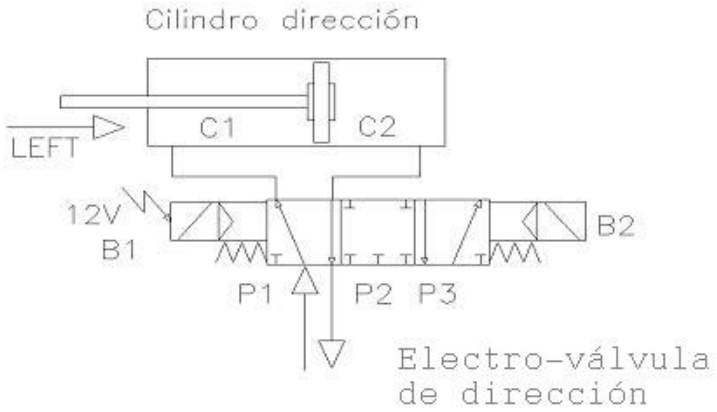


Figura 4.6 : Actuación del sistema electro-hidráulico en el giro hacia la izquierda

- **Giro a la derecha.** El giro hacia la derecha se consigue con la configuración opuesta al giro a la izquierda, es decir excitando la bobina B2 y dejando en reposo B1, figura 4.7. En este

caso, el aceite a presión entra en la cámara C2 y sale a baja presión desde C1 hacia el tanque. El vástago sale del cilindro y la dirección se mueve hacia la derecha.

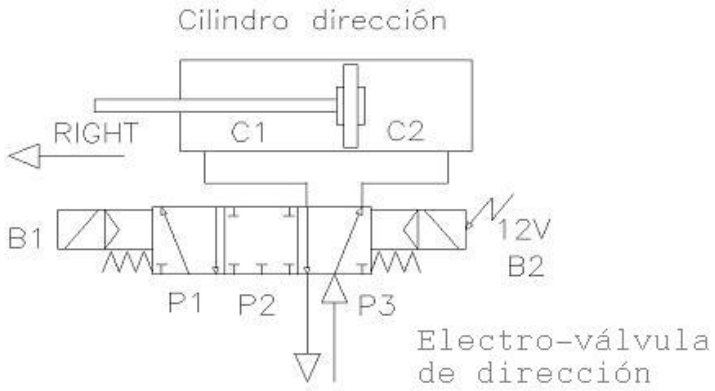


Figura 4.7 : Actuación del sistema electro-hidráulico en el giro hacia la derecha

- **Ausencia de giro.** Si no se excitan ninguna de las dos bobinas, las vías de la electroválvula permanecen cerradas (posición en centros cerrados) y el aceite no entra en ninguna de las dos cámaras, permaneciendo el vástago y la dirección en la misma posición, figura 4.8.

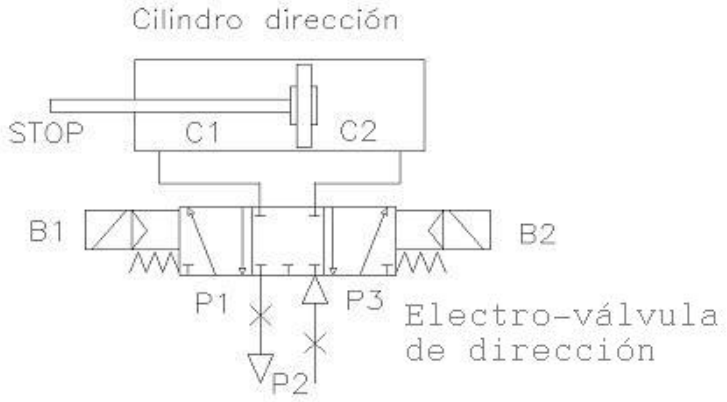


Figura 4.8 : Actuación del sistema electro-hidráulico en ausencia de giro

Al ser la válvula un dispositivo “todo/nada”, no es posible un control proporcional directo de la posición del pistón, esto es, un posicionamiento continuo; sólo sería posible un posicionamiento del pistón en los extremos del mismo. Normalmente, ésta es la funcionalidad

que se espera de una válvula todo/nada y es la que se requiere en la automatización del sistema de actuación del freno y del embrague con dos estados Marcha o Parada. Sin embargo no ocurre así en la automatización de la dirección de DÉDALO, donde es necesario un posicionamiento continuo del cilindro que se corresponda con un determinado ángulo de giro de la dirección. Ahora bien, es posible conseguir un posicionamiento continuo del pistón en el cilindro hidráulico mediante la apertura de la válvula en intervalos variables de tiempo, sin necesidad de disponer de una válvula proporcional. De ahí que el control de la actuación gradual sobre la dirección se lleve a cabo mediante una modulación en anchura de pulso PWM (Pulse Width Modulation). Por ello se aplica un voltaje constante a la electroválvula durante intervalos de tiempo variables logrando así controlar el periodo en el que la válvula está abierta y que el flujo entre en una de las cámaras del cilindro, figura 4.9. La variable PWM se mide en tanto por cien del ancho total del ciclo, de duración total T_a (4.1).

$$PWM = \frac{T_b}{T_a} \times 100 \tag{4.1}$$

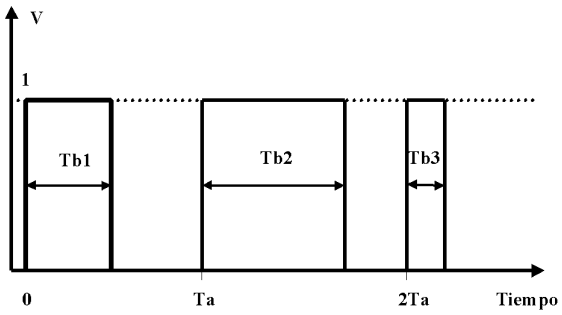


Figura 4.9 : Modulación en anchura de pulso (PWM)

Con estas modificaciones, ya es posible iniciar una conducción automática del tractor a velocidad y marcha fijas; que se puede experimentar de forma inmediata con un control en lazo abierto de la dirección. Ahora bien, para conseguir un comportamiento inteligente y una navegación autónoma y segura es necesario disponer de información del estado del sistema/entorno, siendo indispensable la instalación de dispositivos sensoriales y de proceso a bordo del tractor.

Figura 4.10 : Tractor Agria Hispania 9900 (al llegar de fabrica) y robot DÉDALO (en la actualidad)

4.3 Sistemas sensoriales

La optimización del control del movimiento para conseguir un objetivo de posicionamiento del vehículo, se realiza mediante un control en lazo cerrado o realimentado que necesita disponer de información relativa al sistema de actuación. Teniendo en cuenta las características de los actuadores seleccionados, es necesario incorporar un sensor que proporcione en cada instante la posición del émbolo en cada uno de los cilindros, a fin de determinar el estado de giro y de parada o marcha del vehículo. Igualmente, a fin de disponer de un conocimiento aproximado de su localización relativa, se precisa otro dispositivo sensorial. En ambos casos se trata de sensores propioceptivos que indican la evolución temporal del sistema.

En entornos dinámicos es imprescindible disponer de sensores que nos proporcionen información del conjunto sistema-entorno, sensores exteroceptivos. Al igual que con los actuadores, la selección de los sensores va a depender tanto del tipo de vehículo y entorno de trabajo, como de los objetivos que se pretende abordar. Así, en este caso, al tratarse de vehículos en exteriores, hay que considerar que el robot debe recorrer distancias muy superiores a las de

un entorno de interior. Además, en exteriores las condiciones ambientales muestran un alto grado de variabilidad en: la intensidad luminosa, la resistencia e irregularidades del terreno y en la presencia de pequeños obstáculos (piedras o cascotes). Esta variabilidad incrementa la dificultad del proceso de análisis e interpretación de las señales sensoriales, por lo que se requiere un sistema sensorial fiable y robusto, a prueba de condiciones adversas en la intemperie [Everett, 1995].

En cualquier caso algunas de las consideraciones generales a evaluar en el proceso de selección de los sensores son las siguientes [Dudek y Jenkin, 2000]:

- *Frecuencia de muestreo*

  Cada sensor proporciona las medidas con una frecuencia máxima, tanto en funcionamiento continuo como bajo petición. Por ello es fundamental verificar este límite, ante los objetivos que se persiguen. Así, un sistema de detección de obstáculos con una resolución temporal de 2 (s), no sería de utilidad si el robot se mueve a una velocidad de $10(m/s)$; pues entre cada dos medidas sensoriales consecutivas recorrería ciego una distancia de $20(m)$ pudiendo chocar con cualquier obstáculo imprevisto.

- *Precisión*

  Cada sensor lleva asociado un error nominal en las medidas, que conviene verificar experimentalmente.

- *Robustez*

  Capacidad de hacer frente a condiciones adversas del entorno alejadas de las condiciones ideales de medida, como pueden ser: nivel de ruido, polvo, suciedad, vibraciones.

- *Preproceso de la señal*

  El procesado inicial de la señal, como puede ser un filtrado, facilita su posterior interpretación, teniendo en cuenta que los recursos computacionales del robot son limitados. Por ello es importante tener en cuenta el tiempo de proceso necesario para extraer información relevante de los datos crudos del sensor.

- *Coste, consumo, dimensiones y peso*

  Los sensores deben tener un consumo energético, dimensiones y peso adecuados a las características del vehículo, siendo conveniente que todos sean lo más reducidos posible. El coste es un parámetro adicional que hay que considerar en cuanto al valor añadido final del vehículo.

Teniendo en cuenta las características del tractor DÉDALO, del entorno de trabajo y de las tareas a realizar, la relevancia de las propiedades enumeradas es la siguiente:

- Los vehículos agrícolas son generalmente lentos, por lo tanto la velocidad de muestreo de los sensores no es un punto crítico en su selección.

- La mayoría de las tareas a realizar no requieren una precisión alta, a excepción de la localización; y esta última siempre va a estar sujeta a los errores del controlador, a las holguras mecánicas y a las dimensiones del vehículo.

- Sin embargo los entornos de trabajo en exterior sí implican una exposición a niveles altos de polvo, variaciones de la luminosidad y vibraciones, por ello los sensores seleccionados deben estar bien acondicionados y dar una respuesta fiable a pesar de las bruscas variaciones ambientales.

- Las restricciones de dimensión, peso y localización de los sensores vienen impuestos por el tamaño de los vehículos, ($3{,}2 \times 1{,}0$(m.)) en el tractor DÉDALO.

- Respecto al consumo, el tractor DÉDALO cuenta con un motor diésel y una batería de arranque de 12 (volt.).

La forma más extendida de clasificar los sensores en robótica se fundamenta en la detección del estado interno del robot o del entorno. Los primeros, se denominan sensores internos o *propioceptivos*. El término propioceptivo se usa en biología y psicología para designar la percepción de los estímulos procedentes del interior del propio cuerpo. En sistemas robóticos

los sensores utilizados para medir la intensidad de la batería o la posición de un actuador son claros ejemplos de sensores propioceptivos. Los sensores dedicados a la percepción del entorno se conocen como sensores externos o exteroceptivos, término empleado en biología y psicología para designar la percepción de estímulos procedentes del exterior. Se ha remarcado el subsistema sensorial de localización formado en DÉDALO por un sensor propioceptivo: el odómetro y dos exteroceptivos: DGPS y brújula. Dos razones fundamentales argumentan esta subdivisión. En primer lugar, para la navegación de un robot móvil, es fundamental conocer su posición en el mundo, e imprescindible en aquellos casos en que hay que alcanzar una localización expresada en coordenadas, ya sean absolutas o relativas; de ahí que este tipo de sensores merezcan una atención especial. En segundo lugar, porque el problema de la localización en exteriores es complejo y será necesario dotar al robot de varios sensores de posición que actúen de forma cooperativa.

Las técnicas de situación de robots móviles pueden dividirse en dos grupos, en función del grado de acondicionamiento de la señal utilizada para la localización del vehículo:

1. Medidas directas proporcionadas por sensores como la brújula o los odómetros.

2. Medidas indirectas del sistema-entorno, como localización sobre un mapa o posicionamiento relativo a un conjunto de balizas en tierra o satélites.

El sistema odométrico de determinación de la posición es redundante y complementario al sistema constituido por la brújula y el receptor DGPS. Redundante, porque los odómetros permiten estimar la posición instantánea del robot (x, y, θ) respecto de un sistema de coordenadas relativo al robot que puede transformarse directamente en coordenadas absolutas. La brújula y el receptor DGPS proporcionan directamente las coordenadas absolutas del vehículo. Complementario, porque tanto el método de obtención de las medidas como su frecuencia son muy diferentes [Pozo-Ruz, 2001].

A continuación se presentan los sensores que se han seleccionado e instalado en el tractor DÉDALO.

4.3.1 Sensores propioceptivos

En el caso del tractor DÉDALO las variables de estado interno del sistema son cinco: nivel de energía, posición de la dirección, posición del embrague, posición del freno y distancia recorrida por las ruedas. El sensor del nivel de carga de las baterías es fundamental, pues de la cantidad de energía disponible va a depender el funcionamiento de gran parte de los sensores. El tractor se conduce manipulando con brazos y piernas tres elementos: el pedal de freno, el del embrague y el volante para lograr el giro. En el pedal de freno y en el del embrague interesa distinguir sólo dos estados:

1. pisado a fondo: el vehículo está completamente parado
2. suelto: el vehículo se mueve a velocidad constante

- **Potenciómetros** Para la detección de los estados del embrague (presionado a fondo/suelto) y freno (presionado a fondo/suelto) se han incorporado sendos potenciómetros acoplados a los vástagos de los respectivos cilindros. Los potenciómetros son resistencias variables de alta calidad, cuyo valor de resistencia varía linealmente con la posición del vástago. Incorporando un circuito divisor de tensión a la salida, se obtiene un voltaje proporcional a la posición del vástago.

El ángulo de giro de las ruedas es el que determina la curvatura de la trayectoria del robot, por ello se ha acoplado otro potenciómetro al cilindro del circuito de control de la dirección, de modo que, cuando el pistón se mueve, arrastra el vástago del sensor, midiéndose la posición en que éste se encuentra.

En la figura 4.11 se muestra la colocación del potenciómetro con el cilindro de dirección en el tractor DÉDALO. La relación entre la medida del sensor y la dirección del tractor es compleja, puesto que se trata de un vehículo compuesto por dos cuerpos que giran en torno a una articulación central, manteniendo fijas las ruedas. El hecho de que ambas partes giren simultáneamente obliga a establecer un sistema de medida de ángulos de giro relativos a los dos cuerpos, y no relativos a una referencia fija, figura 4.12.

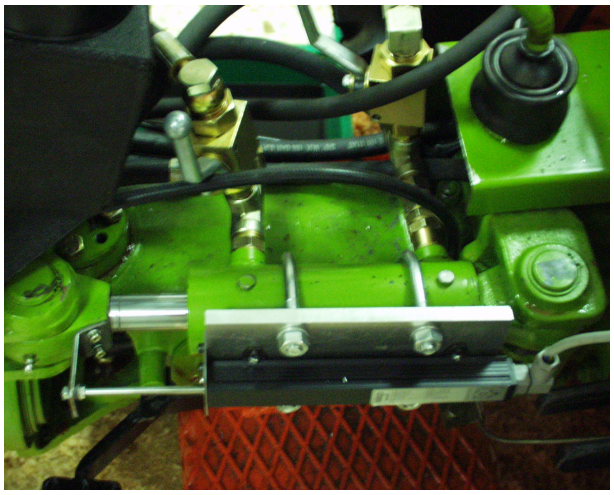


Figura 4.11 : Potenciómetro y cilindro de dirección

Figura 4.12 : Giros en el tractor

El criterio de ángulos de giro elegido trata de la medida del ángulo relativo entre los dos cuerpos del tractor. Para ello se han definido dos sistemas de referencia solidarios con cada una de las partes del tractor. Por simplicidad y comodidad en la experimentación los ejes X e Y se definen en el plano de la rueda delantera y trasera respectivamente, como se ve en la figura 4.13. Se consideran negativos los ángulos correspondientes a giros hacia la izquierda y positivos hacia la derecha. El ángulo de dirección elegido es el suplementario del que forman los ejes X e Y en el plano de las ruedas, figura 4.13.

En la figura 4.14 se muestra la relación obtenida experimentalmente entre el voltaje de salida del potenciómetro, alimentado a 5 (volt.) y el ángulo de giro del tractor α_D, medido

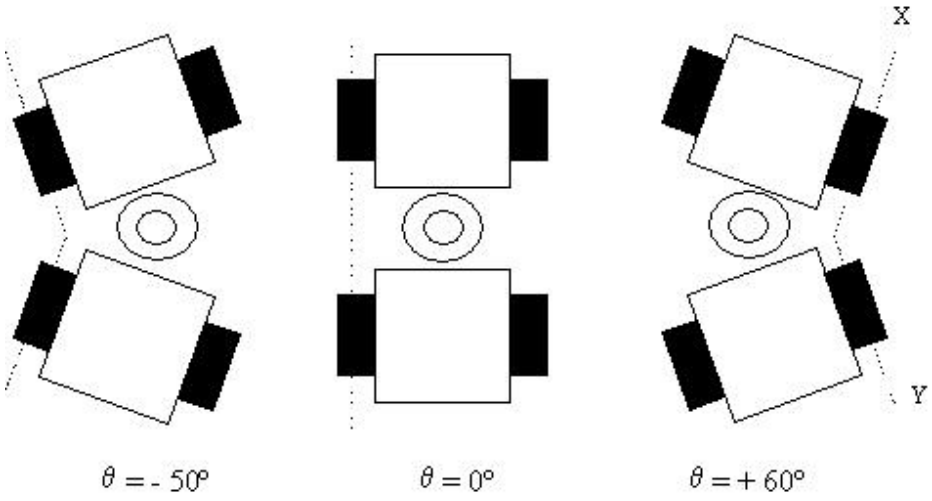


Figura 4.13 : Criterio de calculo del sentido de giro en el tractor DÉDALO

manualmente sobre el suelo.

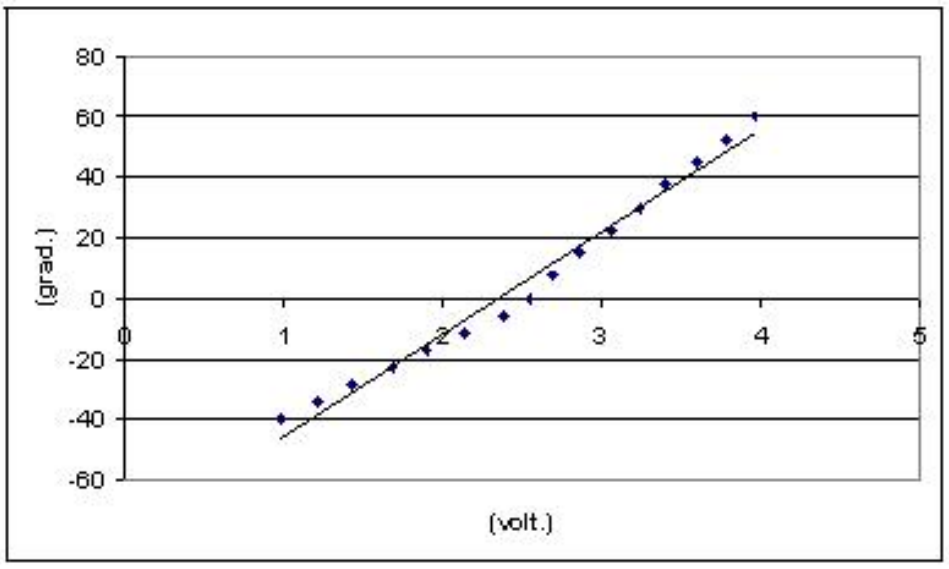


Figura 4.14 : Función de calibración del ángulo de giro (grados) vs. voltaje del potenciómetro (voltios)

Los datos experimentales se ajustan a una línea recta $\alpha_D(V) = 34 \cdot V_d - 80$ con un coeficiente de regresión de $R^2 = 0{,}9836$. Puesto que el giro no es simétrico ($\alpha_D \in [-50, 60]$ (grad.)) la relación podría haberse aproximado mejor mediante dos rectas en vez de una sola. Sin embargo este ajuste resulta irrelevante frente a la imprecisión inherente a la medida experimental del

ángulo.

Sensores odométricos

Los sensores odométricos determinan la localización del vehículo a partir de la medida del desplazamiento sobre el camino recorrido [Everett, 1995]. Para un vehículo que se desplaza con ruedas, el cálculo de la distancia recorrida se puede determinar por la velocidad angular de las ruedas, que a su vez puede obtenerse mediante potenciómetros y codificadores angulares ya sean ópticos, magnéticos, inductivos o capacitivos. Los codificadores ópticos son los que se han utilizado con mayor frecuencia, sobre todo para robots de interiores, sin embargo para aplicaciones en exteriores con polvo, tierra y lluvia, no resultan los más adecuados. En exteriores resulta más apropiado la utilización de detectores de perturbaciones locales de campo magnético por ferro-eléctricos o cerámicas imantadas.

El sistema odométrico de DÉDALO tiene que ser robusto frente al entorno, de ahí que se optase por un disco de tornillos y un sensor capacitivo detector de metales en un intervalo de distancia corto, figura 4.15. Sobre el disco, solidario a cada rueda delantera, se han distribuido 80 tornillos equidistantes entre sí. El sensor capacitivo, fijo al eje de la rueda, detecta el paso de los tornillos, y alimentado con 5 (volt.) devuelve un valor de salida de 0 (volt.) en ausencia de tornillo y de 5 (volt.) si detecta la cabeza del tornillo.

El sensor capacitivo proporciona por tanto, una medida indirecta de la velocidad angular, y a través de ella es posible calcular la distancia recorrida por cada una de las ruedas del vehículo mediante las ecuaciones (4.2)

$$\begin{aligned} \Delta d_{t,R/L} &= N_{t,R/L} \frac{\pi \times d_{R/L}}{n_{R/L}} \\ \Delta s_t &= \frac{\Delta d_{t,L} + \Delta d_{t,R}}{2} \\ \Delta \theta_t &= \frac{\Delta d_{t,R} - \Delta d_{t,R}}{b} \end{aligned} \tag{4.2}$$

Figura 4.15 : Odómetro diseñado para el tractor DÉDALO

Donde $\Delta d_{i,R/L}$ es la distancia recorrida por la rueda derecha o izquierda, $n_{R/L}$ la resolución del odómetro, $d_{R/L}$ el diámetro de la rueda derecha/izquierda, $N_{R/L}$ el número de pasos de odómetro contados en el instante t, b la distancia entre los puntos de apoyo de las ruedas derecha e izquierda y Δs_t y $\Delta\theta_t$ los incrementos de desplazamiento y de ángulo. A partir de las expresiones (4.3) se pueden estimar las coordenadas (x, y, θ) del robot [Borenstein y Feng, 1998].

$$\begin{aligned} \theta_t &= \theta_{t-1} + \Delta\theta \\ x_t &= x_{t-1} + \Delta s_t \cdot \cos\theta_t \\ y_t &= y_{t-1} + \Delta s_t \cdot \sin\theta_t \end{aligned} \tag{4.3}$$

Frente a las ventajas de bajo coste y aparente simplicidad de un sistema odométrico, surge el inconveniente de alta imprecisión en la medida, ya que el error es acumulativo con la distancia recorrida y no se puede utilizar para recorridos mayores de 10 ó 30 metros, dependiendo del número de giros realizados. Además, la navegación en exteriores implica un movimiento sobre terrenos muy distintos, con tramos asfaltados, de tierra batida o en terrenos pedregosos que pueden provocar deslizamientos sin giro o rodadura sin traslación aumentando considerablemente los errores no sistemáticos en la localización relativa del vehículo [Pozo-Ruz, 2001]. Estos sistemas de localización relativa se suelen utilizar en cooperación con otro tipo de sensores de situación o bien mediante ajustes periódicos de la posición con un sistema de balizas (terrestres o aéreas) localizadas con precisión.

4.3.2 Sensores exteroceptivos

La capacidad del robot para detectar el entorno que le rodea es fundamental para la realización de tareas complejas en interacción con un entorno dinámico. Es pues imprescindible que el robot disponga de sensores para la detección de ciertas propiedades de su entorno. En tareas de navegación y reconocimiento de las estructuras presentes en el medio, es necesario representar el conjunto de objetos en la dirección de su movimiento, actualizándolo en tiempo real, para tomar decisiones de guiado localmente. Con este fin, se han instalado en el tractor DÉDALO: 1) un sensor de contacto en el parachoques frontal, 2) un láser de barrido, y 3) dos sensores de inclinación, para detectar desniveles con riesgo de la integridad del vehículo. Además se han integrado dos sensores exteroceptivos pertenecientes al subsistema de localización de DÉDALO: un sensor GPS y una brújula digital.

Receptor GPS diferencial

El sistema GPS es un método de localización global mediante balizas aéreas. Fue desarrollado en la década de los noventa por el Departamento de Defensa de los Estados Unidos con propósitos militares. El sistema de balizas esta formado por un conjunto de satélites

que orbitan en torno a la Tierra con un radio de 26000 (km), emitiendo señales de radio en dos frecuencias de portadora: una de ellas $L1 = 154 \times f_0$ para todos los usuarios y la otra $L2 = 120 \times f_0$ para uso militar exclusivamente ($f_0 = 10{,}23$ (MHz)) [Kaplan, 1996]. En tierra se dispone de un conjunto de receptores que detectan estas señales y a partir de ellas calculan su posición, mediante técnicas de correlación entre el código pseudoaleatorio recibido y el del propio receptor, para calcular el desplazamiento temporal y por lo tanto a la distancia a la que se encuentra el emisor, [Hofmann-Wellenhof et al., 1992]. Una de las ventajas más importantes de este sistema de localización es que se trata de un sistema puramente pasivo y no requiere ninguna modificación del entorno. Sin embargo no se puede utilizar ni en interiores ni bajo tierra o agua. En núcleos urbanos densamente edificados y con edificios de gran altura o en entornos boscosos muy frondosos, la señal GPS no está siempre disponible y si lo está su precisión se ve seriamente reducida [Dudek y Jenkin, 2000].

Los sistemas de localización "GPS diferencial" (DGPS) mejoran la precisión combinando al menos dos receptores GPS que se comunican entre sí, siendo uno de ellos de posición conocida. El posicionamiento diferencial tiene como objetivo minimizar los errores presentes en la señal GPS [Pozo-Ruz, 2001]. Existen tres formas de posicionamiento diferencial DGPS:

1. Usando correcciones locales mediante una estación base cercana al receptor. Esta base genera las correcciones y las transmite al receptor. Esto implica disponer de una base propia y por ello, aunque se trata de la solución más precisa, es también la más cara y sólo es valida para una zona próxima a la base.

2. Mediante un modelo de corrección global que se envía al receptor por un canal de datos vía satélite. Éste es el caso del sistema Omnistar. Este tipo de corrección exige el pago de una suscripción anual para poder recibir la señal de corrección.

3. Utilizando una base que envía los datos de corrección vía radio, gratuitamente, como es el sistema Rasant. La corrección obtenida con este sistema es la de menor coste, pero su calidad es función de la distancia del receptor a la base.

En el mercado existe una amplia variedad de sistemas DGPS con diferentes precisiones y precios [Pozo-Ruz, 2001] para elegir en función de los requerimientos de cada aplicación concreta. En el contexto de la robótica móvil en exteriores, el uso de sensores DGPS junto con otro tipo de sensores de localización relativo, como por ejemplo el odométrico, constituye la elección más adecuada para disponer de un posicionamiento continuo en la navegación, dentro de los límites impuestos por la imprecisión inherente a los sensores de localización seleccionados [Everett, 1995, Dudek y Jenkin, 2000].

En el caso del tractor DÉDALO se ha optado por incorporar el receptor 3100LR12, con correcciones diferenciales del sistema Rasant, figura 4.16. El error en las medidas con el sistema Rasant es muy similar al obtenido con el sistema Omnistar en la zona de Arganda del Rey, con la ventaja de que las correcciones Rasant se envían gratuitamente por radio en FM.

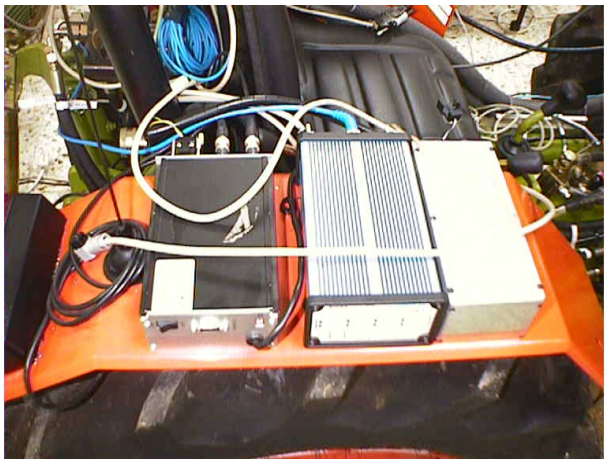


Figura 4.16 : Receptor GPS 3100LR12 más receptor de corrección diferencial Rasant FM

Este receptor, figura 4.16, posee un total de 12 canales y ofrece un posicionamiento con un error inferior a un metro en el $95\,\%$ de los casos e inferior a $0{,}5(m)$ en el $68\,\%$ de los casos.

El receptor DGPS se comunica con el sistema de proceso a bordo a través de un puerto serie utilizando el protocolo de comunicación NMEA 0183. En este protocolo los mensajes comienzan por el carácter \$ y terminan con $\backslash r \backslash n$. Se ha utilizado la trama de mensaje GGA, que incluye posición, tiempo e información relativa al receptor, tabla 4.2. Únicamente hay que configurar el receptor para que reciba las correcciones vía conexión serie desde un dispositivo

externo[1]. La estructura y contenidos completos del mensaje, es la siguiente [Ashtech, 1996]:

```
$GPGGA,1m,m2,c1,m3,c2,d1,d2,f1,f2,M,f3,M,f4,d3*cc
```

Campo	Descripción	Rango
m1	Tiempo UTC: horas, minutos y segundos (hhmmss.ss)	00-235959.50
m2	Latitud en grados y décimas de minuto (ddmm.mmmmm)	0-90
c1	Dirección de la latitud (N:norte, S:sur)	'N'/'S'
m3	Longitud en grados y décimas de minuto (ddmm.mmmmm)	0-180
c2	Dirección de la longitud (E:este, O:oeste)	'E'/'W'
d1	Modo de posición (1:absoluto, 2:diferencial)	1,2
d2	Número de satélites activos	0-12
f1	Dilución de la posición horizontal (HDOP)	0-99.9
f2	Altitud en metros sobre el elipsoide de referencia	-30000.00 a +30000.00
M	Unidad de medida de la altitud de la antena (M metros)	'M'
f3	Separación geoidal en metros	$\pm 999{,}99$
M	Unidad de medida de la separación geoidal (M: metros)	'M'
d3	Edad en segundos de las correcciones diferenciales	0-999
d4	Identificación de la estación de referencia	0-1023
*cc	Checksum	

Tabla 4.2 : Campos del mensaje en la trama GGA

En la trama GGA, el GPS proporciona la longitud y latitud del receptor en grados. Mediante un cambio de coordenadas [Pozo-Ruz, 2001] estos valores se transforman en coordenadas UTM expresadas en metros, obteniéndose así la posición del robot en el plano (x, y). Sin embargo, la orientación se calcula de forma indirecta, y sólo cuando el robot se mueve, a partir del cálculo de la pendiente de la recta que une dos localizaciones consecutivas. Esta medida sólo se obtiene en movimiento y al tratarse de un valor obtenido por diferencia, el error es alto. Cada una de las posiciones usadas para calcular la orientación, P1 y P2 en la figura 4.17 llevan asociadas unas zonas de incertidumbre debido al error en la determinación de la posición GPS. Al calcular el ángulo θ_{GPS} de la recta que une ambos puntos, el abanico de ángulos posibles

[1]Esta configuración es problemática puesto que el puerto por el que el receptor tiene que recibir las correcciones Rasant es el mismo que recibe las consignas de configuración. Por esta razón en el momento en que se le indica desde el PC que va a recibir las correcciones, en protocolo RTCM en lugar de NMEA, ya no es capaz de "entender" los comandos que se le envían desde el PC y por lo tanto no hay manera de indicarle que salve la configuración en la memoria interna. Durante la sesión funciona perfectamente, pero como no se ha guardado la configuración hay que reconfigurarlo cada vez que se enciende el receptor GPS.

es el de todas las rectas que unen cada uno de los puntos dentro de la elipse asociada a P1 con cada uno de los puntos de la asociada a P2. Usando técnicas de propagación de errores [Spiegel y Abellanas, 1988] el error absoluto asociado al ángulo de orientación calculado por las posiciones GPS viene expresado por (4.4).

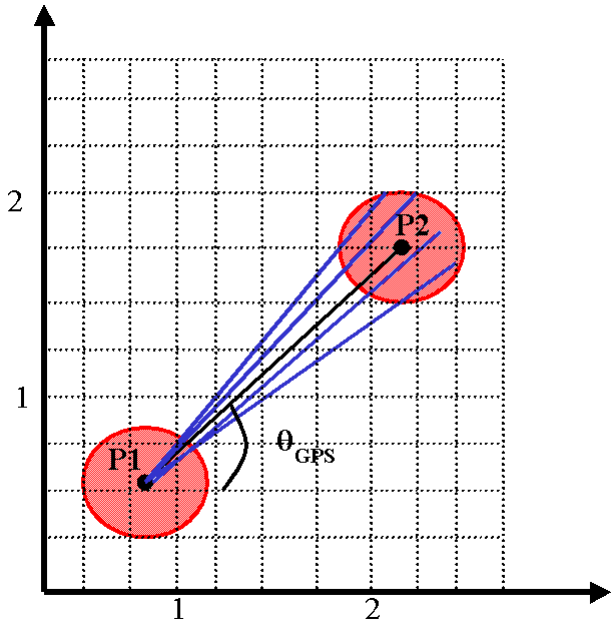


Figura 4.17 : Error en el cálculo de la orientación con localización GPS

$$\Delta\theta_{GPS} = 4 \times |\Delta x| \times \frac{|(x_2 - x_1) - (y_2 - y_1)|}{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{4.4}$$

Brújula digital

Las brújulas magnéticas proporcionan una estimación de la orientación con respecto al norte magnético de la Tierra. A fin de mejorar el error en la medida de la orientación obtenida a través del receptor GPS, se ha instalado a bordo de DÉDALO una brújula de medida de flujo (KVH Industries, modelo C100). Las brújulas de medida de flujo se basan en el concepto de *permeabilidad magnética* μ, $B = \mu \times H$. En el caso de que un material con elevada permeabilidad magnética (buen conductor de las líneas de fuerza magnética) se introduzca en un campo magnético uniforme, y dependiendo de si el material está o no saturado, se pueden

producir dos fenómenos distintos: (1) Si el material no está saturado, las líneas de flujo del campo externo se concentran dentro del material, figura 4.18. (2) Si por el contrario, el material está saturado, el campo magnético externo no se ve afectado por la presencia del material. El funcionamiento de una brújula de medida de flujo se basa en este comportamiento. Básicamente consta de un material de alta permeabilidad magnética con dos bobinas arrolladas en torno a él, una bobina para aplicar una corriente que lleve el núcleo a la saturación y otra bobina encargada de detectar, de acuerdo con la ley de inducción de Faraday, la corriente eléctrica inducida. La magnitud de estos pulsos de corriente inducida varía con la fuerza del campo magnético externo y su orientación con respecto del eje del núcleo y de la bobina detectora.

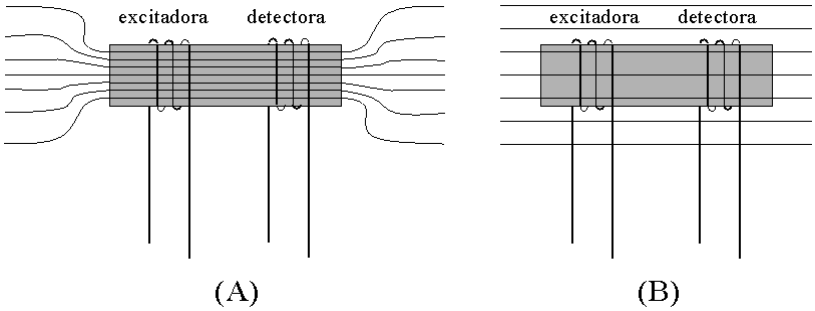


Figura 4.18 : Lineas de flujo magnético externo en los casos de un núcleo no saturado (A) y saturado (B), [Everett, 1995]

La brújula seleccionada posee las características de versatilidad y coste medio. Incluye un sensor toroidal de dos ejes controlado por un microprocesador implementado en la misma placa [Everett, 1995], figura 4.19. La resolución de esta brújula es de $\pm 0{,}10$ grados con una precisión de $\pm 0{,}5$ grados y repetibilidad de $\pm 0{,}2$ grados[2]. El microcontrolador se comunica de forma bidireccional por puerto serie con el procesador a bordo, mediante el protocolo NMEA, enviando datos y recibiendo mensajes de configuración [KVH, 2001]. La brújula puede programarse para envíos a diferentes frecuencias: 600, 60 ó 6 tramas por minuto. La estructura de la trama enviada por la brújula es del tipo:

```
$HCHDM,d,M*cc\r\n
```

[2]Hay que analizar detenidamente la posición de la brújula a bordo del tractor, alejándola de fuentes de interferencia magnética como pueden ser las bobinas de las electroválvulas.

donde *d* es la orientación de la brújula con respecto al norte magnético de la Tierra medido en grados y décimas de grado `(ddd.d)`; *M* corresponde a la medida con respecto al norte magnético y *cc* corresponde al código de redundancia cíclica.

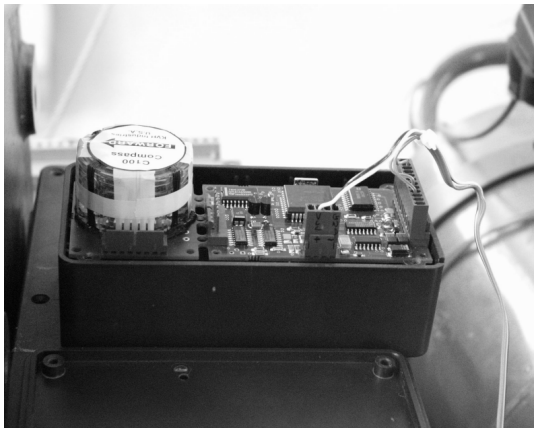


Figura 4.19 : Brújula digital C100.

Sensor de contacto

El sensor de contacto implementa en el robot el "sentido del tacto". Reacciona ante el contacto directo con un objeto, y su intervalo de seguridad es distancia cero. El sensor de choque instalado en DÉDALO está protegido por un parachoques de material flexible, figura 4.21, que incorpora en su interior una fibra óptica que se curva ante una colisión, produciendo una variación del tiempo asociado al recorrido del estado de no colisión. El sensor lleva incorporado un emisor de IR ($930(nm)$) tipo LED. El circuito detector, que consta de un transductor foto-diodo y un amplificador asociado, recibe la luz de la fibra óptica y responde a los cambios que se producen en la señal óptica cuando se aplica una presión sobre la fibra, figura 4.20 enviando una señal a la unidad de control, figura 4.21. Cuando la presión sobre el parachoques cesa, la fibra óptica recobra su forma original y la unidad de control cierra el interruptor [Hergalite, 1996]. Este sensor es inmune a condiciones medioambientales adversas con mucho polvo o lluvia, detectando colisión a lo largo de todo el parachoques.

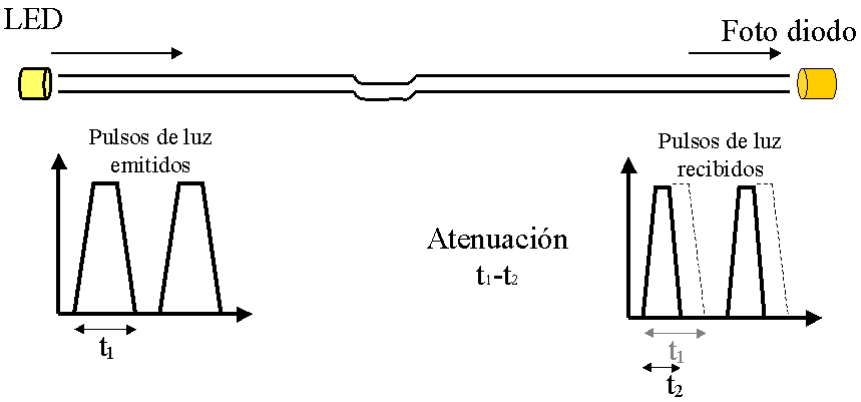


Figura 4.20 : Esquema de funcionamiento del sensor de fibra óptica, [Hergalite, 1996]

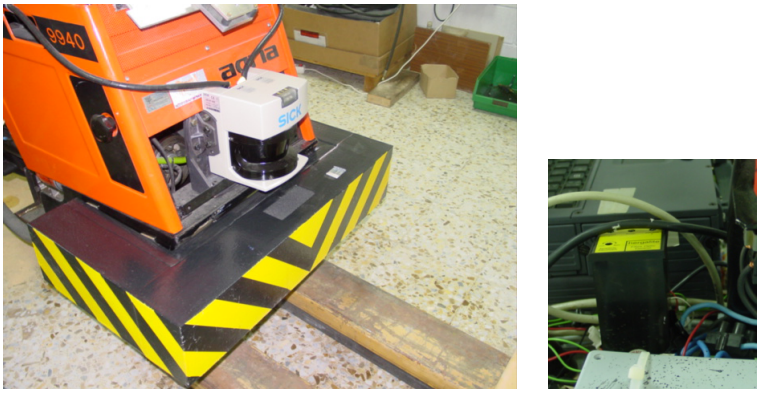


Figura 4.21 : Sensor para detección de colisión y unidad de control.

Sensor láser de barrido

En el tractor DÉDALO se ha integrado un sensor de barrido láser encapsulado herméticamente, que proporciona información de los obstáculos situados en un plano en un intervalo de distancias de 1 (cm) a 30 (m) (SICK, modelo LMS291), figura 4.22. El principio de funcionamiento se basa en la medida del tiempo de vuelo del haz de luz emitido por el sensor y reflejado por cada objeto situado en su campo de detección. El tiempo transcurrido desde que el sensor envía el pulso láser hasta que el receptor lo recibe es proporcional a la distancia entre el sensor y el objeto. El haz de luz láser es reflejado internamente mediante un espejo giratorio, para barrer 180 (grados) con una resolución de 0,5 (grados). El tiempo total de barrido es 26 (ms) y su alcance máximo de 30 (m.) con una resolución de 10 (mm.) y precisión

típica de 50 (mm.). Las medidas proporcionadas por este sensor permiten construir un mapa de obstáculos a una determinada altura del suelo. Los datos medidos en tiempo real se transmiten, vía puerto serie, a una velocidad de 9600 (baudios). Con esta velocidad de transmisión y dado que la trama de medida consta de 559 bytes, con resolución de 0,5 (grados), se obtienen los datos a una frecuencia de 2 (Hz.). El formato de las tramas en la comunicación entre el láser y el ordenador a bordo es el siguiente:

`STX|ADR|LEN|CHD|Data|Status|CRC`

STX es un campo de comienzo de 8 bits, que en los mensajes al ordenador toma el valor `02H`. El campo `ADR`, de 8 bits corresponde a la dirección asignada al sensor ya que se puede atender a varios sensores desde un mismo ordenador. `LEN` indica en 16 bits el número de bytes de datos enviados en el campo `Data`. `CHD` es el byte identificativo del comando enviado, y cuando la trama se dirige al ordenador se completa con la dirección asignada al láser. Cuando se trata de un mensaje con las medidas efectuadas por el láser, el campo `Data` contiene dichas medidas cada una en 2 bytes. Los 8 bytes del campo `Status` registran el estado del sensor (OK, niebla, fallo, etc...) y `CRC` representa el código de redundancia cíclica.

Cuando es el ordenador el que envía un mensaje al sensor láser, éste responde con el siguiente mensaje de reconocimiento: `ACK`.

Figura 4.22 : Láser de barrido SICK modelo LMS291

Sensores de inclinación

Teniendo en cuenta la morfología del terreno por el que va a navegar el tractor DÉDALO se consideró necesario integrar un par de sensores de inclinación que permitiesen medir tanto la inclinación frontal como la lateral del tractor. Con este fin se seleccionaron los sensores de inclinación capacitivos con dieléctrico líquido, SEIKA modelo N3, figura 4.23, que actúan como un condensador cuyo dieléctrico es un fluido con una burbuja de aire. Al inclinarse el sensor la burbuja se desplaza y la capacidad del condensador varía, pues depende de la constante dieléctrica del material entre placas. El sensor incluye un circuito eléctrico integrado que transforma la medida en un voltaje proporcional al ángulo de inclinación del sensor.

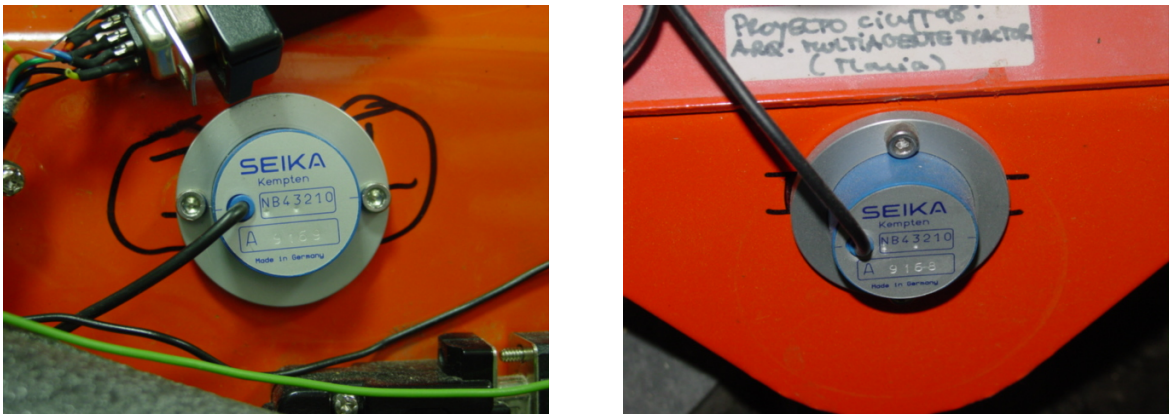


Figura 4.23 : Inclinómetros SEIKA integrados en el tractor DÉDALO: frontal a la derecha y lateral a la izquierda

El rango de medida del sensor N3 es $(-30, +30)$ (grados), con resolución $< 0{,}005$ (grados), y sensibilidad 6 (mvolt.)/(grado). Las ecuaciones (4.5) muestran la calibración de cada uno de los sensores. Donde ζ_l, ζ_f representan las pendientes lateral y frontal y V_l, V_f los respectivos voltajes de salida de los sensores de inclinación.

$$\zeta_l = 175{,}438 \cdot V_l - 432{,}280$$
$$\zeta_f = 172{,}419 \cdot V_f - 427{,}414 \qquad (4.5)$$

A partir de este momento se dispone de un tractor con actuadores y sensores adecuados a las características del sistema, del entorno de trabajo y al tipo de tareas que se pretende abordar. Sin embargo sin un sistema de cómputo capaz de adquirir y procesar las señales de los sensores y de ejecutar el control sobre los actuadores, el vehículo no podrá aún realizar ninguna tarea de modo autónomo.

4.4 Sistema de proceso: ordenador embarcado

El sistema de proceso a bordo del tractor DÉDALO es un ordenador portátil industrial Kontron con un procesador Pentium III a 233 MHz y una ampliación externa de bus PCI, necesaria para conectar la tarjeta de adquisición de datos (PCI-1711). Dispone además de dos ranuras PCMCIA donde se han integrado una tarjeta de red inalámbrica WaveLan y una tarjeta de ampliación de puerto serie (de 2 a 4).

La tarjeta WaveLan permite la comunicación con cualquier nodo de la red de área local inalámbrica del IAI-CSIC y, a través de ésta, con la LAN de cable del IAI-CSIC y con Internet. Actúa del mismo modo que si el sistema de proceso a bordo del tractor estuviera conectado a un punto de la red vía par trenzado. La tarjeta de red inalámbrica verifica el estándar IEEE 802.11 para redes inalámbricas, similar al Ethernet convencional, que define para la transmisión sin hilos los niveles físico y MAC del estándar OSI [Technologies, 1998]. Las características de la tarjeta de red se muestran en la tabla 4.3.

Banda de frecuencias	2.4 (GHz) (2400-2500 (MHz))
Protocolo de acceso al medio	CSMA/CA con ACK
Velocidad de transmisión	2 (Mbps)
Potencia de salida	15 dBm
Estándar	IEEE 802.11 (DSSS)

Tabla 4.3 : Características técnicas de la tarjeta de red inalámbrica

De esta forma cada robot se configura como un nodo de proceso accesible desde cualquier punto de la red interna a través de la configuración de red adoptada. La ventaja de tener en red el sistema de proceso embarcado es ofrecer una comunicación bidireccional con él desde

cualquier puesto de trabajo. Además, al estar el procesador accesible desde cualquier punto de trabajo hace posible el desarrollo de una arquitectura basada en el paradigma cliente-servidor que permita una comunicación flexible y cómoda a los distintos desarrolladores.

4.4.1 Sistema de Comunicación: Paradigma Cliente-Servidor

En el paradigma cliente-servidor se define al nodo cliente como aquel que envía el primer paquete de información; y como nodo servidor, al que permaneciendo a la escucha recibe ese primer paquete. Durante el desarrollo del proceso de comunicación los papeles de emisor y receptor se van intercambiando: tanto el servidor como los clientes pueden enviar y recibir mensajes. Cada uno de ellos se caracteriza por el papel que representa al iniciar el proceso de comunicación, figura 4.24. Las aplicaciones cliente-servidor implican el envío secuencial de los mensajes.

Se ha desarrollado un programa, denominado DÉDALO-Servidor, que reside en el procesador a bordo del tractor. Los distintos programas que actúan como cliente, se comunican con él mediante un conjunto de mensajes con una sintaxis predefinida. Este tipo de comunicación permite que servidor y clientes puedan desarrollarse independientemente siempre y cuando sigan el protocolo de comunicación preestablecido. Puesto que el sistema operativo a bordo del tractor DÉDALO es Windows 98, el interfaz de comunicación en red está basado en el estándar para Windows Sockets [Quinn y Shute, 1996], y todas las aplicaciones han sido desarrolladas en C++ Builder de Borland. La implementación de los programas bajo el paradigma cliente-servidor favorece el modelado de agentes de la arquitectura de control propuesta en esta tesis.

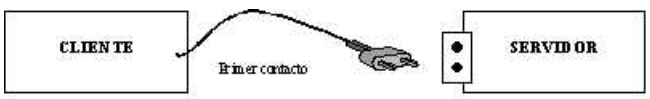


Figura 4.24 : Paradigma de comunicaciones cliente-servidor de la aplicación DÉDALO-Servidor

4.4.2 Aplicación Servidor: DÉDALO-Servidor

El programa DÉDALO-Servidor cumple dos objetivos primordiales:

1. Adquirir las señales sensoriales para enviarlas a los clientes que las soliciten.
2. Ejecutar el ciclo de control básico a nivel de actuadores físicos.

De acuerdo con estos dos objetivos, las competencias del programa DÉDALO-Servidor se resumen en los siguientes puntos:

- Lectura de los voltajes analógicos de cada sensor, proporcionados por la tarjeta A/D (PCI-1711), y preprocesamiento de la información.
- Lectura de las señales de los sensores GPS, brújula y láser por los puertos serie.
- Atención a los mensajes de los clientes.
- Interpretación de los mensajes recibidos.
- Envío a cada cliente de los datos solicitados.
- Realización del ciclo de control del giro y del freno-embrague del robot.
- Control de errores.

El interfaz DÉDALO-Servidor se muestra en la figura 4.25.

Figura 4.25 : Interfaz de la aplicación DÉDALO-Servidor

La comunicación entre DÉDALO-Servidor y los clientes se realiza mediante paso de mensajes predefinidos. El proceso es el siguiente: DÉDALO-Servidor siempre se encuentra a la escucha, cuando un cliente desea establecer conexión con él lo solicita y una vez aceptada la conexión le envía un mensaje de `OK` y permanece abierto el canal de comunicación. Cuando el cliente solicita el envío de algún dato sensorial, manda un mensaje de petición de datos a DÉDALO-Servidor, se registra y comienza el envío de los datos. En cualquier instante el cliente puede solicitar la interrupción en el envío de datos, y una vez recibido ese mensaje se cancela la transmisión de datos al cliente. La sintaxis y semántica asociada a los mensajes se muestra en la tabla 4.4.

Mensajes de control	Solicitud de **envío** de datos		00 + código de la señal
	Solicitud de **interrupción** en el envío de datos		01 + código de la señal
	Control	Volante	80 + 12 + ángulo-volante
		Desactivación	80 + 13 + on/off
Mensajes de percepción	Datos sensoriales	Válvulas	02
		Odometría	03
		DGPS	04
		Inclinómetros	05
		Láser	06
		Brújula	07
		Nivel de batería	08
		Sensor de colisión	09
		Voltajes	11
		Ángulo de las ruedas	12
		Posición freno	13

Tabla 4.4 : Sintaxis y semántica de los mensajes

Los mensajes enviados desde el programa DÉDALO-Servidor tienen todos la siguiente estructura:

1. código que identifica el tipo de señal

2. tiempo

3. medidas propias de cada una de las señales.

En la tabla 4.5 se recogen todas las tramas de estos mensajes.

Válvulas	02 + t + posición-dirección + posición-pedal
Odometría	03 + t + x_{odo} + y_{odo} + θ_{odo} + distancia
DGPS	04 + t + lon + lat + alt + num. sat. + modo
Inclinómetros	05 + t + ζ_f + ζ_l
Láser	06 + t + d0 + d1 + ... + d360
Brújula	07 + t + $\theta_{brujula}$
Batería	08 + t + nivel-bateria
Choques	09 + t + colisión
Voltajes	11 + t + V_1 + V_2 + ... + V_n
Giro Ruedas	12 + t + α_{ruedas}
Pedal Marcha	13 + t + on/off

Tabla 4.5 : Detalle de las tramas de los mensajes

4.4.3 Adquisición de las señales sensoriales

El programa DÉDALO-Servidor realiza un ciclo continuo de lectura de las señales sensoriales, que se ha implementado en cuatro hebras de programación independientes: 1) de lectura de la tarjeta de adquisición de datos, 2) lectura del puerto serie del GPS, 3) lectura del puerto serie de la brújula y 4) lectura del puerto serie del láser. Todas están implementadas como bucles infinitos de lectura, y cada una de ellas tiene asignado un tiempo de “espera” en función de la frecuencia de lectura requerida. Al comienzo y fin de cada bucle se registra el reloj, con precisión de centésimas de segundo, a fin de calcular el tiempo consumido en la ejecución de la tarea asignada a la hebra. La hebra se mantiene “dormida” un periodo de tiempo que es igual al tiempo de espera asignado menos el de ejecución de la tarea.

La hebra de lectura de la tarjeta de adquisición de señales tiene un tiempo de espera de 60 (ms), que se determina, en este caso, en función de las necesidades del sensor más restrictivo: la odometría. Considerando que la velocidad del tractor es de aproximadamente 2(km/h), que

el perímetro de la rueda delantera es de 2765 (mm) y que cada rueda tiene 80 tornillos, resulta que cada pulso leído corresponde a un recorrido de $2765/80 = 34{,}6$ (mm). A 2 (km/h) ($0{,}56$ (m/s)) el tiempo que transcurre entre dos tornillos consecutivos es pues de 60 (ms), esto es la frecuencia mínima a la que se debe leer el odómetro para no perder ninguna lectura.

En el caso de las hebras de lectura del GPS y de la brújula, el tiempo de espera es de 200 (ms) puesto que la frecuencia de ambos sensores es de 5 (Hz.). La frecuencia del láser es de 2 (Hz.) y por lo tanto el tiempo de espera de su hebra es 500 (ms).

DÉDALO-Servidor también se encarga de implementar el control de los actuadores.

4.5 Controladores de los actuadores

En el tractor DÉDALO se han integrado tres circuitos hidráulicos con sendos cilindros en sustitución de los mecanismos de guiado manual, para su navegación desasistida mediante la actuación automática sobre la dirección, el freno y el embrague, que será conjunta para estos dos últimos.

Las válvulas son todo/nada y la variable que determina la acción de control, en cada uno de los circuitos, es el intervalo de tiempo que la electroválvula se mantiene abierta. Este intervalo se regula mediante una modulación en anchura de pulso, esto es, mediante un pulso de voltaje constante (5 (volt.)) pero de anchura variable t.

El control de los actuadores del freno y del embrague en DÉDALO son inmediatos, puesto que únicamente se persigue el posicionamiento en ambos extremos. Sin embargo, el sistema de actuación de la dirección requiere un algoritmo de control complejo ya que es necesario situar el eje de giro de modo continuo a lo largo de todo su recorrido. Un control en lazo abierto de este sistema de actuación resulta inviable debido a las perturbaciones introducidas por las irregularidades y tipo de terrenos (surcos, piedras, desniveles, pendientes, ...), así como por la incertidumbre asociada a los sensores, actuadores y mecanismos de adaptación. De ahí que se requiera un control realimentado capaz de alcanzar y mantener un ángulo de referencia en la dirección. Los métodos tradicionales de control basados en un modelo del sistema, parten de

una descripción analítica mediante un conjunto de ecuaciones integro-diferenciales, precisa y completa del sistema o planta a controlar. A partir de estas ecuaciones, y teniendo en cuenta los requerimientos específicos de funcionamiento del sistema, se diseña el controlador más apropiado [Ogata, 1990].

El sistema automático de dirección del tractor DÉDALO, es difícil de modelar en un marco analítico, pues tanto los dispositivos hidráulicos como los eléctricos (las electroválvulas) que componen el sistema de actuación, presentan inercias y holguras e interacción entre las diferentes partes. El hecho de no poder establecer un modelo analítico preciso de este sistema, cuyo comportamiento es no-lineal, dificulta el uso de técnicas de control clásico que aseguren la respuesta y estabilidad deseadas. Sin embargo, es posible encontrar un modelo cualitativo formado por un conjunto de heurísticos que reflejan el control manual de un conductor experto, integrando en conjuntos borrosos la imprecisión e incertidumbre inherentes al sistema, para conseguir los objetivos de guiado a partir del conocimiento "a priori" de los subsistemas que lo componen y de una experimentación adecuada. Por ello, se propone para el tractor DÉDALO un sistema de control híbrido, figura 4.26, compuesto por un controlador borroso proporcional y un controlador por muestreo cuya misión es mejorar el comportamiento del primero en una región del espacio de estados, la correspondiente a ángulos pequeños [Gutiérrez, 2003]

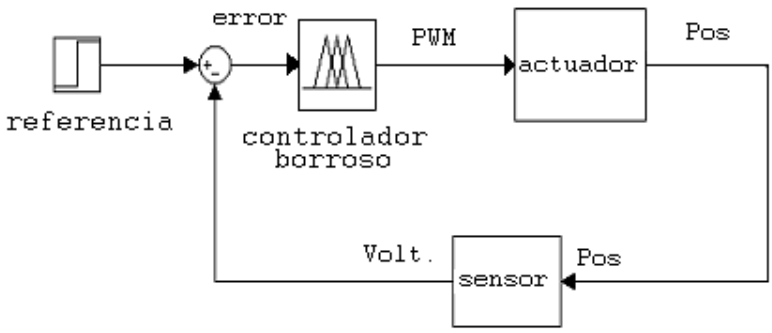


Figura 4.26 : Esquema del controlador del sistema de dirección

4.5.1 Modelo cualitativo de control

Las bases iniciales de la Teoría de Conjuntos Borrosos fueron formuladas por L.A. Zadeh en 1965, y han experimentado desde entonces un amplio desarrollo teórico con aplicaciones en múltiples áreas del conocimiento. La teoría de Zadeh tiene por objetivo la búsqueda de modelos que permitan implementar los mecanismos de razonamiento lingüístico y cuantificar el lenguaje natural. Sin embargo, hasta 1973 Zadeh no presenta la teoría básica para el diseño de Controladores Borrosos [Zadeh, 1973]. En ella se persigue plasmar la experiencia humana de control, clasificación o toma de decisión en un modelo compuesto por un conjunto de heurísticos computacionalmente plausibles y cuantificables. En 1974 Mamdani y colaboradores desarrollan la primera aplicación de un sistema de control basado en algoritmos borrosos para regular una máquina de vapor en el laboratorio, [Mamdani, 1974, Mamdani y Assilian, 1975]. El modelo de control borroso sigue el protocolo de control utilizado por un operario experto y se formula mediante un conjunto de reglas del tipo SI-ENTONCES. El *antecedente* de cada regla (parte SI) contiene una descripción del estado de la planta o sistema y el *consecuente* (parte ENTONCES) la acción de control apropiada.

La investigación y el desarrollo de aplicaciones reales progresan rápidamente [Mamdani, 1993]. Algunas de las aplicaciones más difundidas corresponden a los controladores para trenes del metro desarrollados en los años 80 en Japón [Terano et al., 1994] y los aplicados a electrodomésticos como lavadoras y a cámaras de vídeo [Hirota y (eds.), 1995, Jamshidi et al., 1997]. La mayoría de estos controladores borrosos están muy relacionados con el trabajo de Mamdani, ya que no disponían de un modelo de la planta sino de un conjunto de reglas de control descritas en lenguaje natural que operan sobre términos lingüísticos descritos mediante conjuntos borrosos y que constituyen lo que podemos denominar un sistema experto de control. El éxito y el continuo avance en popularidad de los controladores borroso en la resolución de múltiples problemas de la vida real se debe fundamentalmente al hecho de que la aproximación borrosa es muchas veces la única alternativa posible, ya que la teoría clásica de control es inviable cuando no se dispone de un modelo analítico preciso de la

planta. En la practica, la teoría de control clásico, para afrontar el problema de la complejidad en la descripción del sistema también propone aproximaciones lineales que constituyen una simplificación del sistema, con muy buenos resultados.

El hecho de no tener que construir un modelo clásico de la planta, plantea las siguientes cuestiones en el diseño y análisis de controladores borrosos [Margaliot y Langholz, 2000]:

1. Cómo diseñar sistemáticamente y justificar las reglas de control
2. Cómo definir sistemáticamente las funciones de pertenencia borrosa
3. Cómo analizar propiedades del control en lazo cerrado, como la estabilidad y la robustez

El primer punto es uno de los más conflictivos [Lee, 1990] y se suele resolver utilizando heurísticos basados en un buen conocimiento de la planta y de sus respuestas ante diferentes entradas (aproximación de sentido común). En el caso de que la planta o sistema se controle manualmente por un operario se puede hacer frente al problema desde dos puntos de vista:

- Extrayendo el conocimiento de control del experto en un formato de reglas lingüísticos.
- Almacenando un conjunto amplio de medidas en parejas, estado de la planta y acción de control, que permitan extraer las reglas borrosas.

Las aproximaciones utilizadas para resolver el segundo punto han conducido a la utilización de diferentes métodos para la optimización de los parámetros relativos a las funciones de pertenencia y el peso de cada regla, entre ellos se encuentran los algoritmos genéticos [Geyer-Schulz, 1998] y las redes neuronales [Berenji y Khedkar, 1992, Jang et al., 1997]. Con respecto al último punto, ausencia de un modelo de la planta, el análisis en lazo cerrado es prácticamente imposible excepto por una simulación extensiva. Esta problemática ha llevado al planteamiento de una nueva aproximación denominada *control borroso basado en modelo*. En ella, al igual que en teoría de control clásico, el controlador se diseña a partir de un modelo de la planta, formulado como un modelo borroso del tipo Takagi-Sugeno, que se obtiene bien a partir de una descripción heurística del comportamiento de la planta o bien a partir de un conjunto

de medidas de entradas/salidas de la planta [Takagi y Sugeno, 1985, Driankov y Palm, 1998]. Existe aún otra aproximación al diseño de controladores borrosos, donde el punto de partida vuelve a ser el modelo matemático de la planta descrito por un conjunto de reglas borrosas, pero en este caso se utilizan métodos clásicos, síntesis de Lyapunov [Wang, 1994] o el control predictivo [Babuska y Verbruggen, 1997], para diseñar el controlador más adecuado. Estas aproximaciones generan un controlador no borroso que ya no puede ser interpretado mediante un conjunto de reglas lingüísticas, perdiendo así una de las características fundamentales del mismo, su inteligibilidad por el operario.

De todas las consideraciones anteriores se deduce que el paradigma fundamental del control borroso está en sustituir el algoritmo analítico de control clásico por una base de reglas cuyas variables son términos lingüísticos modelados mediante conjuntos borrosos. La característica fundamental del modelo de control borroso frente a cualquier otro tipo de control clásico radica en la inexistencia de un modelo analítico rígido. Ahora bien, esta afirmación no implica ausencia de un modelo, ya que en la mayoría de los casos éste reside en la mente del operario experto que mediante reglas de naturaleza imprecisa o heurística toma decisiones con información imprecisa e incompleta obteniendo buenos resultados [Klir y Folger, 1988]. Se trata pues de transcribir, si es que existe, el conocimiento y el modo de razonar del operario a un modelo computacional que opera, en primera instancia, con términos del lenguaje natural en vez de valores numéricos. El modelo, que sí existe, se halla embebido y distribuido en el conjunto de relaciones entre los términos lingüísticos utilizados. En cada relación o regla se encapsula una toma de decisión sobre la clasificación o control del sistema en una región del espacio de estados.

En resumen, para analizar y controlar sistemas complejos, no lineales se necesitan *aproximaciones flexibles y modulables que no se basen en un formalismo matemático rígido, sino que por el contrario empleen un marco metodológico tolerante con la imprecisión y las verdades parciales* [Zadeh, 1973]. El objetivo del diseño de un sistema de control es la automatización de las acciones que realiza manualmente un operario para alcanzar o mantener un determinado estado, a través del análisis de un conjunto de variables. Por ello es necesario analizar bajo el paradigma de la Teoría de Conjuntos Borrosos, las variables borrosas del sistema

(etapa de conversión de variables analíticas a borrosas), la relación entre las mismas (reglas) y el modelo de toma de decisión global requerido para el control de un sistema (algoritmo de conversión de variables borrosas a valores numéricos).

Controladores PID borrosos

En Teoría de Sistemas, los controladores más utilizados en aplicaciones industriales son del tipo Proporcional-Integral-Derivativo (PID). Se caracterizan por tener una función de transferencia del tipo: (4.6), [Ogata, 1990].

$$G_C(s) = K_p \left[1 + \frac{1}{T_i s} + T_d s\right] \tag{4.6}$$

donde K_p es la ganancia proporcional, T_i es el tiempo integral y T_d el tiempo derivativo. Si en una representación espacio-temporal, la entrada al controlador es el error $e(t)$, la señal de salida de control correspondiente es (4.7).

$$u(t) = K_p \left[e(t) + \frac{1}{T_i}\int_{-\infty}^{t} e(t)dt + T_d \frac{de(t)}{dt}\right] \tag{4.7}$$

En consecuencia la acción de control de un controlador PID es función del error de la señal de referencia, de su variación en dos instantes consecutivos, y del error acumulado en el tiempo. El diseño de un controlador PID para una planta o sistema, tiene por objetivo determinar los valores de las constantes K_p, T_i y T_d, que permiten obtener la salida deseada en todo el espacio de estados del sistema.

Teniendo en cuenta estas consideraciones es posible mediante técnicas basadas en conjuntos borrosos implementar controladores PID, definiendo un conjunto de reglas que integren la relación existente entre la señal de control (o su variación) y la señales de error y de variación del error. Sin embargo los controladores borrosos de tipo PID no suelen aplicarse, debido fundamentalmente a las dificultades existentes tanto en la generación de una base de reglas eficiente, como en la etapa de sintonía de su elevado número de parámetros [Mudi y Pal, 1999].

En contrapartida cada vez se se extiende más el diseño y aplicación de controladores borrosos del tipo PI y PD [Mann et al., 1599].

- **Controladores borrosos PI.** Un controlador convencional tipo Proporcional-Integral PI se describe mediante la siguiente función temporal (4.8),

$$u(t) = K_p[e(t) + \frac{1}{T_i}\int_{-\infty}^{t} e(t)dt] \tag{4.8}$$

La acción de un controlador PI convencional tiene los siguientes efectos [de Chile, 2002]:

- Incrementa el orden de la respuesta
- No presenta *offset*
- La respuesta es más rápida pero con mayor amplitud en las oscilaciones

Un controlador borroso de tipo PI describe la dependencia de la variación de la señal de control $\Delta u(k) = u(k) - u(k-1)$, con el error $e(t)$ y los cambios del error en el tiempo $\Delta e(k) = e(k) - e(k-1)$ (4.9), [Yager y Filev, 1991].

$$\Delta u(k) = f(e(k), \Delta e(k)) \tag{4.9}$$

Para sistemas no lineales se ha demostrado que los controladores PI borrosos ofrecen mejores prestaciones que los convencionales, ya que estos últimos están diseñados exclusivamente para sistemas lineales. Este tipo de controladores combina la estabilidad inherente a los algoritmos de control proporcional, con una disminución en el error estacionario, propia de los controladores integrales. Sin embargo, para sistemas que incluyen componentes integrales o no lineales su funcionamiento es deficiente ya que genera amplias oscilaciones [Mudi y Pal, 1999].

- **Controladores borrosos PD.** Los controladores Proporcional Derivativo PD proporcionan una alta sensibilidad y contribuyen a incrementar la estabilidad del sistema realimentado.

Además reducen la sobreelongación y permiten el uso de valores altos de la ganancia, al amortiguar el sistema [Malki et al., 1994]. La ley de control implementada por un controlador analógico PD convencional, se define mediante la siguiente expresión (4.10),

$$u(t) = K_p \cdot e(t) + K_d \cdot \frac{de(t)}{dt} \tag{4.10}$$

donde K_p y K_d son las ganancias proporcional y derivativa respectivamente.

El controlador borroso de tipo PD calcula la señal de control en función del error y su incremento (4.11).

$$u(k) = f(e(k), \Delta e(k)) \tag{4.11}$$

La parte proporcional al incremento del error (derivada) da lugar a una reducción en la amplitud de las oscilaciones y produce un efecto de suavizado de la señal, que junto con la estabilidad ofrecida por la parte proporcional, confieren a la señal de salida unas propiedades muy deseables en múltiples aplicaciones de control.

4.5.2 Caracterización del sistema

El sistema hidráulico de dirección de DÉDALO consta de una electroválvula de tres posiciones y el cilindro hidráulico de la dirección. No todos los parámetros del sistema son conocidos: el sistema de dirección y el circuito hidráulico vienen integrados de fábrica en el tractor. Además tanto el subsistema hidráulico como el eléctrico presentan inercias y holguras que hacen difícil el modelado analítico preciso del sistema completo.

Las características fundamentales del funcionamiento del sistema de dirección controlado se formulan en relación a cinco criterios: estabilidad, precisión, sobre-amortiguamiento, rechazo de perturbaciones y rapidez. El primer criterio es la **estabilidad**, aunque ese criterio es exigible en todos los sistemas controlados, cobra especial importancia en ciertas aplicaciones, como ésta. En cuanto a la **precisión** se tolerarán errores en el giro inferiores a ± 5 (grados). A causa

de las dimensiones del vehículo, de que el envío de consignas de conducción es continuo y de que el movimiento es por campo, un error de 5 (grados) resulta inapreciable. Para evitar daños en el sistema es importante que la respuesta sea suave y por lo tanto el funcionamiento en lazo cerrado del sistema sea **sobre-amortiguado**. Otra de las virtudes ineludibles del control en esta aplicación es el **rechazo de perturbaciones**. El desplazamiento del tractor sobre un terreno agrícola implica una continua aparición de perturbaciones, el sistema tiene que ser capaz de funcionar con la presencia de perturbaciones. En cuanto al **tiempo de respuesta**, éste no es una especificación crítica para el funcionamiento del tractor, si bien se buscará la máxima velocidad posible.

Para caracterizar el funcionamiento del sistema se han realizado una serie de ensayos en lazo abierto. En la figura 4.27 se muestran las gráficas correspondientes a los dos ensayos realizados en lazo abierto, lanzando el sistema de dirección y dejándolo girar hacia la derecha y hacia la izquierda respectivamente. En las gráficas se representa la tensión que mide el potenciómetro en voltios frente al tiempo de apertura de la electroválvula en milisegundos.

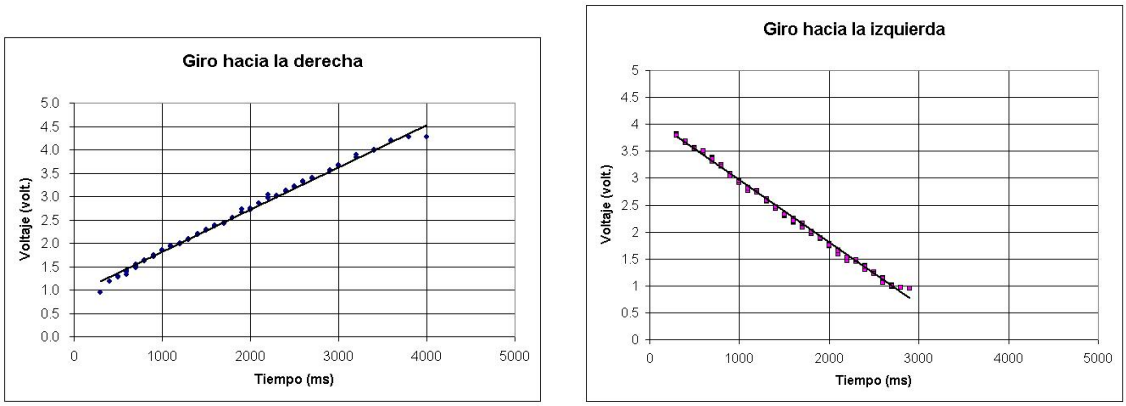


Figura 4.27 : Evolución en el tiempo del giro hacia la derecha y hacia la izquierda

El sistema parece distinguirse por una característica bastante lineal en los giros en ambos sentidos, con una peculiaridad en el arranque de la dirección: presenta una zona “muerta”. La dirección no se mueve hasta que no han transcurrido en torno a 300 (ms) de excitación. Todo indica que esta zona muerta puede ser debida a la inercia mecánica inicial del tractor. Dada la asimetría estructural del sistema, la respuesta para giros hacia la derecha y hacia la izquierda

son sensiblemente diferentes en cuanto a la pendiente de la recta. Por lo tanto, resumiendo:

- Aparecen dos sistemas dinámicos distintos a controlar, según que la dirección gire hacia la izquierda o hacia la derecha.
- Ambos parecen caracterizarse por ser lineales, ecuación (4.12).
- En los dos ensayos aparece una zona muerta en el arranque en la que el sistema se muestra insensible a la excitación, respondiendo posteriormente y manteniendo una evolución uniforme.
- El tiempo necesario para recorrer el giro completo de la dirección es de aproximadamente 4000 (ms) girando hacia la derecha, y de aproximadamente 3000 (ms) girando hacia la izquierda.

$$V_{pot_D} = 0{,}0009 \cdot t + 0{,}9093$$
$$V_{pot_I} = -0{,}0012 \cdot t + 4{,}1154 \tag{4.12}$$

4.5.3 Control borroso P en lazo cerrado

Partiendo del conocimiento adquirido sobre el sistema en los ensayos en lazo abierto, se propone un control borroso proporcional propiciado por la linealidad del sistema de dirección. La variable de entrada del controlador es el "error", calculado en cada ciclo de control como la diferencia entre el ángulo de giro de referencia y el giro real del sistema, y la variable de control-salida la anchura del pulso "PWM" dirigida al sistema, figura 4.28.

La duración del ciclo de control, se ha fijado en $T_a = 1000$ (ms), ya que la respuesta del actuador es lenta, pues a partir de la experimentación se extrae el valor de 3000 a 4000 (ms) en el recorrido de todo el cilindro, dependiendo de cuál sea el sentido. Esta característica unida a la existencia de una zona de respuesta muerta en el sistema, conduce a no seleccionar un tiempo

Figura 4.28 : Esquema del controlador proporcional de la dirección

de ciclo muy corto. Por otro lado el tiempo de ciclo no puede tomar un valor alto, pues en ese periodo el lazo de control estaría inactivo y por tanto “ciego” ante cualquier evento. El valor de 1000 (ms) se ha fijado como solución de compromiso entre los dos extremos considerados.

Variable de entrada.

La variable de entrada “error” se describe mediante 5 términos, definidos por funciones de pertenencia trapezoidales, figura 4.29, mediante las etiquetas lingüísticas N, n, C, p, P: negativo grande (N), negativo pequeño (n), cero (C), positivo pequeño (p) y positivo grande (P).

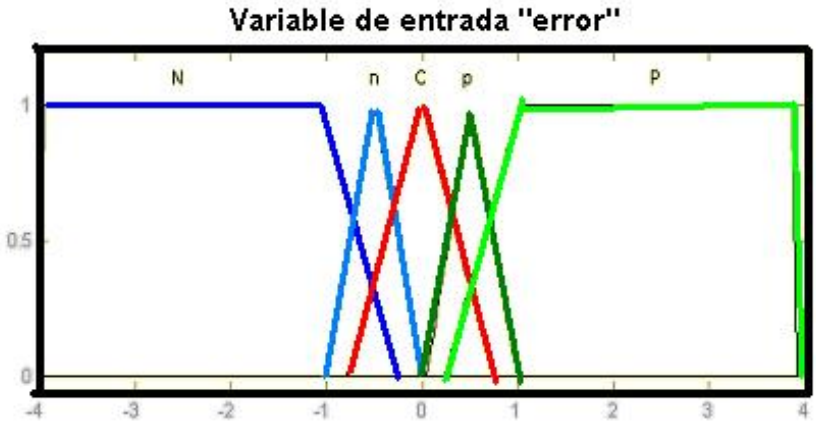


Figura 4.29 : Conjuntos borrosos de la variable de entrada error

La principal característica que se observa en esta definición es una resolución variable en el universo de discurso de la variable error, que puede observarse en la distinta anchura de los

conjuntos. El objetivo de esta propuesta es conseguir que el sistema responda adecuadamente ante errores pequeños, ya que cuando el error es grande basta con definir para la acción de control de la dirección su valor máximo. Sin embargo cuando el error es pequeño es necesario afinar mucho más el valor de la acción de control.

- **Variable de salida.**

La variable de salida “PWM” se calcula como el tanto por cien del tiempo total de ciclo (1000 (ms)) de pulso en alta, sección 4.1.

Los conjuntos borrosos representativos de la variable PWM se muestran en la figura 4.30. Las cinco etiquetas $I, i, C, , d, D$ se corresponden con los términos lingüísticos: muy izquierda (I), izquierda (i), cero (C), derecha (d), muy derecha (D).

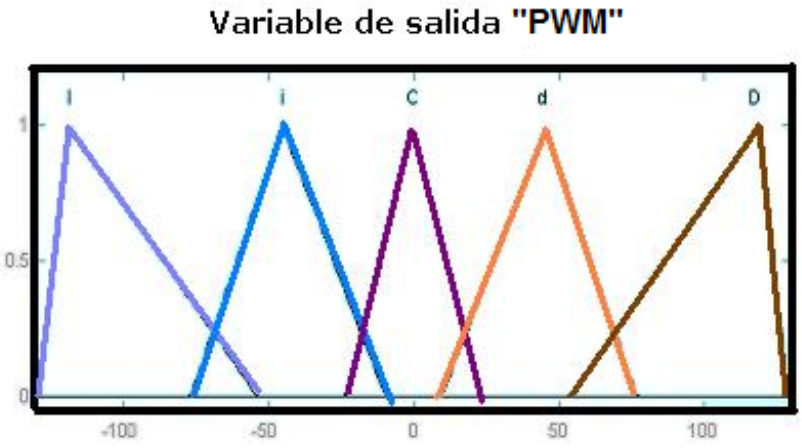


Figura 4.30 : Conjuntos borrosos de la variable de salida PWM

- **Base de conocimiento.** El conjunto de reglas borrosas que se han seleccionado como buenos descriptores del funcionamiento del sistema es el siguiente :

```
R1: IF (error is C) THEN (PWM is C)
R2: IF (error is N) THEN (PWM is I)
```

```
R3: IF (error is P) THEN (PWM is D)
R4: IF (error is p) THEN (PWM is d)
R5: IF (error is n) THEN (PWM is i)
```

La base de reglas se almacena en un fichero de configuración, figura 4.31. Este fichero en formato texto está especialmente orientado para el diseño y ajuste de controladores borrosos, por usuarios no expertos en Teoría de Conjuntos Borrosos, pero expertos en el control de un sistema de forma manual. El fichero se puede editar manualmente o bien mediante un entorno visual, FuzzyShell[3] [Gasós et al., 1990], cuya última versión ha sido programada en Borland C++. El entorno permite definir y manipular tanto variables, como etiquetas lingüísticas y reglas con extrema facilidad al igual que mostrar la representación gráfica de las funciones de pertenencia y de la superficie de control. El disponer de este tipo de herramientas de visualización además de las funciones de interpretación del mismo y de *fuzzification* y *defuzzification*, constituye un valor añadido para el diseño y sintonía del controlador. Se dispone además de un conjunto de funciones en C++, para la interpretación de este fichero de texto genérico, así como para la *fuzzification* de variables y *defuzzification* del conjunto de salida de todas las reglas.

Utilizando el entorno *fuzzy de Matlab* se puede visualizarse la inferencia borrosa a lo largo de todo el universo de discurso, figura 4.32 así como la curva característica del control, figura 4.33.

Resultados del control de dirección borroso P en el tractor.

El control proporcional descrito anteriormente se ha experimentado con el tractor DÉDALO, enviando diferentes referencias a la dirección para comprobar su respuesta en navegación en el campo y almacenando los valores del error en cada una de las trayectorias. A lo largo de estas trayectorias se han registrado, para cada ciclo de control, el ángulo de giro real de las ruedas, el error(t) y la salida de control. Estos valores permiten analizar el funcionamiento del controlador de la dirección del vehículo ante las distintas situaciones de las variables de estado.

[3]FuzzyShell es marca registrada por el CSIC 1643983 y con propiedad intelectual 16045

```
#Fichero con los datos de un controlador borroso

controlador direccion

etiqueta error negativo_grande = -4.0 -4.0 -1.0 -0.2
etiqueta error negativo_pequeño = -1.0 -0.5 -0.5 0.0
etiqueta error cero = -0.8 0.0 0.0 0.8
etiqueta error positivo_pequeño = 0.0 0.5 0.5 1.0
etiqueta error positivo_grande =  0.2 1.0 4.0 4.0

etiqueta PWM muy_derecha = 120 100 100 50
etiqueta PWM derecha = 10 50 50 90
etiqueta PWM cero = -30 -10 10 30
etiqueta PWM izquierda =  -10 -60 -60 -90
etiqueta PWM muy_izquierda = -60 -120 -120 -150

IF ( error = negativo_grande ) THEN  ( PWM = muy_izquierda )
IF ( error = negativo_pequeño ) THEN ( PWM = izquierda )
IF ( error = cero ) THEN ( PWM = cero )
IF ( error = positivo_pequeño ) THEN ( PWM = derecha )
IF ( error = positivo_grande ) THEN ( PWM = muy_derecha )
```

Figura 4.31 : Fichero de texto con la Base de Conocimiento

Para valorar el funcionamiento del controlador se han tenido en cuenta dos parámetros: el error estacionario e_{ss} y el tiempo de retardo t_d. Estos parámetros se utilizan normalmente en Teoría de Control para cuantificar la precisión y la velocidad del sistema, respectivamente [Ogata, 1990]. El tiempo de retardo es el tiempo que requiere el sistema para alcanzar la mitad del valor de referencia (4.13).

$$t_d \Rightarrow \ Output(t_d) = \frac{Referencia}{2} \tag{4.13}$$

El error estacionario es la diferencia entre el ángulo medido y la referencia solicitada una vez que ha finalizado el periodo transitorio (4.14).

$$e_{ss} = \lim_{t \to \infty} e(t) \tag{4.14}$$

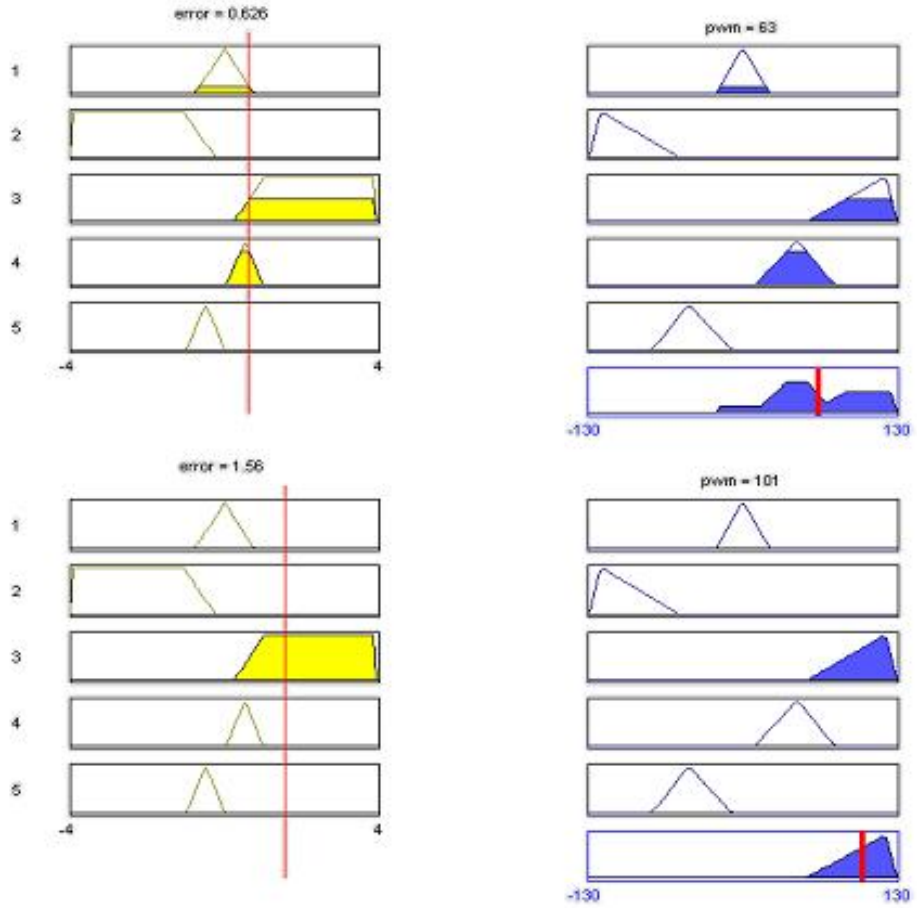


Figura 4.32 : Visualización de la inferencia del controlador P

La figura 4.34 muestra en la parte superior la evolución temporal de la señal de referencia y el valor real del ángulo de la dirección en grados. En la parte inferior se muestran los valores correspondientes de la variable de salida PWM expresados en tanto por ciento del ancho total del pulso del ciclo de control seleccionado.

Los resultados de la gráfica muestran cómo el controlador borroso proporcional cumple las especificaciones propuestas inicialmente, esto es:

- Responde a la función escalón en pocos cientos de milisegundos ($t_d = 1000$(ms.)).
- El comportamiento es sobreamortiguado.
- El error es inferior a ± 5 (grados) ($e_{ss} = 2{,}2$ (grados)).

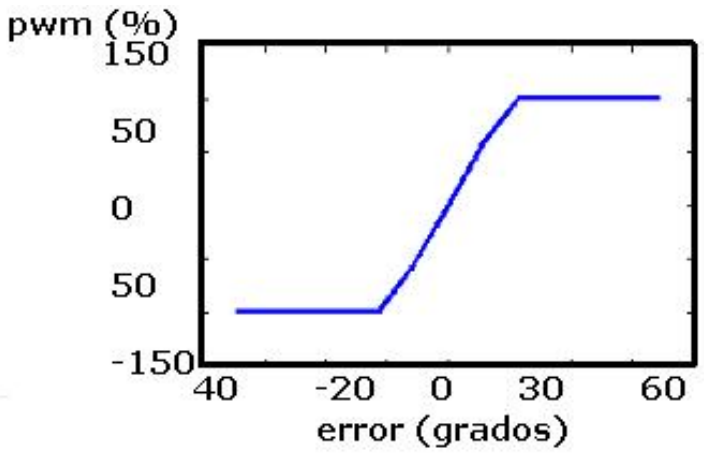


Figura 4.33 : Superfície de control

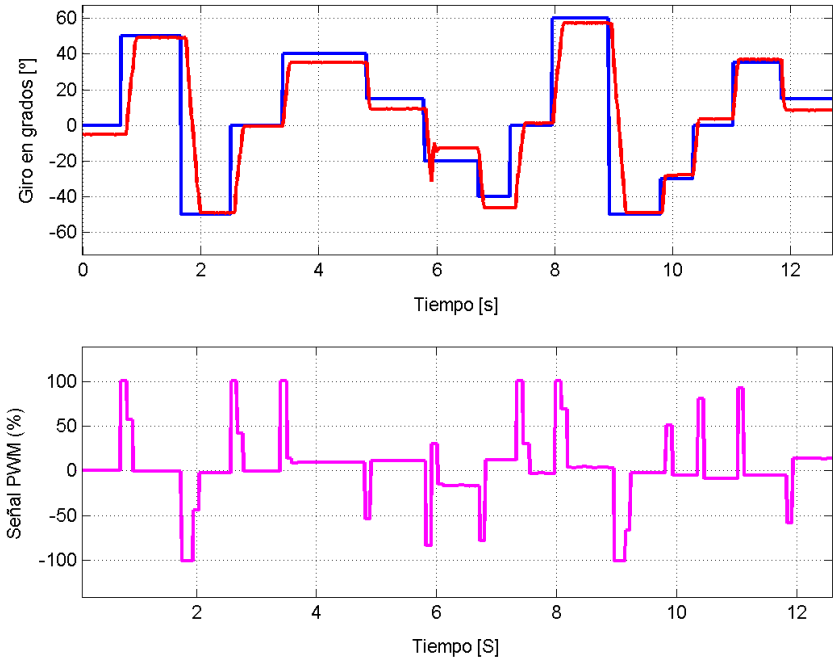


Figura 4.34 : Experimentación del control de dirección del tractor en el campo

- No se ve afectado por la presencia de perturbaciones.

Sin embargo analizando otro ensayo, figura 4.35, se percibe una deficiencia importante en el lazo de control: la insensibilidad del sistema ante pequeños escalones de referencia siempre que se parte de la situación de reposo.

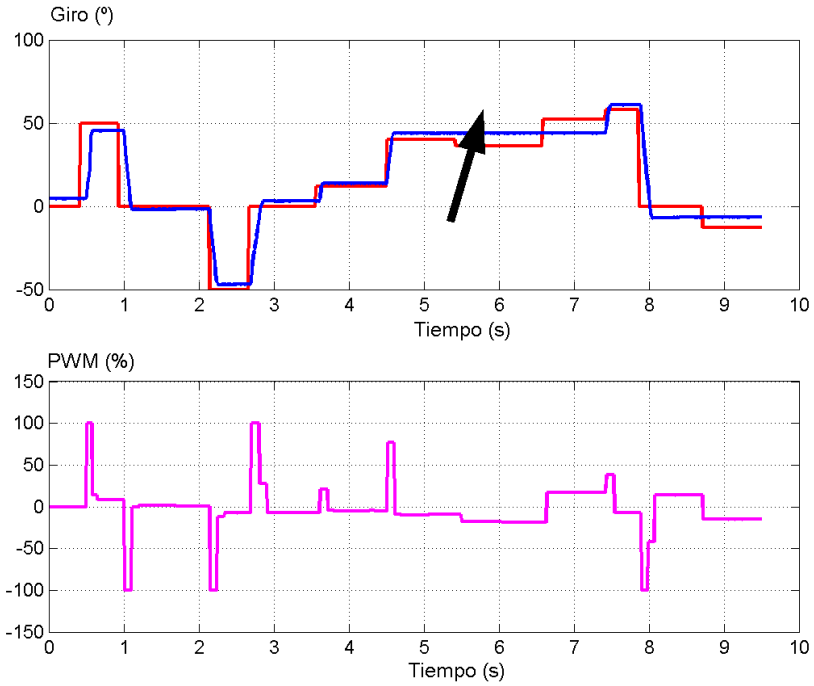


Figura 4.35 : Ensayo del controlador P en el tractor sobre terreno irregular.

Control híbrido

La dificultad del control proporcional para responder ante pequeños cambios en la referencia de giro se debe a una zona muerta que presenta el sistema, analizada al realizar la caracterización experimental del sistema en lazo abierto, figura 4.27. Desde el punto de vista del funcionamiento del sistema, esta zona muerta es insalvable pues obedece a la resistencia del sistema a girar cuando se parte del reposo.

Sin embargo es necesario que el control del sistema de dirección del tractor DÉDALO sea sensible y responda adecuadamente ante funciones de escalón pequeñas, ya que en su defecto no se podría controlar el tractor con suficiente precisión. La solución que se propone constituye una mejora sobre el control proporcional descrito anteriormente, mediante un modelo de control híbrido, de manera que el controlador proporcional actúe en todas las situaciones en las que la variación del ángulo de giro sea grande y active al controlador de alta resolución ante escalones pequeños unidos a estado de reposo.

El controlador híbrido propuesto, figura 4.36, implementa pues dos controladores, seleccionándose cuál es el adecuado en función del estado del sistema y del valor de la referencia. El primero se corresponde con el controlador proporcional descrito anteriormente y el segundo se trata de un controlador por muestreo rápido (de 50 (ms)), que acciona la electroválvula en uno u otro sentido dependiendo del signo del error y la desactiva cuando se ha alcanzado la referencia deseada.

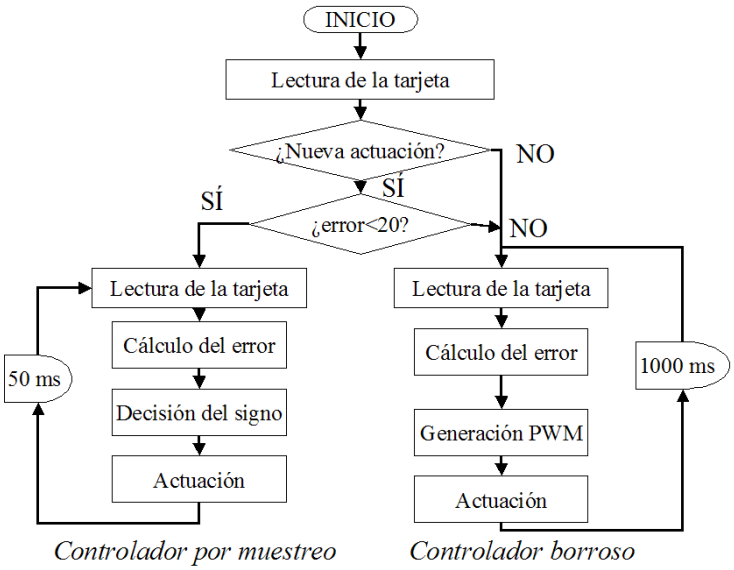


Figura 4.36 : Diagrama de flujo del controlador híbrido de dirección

La implementación del controlador híbrido de DÉDALO tiene dos partes. La primera es la selección del controlador que tiene que aplicarse en función del error que tiene que superar el sistema. Para ello se realiza una comparación entre la señal de referencia y la posición actual de la dirección. Si este error es mayor de 20 (grados) la PWM que se aplique a la válvula será la que calcule el controlador borroso proporcional de tiempo de ciclo 1000 (ms). En caso contrario, error menor que 20 (grados), se activa el controlador por muestreo, con tiempo de ciclo de 50 (ms). Este controlador comprueba el signo del error y dependiendo de éste activa la válvula adecuada. En caso de que el valor absoluto del error sea menor de $2{,}5$ (grados) se considera que se ha alcanzado la referencia y se desactiva el temporizador que implementa el controlador por muestreo. Los pulsos se han implementado con dos temporizadores enlazados,

uno encargado de mantener el pulso en un 1 lógico y el otro en un 0, figura 4.37. El controlador borroso proporcional se implementa utilizando las funciones de la biblioteca FuzzyShell.

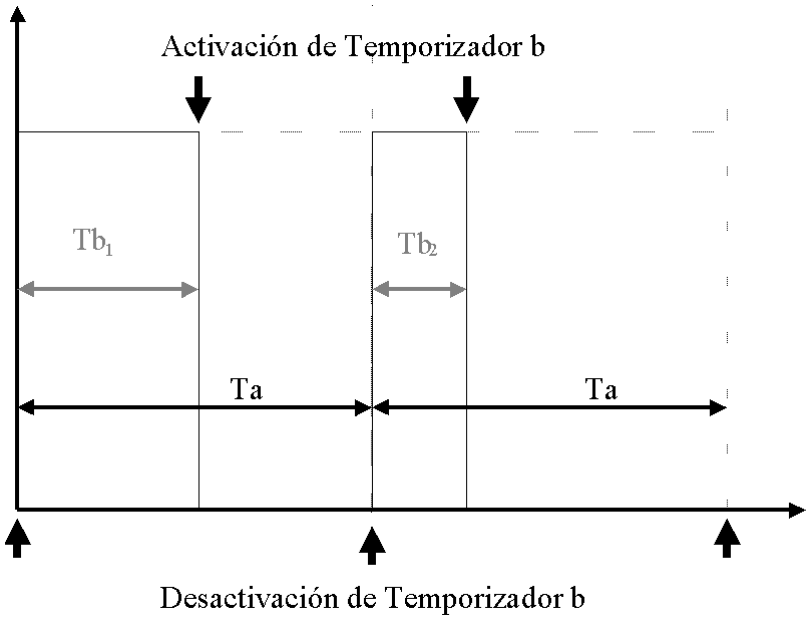


Figura 4.37 : Funcionamiento de los dos temporizadores entrelazados del control A y B

```
TEMPORIZADOR A

(1) Actualiza e(t)
(2) Ejecuta el controlador borroso que calcula PWM
(3) Analiza e(t) para calcular el estado
si e(t)=0 entonces estado=(0,0)
si e(t)<0 entonces estado=(1,0)
si e(t)>0 entonces estado=(0,1)
(4) Escribe 1 en el buffer de la tarjeta
estado define la salida digital sobre la que se escribe.
(5) Activa TEMPORIZADOR B con tiempo de ciclo Tb=Tb*(PWM/100)

FIN DE TEMPORIZADOR A
```

```
_____________________________


___________________

TEMPORIZADOR B

(1) Escribe estado=(0,0)
(2) Se auto-desactiva

FIN DE TEMPORIZADOR B

_____________________________
```

4.5.4 Resultados del control híbrido de dirección

En la figura 4.38 se muestran los resultados obtenidos aplicando el controlador híbrido. Puede observarse que el comportamiento es extremadamente bueno. Se ha conseguido que el sistema responda a escalones de cualquier amplitud, en cualquier punto del espacio de estados.

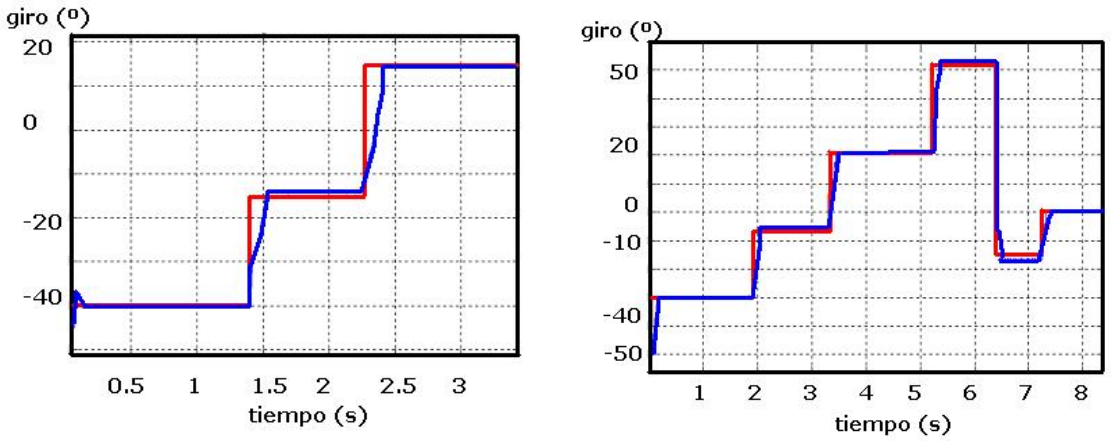


Figura 4.38 : Evolución temporal de la respuesta del sistema de control híbrido de dirección

4.6 Resumen de las características del tractor DÉDALO

La siguiente tabla resume las características del tractor DÉDALO en cuanto a su sistema de actuación y su sistema sensorial se refiere.

CARACTERÍSTICAS		**DÉDALO**
Actuación	**Sist. automatizados**	Dirección, freno y embrague
	Tipo de actuadores	Hidráulicos
	Controladores	Jerárquico borroso
Sensores	**Propioceptivos**	Potenciómetros y sensor batería
	Localización	DGPS-Rasant, odómetros y brújula
	Exteroceptivos	Láser, choques e inclinómetros
Proceso	**PC**	Portátil industrial
	Tarjetas	PCI-1711, WaveLan y Puertos Serie
Programa Servidor		DÉDALO-Servidor

Tabla 4.6 : Resumen de las características sensoriales, de actuación y de proceso del tractor DÉDALO

En este capítulo se ha descrito la etapa de automatización básica de un tractor comercial para su funcionamiento en modo autónomo. La primera fase describe la sustitución de los sistemas de actuación manuales de dirección y tracción por dos actuadores hidráulicos, aprovechando el circuito hidráulico instalado en fábrica del tractor, para el control automático de la dirección y del freno y el embrague.

La selección de los sensores más apropiados al tipo de tareas a realizar, vehículo y entorno constituye la segunda etapa del proceso de automatización. Los sensores instalados en DÉDALO son de dos tipos: propioceptivos y exteroceptivos. Los sensores propioceptivos son dos potenciómetros para la medida del estado del pedal de freno-embrague y del ángulo de giro de la dirección, un sensor de nivel de la batería y un sistema odométrico. Los exteroceptivos son: una brújula y un DGPS, correspondientes junto con los odómetros al subsistema de localización; un sensor láser de barrido, un sensor de colisión y dos inclinómetros.

El sistema de proceso a bordo del tractor DÉDALO es un ordenador portátil industrial, adaptado a las vibraciones del tractor y a las inclemencias meteorológicas. Sobre este

procesador se ha desarrollado una arquitectura de procesos basada en el paradigma cliente servidor, donde el programa denominado DÉDALO-Servidor se ejecuta en el ordenador a bordo, realizando las tareas de adquisición de datos sensoriales y control de los actuadores. Cualquier programa cliente puede comunicarse con DÉDALO-Servidor desde cualquier plataforma conectada a internet, y mediante paso de mensajes predefinidos solicitar datos sensoriales o enviar consignas de control sobre los actuadores.

Finalmente se describen las razones de la elección de un sistema de control borroso de la dirección hidráulica del tractor DÉDALO, así como las características y diseño del mismo. Los sistemas de control borroso se hallan ampliamente difundidos en la comunidad científica y en múltiples aplicaciones industriales por sus buenas prestaciones en sistemas complejos y no-lineales. En la sección 4.5 se describe el controlador híbrido de la dirección y los resultados experimentales que muestran cómo se alcanzan las especificaciones de control requeridas para cualquier valor del ángulo de referencia, sobre firme de tierra en el campo. El error estacionario medio es siempre inferior a 2 (grados) y el tiempo de retardo esta próximo a los 1000(ms).

A partir de este instante se dispone de un tractor automatizado, dotado de mecanismos de actuación automática, sensores propioceptivos, y sensores de localización y de percepción del entorno, para la generación de comportamientos autónomos. Por otro lado, el modelo de arquitectura cliente-servidor implementado para la ejecución de los procesos en interacción directa con los dispositivos físicos, en el ordenador a bordo, permite realizar de forma sencilla la solicitud de información sensorial y el envío de referencias de control desde cualquier otro proceso que se ejecute en una plataforma conectada a internet.

Capítulo 5

AGRO-AMARA: Una arquitectura híbrida basada en agentes

El principal objetivo de esta tesis es el desarrollo de una arquitectura de control para la navegación autónoma de un tractor agrícola. La arquitectura de control debe considerar los requisitos analizados en el capítulo 3 relacionados con los objetivos a realizar, el tipo de vehículo y entorno agrícola de exteriores. El primer requisito apunta a la **orientación a objetivos** de laboreo o de desplazamiento en el campo que debe primar en el modelo de arquitectura. En segundo lugar hay que garantizar la **seguridad** física del vehículo y del medio, ya que se trata de navegar en un entorno abierto y sin señalización donde puede encontrarse con otros vehículos, operarios y animales imprevistos. La seguridad implica: 1) capacidad de reaccionar a tiempo ante imprevistos en el entorno, y 2) robustez ante fallos del sistema.

Otra característica muy deseable es la **versatilidad** o capacidad de hacer frente a un conjunto amplio de entornos y tareas para la resolución de un determinado tipo de problemas. Además,

el diseño de la arquitectura debe permitir la incorporación **gradual** de nuevas habilidades sin perturbar considerablemente los procesos ya existentes, debe ser **escalable** para acometer el reto de enfrentarse con éxito a la incorporación de nuevos sensores, sistemas, entornos y tareas para generar comportamientos cada vez más complejo.

Acorde con la verificación de las características descritas anteriormente, esta tesis propone una arquitectura híbrida basada en agentes como modelo apropiado para la consecución de todos los objetivos de navegación de un robot agrícola en entornos dinámicos. El modelo que se propone -la arquitectura AGRO-AMARA-, tiene como punto de partida los trabajos realizados en el grupo de investigación GPA del IAI-CSIC dentro del campo de las arquitecturas de control [Garcia-Alegre et al., 1995, Guinea et al., 1995, García-Alegre y Guinea, 1992, García-Alegre y Recio, 1998, Cañas y García-Alegre, 1999] materializados en una arquitectura inicial denominada AMARA. Esta arquitectura propone una jerarquía de niveles de competencias que gravitan en torno a una representación del mundo y donde cada nivel está compuesto por un conjunto de agentes especializados en una actuación. Tiene como principios básicos: la modulación de comportamientos, la reusabilidad de habilidades, la facilidad de escalado y la definición explícita de un flujo bidireccional de información: 1) perceptiva de abajo-arriba, que se inicia en las señales sensoriales y puede llegar hasta el usuarios y 2) de acciones de arriba-abajo, que se inicia con la definición de la misión y llega hasta los actuadores. Estas características extraídas de la experimentación con robots de interiores, holonómicos o no, son fundamentales en el diseño de cualquier arquitectura de control.

Este capítulo presenta en detalle la arquitectura propuesta para la navegación autónoma de un tractor agrícola, denominada AGRO-AMARA. En la primera sección, 5.1, se exponen los principios básicos que guían el diseño de la arquitectura, las características de los módulos que la componen y los flujos de información que la caracterizan. La tarea de navegación de un robot agrícola presenta dos facetas bien diferenciadas: por un lado una **navegación global de propósito general**, con un planteamiento común para cualquier otro robot y entorno, cuyo objetivo es asegurar el desplazamiento del vehículo entre dos posiciones cualesquiera del

entorno, considerando la orientación -indispensable dada la escasa maniobrabilidad del tractor-; evitando colisiones con objetos imprevistos y optimizando la secuencia de puntos a seguir o ruta planificada; y por otro una **navegación específica** acotada con precisión por el tipo de laboreo a realizar.

En la sección 5.2 se describe detalladamente la arquitectura global de propósito general propuesta para la navegación autónoma y en la sección 5.3 la navegación específica de laboreo.

5.1 Percepción, representación y acción en la arquitectura AGRO-AMARA

La unidad básica de la arquitectura AGRO-AMARA es el AGENTE, entendido como *"proceso o conjunto de procesos dirigidos a conseguir o mantener un objetivo capaz de habilidades perceptivas, deliberativas y reactivas sin restricción alguna en su complejidad"*[García-Alegre y Recio, 1998]. La definición clásica de comportamiento ([Mataric, 1994]) como "ley de control que agrupa un conjunto de restricciones para alcanzar y mantener un objetivo" se adapta menos a la filosofía planteada, al explicitar únicamente el aspecto de control en su definición, a pesar de que en muchos trabajos ambos términos, agente y comportamiento pueden ser utilizados indistintamente [García-Alegre y Recio, 1998]. La denominación de agente en lugar de comportamiento pretende destacar que la arquitectura está compuesta por módulos con capacidad de percepción y acción, tal y como se planteaba en la sección 3.2 del capítulo 3. Además, el concepto de esquema definido por Arkin [Arkin, 1987] como "especificación genérica de un agente de cálculo" y utilizado por otros autores como [Arbib y Cobas, 1990, Cañas y García-Pérez, 2002], es muy similar a la definición de agente propuesta en este trabajo, con la única diferencia substancial en el origen del mismo, pues el término esquema proviene del campo de la Biología y la Psicología, mientras el término agente surge , en su definición inicial, del área de la Inteligencia Artificial. Por otro lado, Arbib considera que tanto los agentes de Minsky como los comportamientos de Brooks son similares a los esquemas utilizados en sus investigaciones [Arbib, 1989]. El término que se utilice para

definir la unidad formal o bloque estructural básico de la arquitectura, viene determinado fundamentalmente por las raíces e historia del grupo investigador.

En AGRO-AMARA, la noción de agente se ha establecido en los términos más generales posibles, ya que lo fundamental en una arquitectura es el comportamiento observable y no la naturaleza del mecanismo que permite alcanzarlo [MacFarland, 1990]. Esta definición entronca con las ideas de Minsky, para el que los agentes son máquinas con una determinada función [Minsky, 1985]. En AGRO-AMARA se han definido dos tipos de agentes en función del tipo de procesamiento, perceptivo o de planificación y acción que realicen: 1)agentes perceptivos y 2) agentes de actuación. Esta división supone una innovación de AGRO-AMARA con respecto de AMARA. El encapsulamiento de los procesos perceptivos y de acción por separado facilita la reutilización de las habilidades perceptivas lo que resulta especialmente útil en un entorno agrícola donde la percepción es compleja, como se argumenta en los capítulos 2 y 3. La mayoría de los autores no separan procesos perceptivos de procesos de actuación; AGRO-AMARA se diferencia en esto de la mayoría de las arquitecturas híbridas, acercándose a los argumentos de Arkin y Arbib. Este último aporta evidencias neurobiológicas de la separación entre procesos perceptivos y de actuación en ranas [Arbib y Cobas, 1990].

Los agentes perceptivos se dirigen a la elaboración y mantenimiento de los estímulos requeridos en las tareas de control. La percepción se concibe en AGRO-AMARA, como orientada a la acción, y los recursos y mecanismos de control perceptivo vienen determinados por las necesidades del comportamiento [Arkin, 1998].

Los agentes de actuación tienen como finalidad determinar el curso de la acción a seguir, utilizando las percepciones elaboradas por los agentes perceptivos y el soporte proporcionado por una representación del sistema y del mundo. Las percepciones se almacenan en una memoria compartida de tipo pizarra [Hayes-Roth, 1985]. Este modelo, empleado habitualmente en arquitecturas de control [Matellán y Borrajo, 1998, Konolige et al., 1997], permite almacenar estructuras con información con grados de abstracción diferente. Así permite que cada agente trabaje con el tipo de representación que le resulte más adecuado, en la línea del LPS de Saphira [Konolige et al., 1997]. Por lo tanto, existen tantas representaciones del

sistema/entorno como sean necesarias para facilitar la optimización de los planes de actuación, con la única restricción en los tiempos de respuesta del sistema, que a su vez son dependientes de los procesadores y de los procesos de razonamiento involucrados en cada agente. El empleo de una representación de tipo pizarra permite desacoplar los niveles perceptivos de los de actuación, dotando a la arquitectura AGRO-AMARA de mayor flexibilidad, ya que permite que agentes de actuación de alto nivel empleen datos sensoriales crudos o percepciones de bajo nivel su así lo requieren. Esta independencia entre niveles perceptivos y de actuación diferencia AGRO-AMARA de su predecesora AMARA y de las arquitecturas tradicionales deliberativas como RCS o NHCA.

La reutilización de habilidades constituye una de las premisas de la arquitectura híbrida AGRO-AMARA que aquí se propone, en consonancia con el funcionamiento de los centros instintivos descritos en los modelos de comportamiento animal y en diversos estudios etológicos [Vogel y Angermann, 1977, Tinbergen, 1950].

Los agentes en AGRO-AMARA, constan de tres aspectos fundamentales: 1) un conjunto de procesos de cómputo y de comunicación que definen la competencia de cada agente, 2) la representación del sistema y del entorno con distinta granularidad o grado de resolución, 3) el conjunto de entradas y salidas asociadas a cada agente, figura 5.1.

Los procesos integrados en los agentes muestran habilidades perceptivas, deliberativas y reactivas de acuerdo con la definición inicial propuesta para el agente. Los agentes se organizan jerárquicamente en la arquitectura de modo que cada agente inicia la ejecución de aquellos cuya funciones necesita, permitiendo así la reutilización de agentes y facilitando la inclusión de agentes con nuevas habilidades. Desde este punto de vista, este tipo de arquitecturas pueden considerarse como “sociedades” de agentes (agency)[Minsky, 1985]. Sin embargo, aunque se trata de agentes organizados no se fuerza la existencia de niveles fijos estáticos para dotar a la arquitectura de más flexibilidad. En este punto AGRO-AMARA difiere de AMARA que propone la descomposición en niveles. AGRO-AMARA apuesta por un conjunto de agentes sin niveles fijos que se organizan siguiendo el criterio de reutilización de habilidades, en la línea de la arquitectura JDE (Jerarquía Dinámica de Esquemas) [Cañas, 2003] y de las arquitecturas

basadas en comportamientos [Mataric, 1992a].

La representación del mundo aumenta su granularidad conforme aumentan las capacidades de deliberación de los agentes, con la siguiente disminución del detalle de la tarea. La terminología utilizada en estas representaciones se halla más próxima al lenguaje natural de los usuarios finales, en oposición a la representación instantánea del mundo de utilidad en el nivel más bajo de control de dispositivos físicos.

La señales de entrada y salida de los agentes pueden dividirse en dos grupos: 1)señales de activación que proceden o van dirigidas a un agente, necesarias para iniciar el ciclo de ejecución, y 2) los parámetros de modulación del agente, que son opcionales. En AGRO-AMARA la información es compartida por los agentes atendiendo a dos mecanismos clásicos: 1) paso de mensajes con una sintaxis y semántica previamente definida y 2) pizarra o memoria compartida, que implícitamente organiza la información en niveles de abstracción. Las percepciones o estímulos generados o detectados por los agentes perceptivos, se almacenan en la pizarra para ser utilizados por diferentes agentes en el instante que lo precisen.

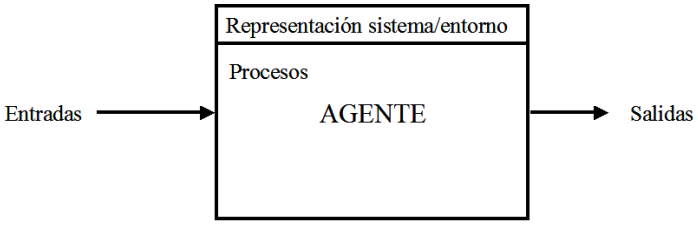


Figura 5.1 : Esquema de un agente

5.1.1 Percepción y representación

En la arquitectura AGRO-AMARA la percepción se articula en torno a un conjunto de agentes perceptivos diseñados para la detección de los estímulos que se consideran clave para la navegación. Estos agentes implementan procesos de percepción orientados a la extracción de características relevantes del entorno o del sistema, y son necesarios para la activación de los agentes de actuación. Los agentes perceptivos, a diferencia de los de actuación, poseen

únicamente dos estados, **inactivo** y **activo**, y el paso de uno a otro estado está condicionado por una señal de activación/desactivación externa, figura 5.2.

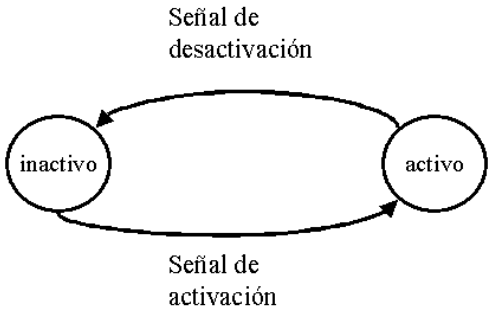


Figura 5.2 : Transiciones de estado en los agentes perceptivos

Las percepciones elaboradas por los distintos agentes perceptivos, dan lugar a un tipo de representación del sistema y del entorno, con distinta granularidad y alcance espacio-temporal. Esta representación se almacena en una zona de memoria compartida o pizarra, para ser utilizada por otros agentes de actuación o perceptivos. La actualización de una determinada representación es responsabilidad del agente perceptivo que la genera. Los agentes perceptivos entroncan con el enfoque *gestaltico* de la percepción que trata de extraer únicamente la información relevante para la activación de un agente y no la reconstrucción completa y detallada del entorno [Monserrat, 1998].

5.1.2 Selección de acción

La acción en AGRO-AMARA se materializa mediante los agentes de actuación, que a lo largo de una cadena de activaciones generan las acciones de control de movimiento de giro y de traslación imprescindibles para el desplazamiento del tractor. Los agentes de actuación se modelan siguiendo el esquema mostrado en la figura 5.1, y pueden encontrarse en 3 estados de activación: **inactivo**, **alerta** y **activo**. En el estado **inactivo** el agente es un proceso o conjunto de procesos que no se ejecutan. El agente **inactivo** pasa al estado de **alerta** al recibir una señal de activación, y al pasar a este estado comprueba cíclicamente si el contexto definido para su ejecución se cumple, si es así pasa al estado **activo**, lo que implica que toma el control; y tomar

el control significa proceder a la toma de decisiones de actuación, bien sea sobre los actuadores o sobre el estado de alerta y modulación de otros agentes. El diagrama de estados de la figura 5.3 incluye además las transiciones de estado **activo** a **alerta**, en el caso de que se dejen de verificar sus condiciones de activación y de estado de **alerta** a **inactivo** al recibir una señal de desactivación.

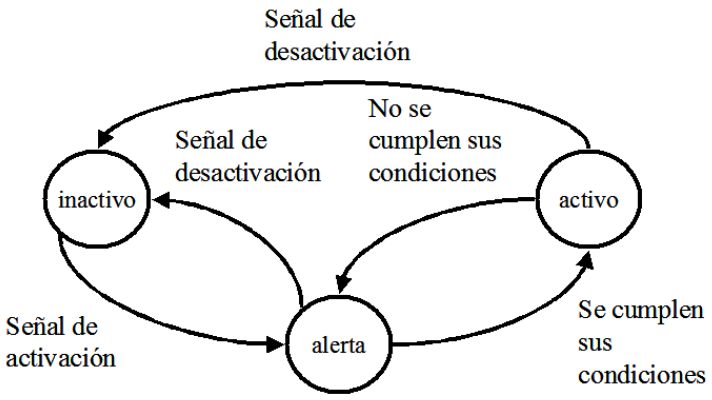


Figura 5.3 : Transiciones de estados en los agentes de actuación

No existe por tanto fusión de comandos, como sucede en la arquitectura DAMM, capítulo 3, sino arbitraje entre comportamientos donde uno de ellos es el que finalmente toma el control. El arbitraje entre comportamientos se alcanza diseñando los agentes de modo que sus contextos de activación sean mutuamente exclusivos [Mataric, 1994]. La existencia de clases perceptivas [Arkin, 1998] facilita en extremo el proceso de arbitraje. Estas clases perceptivas constituyen una partición del espacio de eventos percibidos de acuerdo con los requisitos de los agentes de actuación, por ejemplo: espacio libre u ocupado para el agente **Evitar Obstáculos**.

En cuanto a la planificación, los planes se consideran como un recurso más dentro de la arquitectura y aunque se encapsulan como agentes tienen algunas características similares a las de los agentes perceptivos. El plan, visto como el curso de acción más adecuado, se almacena en la memoria compartida, quedando a disposición de los agentes que lo puedan necesitar. AGRO-AMARA incorpora de este modo las habilidades puramente deliberativas como un agente más dentro de la arquitectura. De este modo aunque los agentes de planificación especifican el curso de acción propuesto son los agentes de navegación los que deciden si el plan se ejecuta tal cual

o no.

5.2 Arquitectura de agentes AGRO-AMARA para navegación global

El problema que resuelve la navegación global es el diseño de los agentes de comportamiento adecuados que logren que el robot navegue con la mínima intervención humana desde su posición actual a un destino objetivo, siguiendo el mejor camino posible y evitando que el robot choque con obstáculos y que vuelque.

El objetivo de la navegación global de un robot agrícola es el desplazamiento seguro entre dos posiciones cualesquiera del campo consideradas origen y destino de la misión, optimizando la ruta y evitando colisiones con obstáculos imprevistos con la finalidad de una intervención mínima del operario. Para resolver este problema se han diseñado una colección de agentes específicos siguiendo las líneas de diseño propuestas en la arquitectura AGRO-AMARA. El agente de actuación **Ir a Punto**, junto con el agente planificador de caminos **Planificar Caminos**, y los agentes de menor nivel modulados por **Ir a Punto**, dan respuesta al problema de navegación planteado.

El objetivo del agente **Ir a Punto** puede descomponerse en tres subobjetivos concurrentes: 1) la navegación hacia el objetivo final, 2) navegación reactiva atendiendo a mecanismos de seguridad y 3) navegación optimizada por el planificador de rutas. Cada uno de estos subobjetivos es alcanzado mediante la intervención de uno o varios agentes, siguiendo la jerarquía mostrada en la figura 5.4. Así el agente **Avanzar** es activado para guiar los desplazamientos hacia un objetivo cercano, **Evitar Obstáculos** y **Parar** actúan para hacer frente a posibles colisiones y vuelcos. El agente **Planificar Caminos** se encarga de la planificación de la ruta, atendiendo a criterios de optimización previamente definidos. Cualquier estrategia de navegación que emplee un plan, necesita en cada instante disponer de una estimación actualizada de la posición del robot en el entorno. En exteriores este problema no es trivial y existen multitud de trabajos dedicados a la obtención de la posición mediante técnicas y

sensores diversos [Pozo-Ruz et al., 2000, Stentz et al., 2002], por ello se ha diseñado un agente perceptivo dedicado exclusivamente a la localización continua del robot en el mundo, el agente perceptivo **Actualizar Posición** que se encarga de elaborar y mantener actualizada en cada ciclo de control la posición del robot.

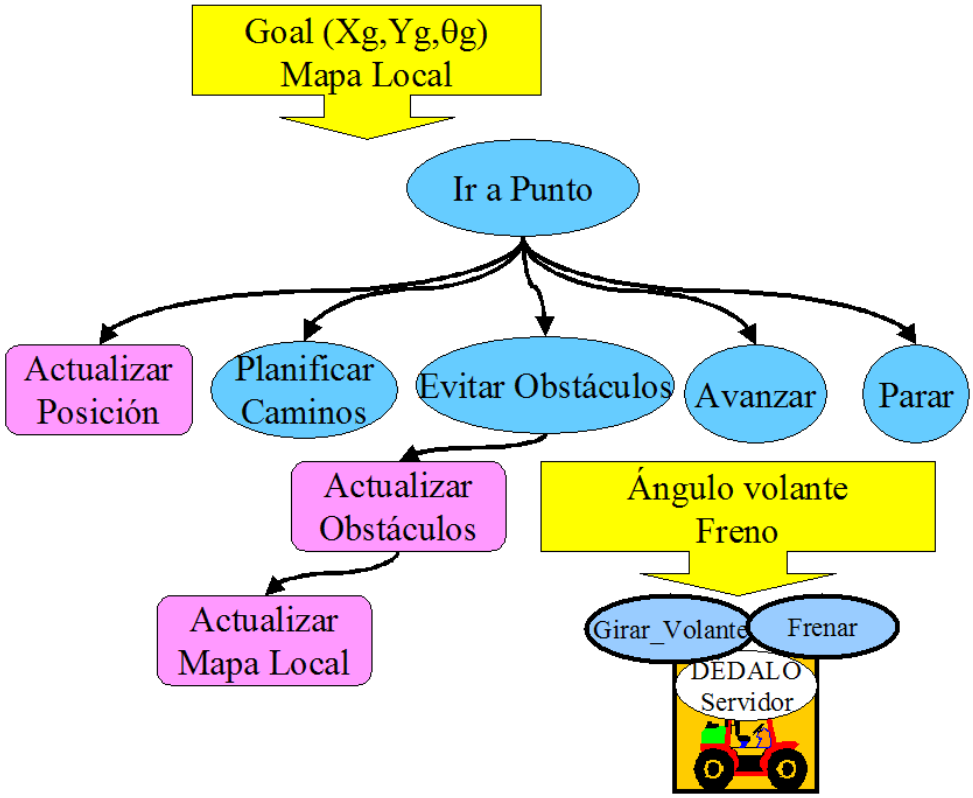


Figura 5.4 : Jerarquía de agentes para la navegación global genérica

Tanto el agente **Avanzar** como el agente **Evitar Obstáculos** requieren información sobre la existencia de obstáculos imprevistos en la trayectoria del robot. Esta percepción de estructuras en el entorno, que obstruyan el camino a seguir por el robot, la realiza el agente **Actualizar Obstáculos**, que opera sobre una representación del entorno de grano medio o mapa local de rejilla, actualizada por el agente perceptivo **Actualizar Mapa Local**. Este último agente se encarga únicamente de mantener el mapa local de rejilla, almacenando el resultado de su procesamiento en la pizarra para su utilización por otros agentes perceptivos o de actuación que operen en base a esta representación local.

La pizarra contiene las variables necesarias por los agentes, tanto de actuación como

perceptivos, actuando de esta manera como canal de comunicación asíncrono entre ellos. La pizarra también proporciona el canal adecuado para la inyección de la información a priori, que como se dijo en el capítulo 3 se considera importante en el diseño de una arquitectura para un robot móvil agrícola. Así el operario humano puede introducir en la pizarra información necesaria para los agentes. En el caso de la navegación global general, el humano introduce el mapa global estático y la posición objetivo. El mapa global estático puede ser una fotografía aérea de la zona georreferenciada (fotografías aéreas pueden obtenerse en tiendas especializadas) o un mapa georreferenciado. Sobre este mapa el operario define las zonas prohibidas de paso que empleará el agente **Planificar Caminos**. También sobre el mapa el operario selecciona el punto objetivo destino.

A continuación se describen en detalle cada uno de los agentes de la arquitectura, implicados en la navegación global del robot, mostrados en la figura 5.5. En primer lugar se describe la función y restricciones de los agentes perceptivos, ya que sin ellos no se produciría la activación de determinados agentes de actuación. La exposición de los agentes que componen la arquitectura, comienza por los de nivel inferior, esto es los que se encuentran más próximos a los actuadores y poseen una representación temporal instantánea o de corto alcance, para finalizar con los de mayor grado de abstracción y representación espacio-temporal de largo alcance. Estos últimos poseen unos mecanismos de deliberación y un lenguaje (vocabulario y sintaxis) más próximos a los del operario, facilitando la comunicación a través de la interfaz hombre-máquina para la activación de cada misión de laboreo y visualización del estado del sistema/entorno en cada instante. Los dos agentes de locomoción básicos **Girar Volante** y **Frenar** corresponden a los controladores de bajo nivel de los dispositivos de dirección y tracción. Estos agentes se explicaron con detalle en el capítulo 4, sección 4.5, al actuar directamente sobre los dispositivos físicos del robot.

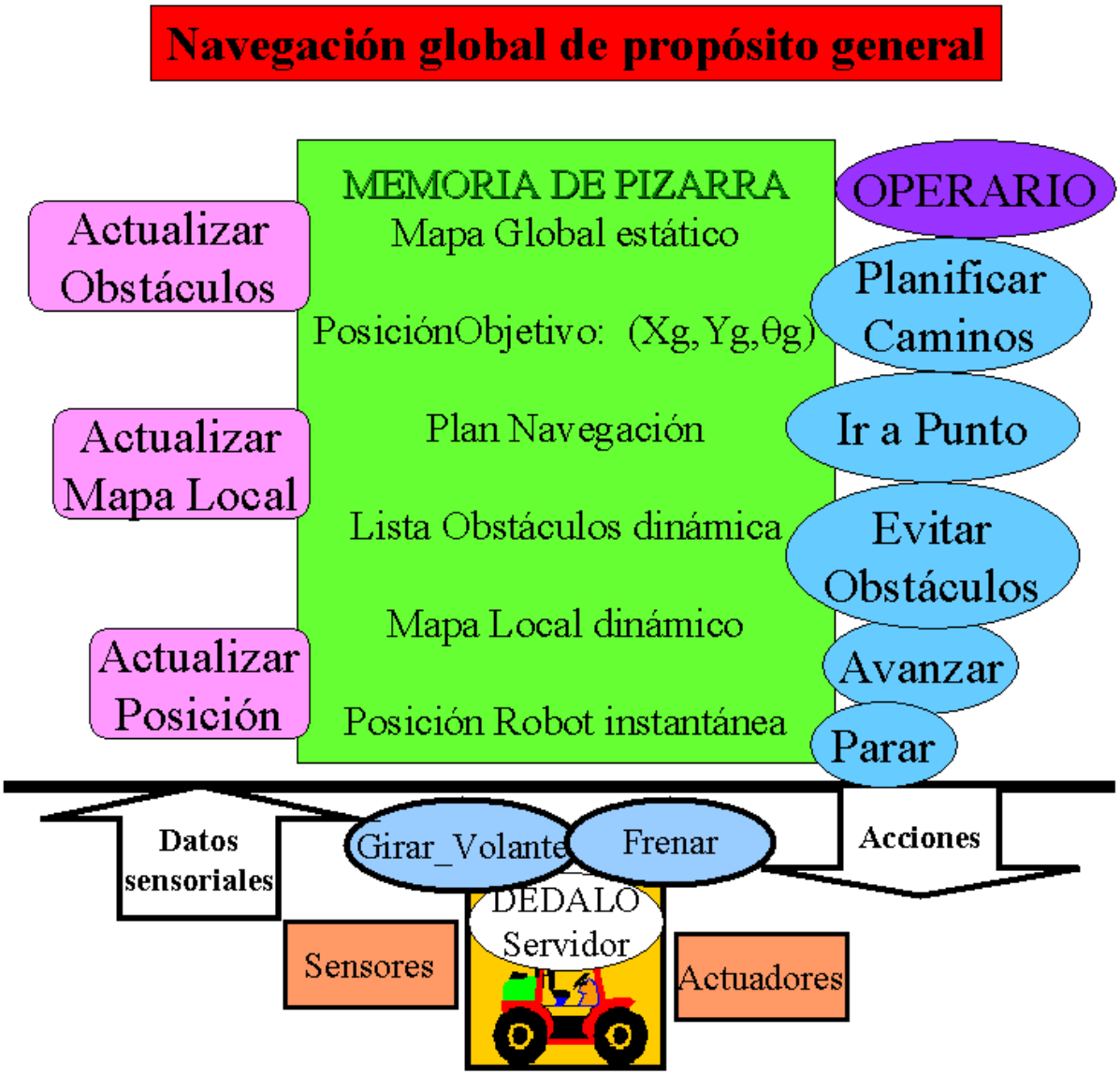


Figura 5.5 : Arquitectura de agentes para la navegación global de propósito general. Las elipses de color azul, situadas a la derecha de la figura corresponden a los agentes de actuación. Los rectángulos con esquinas redondeadas de color rosa, situados a la izquierda son los agentes perceptivos. En el centro de la imagen en color verde, se muestra el contenido de la pizarra. La zona inferior de la figura corresponde al servidor del robot, con los dos agentes básicos de locomoción **Girar Volante** y **Frenar**.

5.2.1 Agentes perceptivos de la arquitectura de navegación global

A continuación se describen los agentes perceptivos desarrollados para la navegación global general: **Actualizar Posición**, **Actualizar Mapa Local** y **Actualizar Obstáculos**.

Agente Actualizar Posición

El agente **Actualizar Posición** tiene como objetivo la obtención y el mantenimiento de la posición del robot, imprescindible para conseguir abordar la navegación autónoma del tractor. La navegación en exteriores implica en la mayoría de los casos: 1) distancias muy superiores a las que se encuentran en dominios de interior [Nehmzow y Owen, 2000] y 2) vehículos pesados y de gran dimensión, por ello las precisiones requeridas son generalmente inferiores a las de un entorno de interior. En el ámbito de la Agricultura de Precisión, se ha enfatizado en extremo la necesidad de alta precisión en el posicionamiento, incorporando sensores muy precisos pero de elevado coste [Thuilot et al., 2002], receptor RTK-GPS, en los sistemas de guiado y de asistencia a la conducción. Este interés en el posicionamiento preciso está motivado por el tipo de tarea a realizar, esto es, describir trayectorias con mayor precisión que la conseguida en una conducción manual [Trimble, 2002, Integrinautics, 2002, Beeline, 2002]. En labores agrícolas específicas como el arado o la siembra de hortalizas o viñas es necesaria una precisión alta, sin embargo en tareas de navegación global o de navegación orientada a facilitar tareas agrícolas, una precisión submétrica es suficiente.

Además, si el objetivo perseguido es una navegación autónoma, el sistema de posicionamiento tiene que ser robusto frente a fallos e imprecisiones en los sensores de localización. Por ello es fundamental que el agente **Actualizar Posición** sea capaz de tomar decisiones que conduzcan a la resolución de este tipo de contingencias, por el riesgo que suponen para la integridad del tractor y del entorno.

Atendiendo a las dos consideraciones anteriores -precisión submétrica y robustez en el posicionamiento-, el agente **Actualizar Posición** recibe información relativa a su localización de diferentes sensores, estimando la posición del robot en cada instante de tiempo. La estima se realiza analizando el funcionamiento de cada uno de los sensores de localización y empleado en el cálculo de la posición aquél o aquéllos que se consideran mejores en ese instante, no se realiza por lo tanto fusión sensorial. Este enfoque de integración de dispositivos de posicionamiento redundantes es común en otros robots autónomos dedicados a tareas agrícolas

[Hague et al., 1999a, Pilarski et al., 2002, Stentz et al., 2002].

- **Entradas, salidas, representación y procesos en el agente Actualizar Posición.** El agente **Actualizar Posición** tiene como única entrada una señal de activación y como salidas la posición estimada del robot *Posición*, (x_R, y_R, θ_R) y el modo de posicionamiento. Este modo de posicionamiento hace referencia a los sensores empleados en la estima de la posición en ese instante. El algoritmo de actualización de la posición del robot requiere en cada instante, las medidas sensoriales de la brújula digital, del receptor GPS y del sistema odométrico. En la figura 5.6 las señales sensoriales se han representado mediante rectángulos de color naranja.

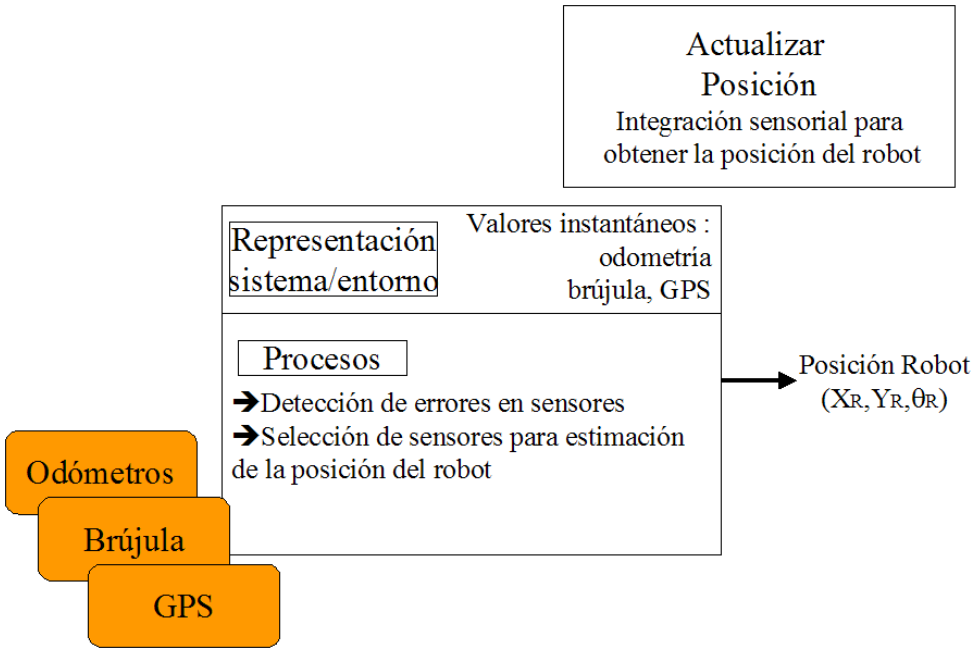


Figura 5.6 : Esquema de entradas, salidas, representación y procesos en el agente **Actualizar Posición**.

Para el diseño del algoritmo de cálculo de la posición, se han tenido en cuenta las características de los sensores de localización integrados en el tractor, sección 4.3. El receptor DGPS y la brújula digital proporcionan la posición y orientación absolutas respectivamente. Por el contrario, la odometría constituye un sistema de posicionamiento relativo que permite obtener una estimación de la distancia recorrida respecto a la posición inicial. En la siguiente tabla, tabla 5.1, se resumen las propiedades más relevantes de cada uno de los sistemas sensoriales de

posicionamiento a bordo del tractor.

SENSORES	**GPS**	**Brújula**	**Odometría**
Medida	absoluta	absoluta	relativa
Frecuencia	5(hz.)	5(hz.)	50(hz.)
Errores	acotados	acotados	no acotados

Tabla 5.1 : Características de los sistemas de posicionamiento.

El agente **Actualizar Posición** tiene como entradas las estimaciones instantáneas proporcionadas por los sensores de localización y utiliza un algoritmo de fusión sensorial para obtener la localización actual del vehículo. Se parte de la hipótesis de que el receptor GPS con corrección diferencial proporciona una localización mucho más precisa que el sistema odométrico, pero con posibles pérdidas de la señal, pues el error está acotado. El agente **Actualizar Posición** procede a la detección de posibles fallos en cada uno de los sensores, con el fin de activar el proceso de cálculo más adecuado a la precisión de la información recibida, tabla 5.2. La variable modo de posicionamiento alude al conjunto de sensores que participan, en cada caso, en el proceso de integración sensorial.

Modo	GPS	Brújula	Odometría	Posición robot
0	OK	OK	–	DGPS + Brújula
1	OK	X	–	DGPS
2	X	OK	OK	Odometría + Brújula
3	X	X	OK	Odometría
4	X	X	X	Fallo

Tabla 5.2 : Modos de posicionamiento y sensores de localización.

Se considera que el GPS proporciona una posición incorrecta en las situaciones siguientes: 1) si no dispone de la corrección diferencial, 2) cuando el número de satélites detectados es reducido, y 3) mediante cálculos indirectos partiendo de la hipótesis de que el robot al moverse a velocidad constante, recorre una distancia estimada en el intervalo de tiempo medido. Por lo tanto si dos medidas consecutivas del receptor GPS violan esta última hipótesis, sobre un umbral predefinido, se infiere un posible fallo en el GPS. Estos fallos pueden estar producidos por zonas de sombra del GPS, o fallos esporádicos debidos a errores multicamino etc... La

misma hipótesis se utiliza para detectar fallos en la brújula.

Respecto a la odometría esta hipótesis se verifica siempre puesto que se trata de un sensor que calcula incrementos. Sin embargo, ya que el error en la odometría aumenta con la distancia recorrida y no está acotado, se considera que la información que proporciona no es fiable a partir de cierta distancia, que depende del número de giros realizados y cuyo valor se puede estimar experimentalmente. Al tratarse de una navegación en exteriores los errores odométricos son aún mayores que en interiores, pues el tipo de terreno favorece el deslizamiento en unas ocasiones y la rodadura sin deslizamiento en otras ante badenes, surcos y agrupamientos de piedras[Borenstein y Feng, 1998].

El diagrama de flujo de información en el agente **Actualizar Posición**, figura 5.7, y el pseudocódigo del anexo A, describen en detalle el funcionamiento de este agente perceptivo.

El tiempo del ciclo de operación del agente **Actualizar Posición** viene determinado por la frecuencia de las lecturas proporcionadas por los sensores. En este caso, no puede ser inferior a 200 (ms) ya que la frecuencia de actualización de la señal más lenta (GPS y brújula) es de 5 (Hz.).

Agente Actualizar Mapa Local

El uso de mapas locales basados en rejillas de ocupación es común en robótica, al ser una herramienta de representación muy adecuada para fusionar información de diferentes sensores y representar la incertidumbre inherente a las medidas sensoriales. Esta incertidumbre se propaga y afecta a los resultados posteriores obtenidos al operar con medidas de sensores reales. La generación y actualización del agente *Mapa Local* de rejilla se realiza a partir de las medidas crudas proporcionadas por el sensor láser de barrido y de la estima de la posición del robot actualizada por el agente **Actualizar Posición**. Los datos crudos que proporciona este dispositivo sensorial, se agrupan en nubes de puntos con un valor de ruido asociado, sobre un plano horizontal [Prassler et al., 2000]. El mapa de rejilla ofrece además la ventaja de almacenar información sobre regiones del espacio que actualmente se encuentran fuera del alcance del

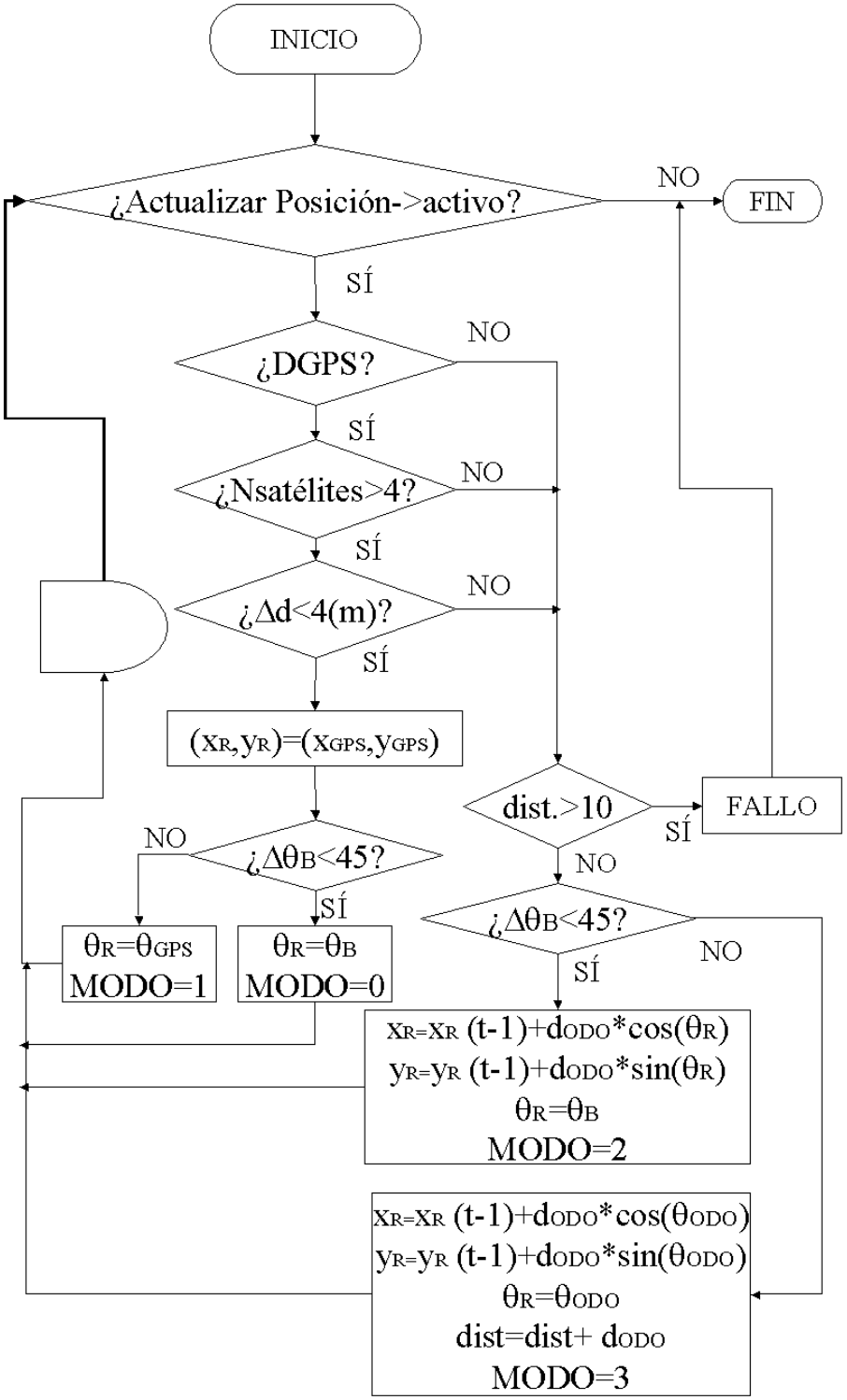


Figura 5.7 : Diagrama de flujo de información en el agente **Actualizar Posición**

sensor pero que fueron detectadas en instantes anteriores, proporcionando una información adicional de gran utilidad.

El mantenimiento de un mapa local de rejilla es fundamental para la elaboración de estímulos o estructuras relevantes para la navegación, como son los obstáculos. El agente **Actualizar Mapa Local** mantiene actualizado el *Mapa Local* de rejilla en relación a la ocupación de las celdillas que lo forman. Necesita la información sensorial cruda que proporciona el láser de barrido y la percepción *Posición*, actualizada por el agente **Actualizar Posición**, descrito anteriormente. El agente **Actualizar Mapa Local** dispone de una entrada, la señal de activación, y como única salida, la actualización de ocupación de celdillas en el *Mapa Local*. Una vez realizado este proceso el agente **Actualizar Mapa Local** cambia a estado **activo** y envía una señal de activación al agente **Actualizar Posición** con el fin de que la percepción *Posición* esté siempre actualizada, figura 5.8.

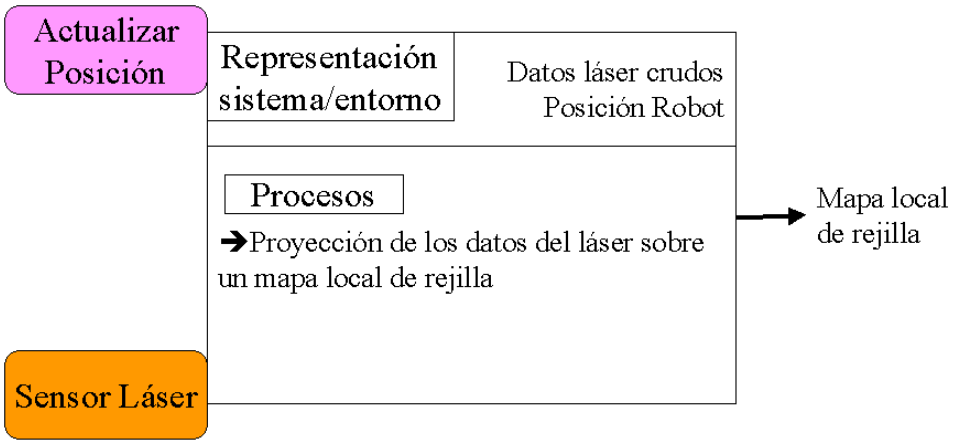


Figura 5.8 : Esquema de entradas, salidas, representación y procesos en el agente **Actualizar Mapa Local**

■ **El *Mapa Local*.** El *Mapa Local* constituye una representación del estado del entorno próximo al robot, y se ha representado mediante un cuadrado reticulado de 40×40 (m.) con celdillas de 20 (cm.) de lado, donde cada celdilla tiene asociado un valor de ocupación. Las dimensiones del mapa vienen determinadas por el alcance del sensor láser de barrido. El tamaño de la celdilla se ha seleccionado considerando el tamaño de los obstáculos que se esperan

encontrar en el campo, árboles, personas, vehículos, cuyo tamaño visto desde la altura a la que se ha colocado el sensor láser supera los 20 (cm.) que mide cada celdilla.

Se han definido tres valores lingüísticos para indicar el grado de ocupación de las celdillas:$\{ocupado, libre, desconocido\}$. Estos valores se calculan a partir de las últimas tres medidas de distancia sistema-obstáculo proporcionadas por el sensor láser, para cada una de las celdillas. De este modo el mapa de rejilla actúa como un *filtro paso bajo* que rechaza las lecturas con ruido a expensas de una pequeña latencia. En este caso, y teniendo en cuenta la precisión del sensor, la latencia es muy pequeña ($3 \times 500 = 1500$ (ms.)) y sólo son necesarias las tres últimas medidas para filtrar adecuadamente el ruido.

La figura 5.9 corresponde a una instantánea del *Mapa Local*. Las celdillas libres se han representado en color blanco, las ocupadas en color negro y en color gris las que poseen un estado desconocido. Puede observarse que la posición en la que se encuentra el robot (centro del mapa) se considera "libre", bajo la hipótesis de que físicamente no pueden coexistir robot y obstáculo en una misma celdilla; incluso en el caso de choque, el obstáculo se situaría en celdillas vecinas a las que ocupa el robot. Este mapa local se mueve solidario al robot, de forma que al desplazarse el robot una distancia prefijada, el mapa se centra en la nueva posición del robot.

El agente **Actualizar Mapa Local** actualiza el*Mapa Local* traduciendo, mediante un *modelo sensorial*, las medidas obtenidas por el láser de barrido en un valor de ocupación asociado a aquellas celdillas afectadas por la medida y posteriormente mediante una *regla de actualización*, actualizando el estado de las celdillas.

- **Modelo del sensor láser de barrido.** El modelo sensorial propuesto considera al haz láser como un sector circular, de modo que el valor medido por el láser en cada ángulo, se define como radio del sector en la dirección y sentido indicados. Las celdillas que en un instante dado del barrido intersecan con el arco del sector se consideran ocupadas y el resto de las que se encuentran en el interior del sector se consideran libres, figura 5.10.

Cada medida del láser de barrido se caracteriza por un par (ρ, θ), por lo tanto, teniendo

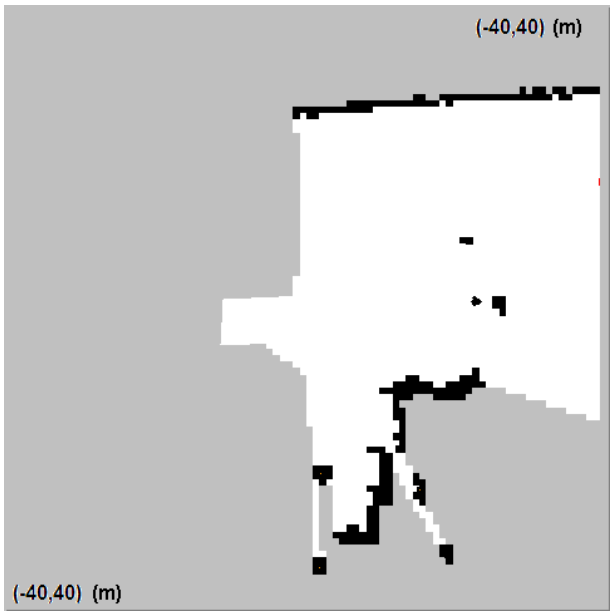


Figura 5.9 : Representación de estados de ocupación en el *Mapa Local* instantáneo. Se representa el espacio (libre/ocupado/desconocido) en color (blanco/negro/gris) respectivamente.

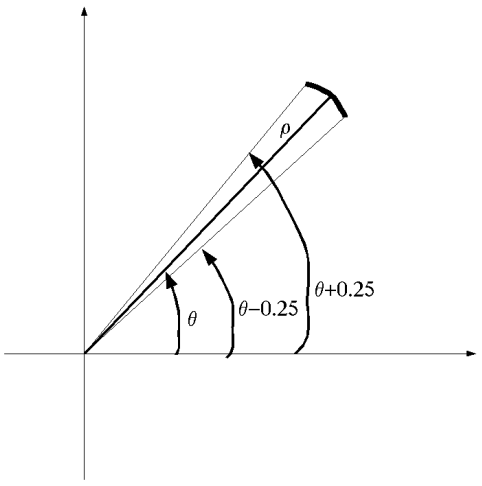


Figura 5.10 : Modelo del láser de barrido para la actualización del mapa de rejilla

en cuenta las características del láser descritas en el capítulo 4, sección 4.3, se asigna a todos los puntos (x_o, y_o) que verifican la ecuación (5.1) siendo $\psi \in [\theta - 0{,}25, \theta + 0{,}25]$ (grados) un valor numérico de ocupación $+1$, que permita realizar cálculos sobre la ocupación a lo largo del

tiempo.

$$\begin{aligned} x_o &= x_L + \rho cos(\psi) \\ y_o &= y_L + \rho sin(\psi) \end{aligned} \tag{5.1}$$

A las celdillas incluidas en el sector circular se les asigna un valor de ocupación -1. El resto de las celdillas no se ven afectadas en su valor de ocupación.

- **Actualización del nivel de ocupación de una celdilla.** La actualización del valor numérico de ocupación de cada celdilla se logra mediante un proceso de "votación" sobre los tres últimos valores de la memoria de ocupación que posee cada celdilla [Cañas y García-Pérez, 2002]. Así, el estado de la celdilla se define ocupado si al menos dos de los tres valores son $+1$, libre si al menos dos de los tres son -1 y desconocido en el resto de los casos. Esta regla de actualización dota de dinamismo al mapa de rejilla, ya que permite incorporar con rapidez tanto los nuevos obstáculos como los nuevos espacios libres, pero a la vez le proporciona cierta robustez frente a fallos espúreos en el sensor al tomar los tres últimos valores.

- **Entradas, salidas, representación y procesos del agente Actualizar Mapa Local.** El agente **Actualizar Mapa Local** dispone de una única entrada, la señal de activación, figura 5.8 y una única salida, la actualización del mapa. Sus procesos operan con la representación instantánea proporcionada por los datos láser crudos y la posición del robot. En la figura 5.11 se muestra el diagrama de flujo de información del agente **Actualizar Mapa Local**. En primer lugar, una vez que **Actualizar Mapa Local** recibe la señal de activación, envía una señal de activación al agente **Actualizar Posición**, pues necesita disponer de esta información actualizada. A continuación inicia su ciclo de ejecución, con un tiempo asociado a este proceso de 500 (ms.). El valor de este periodo de actualización viene determinado por la frecuencia de barrido del sensor láser (500 (ms.)), y no existe ninguna otra razón para actualizar el mapa local

a mayor frecuencia, ya que de este modo se han sincronizado los ritmos de actualización del láser y del mapa local asegurando que ni se pierden ni se repiten medidas láser.

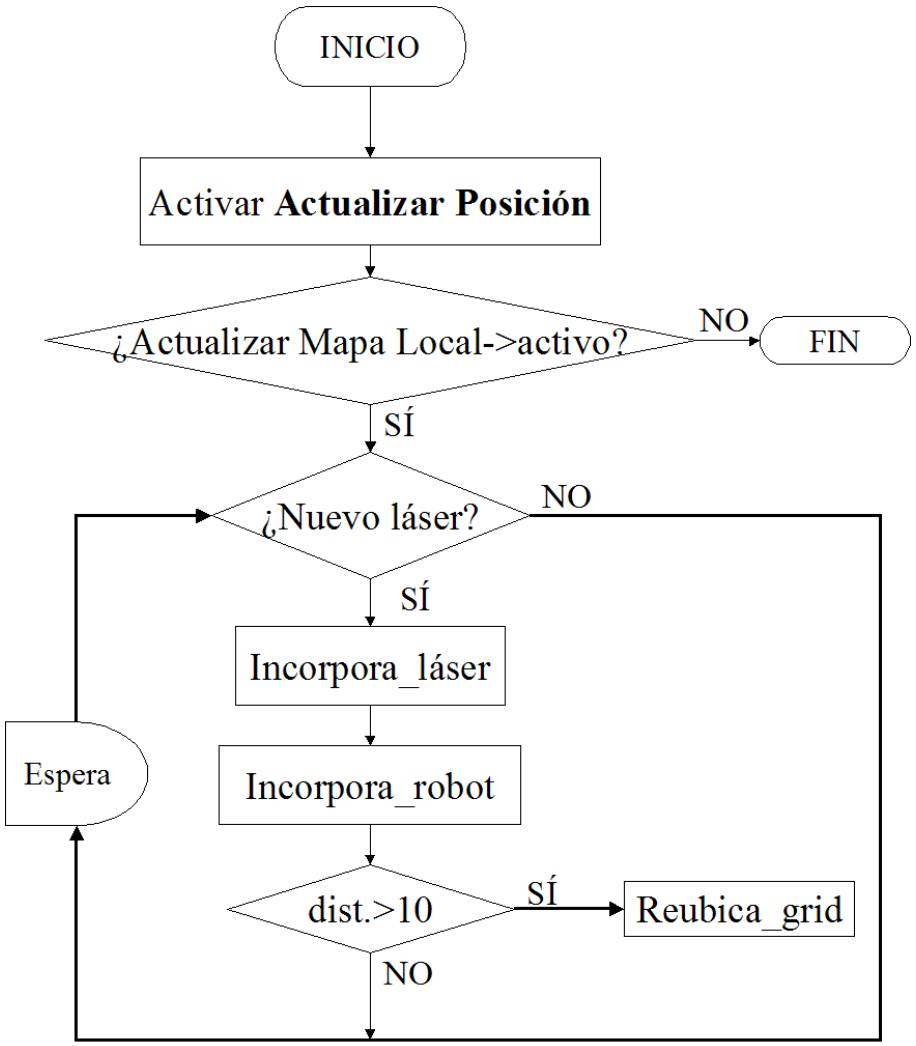


Figura 5.11 : Diagrama de flujo de información en el agente **Actualizar Mapa Local**

En cada ciclo de ejecución **Actualizar Mapa Local** comprueba que el agente **Actualizar Posición** se encuentra en estado activo, y si no es así genera una señal de FALLO finalizando el proceso. En caso contrario, y si dispone de nuevos datos del láser, los incorpora al *Mapa Local*, incorporando también el objeto robot conforme a su posición y dimensiones. Tras incorporar toda la información relativa a la ocupación del mapa, proporcionada por el sensor láser y por la *Posición* del robot, se considera actualizada la representación *Mapa Local* y se procede a reubicar el *Mapa Local* centrándolo en la posición que ocupa el robot. Este proceso final, solo se

realiza si el campo visual del sensor láser presenta mayoritariamente zonas que no se encuentran en el *Mapa Local*. Finalmente, se calcula el tiempo invertido por los diferentes procesos y se activa un proceso de espera con un tiempo asociado igual a $500 - t_{proceso}$ (ms.).

Agente Actualizar Obstáculos

La navegación de robots requiere de estrategias perceptivas y de toma de decisión que tengan en cuenta todas las características y restricciones impuestas tanto por el tipo de vehículos como por los entornos naturales [Devy et al., 1995]. Las distancias de las estructuras a percibir, unido a las dimensiones, velocidad y capacidad de maniobra de los robots de exteriores, hace que su dotación sensorial y de actuación sea muy diferente a la de aplicaciones en interior. Además, la mayoría de los robots de interior son holonómicos, sin embargo el modelo cinemático más habitual en vehículos de exteriores es el tipo coche y estos vehículos son de grandes dimensiones. Su reducida capacidad de maniobra en espacios pequeños obliga a planificar con suficiente antelación la maniobra para alcanzar una posición con una orientación determinada.

En el caso de robots de exteriores la estrategia más sencilla a seguir ante la aparición de un obstáculo -que puede ser dinámico- en la trayectoria que sigue el robot, consiste en detener el vehículo hasta que éste desaparezca y si el obstáculo permanece, solicitar la asistencia de un operario [Stentz et al., 2002]. Con la finalidad de incrementar la autonomía del robot, en este trabajo se han investigado estrategias que incluyan evitar los obstáculos estáticos imprevistos, si existe suficiente espacio libre, en vez de esperar o enviar un mensaje al operario. Los algoritmos de navegación adecuados para distancias cortas normalmente no son fáciles de transportar a entornos de exteriores por las grandes diferencias de los entornos y de las respuestas de los sensores y actuadores que se utilizan en cada caso [Nehmzow y Owen, 2000]. Para facilitar las maniobras de algunos vehículos de exteriores, es necesario detectar los obstáculos con suficiente antelación para iniciar a tiempo la maniobra más conveniente para evitarlos [Miura et al., 1999], aunque esto suponga en algunos casos dar un rodeo. Si sólo se dispone de una representación instantánea de los obstáculos, las técnicas utilizadas habitualmente en interiores para evitar obstáculos no proporcionan la anticipación necesaria para llevar a cabo la

maniobra correctamente y pueden dar lugar a trayectorias inapropiadas. Una de estas situaciones se muestra en la figura 5.12, donde el robot avanza guiado por un control reactivo sencillo para evitar obstáculos, que le obliga a rodearlos por la dirección y sentido en la que percibe mayor espacio libre. Ahora bien, ante un obstáculo móvil que cruza la trayectoria del robot de derecha a izquierda, esta estrategia de evitar obstáculos, da lugar a un comportamiento de persecución no deseable. Si se dispusiese de información relativa al movimiento del obstáculo, una estrategia apropiada en este caso sería acelerar para pasar antes que el obstáculo o bien frenar para dejarle paso al objeto y continuar posteriormente. En ambos casos, además de evitar la colisión con el obstáculo se evita la desviación respecto de la trayectoria óptima que seguía el robot.

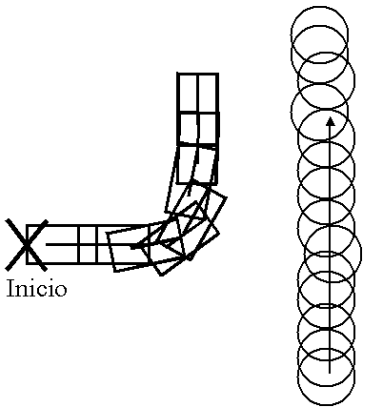


Figura 5.12 : Desviación incorrecta del vehículo para evitar un objeto dinámico al disponer únicamente de información instantánea del mundo

Este ejemplo ilustra la conveniencia de identificar y clasificar los obstáculos en estáticos y dinámicos, caracterizando los parámetros de movimiento de los objetos, a fin de activar la mejor estrategia para evitar colisión en el tiempo más corto posible [Prassler et al., 2000], generando actuaciones semejantes a las de un conductor experto. El objetivo del agente **Actualizar Obstáculos** es desarrollar una estrategia de detección y caracterización de obstáculos imprevistos a partir de la estimación de la velocidad de los objetos. A partir de esta información perceptiva materializada en la percepción *Obstáculos*, el robot puede activar estrategias de navegación selectivas.

- **Entradas, salidas, representación y procesos del agente Actualizar Obstáculos.** El agente **Actualizar Obstáculos**, tiene como entradas: 1) el *Mapa Local*, 2) la *Posición del robot* actualizada y 3) la señal de activación. Por ello, una vez activado inicia a su vez la activación de los agentes **Actualizar Mapa Local** y **Actualizar Posición**, necesarios para proceder a una localización correcta de los obstáculos, figura 5.13. El agente **Actualizar Obstáculos** mantiene actualizada la percepción *Obstáculos* que contiene información sobre las características de movimiento, incluyendo una estimación de la velocidad, y localización de los obstáculos detectados en el entorno próximo, 3 (m) de radio, con el sensor láser de barrido.

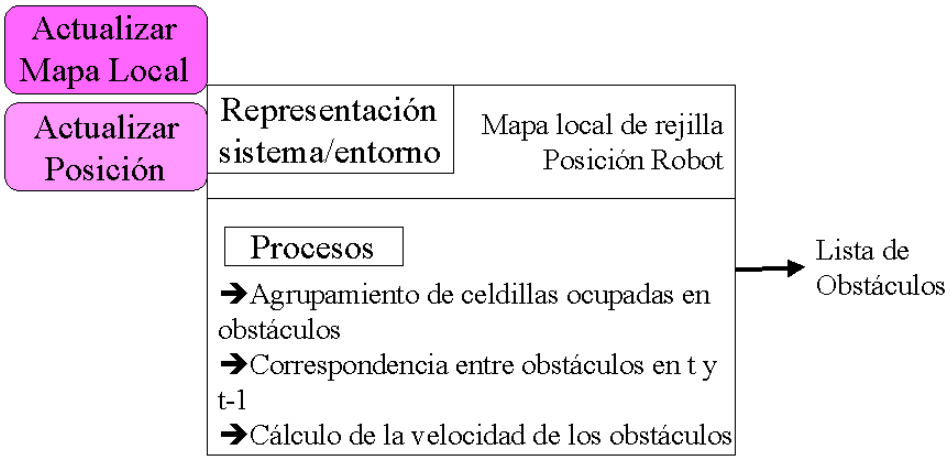


Figura 5.13 : Esquema de entradas, salidas, representación y procesos en el agente **Actualizar Obstáculos**

El diagrama de flujo de información del agente **Actualizar Obstáculos** se muestra en la figura 5.14. Los procesos que contempla se desencadenan siguiendo una secuencia. Primero, envía las señales de activación a los agentes **Actualizar Posición** y **Actualizar Mapa Local**, para garantizar que las percepciones *Posición* y *Mapa Local* están actualizadas. Tras inicializar los obstáculos comienza el ciclo iterativo de ejecución, con un tiempo de 500 (ms.) limitado por la frecuencia de actualización del *Mapa Local* que es de 2 (Hz.). Este ciclo se repite mientras el agente **Actualizar Obstáculos** se encuentra en estado **activo**. En cada iteración **Actualizar Obstáculos** comprueba si alguno de los agentes **Actualizar Posición** y **Actualizar Mapa Local** se halla **inactivo**, en cuyo caso genera una señal de FALLO y finaliza el proceso. Si no es así, actualiza el *Mapa Local* y continúa el proceso iterativo.

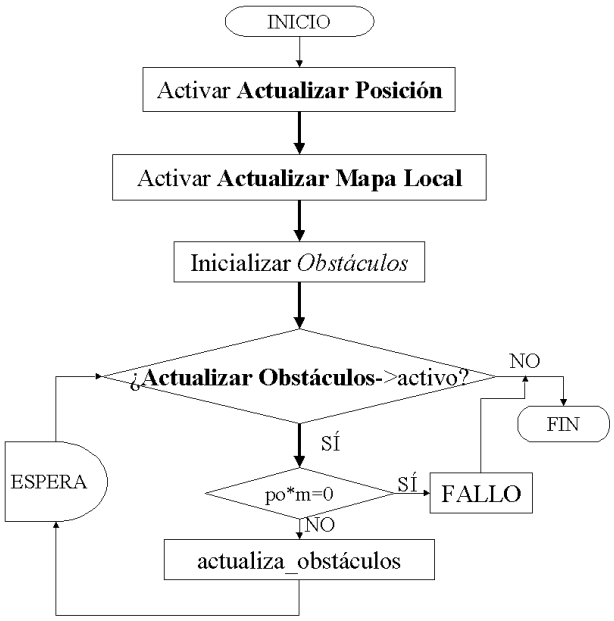


Figura 5.14 : Diagrama de flujo de información del agente **Actualizar Obstáculos**

- **Obtención y caracterización de *Obstáculos*.** A partir del *Mapa Local* es posible agrupar las celdillas ocupadas en conjuntos de celdillas conexas, que se denominarán *Obstáculos*, aplicando un algoritmo de segmentación basado en la detección de las celdillas colindantes. Los *Obstáculos* se almacenan, en lo que se denomina Lista de Obstáculos ($OL(t) = O_1(t), O_2(t), ..., O_n(t)$), donde n es el número de *Obstáculos* detectados en el tiempo t. Cada *Obstáculo* se halla caracterizado por las coordenadas y velocidad de su centroide y tiene el siguiente conjunto de propiedades (5.2):

$$O_i(t) = \begin{bmatrix} ID_i \\ celdilla_i\{j,k\} \\ X_{centroide}\{r\} \\ Y_{centroide}\{r\} \\ v_{centroide}\{s\} \\ \theta_{centroide}\{s\} \end{bmatrix} \tag{5.2}$$

La primera de las propiedades constituye la identificación del objeto, la segunda representa las celdillas que pertenecen al obstáculo, la tercera y la cuarta corresponde a las coordenadas

cartesianas del centroide (definido según las ecuaciones (5.3)) y las dos últimas definen el módulo y la dirección del vector velocidad del centroide. Las coordenadas del centroide y el vector velocidad se almacenan en una memoria FIFO, donde $1 \leq r \leq 10$ es el índice de las últimas 10 posiciones y $1 \leq s \leq 9$ el índice del vector velocidad.

Esta representación de los *Obstáculos* permite establecer la correspondencia entre los *Obstáculos* detectados en dos instantes de tiempo consecutivos. Una vez que el proceso de correspondencia ha finalizado, se actualizan los parámetros de los *Obstáculos*. Si en la nueva lista de obstáculos aparece algún conjunto de celdillas conexas que no corresponde a ninguno de los existentes previamente, éste se considera un obstáculo nuevo.

- **Proceso de segmentación de los *Obstáculos*.** El proceso de segmentación transforma la representación de celdillas del *Mapa Local* en una representación de celdillas agrupadas u *Obstáculos* con una granularidad y nivel de abstracción superior. Esta transformación se realiza mediante un algoritmo de agrupamiento por vecinos más próximos a fin de constituir conjuntos de celdillas ocupadas adyacentes. Se trata de un algoritmo de búsqueda de las celdillas ocupadas adyacentes, que almacena simultáneamente en un registro las celdillas ya visitadas para reducir el tiempo de cómputo. Una vez que se han visitado todas las celdillas de un grupo, el proceso se dirige a la siguiente celdilla no explorada y el proceso se repite de forma iterativa. La búsqueda se realiza únicamente sobre las celdillas ocupadas para reducir al máximo el coste computacional.

Al final del proceso de segmentación se obtiene un conjunto de *Obstáculos*, donde las coordenadas de su centroide se calculan mediante las expresiones (5.3):

$$\begin{aligned} X_{cent_i}(t) &= \frac{\sum_{k=1}^{M} Xcel_k(t)}{M} \\ Y_{cent_i}(t) &= \frac{\sum_{k=1}^{M} Ycel_k(t)}{M} \end{aligned} \tag{5.3}$$

donde M es el número de celdillas del *Obstáculo* i, donde $1 \leq i \leq N$, siendo N el número

de *Obstáculos* en la lista. El vector velocidad se define por su módulo y dirección (5.4), y se utilizará en el proceso de correspondencia entre *Obstáculos*.

$$\begin{aligned} & v_{cent_i} = \\ & \frac{\sqrt{[X_{cent_i}(t) - X_{cent_i}(t-1)]^2 + [Y_{cent_i}(t) - Y_{cent_i}(t-1)]^2}}{\Delta t} \\ & \theta_{cent_i} = \arctan\left[\frac{Y_{cent_i}(t) - Y_{cent_i}(t-1)}{X_{cent_i}(t) - X_{cent_i}(t-1)}\right] \end{aligned} \tag{5.4}$$

Inicialmente $v_{cent} = 0$ y $\theta_{cent} = 0$ para todos los *Obstáculos* detectados. Una vez que se ha establecido la primera correspondencia se actualizan los valores iniciales de la velocidad.

- **Correspondencia y seguimiento de *Obstáculos*.** Una vez que se han calculado las posiciones y velocidades de los *Obstáculos*, ecuaciones (5.3) y (5.4), es necesario establecer la correspondencia entre los *Obstáculos* detectados en t con los existentes en $t-1$ con el fin de poder actualizar sus propiedades. El algoritmo de correspondencia realiza una búsqueda en la lista de *Obstáculos* en t, para seleccionar aquellos cuyas posiciones están más próximas que un umbral a las posiciones predichas en t, para los *Obstáculos* registrados en $t-1$. La predicción de estas posiciones se lleva a cabo a partir de las expresiones (5.5):

$$\begin{aligned} XPRED_{cent_i}(t) &= X_{cent_i}(t-1) + [v_{cent_i} \cdot \cos(\theta_{cent_i})] \cdot \Delta t \\ YPRED_{cent_i}(t) &= Y_{cent_i}(t-1) + [v_{cent_i} \cdot \sin(\theta_{cent_i})] \cdot \Delta t \end{aligned} \tag{5.5}$$

Todos aquellos *Obstáculos* cuyas posiciones predichas en t se encuentren dentro de un círculo de radio R centrado en la posición del *Obstáculo* i en t son candidatos a ser la "pareja" de ese *Obstáculo*. Aquel candidato que comparta mayor número de celdillas con él será seleccionado como el candidato más adecuado. Al nuevo obstáculo detectado en t le corresponderá su identificación ID y consecuentemente se actualizará el valor del vector velocidad.

El proceso de correspondencia se muestra en la figura 5.15, donde se presentan dos obstáculos "viejos" O_0 y O_1 y uno nuevo O_a. Los círculos fijan el umbral alrededor de la posición predicha en t para los *Obstáculos* O_0 y O_1. Puesto que el centroide de O_a se encuentra dentro del círculo correspondiente a O_1 y fuera del de O_0, se selecciona O_1 como su pareja. Por lo tanto se considera que O_a se corresponde con O_1 en t y se le asigna su identificador $ID = 1$.

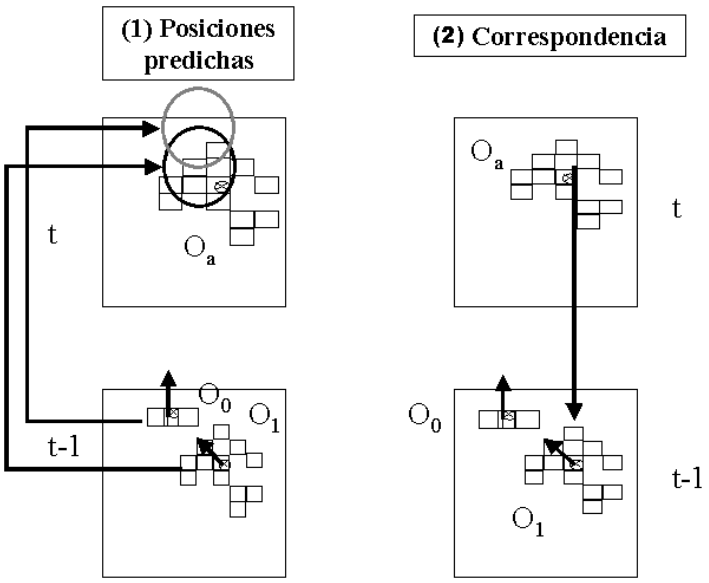


Figura 5.15 : Correspondencia entre *Obstáculos* en tiempos consecutivos. Los *Obstáculos* se representan mediante agregados de celdillas cuadradas. Los círculos corresponden a las posiciones predichas para los *Obstáculos* más próximos ya registrados en $t - 1$.

Los *Obstáculos* en t para los cuales no se ha encontrado correspondencia con ninguno de los registrados en $t - 1$ se consideran nuevos *Obstáculos* y su velocidad se inicializa a 0. Los *Obstáculos* en $t - 1$ que no se corresponden con ninguno en t se eliminan de la lista de *Obstáculos*. Al final del proceso de correspondencia, se obtiene la nueva lista de *Obstáculos* cuyas propiedades hay que actualizar. Los componentes de la velocidad se actualizan conforme a las expresiones (5.4).

Algunas de las ventajas del algoritmo propuesto son:

- La mayor robustez del algoritmo en el cálculo de la correspondencia al considerar la velocidad de los *Obstáculos* en este proceso, dado que se busca la correspondencia entre *Obstáculos* alrededor de la posición predicha, que se ha calculado a partir del valor de la

velocidad.

- La primera selección de candidatos reduce la búsqueda ya que se calcula el grado de solapamiento de celdillas únicamente para los *Obstáculos* más próximos.
- El criterio de selección de candidatos por solapamiento máximo de celdillas evita las asociaciones incorrectas entre *Obstáculos*.

- **Actualización de la velocidad de los *Obstáculos*.** El vector velocidad se calcula a partir del módulo v_{cent} y de la dirección de movimiento θ_{cent}. Los valores de la velocidad en las últimas 9 iteraciones se registran desde la primera iteración en una matriz de dos columnas $(v_{cent}, \theta_{cent})$.

Un aspecto importante al estimar la velocidad es la posibilidad de distinguir entre la observación de un objeto nuevo, que entra en el campo de visión del sensor, por estar ocluido o fuera de rango en instantes anteriores, y un objeto móvil [Prassler et al., 2000]. Para solventar esta cuestión se utiliza el estado "desconocido" previamente asignado a celdillas del *Mapa Local*. Se define una variable *nueva_aparición* que toma el valor CIERTA si más de la mitad de las celdillas que forman el *Obstáculo* se encuentra en el estado "desconocido" y FALSA en caso contrario. De este modo si *nueva_aparición* es CIERTA significa que la mayoría de las celdillas del *Obstáculo* son de nueva aparición en el *Mapa Local* y por lo tanto no debe calcularse aún el valor de la velocidad puesto que no se dispone de suficiente información. Únicamente se calcula el valor de la velocidad cuando la variable *nueva_aparición* es FALSA en las dos últimas iteraciones.

Los *Obstáculos* cuya velocidad es menor que $0{,}04$ (m/s) (para una latencia de 10×500 (ms) y resolución de la celdilla de 20 (cm)) se definen como estáticos. Obviamente en estos casos no se puede detectar variación alguna en la posición entre los instantes consecutivos t y $t-1$. Si el valor estimado del módulo de la velocidad se halla próximo a 0, se procede a recalcular el valor de la velocidad, pero esta vez entre los instantes t y $t-2$. Puesto que siempre se almacenan las últimas 10 posiciones del centroide para cada uno de los *Obstáculos* puede

procederse iterativamente hasta llegar a recalcular la velocidad entre los instantes t y $t-9$, con la intención de encontrar en algún momento un valor superior a $0{,}04$ (m/s).

5.2.2 Agentes de actuación

En este apartado se presentan los agentes de actuación diseñados y verificados de forma gradual desde el primero hasta el último, que es un planificador dedicado a la optimización de la trayectorias de navegación. Todos ellos siguen el mismo diseño modular esquematizado en la figura 5.1, con las entradas y salidas del agente, la representación interna del mundo en la que basa sus procesos de razonamiento para la toma de decisiones que se dirigen hacia otros agentes o se registran en la memoria global compartida del sistema o pizarra, y los procesos que dan lugar a su comportamiento.

Cada agente surge ante la necesidad de incrementar una habilidad que no puede conseguirse con la activación y modulación de los agentes de actuación existentes previamente. Los agentes **Parar**, **Evitar Obstáculos** y **Avanzar** constituyen los bloques básicos que soportan los comportamientos observables de navegación cada vez más complejos. La navegación global general en AGRO-AMARA se sustenta en las habilidades básicas de estos agentes. Este conjunto básico de agentes permiten la navegación global del tractor como se verá en los resultados experimentales del capítulo 6.

Agente Parar

El agente **Parar** es el bloque básico imprescindible para garantizar una navegación segura desde los primeros desplazamientos. Su objetivo es la parada inmediata del tractor siempre que se perciba una situación de peligro para la integridad del sistema/entorno. Es el agente de seguridad básico ante colisiones y vuelcos.

Una vez que pasa al estado **activo** el agente **Parar** envía la señal de parada a DÉDALO-Servidor. Su activación proviene de la detección de una colisión o de una inclinación lateral (p_l) o frontal (p_f) superior al umbral de seguridad, fijado en $15\,\%$. El agente **Parar** tiene como

entradas las lecturas sensoriales del sensor de colisión alojado en el parachoques frontal y las lecturas de los dos inclinómetros, tal y como se muestra en la figura 5.16.

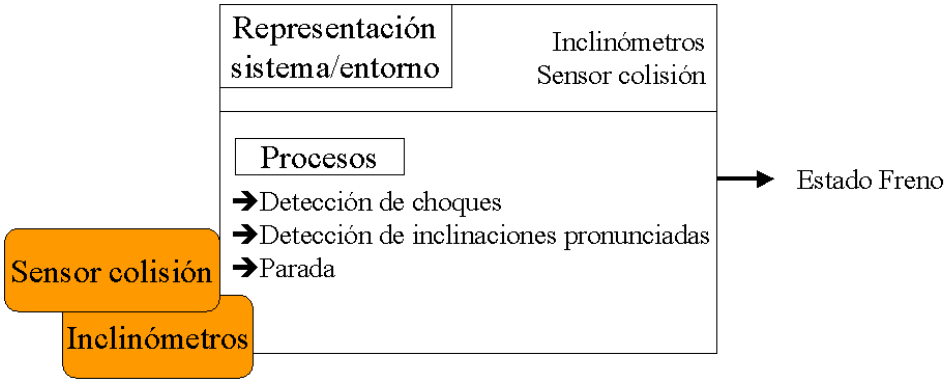


Figura 5.16 : Esquema de entradas,salidas, representación y procesos del agente **Parar**

Parar es un agente de actuación, por lo tanto cuando recibe la señal de activación de otro agente, se produce una transición del estado **inactivo** al de **alerta**. En el estado de **alerta**, el agente **Parar** comprueba si se da su contexto de activación, que en este caso es la detección de colisión o de inclinación superior al umbral. Si se verifican estas condiciones, el agente **Parar** pasa al estado **activo** y ejecuta los procesos que detienen el tractor. La toma de decisión es puramente reactiva y la acción es el envío al servidor de la señal de parada inmediata del robot. Si dejan de producirse las condiciones anteriores, pasa de nuevo al estado de **alerta**. Como el resto de agentes, el agente **Parar** se ejecuta cíclicamente, con un tiempo de ciclo de $T = 500$ (ms.). El diagrama de flujo de información de la figura 5.17 muestra el modo de operación de este agente.

Agente Evitar Obstáculos

La habilidad más deseada en un robot móvil es la de navegar de modo autónomo [Nehmzow y Owen, 2000], que normalmente implica el desplazamiento desde una localización inicial a una final evitando los obstáculos que puedan aparecer en su trayectoria. En el campo de la robótica se han desarrollado múltiples y variados algoritmos para evitar obstáculos

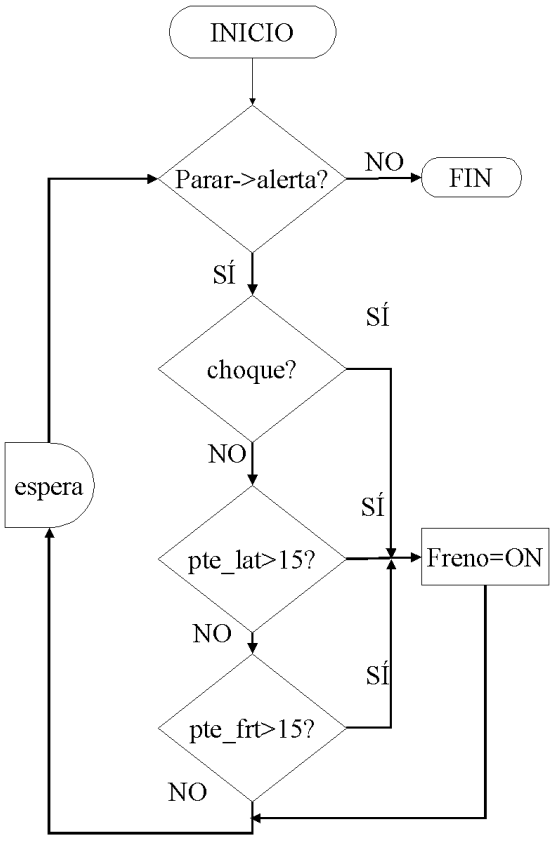


Figura 5.17 : Diagrama de flujo de información del agente **Parar**

[Fox et al., 1997, Simmons, 1996, Borenstein y Koren, 1991], la mayoría de ellos basados en la medida instantánea de la distancia entre el obstáculo y el robot. Estos algoritmos se han utilizado con éxito en interiores, mayoritariamente con obstáculos estáticos y en algún caso con móviles. Su eficacia depende de la rapidez del lazo de control reactivo implementado en cada caso y de la capacidad de maniobra de los robots de interiores, habitualmente de dimensiones reducidas y holonómicos.

Las aplicaciones en exteriores, entre las que se encuentra la Agricultura de Precisión, requieren estrategias de percepción y decisión más complejas para hacer frente a las dificultades inherentes a los entornos parcialmente estructurados [Devy et al., 1995]. Como se comentaba al describir el agente **Actualizar Obstáculos**, si se quiere aumentar la autonomía del robot es necesario dotarle con mecanismos para evitar los obstáculos que aparezcan en su camino. El agente **Evitar Obstáculos** se encarga de esta tarea, basado en la representación de *Obstáculos*

elaborada por el agente **Actualizar Obstáculos**.

- **Entradas, salidas y representación en el agente Evitar Obstáculos.** El agente **Evitar Obstáculos** garantiza la ausencia de colisiones entre el robot y los obstáculos imprevistos que pueden aparecer en su trayectoria, razonando para tomar una decisión sobre cual de las posibles alternativas constituye la estrategia más eficaz en cada caso.

  - **Entradas y salidas.** Como se muestra en la figura 5.18 el agente **Evitar Obstáculos** tiene como entradas la señal de activación, proveniente de otro agente, y las percepciones *Obstáculo* y *Posición*, y como salidas los valores del ángulo de giro de la dirección y la posición del freno, que se envían a los controladores de bajo nivel del robot. Puesto que requiere las percepciones *Obstáculo* y *Posición*, el agente **Evitar Obstáculos** envía una señal de activación a los agentes encargados de mantenerlas actualizadas, **Actualizar Obstáculos** y **Actualizar Posición**, respectivamente.

  - **Contexto de activación.** La condición que permite al agente **Evitar Obstáculos** pasar del estado **alerta** al estado **activo** es la complementaria a la de los agentes **Parar** y **Avanzar**, es decir que no se haya detectado colisión, que $p_f, p_l < 15$ y que exista algún obstáculo que ocupe más de 3 celdillas del *Mapa Local* en un círculo de radio 4 (m.) centrado en el robot. 4 (m.) es el espacio necesario para que el robot DÉDALO, en función de su capacidad de giro, pueda evitar un obstáculo con seguridad.

- **Procesos en el agente Evitar Obstáculos.** Cuanto mayores sean las dimensiones del robot más amplio será el rodeo que tiene que dar para esquivar un obstáculo, tal y como se muestra en simulación en la figura 5.19. Por lo tanto el coste de evitar un obstáculo en tiempo y espacio recorrido es alto. Por ello, el agente **Evitar Obstáculos** aprovecha la percepción *Obstáculo* para diseñar una estrategia de comportamiento diferente si se trata de un *Obstáculo* móvil o bien de uno estático. La estrategia es como sigue: si el *Obstáculo* es dinámico el robot se para y espera a que desaparezca. Si el *Obstáculo* es estático, el robot le rodea. Además se definen dos zonas de

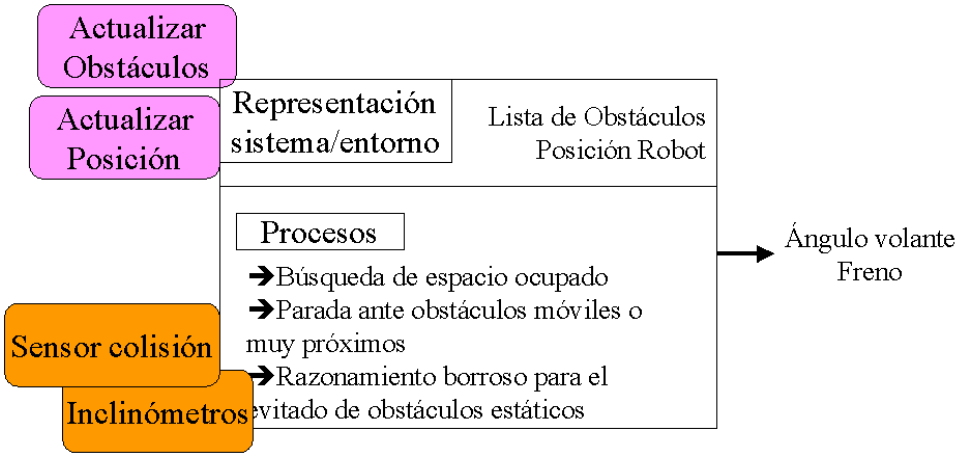


Figura 5.18 : Esquema de entradas, salidas, representación y procesos del agente **Evitar Obstáculos**

operación, una próxima al robot, de alta seguridad y otra más lejana, de seguridad, tal y como se muestra en la figura 5.20. Cualquier *Obstáculo* detectado en la zona de alta seguridad hace que el robot se pare, puesto que no existe espacio libre para poder realizar una maniobra y evitarlo. Los *Obstáculos* que entran en la zona de seguridad se evitan siempre que sean estáticos.

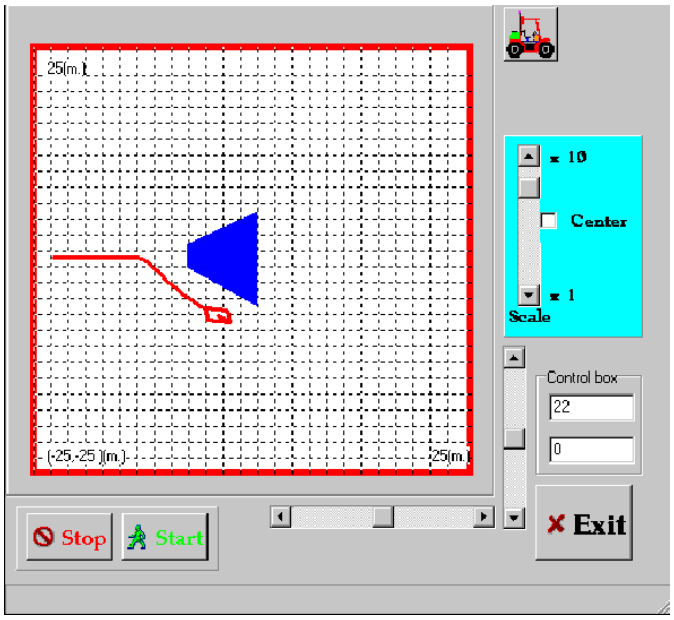


Figura 5.19 : Trayectoria generada por el robot para rodear un obstáculo en simulación

Para rodear los *Obstáculos* se sigue una estrategia borrosa de control. El controlador calcula el ángulo de giro en función de la posición del *Obstáculo* con respecto al robot. Como puede

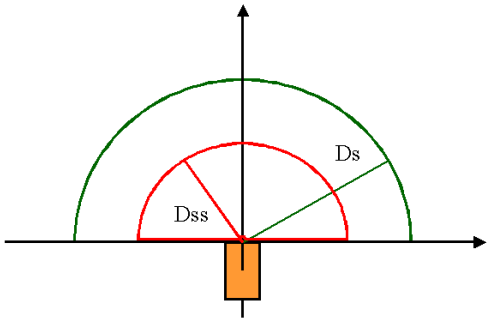


Figura 5.20 : Zonas de seguridad (en color verde) y de máxima seguridad (en color rojo) definidas para el agente **Evitar Obstáculos**

existir más de un *Obstáculo* dentro de la zona de seguridad, se obtiene la salida del controlador para cada *Obstáculo* y se elige el valor medio como salida del controlador. El controlador borroso tiene por lo tanto una única entrada -el ángulo de la línea recta que une el centroide de un obstáculo con el robot (γ)-, y una única salida para cada obstáculo -el ángulo de giro del vehículo (α)-, figura 5.21.

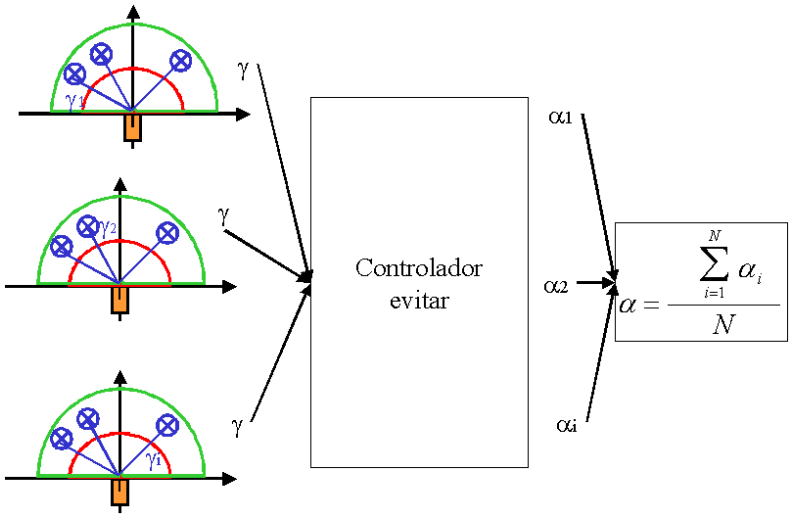


Figura 5.21 : Controlador borroso implementado en el agente **Evitar Obstáculo**

Se calcula la salida del controlador: α_i para cada *Obstáculo* que se encuentre dentro de la zona de seguridad, es decir $O_i/d(O_i, R) < d_S$. La salida final se define mediante

la expresión, de manera análoga a los campos de potencial definidos por Borenstein en [Borenstein y Koren, 1991]:

$$\alpha = \frac{\sum_{i=1}^{N} \alpha_i}{N}$$

La variable de entrada del controlador borroso, γ se describe mediante tres etiquetas lingüísticas, “derecha”, “centro” e “izquierda” y corresponden al valor borroso de la localización del *Obstáculo* con respecto del robot. Se representan mediante las funciones de pertenencia que se muestran en la figura 5.22.

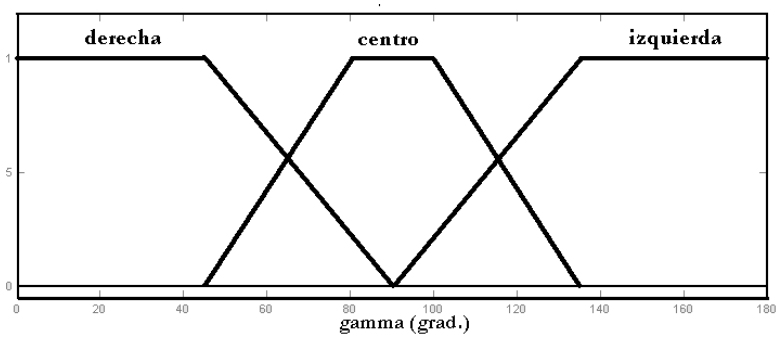


Figura 5.22 : Conjuntos borrosos de la variable γ

La variable de salida es el ángulo de giro del robot, α, que se define mediante dos etiquetas lingüísticas: “derecha” e “izquierda” representadas por las funciones de pertenencia monótonas de la figura 5.23.

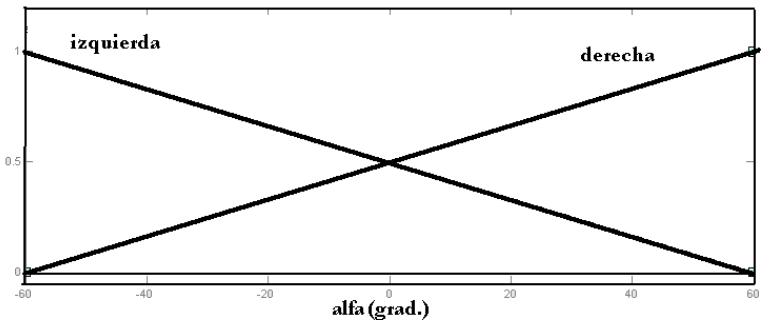


Figura 5.23 : Conjuntos borrosos de la variable α

Tres reglas son suficientes para modelar el control de la maniobra de evitar *Obstáculos*, tabla 5.3.

IF	$gamma = izquierda$	THEN	$\alpha = derecha$
IF	$gamma = centro$	THEN	$\alpha = derecha$
IF	$gamma = derecha$	THEN	$\alpha = izquierda$

Tabla 5.3 : Reglas del controlador borroso implementado en el agente **Evitar Obstáculos**

Tal y como se indica en el diagrama de flujo de información del agente **Evitar Obstáculos**, figura 5.24, una vez que éste recibe la señal de activación y cambia de estado **inactivo** al estado **alerta**, envía las señales de activación a los agentes **Actualizar Obstáculos** y **Actualizar Posición**. A continuación ejecuta las rutinas de iniciación propias (apertura del fichero de texto con la base de reglas del controlador y enlace de variables). Mientras el agente **Evitar Obstáculos** permanece en estado **alerta** o **activo** ejecuta cada 500 (ms.) el siguiente ciclo. Primero comprueba si existe o no colisión solicitando las medidas del sensor de choques y de los de inclinación, para verificar si las pendientes lateral (p_l) y frontal (p_f) son mayores de 15. En cualquiera de los tres casos (colisión, pendiente lateral superior al umbral o pendiente frontal superior al umbral) pasa el estado **alerta**. Para cada *Obstáculo* presente comprueba en primer lugar si se encuentra dentro de la zona de máxima seguridad: $d(O_i, R) < d_{SS}$ ($d_{SS} = 1$ (m.) en los experimentos). Si esto es cierto, pasa al estado **activo**, envía la señal de parada y ya no analiza el resto de los *Obstáculos*, pues basta que exista uno en la zona de máxima seguridad para parar, ya que el robot DÉDALO no tiene en este caso suficiente espacio para esquivarlo. Espera el tiempo de ciclo 500 (ms) y comienza de nuevo.

Si el *Obstáculo* verifica que $d(O_i, R) > d_{SS}$, no está en la zona de máxima seguridad, entonces se comprueba si se encuentra dentro de la zona de seguridad, $d(O_i, R) < d_S$ ($d_S = 3$ (m.)). Si el *Obstáculo* está fuera de esta zona, se pasa al siguiente *Obstáculo*. Si está dentro, se examina su velocidad. En caso de que el *Obstáculo* sea móvil, $v_i \neq 0$ el agente pasa a estado **activo**, envía la señal de frenado y ya no analiza el resto de los *Obstáculos* sino que espera el tiempo de ciclo correspondiente. Si el *Obstáculo* es estático, calcula con el controlador borroso el ángulo de giro correspondiente y pasa al siguiente *Obstáculo*.

Si una vez finalizado el análisis de todos los *Obstáculos*, ninguno está dentro de la zona de seguridad o de máxima seguridad, las condiciones de ejecución de **Evitar Obstáculos** no se

cumplen, y por lo tanto pasa al estado **alerta**. En caso contrario pasa al estado **activo** y calcula el valor final del ángulo de giro del robot. Espera el tiempo de ciclo correspondiente y se ejecuta de nuevo.

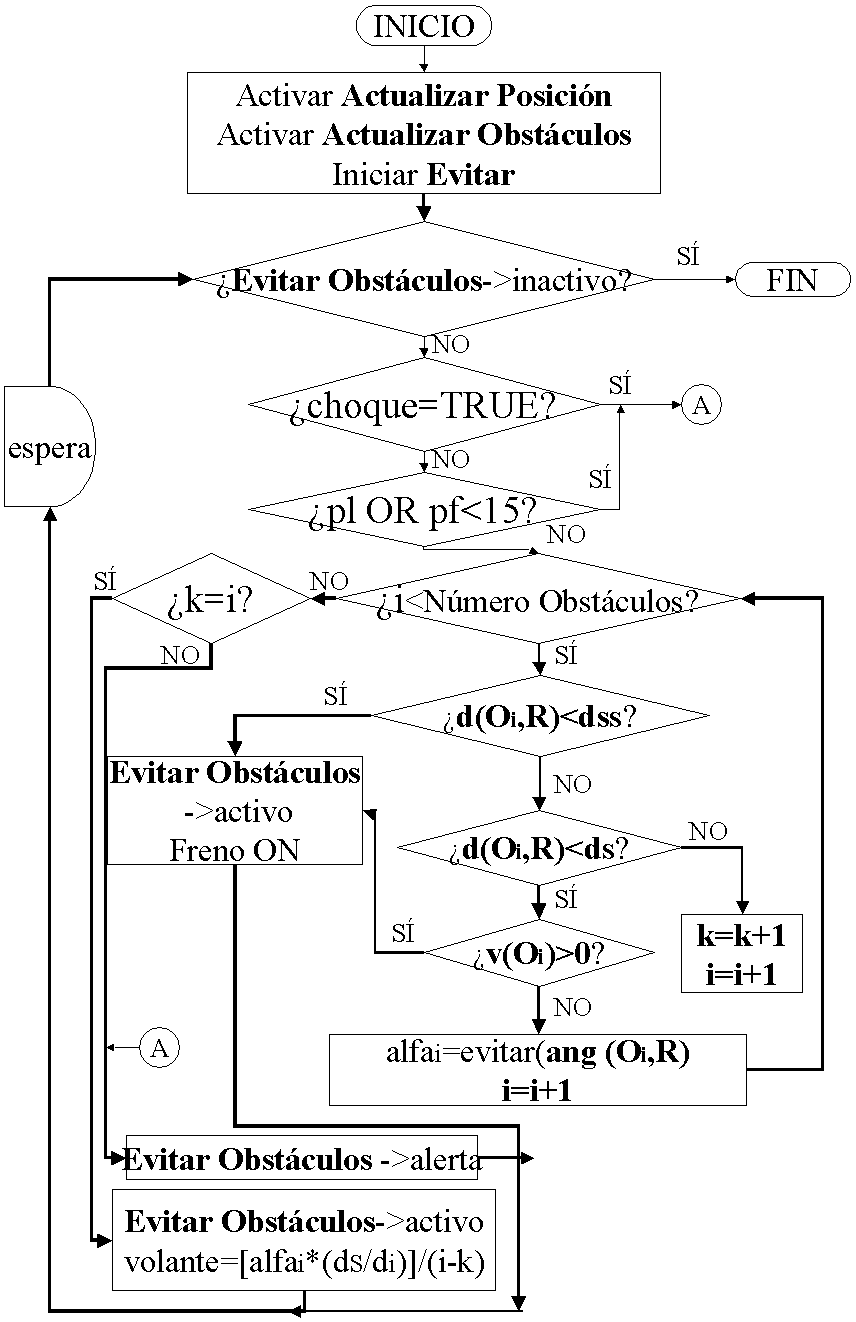


Figura 5.24 : Diagrama de flujo de información del agente **Evitar Obstáculos**

Agente Avanzar

El objetivo del agente **Avanzar** es dirigir al tractor desde una posición inicial expresada como (x_R, y_R, θ_R) hasta una posición final con una orientación determinada, lo que se denominará en adelante *objetivo orientado* (x_G, y_G, θ_G). La mayoría de los robots comerciales y de investigación en interiores son holonómicos, teniendo desacoplados los movimientos de traslación y rotación. En estos casos alcanzar un objetivo orientado consiste en alcanzar las coordenadas 2D, y una vez allí, rotar para lograr la orientación deseada. Los robots no-holonómicos, como son los vehículos y tractores comerciales, se desplazan y giran simultáneamente para conseguir llegar a un destino con la orientación deseada.

Conducir y aparcar un vehículo no-holonómico requiere una maniobra que puede ser compleja y no trivial para alcanzar una localización final con una orientación precisa. Muchos de los trabajos dedicados a la navegación de robots no-holonómicos [Manikonda et al., 1995, Scheuer y Fraichard, 1996, Paromtchik et al., 1997] se dirigen hacia el diseño de una trayectoria analítica compuesta por segmentos rectilíneos, conectados entre sí mediante arcos de circunferencia o clotoides [Scheuer y Fraichard, 1996]. Este planteamiento del problema de la navegación implica dos pasos, el primero la planificación de una trayectoria que pueda ser realizada por el robot y el segundo el desarrollo de un algoritmo de control para seguir la trayectoria. Esta secuencia de planificación-ejecución tiene el inconveniente bien conocido de la carencia de capacidad de reacción. Dentro de este contexto, Minguez et al. [J. Mínguez, 2002] contemplan las restricciones cinemáticas del robot dentro de la representación espacial del entorno, lo que permite que el robot se mueva en este espacio modificado utilizando estrategias reactivas. Sin embargo no abordan el problema de alcanzar un objetivo orientado.

Ollero et al. [Ollero et al., 1997] desarrollan un método de seguimiento explícito de caminos donde un sistema de reglas borrosas controla el seguimiento de un trayectoria por un robot holonómico pequeño, obteniendo buenos resultados en el seguimiento de tramos curvos. En lugar de diseñar una ruta compleja y precisa que el robot tiene que seguir, en otros estudios se plantea el uso de algoritmos basados en lógica borrosa para conducir al robot desde una posición

inicial a un objetivo orientado [Gasós et al., 1991, Fraichard y Garnier, 2001], sin necesidad de planificar la trayectoria global.

Los vehículos agrícolas no-holonómicos tienen una cinemática y una dinámica compleja y no lineal, y es prácticamente imposible modelar la interacción entre las ruedas y el terreno [Fraichard y Garnier, 2001], [Lindgren et al., 2002]. Los controladores borrosos funcionan bien en sistemas no lineales, con alto grado de incertidumbre y conocimiento incompleto. Además son especialmente adecuados encapsular en un conjunto de heurísticos las estrategias de un conductor experto sin necesidad de un modelo analítico completo.

En la figura 5.25 se muestra la necesidad de considerar objetivos orientados. El ejemplo presenta la navegación en un circuito cerrado, donde el vehículo tiene que alcanzar el objetivo, etiquetado en la figura como G. Para ello, incluso en el caso de que la orientación del punto G fuese indiferente, es mucho más eficaz alcanzar el punto intermedio marcado con la orientación etiquetada como 2, que la etiquetada como 1.

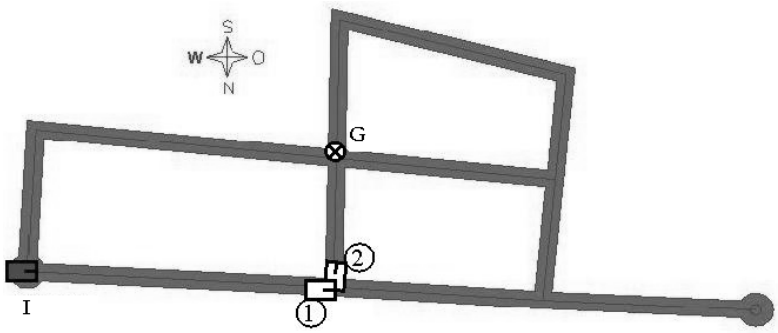


Figura 5.25 : Circuito cerrado en el IAI-CSIC donde se muestra la posición inicial del vehículo, la final y una posición intermedia con dos orientaciones distintas.

■ **Entradas, salidas y representación en el agente Avanzar.** El objetivo del agente **Avanzar** es conducir al robot desde una posición inicial (x_R, y_R, θ_R) hasta un objetivo orientado (subobjetivo dentro de la ruta global) (x_G, y_G, θ_G). Para ello se ha desarrollado un controlador de reglas borrosas basado en el conocimiento de un conductor experto, observado en un conjunto amplio de experimentos.

Las entradas del agente **Avanzar** son la señal de activación y el objetivo al que tiene que

dirigir el robot. Las salidas del agente **Avanzar** son: 1) el ángulo de giro y 2) la posición del freno, como se muestra en la figura 5.26. Para ello debe disponer de las percepciones *Posición* y *Obstáculos* actualizadas, y por ello debe activar a los agentes perceptivos **Actualizar Posición** y **Actualizar Obstáculos**.

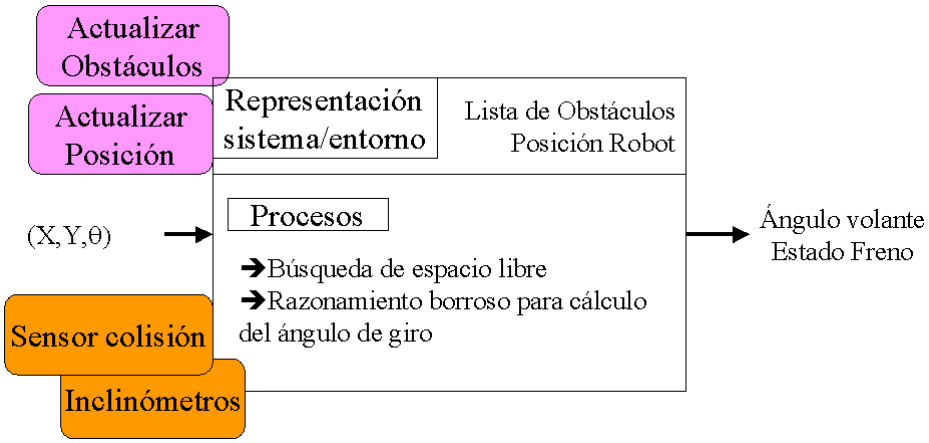


Figura 5.26 : Esquema de entradas, salidas, representación y procesos del agente **Avanzar**

- **Contexto de activación.** Una vez que el agente **Avanzar** ha recibido la señal de activación, pasa del estado **inactivo** al estado **alerta**. Una vez en estado de **alerta** comprueba si se verifican su contexto de ejecución, figura 5.33. El contexto de ejecución de **Avanzar** es el complementario de **Parar** y **Evitar Obstáculos**, siguiendo la idea de las clases perceptivas que dividen el espacio de percepción de estos agentes en clases disjuntas para evitar la activación y ejecución simultánea de dos o más agentes. Las condiciones de ejecución de **Avanzar** son : 1) inexistencia de colisión, 2) $p_f, p_l > 15$ y 3) inexistencia de obstáculos de tamaño mayor de 3 celdillas en un círculo de radio 4 (m) centrado en el robot. Si estas tres condiciones se cumplen, el agente **Avanzar** pasa al estado **activo**, procediendo a calcular el ángulo de giro, como se muestra en el diagrama de flujo de la información de la figura 5.33.

- **Procesos en el agente: controlador borroso del agente Avanzar.** El modelo de razonamiento borroso para obtener las consignas de giro puede formularse como un conjunto de reglas borrosas obtenidas de: (a) un experto humano, (b) las acciones de control de un

operador, (c) un modelo borroso del proceso y (d) un proceso de aprendizaje [Sugeno, 1987]. El controlador borroso del agente **Avanzar** se obtuvo observando y mimetizando (apartado (b) anterior) las acciones de un conductor sobre el volante del vehículo en un conjunto de reglas lingüísticas.

El primer paso consiste en el análisis de las variables geométricas implicadas en el problema y de las percepciones que el conductor define como relevantes. En la figura 5.27 se muestran las variables implicadas en el problema.

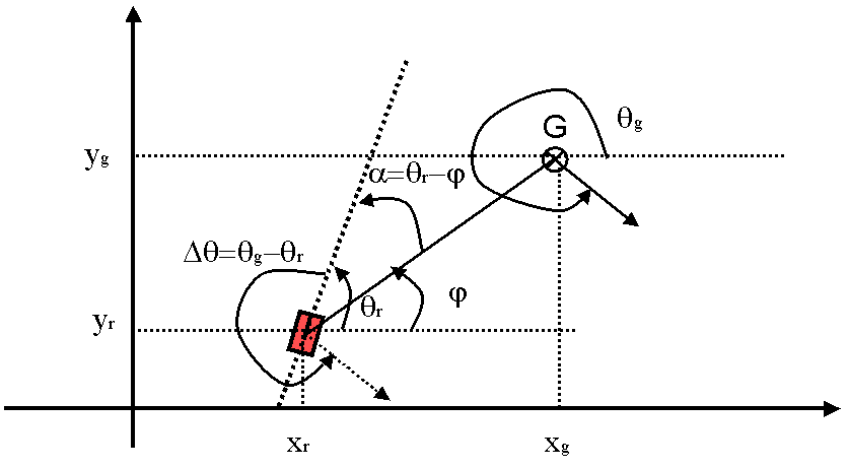


Figura 5.27 : Esquema de las variables geométricas relevantes para la maniobra de avance hacia un objetivo orientado de un vehículo no holonómico

EL segundo paso es capturar la habilidad humana para realizar la maniobra con autonomía en forma de reglas. Para todo ello, se ha utilizado un entorno de simulación donde el robot conducido mediante comandos enviados por el experto humano tiene que alcanzar un objetivo orientado. Durante el movimiento del vehículo se almacenan las variables geométricas y los comandos de giro recibidos. En la figura 5.28 se muestran los valores de $\Delta\theta = \theta_G - \theta_R$ (diferencia entre la orientación objetivo y la actual) como triángulos, los de $\alpha = \theta_R - \phi$ (diferencia entre la orientación del robot y la alineación con el objetivo) como cuadrados y los comandos de giro como círculos, todos ellos en función de la distancia robot-objetivo.

En la representación de la figura 5.28 pueden distinguirse claramente tres zonas, que se han denominado: “Aproximación” , “Preparación” y “Orientación”. La primera de las zonas se corresponde con comandos que dan lugar a una disminución del ángulo α, lo que significa que

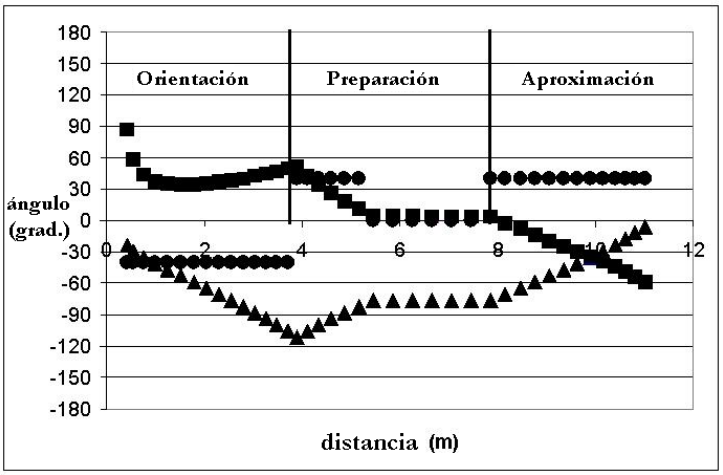


Figura 5.28 : Intervalo de valores de las variables relevantes durante la conducción manual del robot en el simulador

cuando el robot está alejado del objetivo su estrategia sólo pretende aproximarse a él, sin tener en cuenta la orientación. En la figura 5.28 esta zona corresponde a valores positivos del ángulo de giro ordenado, que hace que α, que es negativo, aumente hasta alcanzar un valor próximo a 0.

En la segunda zona las órdenes de giro provocan una desviación del robot en sentido contrario al de la orientación del objetivo, esto es aumenta el valor absoluto de $\Delta\theta$, de modo que se prepara la aproximación al objetivo con la orientación adecuada. En el último paso las órdenes conducen al robot al objetivo reduciendo simultáneamente la distancia y la diferencia angular.

De los experimentos también se deduce que las distancias que fijan las fronteras entre las tres zonas dependen del valor absoluto de $\Delta\theta$, resultado evidente puesto que cuanto mayor es el error de orientación, antes debe iniciarse la maniobra.

Una vez analizadas las variables relevantes se genera el sistema de reglas borrosas, figura 5.29, donde las variables de entrada del sistema son:

1. **diferencia_orientación**: describe la diferencia entre la orientación del robot y la del objetivo. Su universo de discurso se sintetiza en tres etiquetas lingüísticas: *negativa*, *cero*

y *positiva*, figura 5.30.

2. **diferencia_alineación**: representa la diferencia entre la orientación actual del robot y la pendiente de la recta que une el robot con el objetivo. Sus etiquetas lingüísticas son las mismas que las de la variable **diferencia_orientación**, figura 5.30.

3. **distancia**: es una variable escalar que representa la separación entre el robot y el objetivo. Se describe mediante tres términos lingüísticos: *grande*, *media* y *pequeña*.

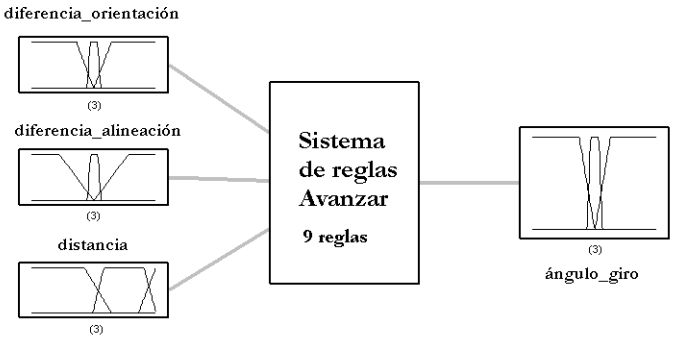


Figura 5.29 : Controlador borroso del agente **Avanzar**

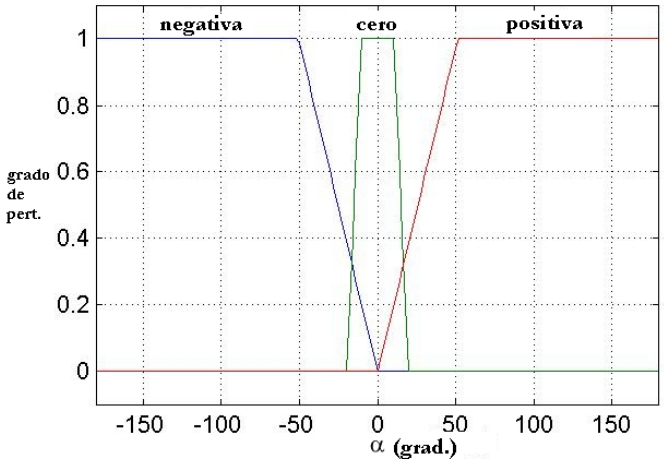


Figura 5.30 : Conjuntos borrosos de las variables de entrada **diferencia_alineación** y **diferencia_orientación**

La **distancia**, se utiliza para determinar la frontera entre las zonas de “Aproximación”,

"Preparación" y "Orientación"; y la frontera depende del valor absoluto de $\Delta\theta$. Por este motivo se ha dividido $\Delta\theta$ en tres intervalos:

1. $|\Delta\theta| < 70$
2. $70 < |\Delta\theta| < 100$
3. $|\Delta\theta| > 100$

En cada uno de estos tres intervalos se definen diferentes anclajes para las funciones de pertenencia de la variable **distancia**. Esto quiere decir que el significado de los términos lingüísticos de *grande*, *media* y *pequeña* para esta variable depende de la diferencia entre la orientación del robot y la del objetivo. Las funciones de pertenencia de la variable **distancia** en cada intervalo se muestran en la figura 5.31.

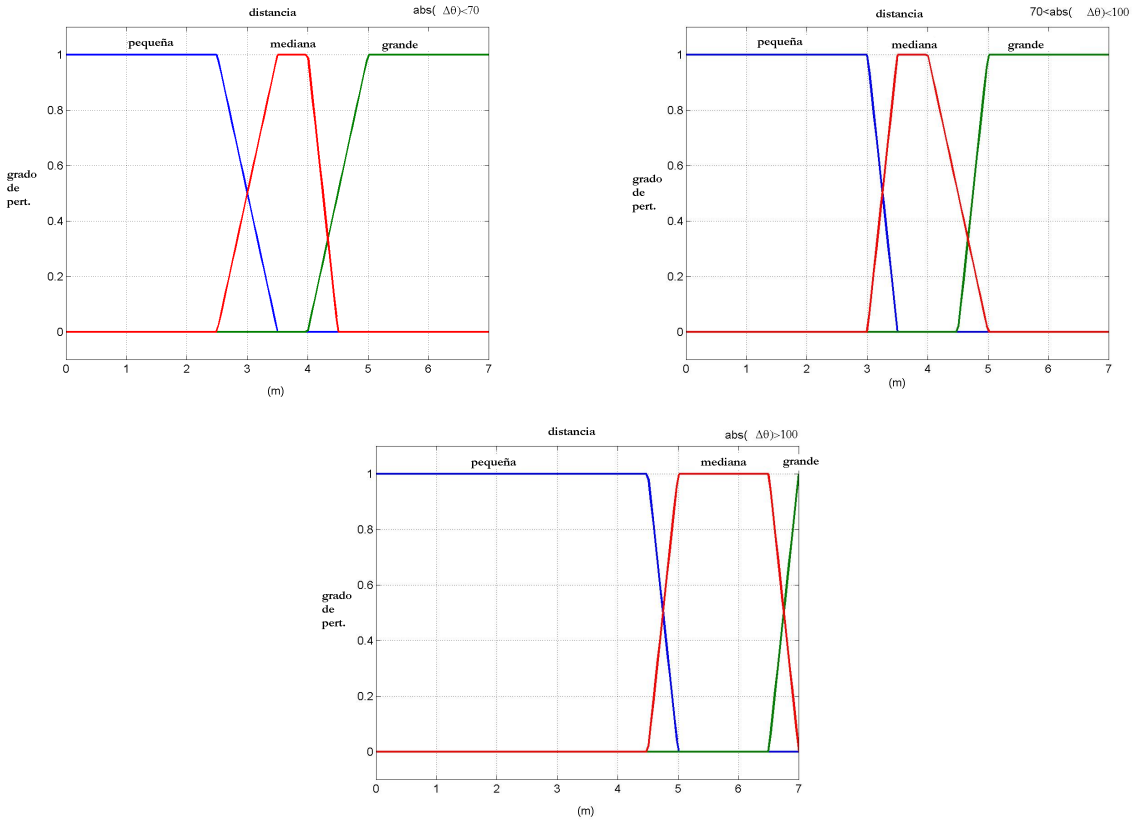


Figura 5.31 : Conjuntos borrosos de la variable **distancia** en los tres intervalos.

La única variable de salida es el **ángulo_giro** que se envía al servidor DÉDALO en cada ciclo

de control. Su universo de discurso se representa mediante tres etiquetas lingüísticas: *derecha*, *centro* e *izquierda*, figura 5.32.

Resumiendo, el controlador borroso del agente **Avanzar** se describe mediante tres variables de entrada y una de salida. Las nueve reglas de la base de conocimiento combinan los valores de las tres variables de entrada, como indica la figura 5.29. Estas reglas pueden agruparse, a su vez, en tres grupos:

1. En el primer grupo se incluyen las reglas en las cuales la variable **distancia** toma el valor lingüístico *larga*, que corresponde a la zona "Aproximación" (reglas 1, 2 y 3 en la tabla 5.4).

2. En la segunda fase el valor lingüístico de la variable **distancia** en el antecedente es *mediana* (reglas 4, 5 y 6) correspondiendo con la zona de "Preparación".

3. La fase que corresponde a la zona de "Orientación" (reglas 7 a 9) es aquella en la cual la variable **distancia** toma el valor *pequeña*.

El sistema de control difuso se ha desarrollado con la ayuda del entorno de programación FuzzyShell [Gasós et al., 1990].

	distancia	**diferencia_alineación**	**diferencia_orientación**	**ángulo_giro**
1	*grande*	*negativa*		*izquierda*
2	*grande*	*cero*		*centro*
3	*grande*	*positiva*		*derecha*
4	*mediana*		*negativa*	*izquierda*
5	*mediana*		*cero*	*centro*
6	*mediana*		*positiva*	*derecha*
7	*pequeña*		*negativa*	*derecha*
8	*pequeña*		*cero*	*centro*
9	*pequeña*		*positiva*	*izquierda*

Tabla 5.4 : Base de conocimiento del controlador borroso del agente **Avanzar**

El agente **Avanzar** funciona en un ciclo de control que se repite cada 500 (ms.). Se inicia cuando recibe de cualquier otro agente la señal de activación. En ese momento pasa del estado **inactivo** al estado **alerta** y envía las señales de activación a los agentes **Actualizar Posición**

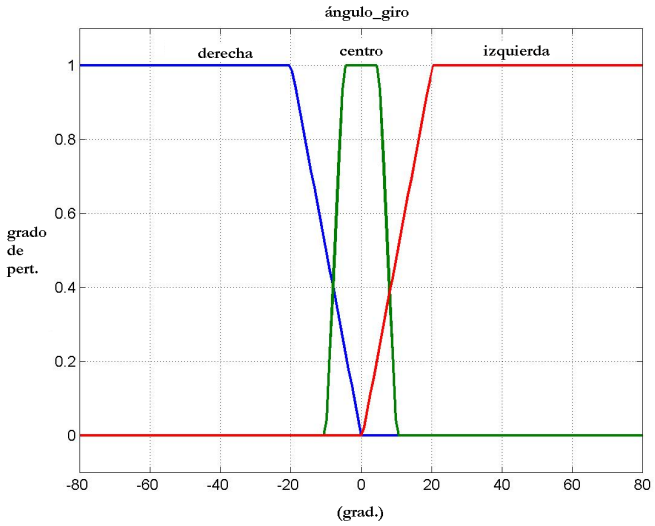


Figura 5.32 : Funciones de pertenencia de la variable de salida **ángulo_giro**

y **Actualizar Obstáculos**, figura 5.33. A continuación ejecuta las rutinas de inicialización propias, para abrir el fichero de texto del controlador y asociar las zonas de memoria correspondientes a las variables y comienza su ejecución iterativa, que terminará cuando reciba la señal de desactivación procedente de cualquier otro agente.

Agente Planificar Caminos

Como se discutió en el capítulo 3, las capacidades de deliberación de los agentes permiten optimizar la toma de decisiones [García-Alegre et al., 1993]. Hasta hace pocos años su utilización se veía restringida por el coste computacional asociado a los procesos deliberativos, hoy en día el coste de un sistema de proceso se ha reducido hasta extremos insospechados y su capacidad de proceso ha aumentado exponencialmente. De ahí que cada vez más se integren procesos deliberativos en las arquitecturas de control; que fueron diseñadas inicialmente con componentes altamente reactivos, para dar respuestas rápidas independientemente de su grado de precisión.

El objetivo de la navegación global de propósito general implica la consecución de un desplazamiento seguro del robot de un punto inicial a un punto final siguiendo una trayectoria.

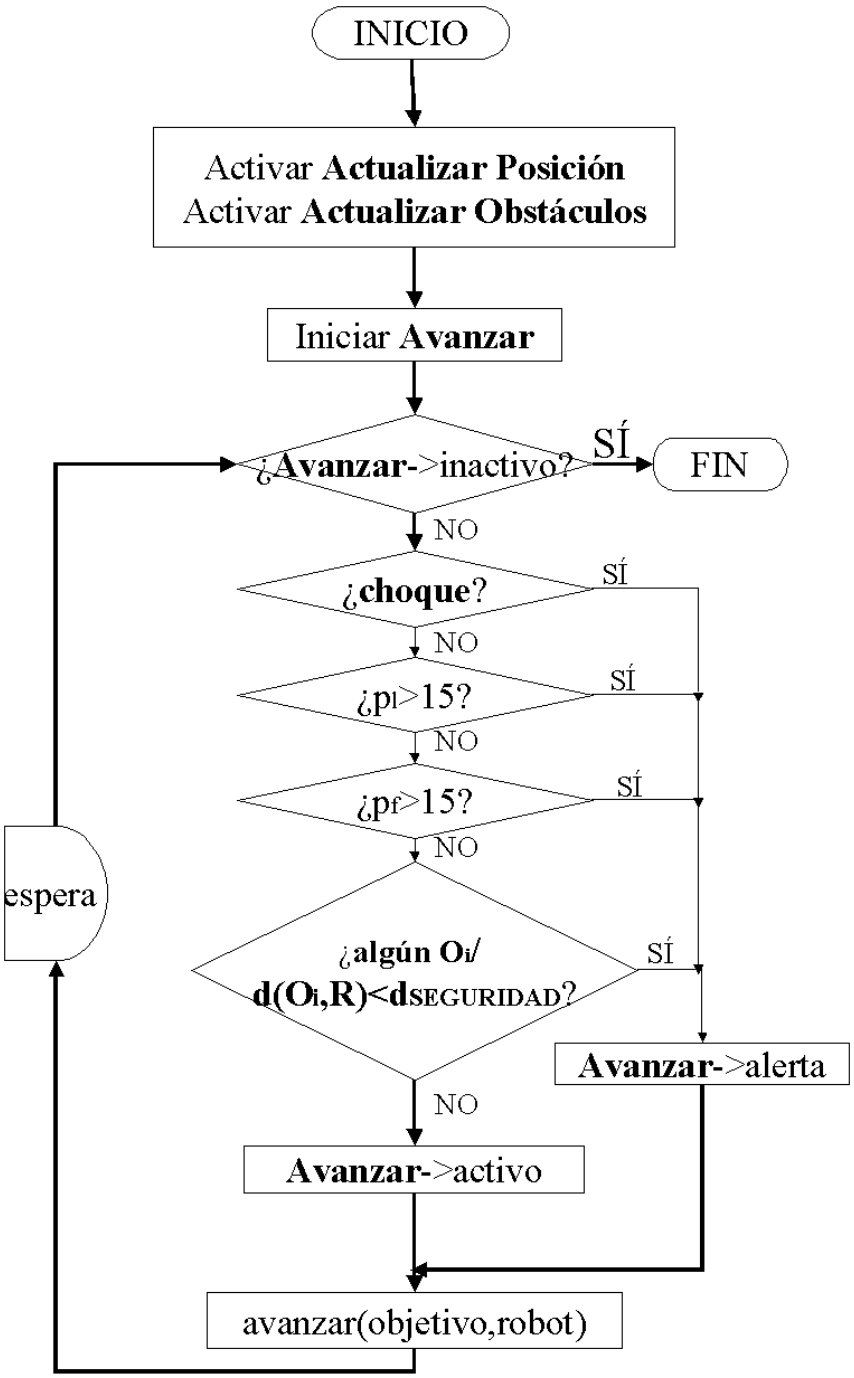


Figura 5.33 : Diagrama de flujo de información del agente **Avanzar**

El agente **Planificar Caminos** está diseñado para buscar la trayectoria óptima de acuerdo con unos criterios o heurísticos previamente establecidos, que pueden variar de una aplicación a otra. El conocimiento implícito en los heurísticos se traduce posteriormente en una función analítica de coste, que determina cual es el camino óptimo entre todas las alternativas posibles

de caminos seguros.

- **Entradas, salidas y representación del agente Planificar Caminos ROSEN-DOS.** El objetivo del agente **Planificar Caminos** es elaborar un *Plan de Navegación* que permita llegar con eficiencia al destino final. Para ello necesita en primer lugar conocer el destino final, el mapa del entorno donde estén representadas las zonas prohibidas (Mundo de obstáculos estáticos) y la posición inicial del robot.

El destino final, el mapa y la señal de replanificación constituyen las entradas del agente, como se muestra en la figura 5.34 y en este caso se trata de un conocimiento que es inyectado al sistema por el operario, marcando sobre la imagen digital georreferenciada del entorno. Para poder disponer de la *Posición* del robot, el agente **Planificar Caminos** tiene que enviar la señal de activación al agente **Actualizar Posición**, encargado de elaborar y mantener esta percepción. La única salida del agente **Planificar Caminos** es un *Plan de Navegación* que consiste en un conjunto de posiciones 2D intermedias que conectan el punto inicial con el destino final o trayectoria tentativa. Esta trayectoria se almacena en la pizarra como una representación a utilizar por otros agentes de actuación. El agente **Planificar Caminos** es un agente totalmente deliberativo, que procesa la información de las entradas transformándola en otra con una estructura distinta, que constituye su única salida. En este sentido, realiza un procesamiento similar al de los agentes perceptivos: **Actualizar Posición**, **Actualizar Mapa Local** y **Actualizar Obstáculos**, pues no produce modificación alguna en el entorno pero sí en la memoria del sistema.

- **Procesos en el agente Planificar Caminos** El planificador integrado en este agente, denominado ROSEN-DOS, constituye una versión más completa del planificador ROSEN diseñado y desarrollado en el IAI [García-Alegre et al., 1993]. ROSEN es un planificador de propósito general, basado en un algoritmo de búsqueda A*, que utiliza un mapa de zonas prohibidas definidas como polígonos de n lados. Devuelve la ruta más corta entre el punto inicial y el objetivo mediante una lista de posiciones intermedias, que se conectan mediante

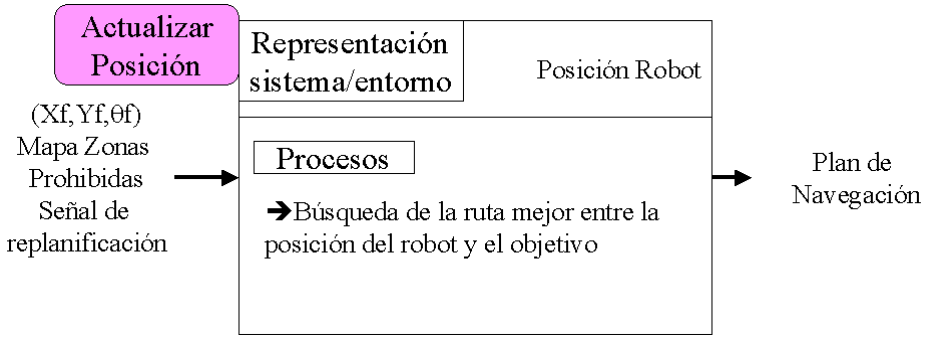


Figura 5.34 : Esquema de entradas, salidas, representación y procesos del agente **Planificar Caminos**

segmentos rectilíneos dando lugar a la trayectoria tentativa óptima. En la etapa actual, el operario selecciona y marca de forma interactiva mediante el ratón las zonas de navegación prohibidas sobre una imagen aérea georreferenciada.

El mapa global de obstáculos, el destino final y la posición actual del robot son las entradas del agente **Planificar Caminos**, y su salida es la trayectoria óptima.

Al aplicar **Planificar Caminos**, diseñado para interiores, en la navegación en exteriores, se comprobó que el heurístico de recorrido con mínima distancia no era suficiente. Al tratarse de un robot no-holonómico, el camino más corto no siempre resulta ser el más adecuado si el vehículo tiene que realizar muchos giros. Además, el planificador ROSEN proporciona las posiciones intermedias mediante coordenadas 2D sin orientación y sin embargo el destino final es una posición orientada. Estas consideraciones, llevaron a plantear un conjunto de modificaciones en la función de coste inicial del planificador de forma que pudiese:

1. penalizar los giros
2. tener en cuenta la orientación del objetivo

El planificador resultante se denomina ROSEN-DOS y es un aporte de esta tesis.

- **Funciones de coste.** El planificador ROSEN calcula a partir del mapa de polígonos, el grafo de visibilidad y busca en esta nueva representación geométrica del mundo, el camino más corto entre vértices visibles utilizando una función de coste del tipo (5.6) donde se pondera

tanto la distancia del camino recorrido desde el origen al vértice actual, d_{ini}, como la distancia en línea recta desde dicho vértice al objetivo d_{fin}.

$$F_{coste} = d_{ini} + d_{fin} \tag{5.6}$$

En el planificador ROSEN-DOS esta función de coste se ha modificado para penalizar los giros en la búsqueda del camino óptimo. Tratando de que la modificación sea mínima se ha optado por modular los dos sumandos de la función de coste de ROSEN, mediante dos funciones que dependen de los ángulos α_{ini}, y β_{fin}. La función de coste resultante es la siguiente (5.7):

$$\tilde{F}_{coste} = d_{ini} \cdot F(\alpha)_{ini} + d_{fin} \cdot F(\beta)_{fin} \tag{5.7}$$

La función $F(\alpha)_{ini}$ pondera el ángulo que forman los puntos: actual, A_0, anterior A_{-1} y candidato A_1, como muestra la figura 5.35, favoreciendo la máxima alineación y en consecuencia penalizando los giros. Al tener en cuenta el ángulo de la recta delimitada por A_0 y A_{-1} se está considerando implícitamente que el robot llega a A_0 con una orientación próxima a la pendiente de la recta que une ambos puntos, α_1 en la figura 5.35. De este modo si el robot parte del punto A_0 con una orientación inicial α_1, para llegar a A_1 tiene que girar un ángulo correspondiente al de la recta que une A_0 y A_1, α_2, menos su ángulo inicial, es decir un giro de $\alpha_2 - \alpha_1$. La función $F(\alpha)_{ini}$ se define mediante la expresión (5.8):

$$F(\alpha)_{ini} = 1 - \frac{1}{2} \cdot \cos(|\alpha_2 - \alpha_1|) \tag{5.8}$$

De la misma manera $F(\beta)_{fin}$ pondera el ángulo con el que llegaría el robot al destino final, navegando en línea recta desde A_0 y la orientación deseada en el objetivo. Se pretende favorecer los caminos en los que el robot llega con una orientación próxima a la orientación objetivo del destino. En este caso entran en juego tres ángulos:

1. el ángulo estimado de llegada del robot desde el punto actual A_0 al punto candidato A_1, denominado β_1 en la figura 5.36,

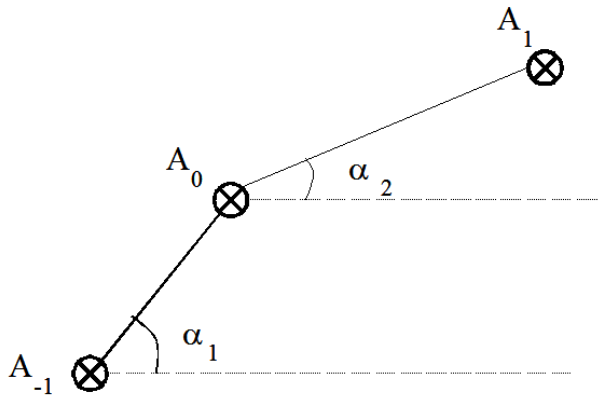


Figura 5.35 : Descripción gráfica de los ángulos de la función $F(\alpha)_{ini}$

2. el ángulo con el que llegaría el robot desde A_1 al objetivo si fuera en línea recta, denominado β_2,

3. el ángulo objetivo de llegada al punto destino, establecido por el operario.

La función $F(\beta)_{fin}$ se define mediante la expresión (5.9):

$$F(\beta)_{fin} = 1 - \frac{1}{2} \cdot \cos(|\beta_2 - \beta_1 - \theta_{goal}|) \tag{5.9}$$

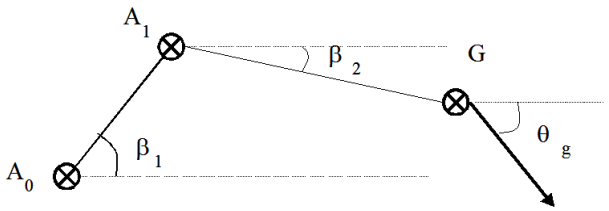


Figura 5.36 : Descripción gráfica de los ángulos en la función $F(\beta)_{fin}$

En la figura 5.37 se presentan cuatro ejemplos de *Plan de Navegación* elaborados por el planificador ROSEN-DOS. Se ha representado en color negro la ruta propuesta, en malla de color azul las zonas prohibidas y con un círculo de color rosa y una flecha el destino final y su orientación respectivamente. Los ejemplos (a) y (b) muestran puntos objetivo semejantes pero con orientaciones opuestas. El **Planificador de Caminos** ROSEN-DOS, al ser sensible

a la orientación del destino final, tiene en cuenta estas diferencias y propone rutas diferentes para cada caso. En el caso (a) la ruta coincide con el camino de mínima distancia, sin embargo en el caso (b) la penalización sobre los giros da lugar a una ruta más larga pero con menos giros. En definitiva, la función de coste establece un compromiso entre distancias y giros, y este compromiso se explicita en las funciones, $F(\alpha)_{ini}$ y $F(\beta)_{fin}$.

El coseno del ángulo penaliza los giros, sin embargo tomar sólo $1 - \cos$ implicaría que para ángulos muy favorables (puntos alineados) el factor de peso sería 0 con independencia del valor de la distancia. En estos casos podrían generarse trayectorias con un recorrido muy grande, evidentemente no deseables. Para evitar estos resultados absurdos, se introduce el factor de peso $1/2$ multiplicando al coseno, para forzar un compromiso más razonable entre ángulo y distancia. Las gráficas (c) y (d) de la figura 5.37 muestran cómo en el caso (c) el compromiso se resuelve en favor del ángulo y en el caso (d) la distancia pesa más en la decisión sobre la trayectoria propuesta al ser la diferencia de distancias entre los dos caminos posibles superior que en el caso (c).

Dependiendo de las dimensiones del vehículo y del espacio disponible se puede dar más peso a la distancia o al ángulo modificando únicamente el factor que multiplica los cosenos en las ecuaciones (5.8) y (5.9). En el caso del robot DÉDALO en el entorno del IAI-CSIC este factor se ha establecido en $1/2$ de modo experimental. Valores más próximos a la unidad penalizan más los giros y menores penalizan más la distancia recorrida en cada trayectoria.

El agente **Planificar Caminos** elabora un plan de actuación que coloca en la pizarra, de la misma manera que los agentes perceptivos colocan las percepciones que elaboran. En este sentido **Planificar Caminos** no controla directamente la actuación del robot.

Para mantener el funcionamiento iterativo característico de los agentes sin tener que replanificar constantemente, muestrea cada 5 (s.) una variable denominada *nuevo_plan*, elaborada por el agente **Ir a Punto**. Si esta variable es CIERTA entonces replanifica, solicitando como entrada el valor de la *Posición* actual. Esta posición actual será la posición inicial de entrada al planificador, manteniéndose el objetivo anterior. Si la variable toma el valor FALSO, espera el tiempo de ciclo de 5 (s.) sin realizar proceso alguno, tal y como muestra

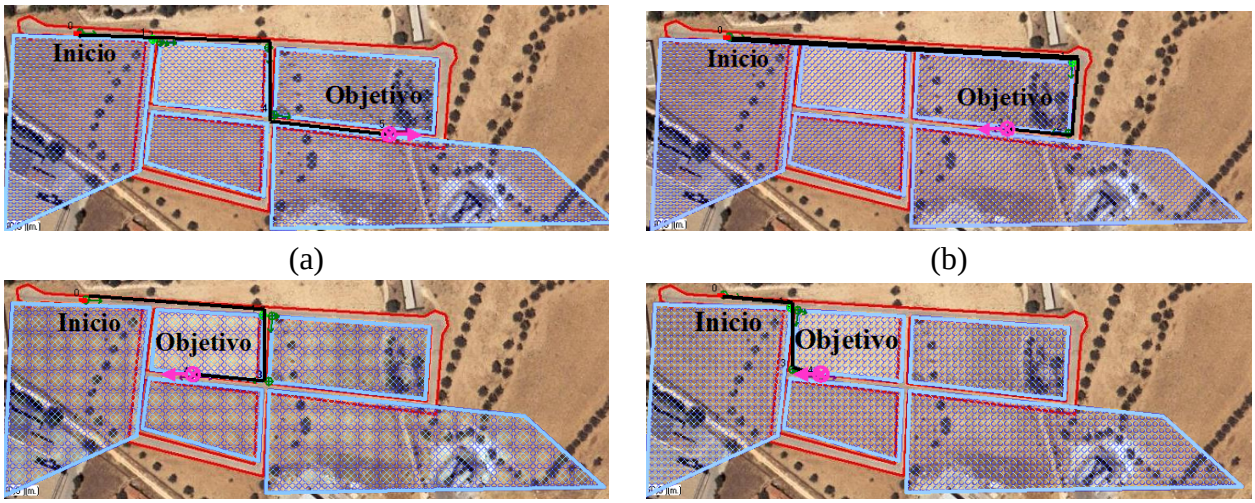


Figura 5.37 : Ejemplos de trayectorias obtenidas con el agente **Planificar Caminos** ROSEN-DOS

el diagrama de flujo de información de la figura 5.38. La trayectoria óptima resultado de la planificación, constituye la salida del agente **Planificar Caminos** y se almacena en la memoria global compartida o pizarra.

Agente Ir a Punto

El agente **Ir a Punto** tiene por objetivo gestionar la navegación local, lo que implica el control del desplazamiento del robot desde su posición actual a una posición destino definida por el triplete (x_g, y_g, θ_g) optimizando localmente la trayectoria con un mapa global y evitando colisionar con obstáculos imprevistos. Para ello **Ir a Punto** utiliza la habilidad de los agentes **Parar, Evitar Obstáculos, Avanzar** y **Planificar Caminos**, activándolos y modulándolos adecuadamente de acuerdo con el entorno. Esta reutilización, tanto de agentes perceptivos como de actuación, es una de las capacidades que contempla la arquitectura AGRO-AMARA.

Para que el robot navegue satisfactoriamente, el agente **Ir a Punto** tiene que realizar un conjunto de de tareas que se desglosan en:

- Seleccionar en cada instante un subobjetivo o posición, para el agente **Avanzar**.
- Analizar si es necesario o no elaborar un plan nuevo.

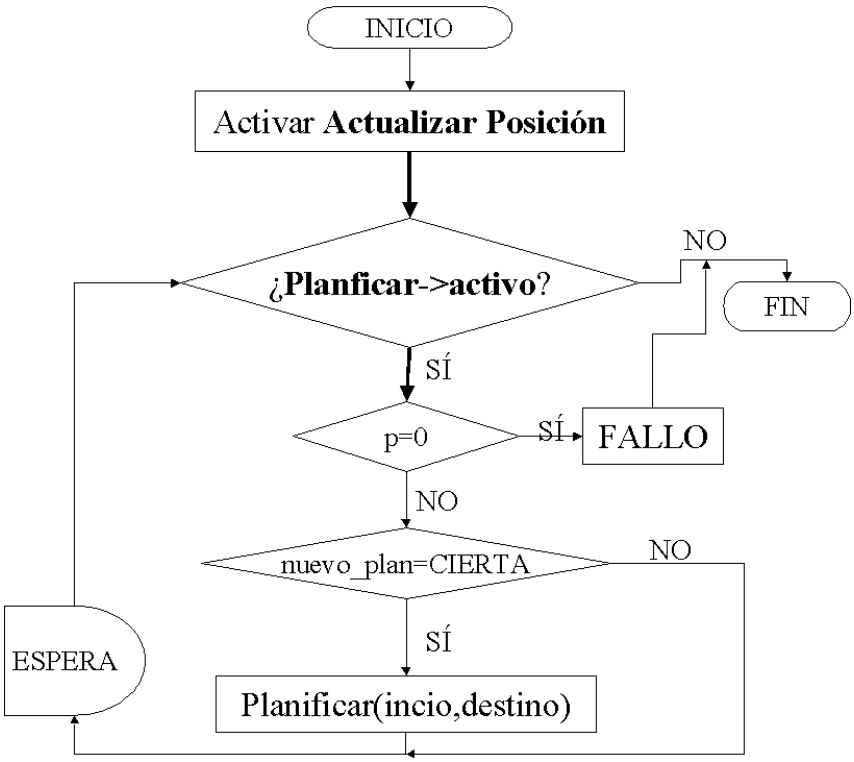


Figura 5.38 : Diagrama de flujo de información del agente **Planificar Caminos**

- Evitar situaciones sin salida. Si se encuentra sin salida, enviar mensaje de alerta al operario, parar todos los procesos y detener el robot.
- Gestionar los conflictos en la activación de los agentes; tanto en ausencia de señal de control, como en la activación simultánea de más de un agente de actuación.
- Decidir si se ha conseguido el destino con la precisión requerida o no y finalizar el proceso de navegación.

La figura 5.39 ilustra el esquema de entradas, salidas, representación y procesos del agente **Ir a Punto**. Las entradas del agente son: 1) un mapa con las zonas de paso prohibidas, descritas como polígonos con un número arbitrario de lados y 2) la posición objetivo en coordenadas absolutas con la orientación deseada. En la actualidad esta información la proporciona el usuario considerado como un agente más de la arquitectura.

Las salidas del agente **Ir a Punto** son las siguientes: 1) la posición instantánea de destino

que constituye la entrada del agente **Avanzar**, 2) la señal de replanificación que indica si es o no necesario elaborar un plan nuevo y constituye a su vez una entrada del agente **Planificar Caminos**, 3) la señal de objetivo conseguido, 4) la señal de fallo en la ejecución, y 5) las señales de activación de los agentes implicados: **Avanzar**, **Evitar Obstáculos**, **Parar**, **Planificar Caminos** y **Actualizar Posición**.

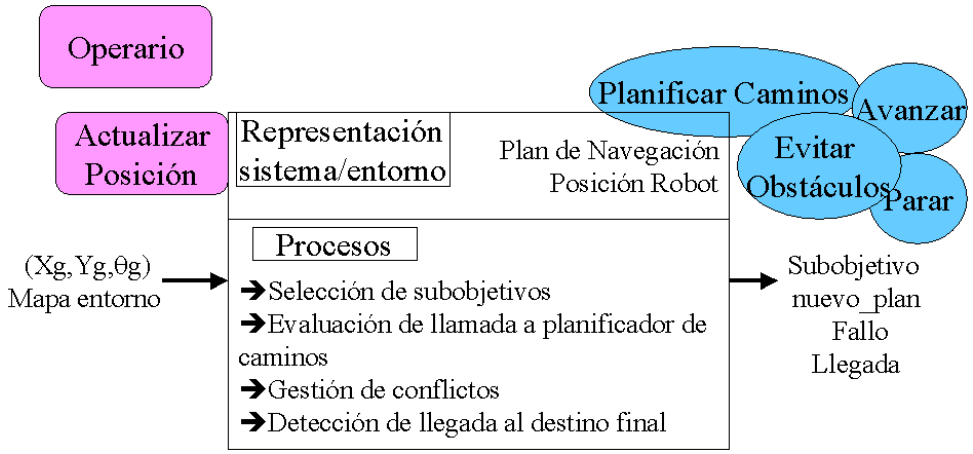


Figura 5.39 : Esquema de entradas, salidas, representación y procesos del agente **Ir a Punto**

- **Selección de un subobjetivo.** Para seleccionar cada subobjetivo, el agente **Ir a Punto** parte del plan de navegación elaborado por el agente **Planificar Caminos**, explicitado mediante un conjunto de posiciones intermedias que conectan la posición inicial con el destino final. En cada ciclo de control, **Ir a Punto** compara la *Posición* actual del robot con la posición final y decide si la distancia de separación es “grande”. Si se considera alcanzada la posición, por ser esta distancia “muy pequeña”, el agente **Ir a Punto** toma como nuevo objetivo final la siguiente posición del *Plan de Navegación*.

El agente **Ir a Punto** diferencia entre el destino final y los subobjetivos o posiciones intermedias, considerando que no es necesario alcanzar estos puntos intermedios con tanta precisión como el destino final. Para decidir si se ha alcanzado o no el objetivo, se calcula la distancia entre la posición actual del robot y la del objetivo, (ya sea subobjetivo o destino final), si esta distancia es menor que un determinado umbral se considera que el robot ha alcanzado el

objetivo. Este umbral es distinto si se trata de alcanzar un subobjetivo que si se trata del destino final, requiriendo una precisión mayor en este último caso.

- **Replanificación y salida en situaciones conflictivas.** Una cuestión que siempre se plantea en torno a los planificadores es cuándo es necesario elaborar un plan nuevo [Murphy, 2000]. Habitualmente se manejan dos criterios para decidir si es necesario o no elaborar un plan: 1) si se detecta que el mundo sobre el que se razonó para elaborar el último plan y el mundo que actualmente perciben los sensores no coinciden y 2) cuando se detecta que el robot se ha desviado mucho de la trayectoria tentativa. En el caso de la arquitectura AGRO-AMARA para robots agrícolas, el mapa utilizado para elaborar el plan está basado en entidades abstractas definidas por el operario como zonas "prohibidas", en cierto sentido similares a las zonas de sombra de radiofrecuencia que contempla [Rosenblatt y Payton, 1989]. Al tratarse de entidades abstractas, no siempre pueden detectarse con los sensores del robot. Por esta razón, el *Plan de Navegación* es una propuesta de camino tentativo optimizado de acuerdo a unos criterios, que varían con la aplicación, y tiene que completarse con estrategias de navegación reactiva, que en el caso de AGRO-AMARA son realizadas por los agentes **Avanzar**, **Parar** y **Evitar Obstáculos** los cuales disponen de un mapa local del entorno actualizado en cada ciclo de control.

Para detectar que el robot se encuentra "perdido" se compara el tiempo transcurrido desde que alcanzó el subobjetivo anterior con el tiempo estimado para la realización de ese recorrido, pues se conoce la distancia entre las dos posiciones y la velocidad constante del robot (5.10). Si el tiempo transcurrido supera al tiempo estimado, por encima de un determinado umbral (5.11) se considera que existe un problema.

$$\Delta t_{esperado} = \frac{d(G_{i-1}, G_i)}{v_R} \tag{5.10}$$

$$\Delta t_{transcurrido} > \Delta t_{esperado} + \Delta t_{umbral} \tag{5.11}$$

Ante esta situación se puede inferir que el robot se encuentra en problemas y para ello se

comprueba si, desde la última ejecución del agente **Ir a Punto**, la posición del robot ha variado o no. Si ha variado la posición, el robot se encuentra "perdido" y es necesario elaborar un nuevo plan a partir de la posición actual del robot. Una vez determinado el estado de movimiento del tractor, el agente **Ir a Punto** asigna a la variable *nuevo_plan*, el valor de CIERTO, a fin de que el **Planificador de Caminos** elabore un nuevo *Plan de Navegación*. Si el estado del robot fuese parado, se envía una señal de FALLO al operario y se detiene el proceso de navegación local. Esta situación se produce ante objetos "móviles" que bloquean constantemente el camino del robot, moviéndose con rapidez y de forma aleatoria en su campo visual, por ejemplo un niño.

- **Gestión de conflictos entre agentes.** Aunque los agentes no son extremadamente complejos y sus condiciones de activación se han diseñado de forma que los contextos de activación sean mutuamente exclusivos, siempre pueden producirse conflictos de toma de control entre agentes. Así, si ninguno de los tres agentes de actuación, **Parar**, **Evitar Obstáculos** y **Avanzar**, está en estado **activo**, se llegaría a una situación de vacío de control. Pero puede también darse el caso opuesto: que más de un agente se encuentre en estado **activo**, generando un conflicto en el control. Por ello, el agente **Ir a Punto** debe disponer de las competencias necesarias para verificar que en cada instante, sólo un agente se encuentra en estado **activo** y actuar adecuadamente cuando esto no suceda.

Con este objetivo el agente **Ir a Punto** tiene implementadas unas estrategias de gestión de conflictos. Para ello analiza, en cada iteración, los valores de activación de los tres agentes de actuación básicos. Si la situación es de vacío de control, **Ir a Punto** opta por la solución más conservadora, forzar la activación del agente **Parar**, por el riesgo nulo que implica. Si se trata de un conflicto de toma de control entre agentes, se toma una decisión de asignación de control, de acuerdo a unas prioridades previamente establecidas. Si uno de los agentes activos es **Parar**, se obliga al resto de los agentes activos a permanecer en estado de **alerta**. Si el agente **Parar** no está **activo**, es el agente **Evitar Obstáculos** el que se mantiene en estado **activo** y el agente **Avanzar** es forzado a pasar al estado de **alerta**. Igual que en el caso de vacío de control, las prioridades se han establecido siguiendo un criterio conservador, primando

ante todo la seguridad. Otra situación conflictiva se produce cuando se detecta algún fallo en el funcionamiento de cualquiera de los agentes. En este caso se envía una señal de FALLO al operario y se detiene la ejecución.

El diagrama de flujo de información de la figura 5.40 y el de pseudocódigo del agente **Ir a Punto**, descrito en el anexo A muestran las entradas, salidas, representación y procesos de este agente. El agente **Ir a Punto** al activarse envía, a su vez, señales de activación a los agentes de actuación **Parar**, **Evitar Obstáculos**, **Avanzar** y **Planificador de Caminos** y al agente perceptivo **Actualizar Posición.** Después inicia su ciclo de ejecución con un tiempo de iteración de 1 (s.). En cada iteración comprueba en primer lugar si se ha llegado al destino final, si es así finaliza el proceso satisfactoriamente, con una señal de OK. Si aún no ha llegado al destino final, comprueba si se ha elaborado un *Plan de Navegación* nuevo, y de ser afirmativo actualiza el subobjetivo. Si continúa con el *Plan de Navegación* anterior, comprueba si el robot ha llegado al subobjetivo en curso, y si es así, establece un nuevo subobjetivo. En caso contrario analiza si el robot se ha perdido, si hace falta replanificar o bien si existe algún fallo.

En los casos en que el tiempo transcurrido es inferior al esperado se pasa a un análisis más detallado de los posibles conflictos de control. Para ello se comprueban los valores de activación de los agentes, que se encuentran almacenados en una variable global de tipo entero, de manera que un valor 0 corresponde al estado **inactivo**, 1 al estado **alerta** y 2 a **activo**. De esta forma, a través de un conjunto de operaciones de multiplicación entre estos valores, se detectan las situaciones de vacío o de conflicto en el control. En caso de que alguno de los agentes haya tenido un fallo y esté **inactivo**, el producto de los valores de activación de todos los agentes será 0, y el agente **Ir a Punto** generará un mensaje de FALLO y suspenderá la ejecución. Si esto no ocurre, se analizan los estados de los agentes motores, y si existe un vacío de control (ningún agente en estado **activo**), el producto de los valores asignados a los tres estados será 1. En este caso se fuerza la activación del agente **Parar**. Si el producto fuese un 8, significaría que los tres están activos y se fuerza también la activación del agente **Parar**, obligando a realizar una transición al estado de **alerta** a los agentes **Evitar Obstáculos** y **Avanzar**. Si el resultado es 4 significa que hay 2 agentes activos. En este caso se analiza el valor de estado del agente

Parar. Si el resultado es un 2, el agente **Parar** toma el control y si no es 2, lo toma el agente **Evitar Obstáculos**. Si el producto es 2 (un único agente activo) se realiza una espera igual al tiempo de ciclo menos el tiempo transcurrido en la iteración y se vuelve a ejecutar el agente **Ir a Punto**.

5.3 Arquitectura para navegación de laboreo autónoma

La mayor parte de la navegación que realiza un robot agrícola va dirigida al seguimiento preciso de una trayectoria en concordancia con la misión de laboreo, por ejemplo la fumigación de un campo de olivos, recolección automática de hortalizas y frutas, cosechado de cereal, etc... Este tipo de navegación está sometida a unas restricciones que no existen en las estrategias de navegación más generales.

La Navegación de Laboreo tiene como finalidad elaborar las trayectorias a seguir por el robot en una labor agrícola, así como controlar la correcta ejecución de las mismas. Las trayectorias a seguir por el robot en la navegación de laboreo poseen unas características distintas a las de una navegación global, y están sujetas a las restricciones y requisitos impuestos inicialmente por el agricultor. En primer lugar, el objetivo básico es diferente, pues en la navegación global se trata de alcanzar un destino mientras que en la navegación de laboreo lo fundamental es recorrer los distintos tramos de la trayectoria planificada de acuerdo a las restricciones impuestas por la tarea, que serán diferentes si se trata de fumigar, recolectar, plantar o fertilizar. En la navegación de laboreo la planificación tiene que atender a criterios de optimización muy diferentes de los clásicos de la navegación global, que persiguen minimizar el coste energético y de tiempo en el recorrido. Así, en el caso de fumigación de un campo de frutales uno de los criterios a considerar por el planificador es que la trayectoria tentativa debe cubrir la totalidad de la superficie del campo.

En este sentido la navegación de laboreo es mucho más estricta que la navegación global, ya que al objetivo de llegar al destino final se añade el de recorrer los segmentos de cada trayectoria de acuerdo a unas especificaciones precisas. Sin embargo, el entorno en el que se desarrolla la

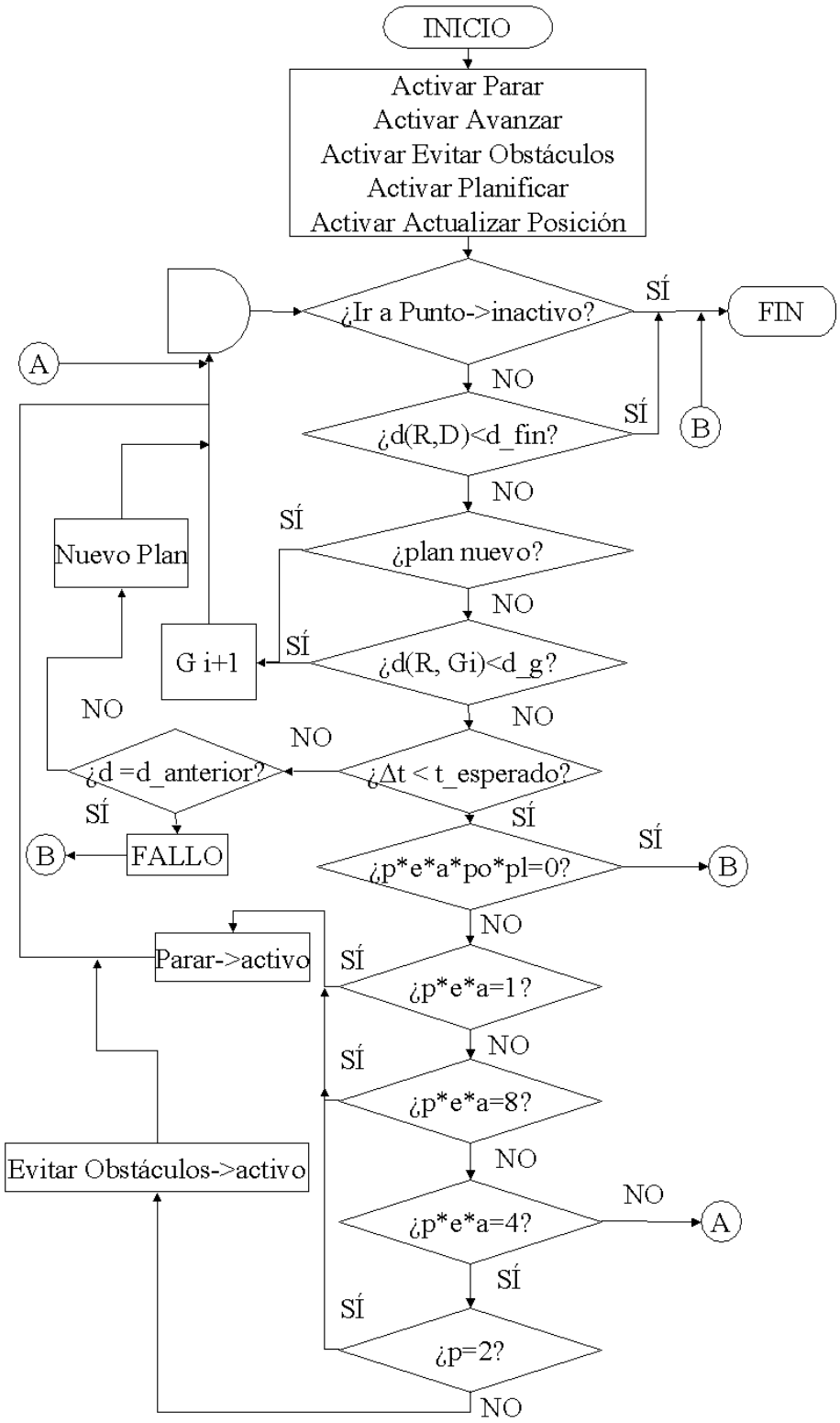


Figura 5.40 : Diagrama de flujo de información del agente **Ir a Punto**

navegación de laboreo está muy estructurado, conociéndose la topología de plantación de los distintos cultivos (frutales, hortalizas, cereales, olivos); y esto permite disponer desde el inicio de una información muy valiosa para el diseño del control de las trayectorias, dirigido a la navegación específica que se persigue. Este alto grado de estructuración, donde el cultivo forma hileras y no existe aleatoriedad en su distribución constituye una gran ayuda en las expectativas de aparición de los estímulos.

Respecto de la capacidad de reacción ante imprevistos, el robot va a encontrar un reducido repertorio de acciones a realizar ante un imprevisto: 1)parada o 2) desvío si se dispone de suficiente espacio libre y el objeto es estático. Siguiendo con el ejemplo anterior, la reducida distancia entre hileras en los campos de frutales conducen a que la única acción posible sea detener el vehículo.

A pesar de estas diferencias, la navegación global y la navegación de laboreo tienen muchos puntos en común. En primer lugar comparten la necesidad de seguridad ante imprevistos, pues el hecho de que el entorno sea más estructurado no exime de la posibilidad de que aparezcan obstáculos imprevistos o fallos en el sistema. En segundo lugar el objetivo de llegar a cierto destino se mantiene, aunque la flexibilidad en la ejecución de la ruta disminuya considerablemente. Debido a las diferencias entre ambos tipos de navegación, la navegación de laboreo hace uso, en los niveles superiores de abstracción, de estrategias de navegación que difieren de las de la navegación global. No obstante algunos agentes pueden reutilizarse, como los de bajo nivel en contacto con los dispositivos físicos, optimizando de este modo el tiempo de diseño y desarrollo de la arquitectura para la navegación de laboreo. La arquitectura híbrida basada en agentes propuesta y desarrollada en esta tesis, AGRO-AMARA, contempla la utilización de los mismos agentes modulados de manera distinta a fin de conseguir comportamientos diferentes.

El objetivo de la navegación autónoma de laboreo, consiste en recorrer un campo de cultivo en su totalidad, realizando una secuencia de desplazamientos siguiendo segmentos rectilíneos y curvos, como el que se muestra en la figura 5.41, evitando colisionar con obstáculos imprevistos y afrontando posibles fallos del sistema. Este objetivo general puede descomponerse en los

siguientes subobjetivos:

1. Recorrer con suficiente precisión una línea recta
2. Girar para cambiar de una línea recta a otra situada a una distancia conocida
3. Mantener la seguridad del sistema

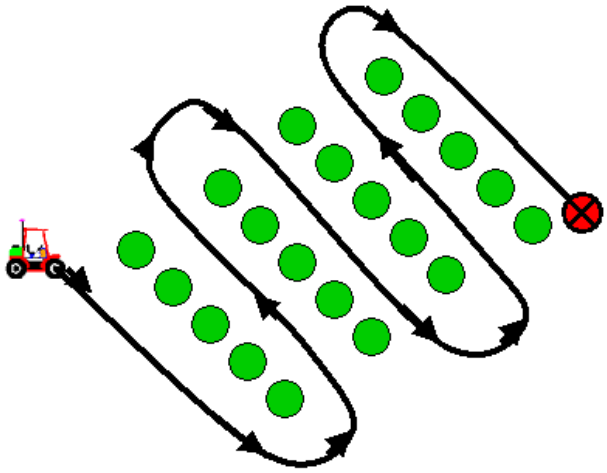


Figura 5.41 : Ruta típica de la navegación de laboreo

El agente **Recorrer Campo** es el encargado de coordinar la ejecución de una navegación de laboreo segura de acuerdo con las especificaciones formuladas por el agricultor, desempeñando el mismo papel que el agente **Ir a Punto** en la navegación global, figura 5.42.

La figura muestra la jerarquía propuesta en AGRO-AMARA para la navegación de laboreo. **Recorrer Campo** es el agente de más alto nivel, y puesto que la trayectoria a realizar por este agente se compone de líneas rectas y arcos de circunferencia, para pasar entre líneas de cultivo, el agente **Recorrer Campo** depende de los agentes de actuación **Seguir Recta** y **Seguir Arco de Circunferencia** para controlar ambos tramos de la navegación. Además necesita conocer en todo momento cuál es la *Posición* del robot, razón por la cual activa el agente perceptivo **Actualizar Posición**, utilizado también en la navegación global. Tanto el agente **Seguir Recta** como el agente **Seguir Arco de Circunferencia** han de garantizar una navegación sin colisiones. Los agentes **Parar** y **Evitar Obstáculos** se encargan de este

Figura 5.42 : Esquema completo de los agentes de la navegación específica de laboreo. Las elipses de color azul situadas a la derecha de la figura, corresponden a los agentes de actuación. Se han destacado en color amarillo los agentes de actuación nuevos que no existían en la navegación global. Los rectángulos con esquinas redondeadas en color rosa situados a la izquierda son los agentes perceptivos. En el centro de la imagen en color verde se muestra el contenido de la pizarra. La zona inferior de la figura muestra el servidor del robot, con los dos agentes básicos de locomoción **Girar Volante** y **Frenar**.

cometido y al igual que en la navegación global, requieren el conocimiento de los *Obstáculos* actualizados por el agente perceptivo **Actualizar Obstáculos**, que utiliza la representación *Mapa Local*, mantenida a su vez por el agente **Actualizar Mapa Local**. El agente **Seguir Recta** reutiliza la funcionalidad motora del agente **Avanzar**, modulándolo convenientemente para describir trayectorias rectilíneas, mediante un algoritmo similar al conocido en Robótica

como “persecución pura”. Por otro lado, el agente **Seguir Arco de Circunferencia** utiliza las capacidades del agente básico **Girar**, que fuerza giros de 180 grados, haciendo que el robot recorra trayectorias semicirculares. Un agente perceptivo nuevo completa la navegación de laboreo, se trata del agente **Actualizar Posición en Campo** que mantiene la percepción de la situación relativa del robot en el campo de la labor, es decir si está o no en una línea de cultivo.

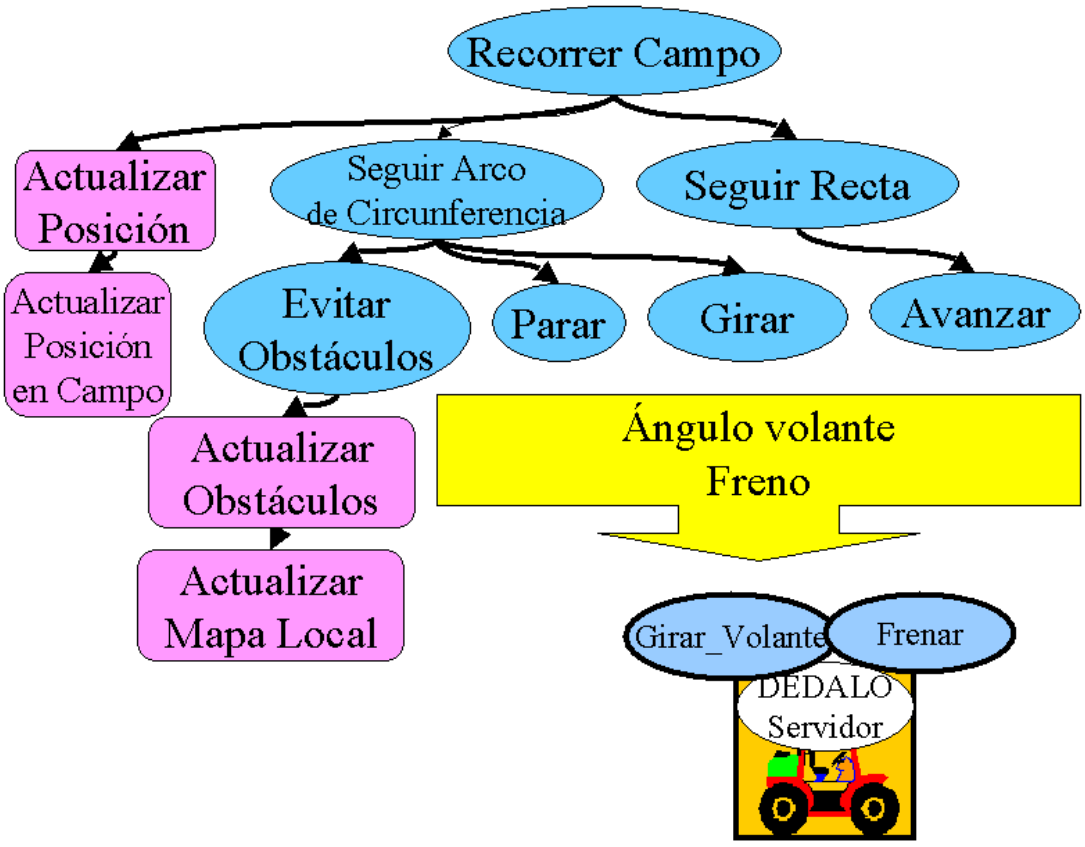


Figura 5.43 : Jerarquía de agentes en AGRO-AMARA para la navegación específica de laboreo

A continuación se describen detalladamente los agentes exclusivos de la navegación de laboreo, pues los comunes a la navegación global se presentaron en el apartado 5.2 de este capítulo.

5.3.1 Agentes perceptivos

El único agente perceptivo que la navegación de laboreo no comparte con la general es el agente **Actualizar Posición en Campo** que se describe a continuación.

Agente Actualizar Posición en Campo

Un recorrido típico de un tractor por un campo de cultivo se compone de trayectorias rectas y curvas para cambiar de una línea de cultivo a otra [Stoll, 2003, Thuilot et al., 2002]. Este entramado de trayectorias se repite en muchos tipos de cultivos, ya sea cereal [Pilarski et al., 2002], frutales [Stentz et al., 2002], cultivos de huerta [Astrand y Baerveldt, 2002] o jardinería (grandes extensiones de césped) [Stoll, 2003].

En la arquitectura para la navegación de laboreo propuesta en esta tesis, se han diseñado dos agentes para el control de las trayectorias de laboreo: el agente **Seguir Recta** en los tramos rectilíneos y el agente **Seguir Arco de Circunferencia** en las trayectorias curvas que conectan dos líneas consecutivas de cultivo. Para que estos dos agentes puedan "distinguir" cuándo tienen que tomar el control de la navegación es necesario identificar perfectamente las condiciones del cambio. Por esta razón el agente **Actualizar Posición en Campo** mantiene constantemente actualizada la percepción *En Línea*, que toma el valor CIERTO si el robot se encuentra dentro del área de navegación en línea recta y FALSO si está en una zona de transición entre dos líneas de cultivo.

La percepción *En Línea* es utilizada por los agentes **Seguir Recta** y **Seguir Arco de Circunferencia** para determinar su valor de activación: si es CIERTA responderá al contexto de activación de **Seguir Recta** y si es FALSA a la de **Seguir Arco de Circunferencia**. El hecho de encapsular el agente perceptivo **Actualizar Posición en Campo** añade modularidad al sistema, de manera que este agente puede tener diversas implementaciones dependiendo del tipo concreto de tarea y cultivo. Esta característica es especialmente relevante, ya que dependiendo del tipo de tarea agrícola que se pretenda realizar, tanto el criterio de decisión como los sensores más adecuados van a variar. No es lo mismo detectar los surcos en un campo de coliflores,

donde pueden distinguirse mediante visión [Hague et al., 1999b], que detectarlos en campos de frutales, donde además de un sensor de visión puede ser útil un sensor láser, o en campos de cereales, donde no existe una estructura real que corresponda a los surcos y por lo tanto las líneas rectas deben determinarse a partir de la localización precisa del robot sobre un mapa. Además, encapsular la percepción *En Línea* independientemente de los agentes de actuación permite que otros agentes puedan disponer de ella sin necesidad de volver a calcularla.

Por lo tanto el objetivo del agente **Actualizar Posición en Campo** es elaborar y mantener actualizada la percepción *En Línea*. El agente **Actualizar Posición en Campo** necesita conocer: 1) el plan de navegación de laboreo, denominado *Plan de Navegación de Laboreo*, diseñado a priori por el usuario del sistema e introducido manualmente marcando las posiciones consecutivas por las que el tractor debe pasar, sobre una imagen digital georreferenciada del campo de labor, 2) la percepción *Posición*, actualizada por el agente **Actualizar Posición** que es activado por el agente **Actualizar Posición en Campo**.

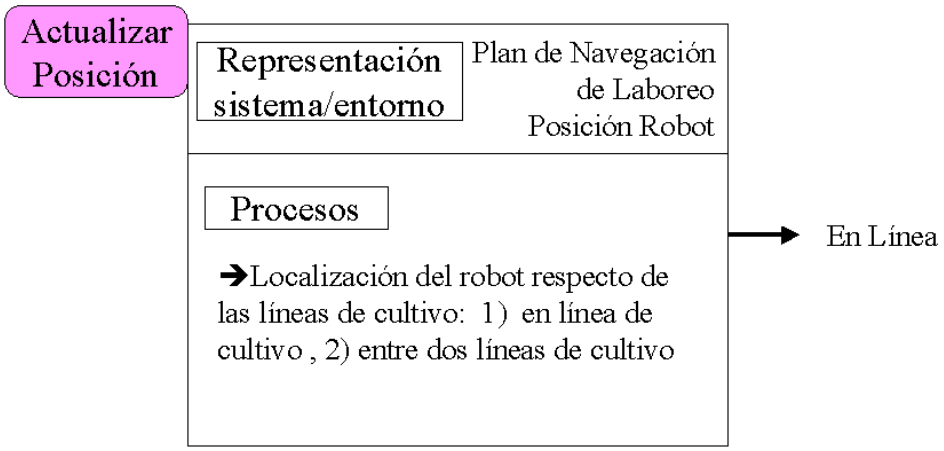


Figura 5.44 : Esquema de entradas, salidas, representación y procesos del agente **Actualizar Posición en Campo**

El agente **Actualizar Posición en Campo**, conociendo el *Plan de Navegación de Laboreo*, el punto de ejecución del plan y la posición del robot (*Posición*), establece una división del espacio de laboreo en tres zonas distintas: (1) zona próxima al final de la trayectoria de navegación en curso (próxima al punto fin de línea), (2) zona próxima a la recta que define

la línea de cultivo y 3) resto del espacio, figura 5.45.

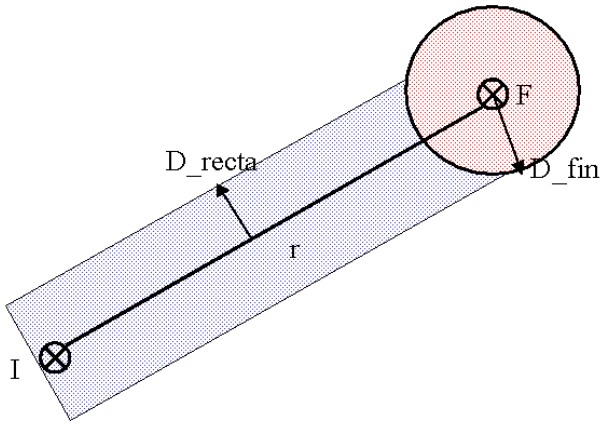


Figura 5.45 : Configuración del espacio de laboreo del agente perceptivo **Actualizar Posición en Campo**. En color azul la zona de la recta y en color rojo la zona de fin de recta. El resto del espacio está en color blanco

El agente **Actualizar Posición en Campo** comprueba en primer lugar si la *Posición* actual del robot está más próxima al punto fin de línea F que un umbral D_{fin} , fijado en 2 (m.) (este umbral se ha fijado considerando las dimensiones del robot). Si se verifica esta condición, asigna el valor FALSO a la percepción *En Línea* y detiene la ejecución del agente el tiempo de espera establecido. Si la condición no se cumple, pasa a verificar si el robot se halla próximo al segmento $\overline{IF}$. Para ello comprueba si la distancia entre la *Posición* del robot y la recta es menor que la distancia umbral d_r. Esta distancia viene determinada por la expresión (5.12):

$$d(R,r) = \frac{|-(y_f - y_i)\cdot x + (x_f - x_i)\cdot y + (x_i \cdot y_f - y_i \cdot x_f)|}{\sqrt{(x_f - x_i)^2 + (y_f - y_i)^2}} \tag{5.12}$$

Si $d(R,r) \leq d_r$ entonces el robot está siguiendo la recta correctamente y *En Línea* se actualiza con valor CIERTO. En caso contrario el robot no está siguiendo la recta ni tampoco se encuentra próximo al fin de línea. Entonces si el valor anterior de *En Línea* era FALSO, el robot se encuentra girando hacia una nueva recta alejado del fin de línea, por lo tanto se mantiene dicho valor. Sin embargo, si el valor anterior de *En Línea* era CIERTO el robot se ha alejado mucho de la recta ideal y es necesario enviar una señal de error y detener la ejecución. El diagrama de flujo de información mostrado en la figura 5.46 y el pseudocódigo del agente

Actualizar Posición en Campo, anexo A, clarifican la ejecución de los procesos embebidos en el agente.

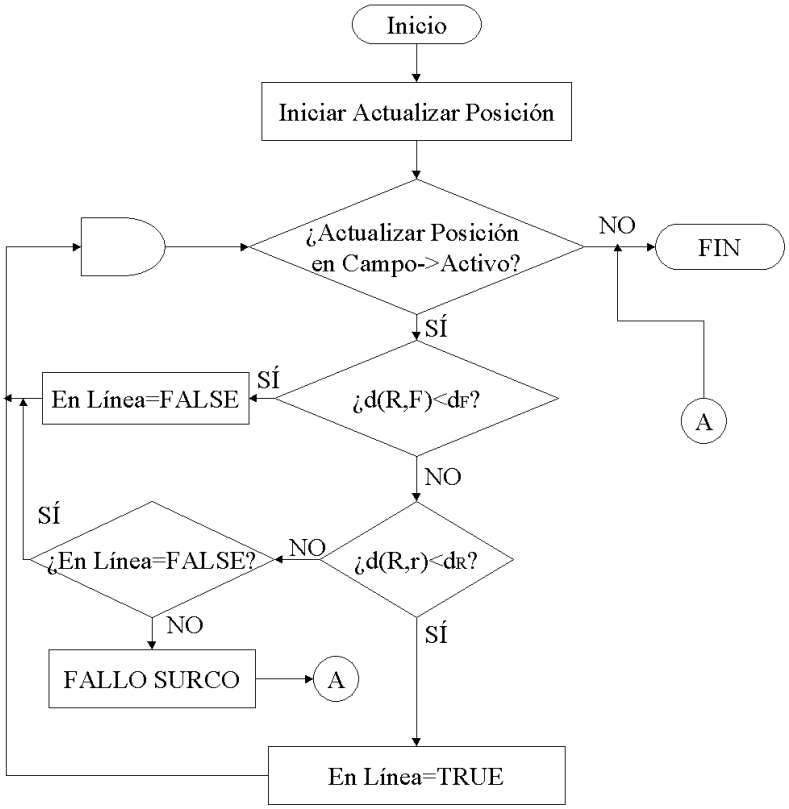


Figura 5.46 : Diagrama de flujo de información del agente **Actualizar Posición en Campo**

5.3.2 Agentes de actuación

Girar

El agente **Girar** tiene asignado el control de los giros entre líneas de cultivo, ya sean éstas consecutivas o no. Es un agente muy especializado en una tarea que tiene por objetivo forzar al máximo el giro, y en ello estriba su diferencia con el agente **Avanzar**, dotado de mayor flexibilidad. El objetivo del agente **Girar**, una vez asignado un radio de giro, es enviar la consigna de giro de volante más adecuada para que el robot se desplace desde su posición inicial (x_i, y_i, θ_i) a una posición final (x_f, y_f, θ_f) tal que $\theta_f = \theta_i + \pi$ y $r_{giro} = \frac{\sqrt{(x_f - x_i)^2 + (y_f - y_i)^2}}{2}$.

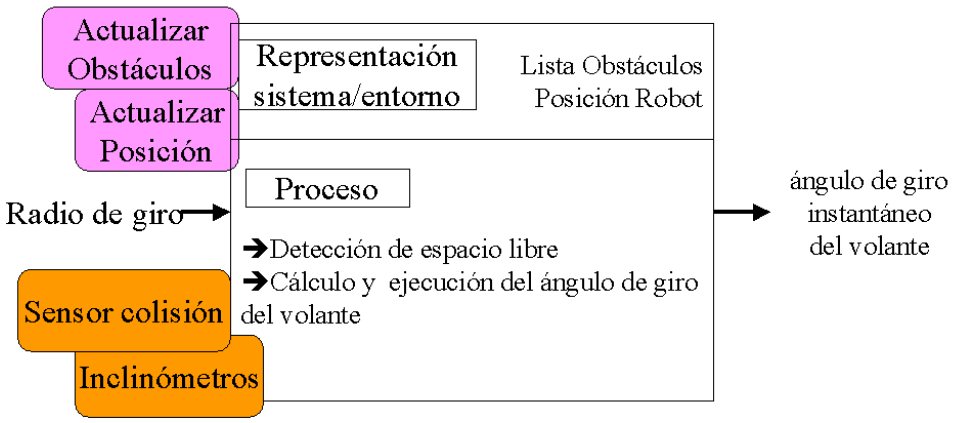


Figura 5.47 : Esquema de entradas, salidas, representación y procesos del agente **Girar**

- **Entradas, salidas y representación del agente Girar.** El agente **Girar** necesita como parámetro modulador de su comportamiento el radio de giro deseado r_{giro}, proporcionado por el agente que lo active. Cada ciclo de control del agente **Girar** devuelve un valor del ángulo de volante, que es enviado a los controladores de bajo nivel del robot.

- **Contexto de activación.** Una vez que el agente **Girar** pasa al estado de **alerta** empieza la comprobación cíclica de su contexto de activación, el mismo que el del agente **Avanzar**; es decir, ausencia de colisión, valores de las pendientes lateral y frontal menores de $15\,\%$ y ausencia de *Obstáculos* que ocupen más de 3 celdillas adyacentes a distancia menor de 4 (m.) del robot. Para ello, al igual que el agente **Avanzar**, necesita disponer de las percepciones *Posición* y *Obstáculos*, por lo que envía la señal de activación a los agentes **Actualizar Posición** y **Actualizar Obstáculos**, figura 5.47.

- **Procesos del agente Girar.** En cada iteración el agente comprueba en primer lugar que el radio de giro r_{giro} se encuentra dentro del intervalo de valores permitidos por las restricciones cinemáticas del robot. El radio de giro mínimo del robot se ha calculado experimentalmente, trazando un círculo a la derecha y otro a la izquierda con el volante girado al máximo en cada sentido. Los radios de giro mínimos son $R_{min} = 4{,}7$ (m.) para giros hacia la derecha y $R_{min} = 5{,}0$ (m.) para giros a la izquierda, que corresponden con ángulos de giro máximos de

$\alpha_{max} = 60$ (grad.) a la derecha y $\alpha_{max} = -50$ (grad.) a la izquierda.

Si el radio de giro requerido es menor que el radio mínimo correspondiente se envía un mensaje de error al operario; en un futuro el agente **Girar** incorporará una estrategia alternativa para la resolución autónoma de esta situación. Si r_{giro} se encuentra en el intervalo de valores permitidos, se procede a calcular el sentido del giro. Para ello se compara la pendiente de la recta que une la posición del robot y el destino ψ, con el ángulo de orientación del robot θ_R de manera similar al procesamiento que sigue el agente **Avanzar**. Si $(\psi - \theta_R) > 0$, se realiza un giro hacia la izquierda y en caso contrario hacia la derecha. Una vez calculado el sentido de giro hay que calcular el valor del ángulo de volante necesario para alcanzar la posición deseada. Para ello se parte de un modelo cinemático del vehículo y se construye, en la etapa inicial de diseño, una tabla analítica que relaciona los valores de giro con el ángulo de volante requerido para realizarlo. El agente **Girar** busca en la tabla el valor de ángulo de volante α_v necesario para un r_{giro} dado. El modelo cinemático utilizado es el modelo de la bicicleta simplificado.

Una vez que el agente **Girar** ha encontrado en la tabla el ángulo de volante α_v, correspondiente con el radio de giro r_{giro} requerido, lo envía a DÉDALO-Servidor y espera el tiempo de ciclo establecido antes de iniciar una nueva iteración, como se observa en el diagrama de flujo de la figura 5.48.

Seguir Recta

Navegar en línea recta correctamente es uno de los objetivos requeridos en la navegación de laboreo. El agente **Seguir Recta** se responsabiliza de esta tarea en la arquitectura AGRO-AMARA. Así, dado un segmento de una línea recta definido por sus puntos inicio de línea y fin de línea $\overline{IF}$, el agente **Seguir Recta** organiza el conocimiento para el control de la estrategia de seguimiento preciso de la trayectoria rectilínea delimitada por los dos puntos. Además, mantiene en todo momento los objetivos básicos de seguridad frente a colisiones y fallos en el funcionamiento del sistema. Para ello el agente **Seguir Recta** reutiliza las habilidades de los agentes de navegación local **Avanzar**, **Evitar Obstáculos** y **Parar**, igual que **Ir a Punto**.

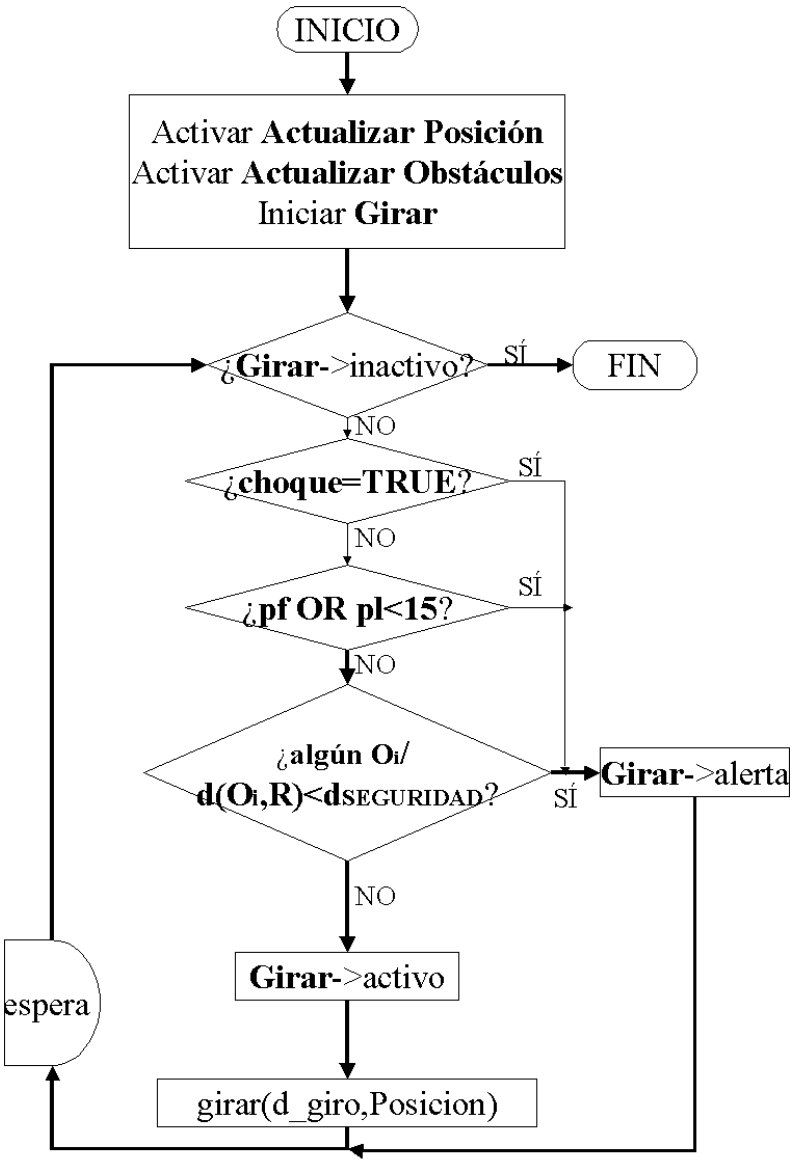


Figura 5.48 : Diagrama de flujo de información del agente **Girar**

Sin embargo para forzar el seguimiento preciso de una trayectoria rectilínea, **Seguir Recta** los modula de forma diferente a como lo hace **Ir a Punto**. Este caso es un ejemplo de cómo la arquitectura AGRO-AMARA permite mediante la modulación conseguir comportamientos diferentes con el mismo conjunto de agentes básicos.

El agente **Seguir Recta** es el agente de navegación activo cuando el robot se encuentra en una línea de cultivo, es decir, cuando la percepción *En Línea*, elaborada por el agente **Actualizar Posición en Campo**, toma el valor CIERTO. Tal y como muestra el esquema de entradas y

salidas del agente **Seguir Recta**, figura 5.50, las entradas necesarias son los puntos inicio de línea y fin de línea que definen la línea recta objetivo. La salidas del agente **Seguir Recta** son: 1) objetivo sobre la línea recta, que constituye en cada instante el subobjetivo concreto del agente **Avanzar** y 2) una señal de fallo. En consecuencia, el agente **Seguir Recta** modula a los agentes de actuación **Avanzar**, **Evitar Obstáculo** y **Parar** y a los agentes perceptivos, **Actualizar Posición en Campo** y **Actualizar Posición**. La figura muestra la jerarquía para el agente **Seguir Recta**.

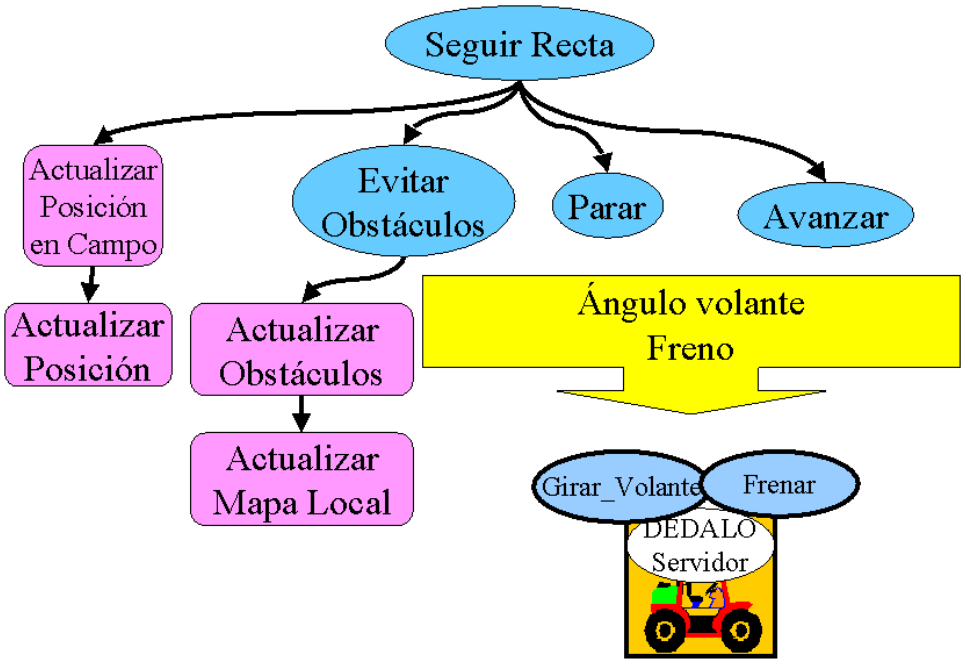


Figura 5.49 : Jerarquía del agente de actuación **Seguir Recta**

La estrategia del agente **Seguir Recta** se basa en una idea clásica para el seguimiento de trayectorias y se halla presente en algoritmos de uso tan extendido como el de persecución. Esta idea es la de discretizar el camino y describirlo como un conjunto de puntos a los que el robot tiene que ir dirigiéndose consecutivamente. En este caso el camino se discretiza en segmentos de 5 (m.) de longitud, medida apropiada para el tamaño del vehículo automatizado DÉDALO. Dados los puntos I, F que definen la línea recta objetivo, se calcula la distancia entre ambos d_r y su pendiente β, a partir de las ecuaciones (5.14).

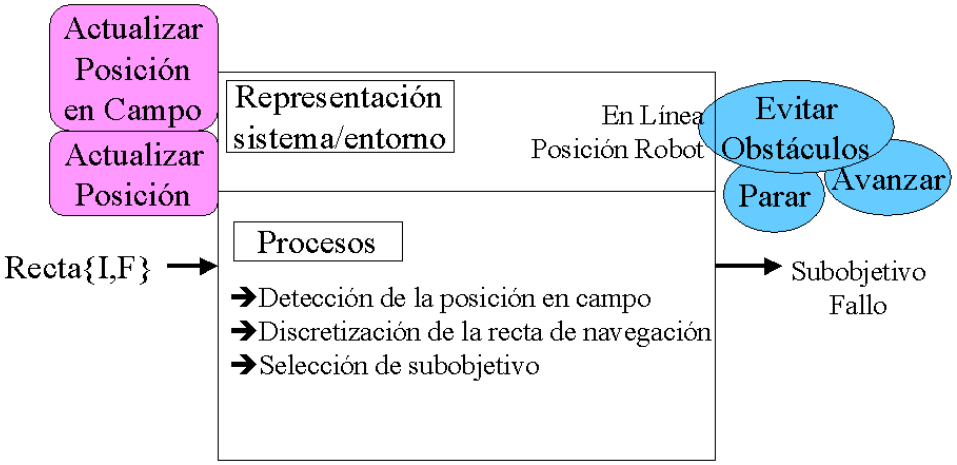


Figura 5.50 : Esquema de entradas, salidas, representación y procesos del agente **Seguir Recta**

$$d_r = \sqrt{(x_f - x_i)^2 + (y_f - y_i)^2} \tag{5.13}$$

$$\beta = \arctan(\frac{y_f - y_i}{x_f - x_i}) \tag{5.14}$$

El segmento de la línea recta, se discretiza en subsegmentos de 5 (m.) de longitud, obteniéndose los siguientes puntos subobjetivo: G_k donde $k = 1, ..., \frac{d_r}{5}$ (5.15):

$$G_k = \left\{ \begin{array}{l} x_k = x_{k-1} + \frac{d_r}{5} \cdot \cos(\beta) \\ y_k = y_{k-1} + \frac{d_r}{5} \cdot \sin(\beta) \\ \theta_k = \beta \end{array} \right\} \tag{5.15}$$

Una vez discretizada la recta, el agente **Seguir Recta** opera de modo similar al agente **Ir a Punto**, comprobando en cada iteración si se ha alcanzado el subobjetivo, y si es así, envía el siguiente subobjetivo al agente **Avanzar**. Tal y como se muestra en el diagrama de flujo de información de la figura 5.51, y en del pseudocódigo correspondiente a los procesos del agente **Seguir Recta**, anexo A; el agente **Seguir Recta** inicia sus procesos activando a los agentes **Actualizar Posición**, **Actualizar Posición en Campo**, **Avanzar**, **Evitar Obstáculos** y **Parar**. Una vez activados, y siempre que se encuentre en estado **activo** o de **alerta**, el agente

Seguir Recta ejecuta sus procesos cíclicamente. Así, en cada iteración confirma si se verifica su contexto de ejecución, comprobando si la variable *En Línea* es CIERTA; si no lo es pasa al estado de **alerta** y espera su tiempo de ciclo antes de reiniciar su ejecución. Si *En Línea* es CIERTA, comprueba que el agente **Avanzar** se encuentra en estado de **alerta** o en **activo**. También hay que comprobar si es necesario volver a discretizar la recta, ya que cuando se cambia de una línea de cultivo a la siguiente, la recta también cambia. Para detectar este cambio se comparan los puntos inicio de línea y fin de línea actuales con los de la iteración anterior. El resto de la secuencia de procesos es similar a la del agente **Ir a Punto**.

Agente Seguir Arco de Circunferencia

Las trayectorias a seguir en una navegación de laboreo se componen, en la inmensa mayoría de los casos, de tramos rectilíneos y tramos curvos correspondientes a cambios entre dos líneas de cultivo diferentes. Este cambio implica usualmente un giro de 180 grados, esto es, un cambio de sentido del vehículo [Thuilot et al., 2002, Pilarski et al., 2002]. El objetivo del agente **Seguir Arco de Circunferencia** es, partiendo de la posición fin de línea de una de las líneas de cultivo, situar el tractor al comienzo de la siguiente línea de cultivo (punto inicio de línea) con la orientación adecuada, evitando colisiones y vuelcos.

- **Entradas, salidas y representación del agente Seguir Arco de Circunferencia.** En la figura 5.52 se muestra el esquema de entradas, salidas, representación y procesos del agente **Seguir Arco de Circunferencia**. Para iniciar los procesos, el agente **Seguir Arco de Circunferencia** necesita que las percepciones *En Línea* y *Posición* estén actualizadas y por lo tanto procede a activar los agentes perceptivos **Actualizar Posición** y **Actualizar Posición en Campo**. Las entradas del agente son las dos rectas entre las que debe describir el arco de circunferencia. Las salidas de **Seguir Arco de Circunferencia** son, figura 5.52: 1) el diámetro de la circunferencia asociada al giro que se envía al agente **Girar** y 2) una señal de fallo si las líneas de cultivo se encuentran más próximas que el diámetro mínimo asociado al giro máximo del tractor.

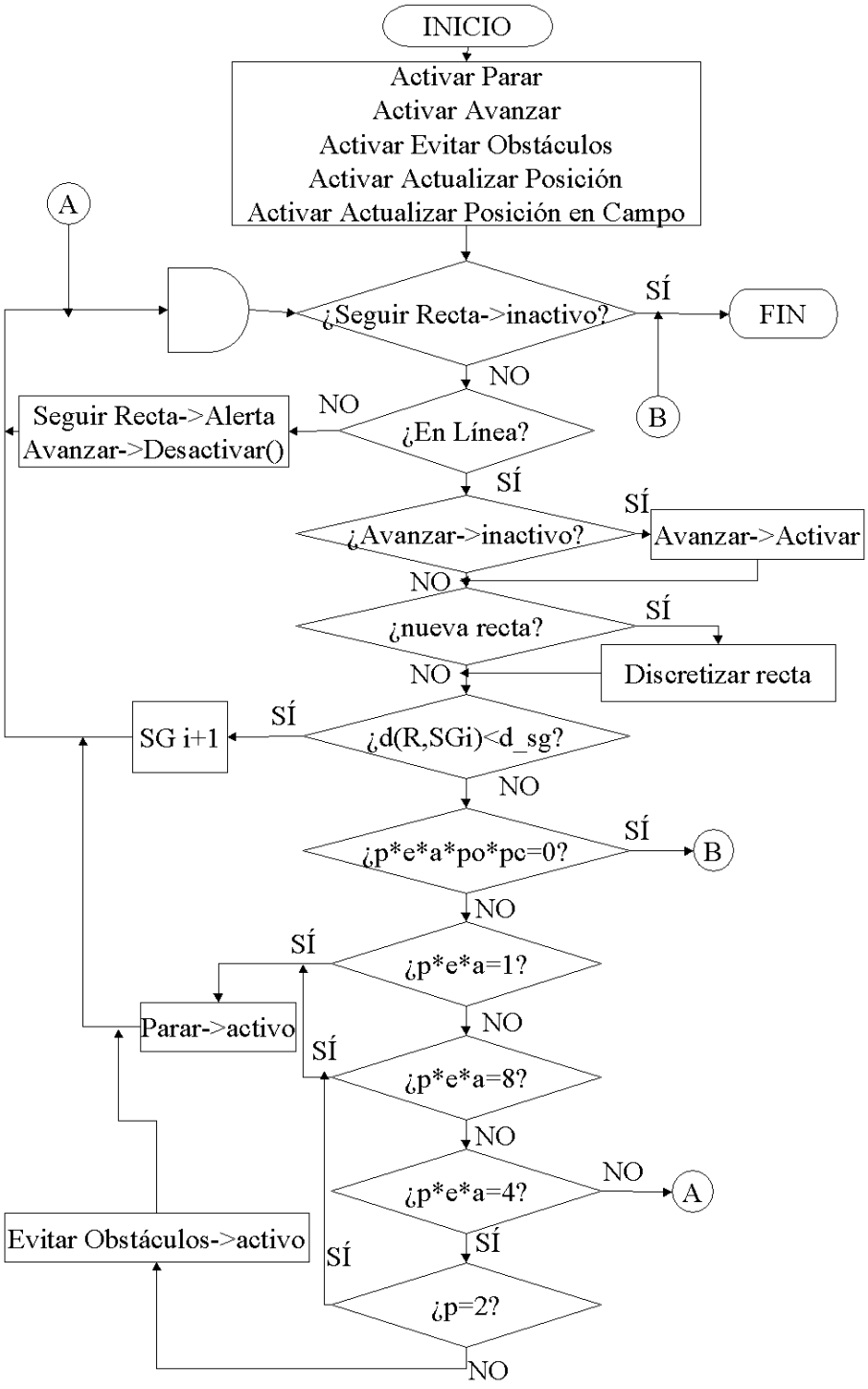


Figura 5.51 : Diagrama de flujo de información del agente **Seguir Recta**

- **Contexto de activación.** Puesto que los agentes **Seguir Arco de Circunferencia** y **Seguir Recta** se alternan en el control del robot durante la navegación de laboreo, el contexto

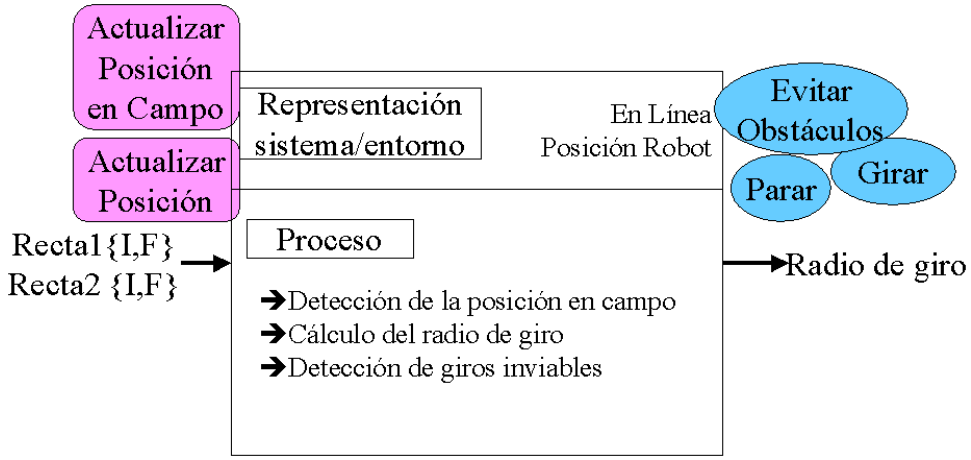


Figura 5.52 : Esquema de entradas, salidas, representación y procesos del agente **Seguir Arco de Circunferencia**

de ejecución del agente **Seguir Arco de Circunferencia** será el complementario del contexto del agente **Seguir Recta**. Por lo tanto, el agente **Seguir Arco de Circunferencia** tomará el control de la navegación cuando *En Línea* tenga valor FALSO. A su vez, el agente **Seguir Arco de Circunferencia** modula los agentes **Parar**, y **Evitar Obstáculo**, encargados de la seguridad, y el agente **Girar**, quien sustituyendo al agente **Avanzar** efectúa el giro de 180 grados necesario para cambiar de línea de cultivo.

- **Procesos del agente Seguir Arco de Circunferencia** El modo de operar del agente **Seguir Arco de Circunferencia** se esquematiza en el diagrama de flujo de información, figura 5.54. En cada ciclo se comprueba en primer lugar el valor de la percepción *En Línea*: si es CIERTA, el agente **Seguir Arco de Circunferencia** pasa al estado de **alerta**. Si es FALSA, se cumplen sus condiciones de ejecución y pasa a comprobar el estado del agente **Girar**, para activarlo si es que está **inactivo**. El siguiente paso es verificar que la distancia entre la posición fin de línea de la línea de cultivo actual y la siguiente permite efectuar el giro. Si no es así porque $d(F_i, I_{i+1}) \leq 5$ (m.), envía una señal de error y detiene la ejecución. Si hay suficiente espacio libre, se comprueba el estado de los agentes. Si no hay errores se procede al cálculo del sentido de giro, que vendrá definido por la distancia relativa de la posición destino (inicio de línea) con respecto a la *Posición* del robot.

Tal y como se muestra en la figura 5.53, se calcula la pendiente de la recta que une al robot con el subobjetivo α_r (posición inicio de línea de la siguiente línea de cultivo) y se le resta la orientación del robot θ_R. Si esta diferencia es positiva el robot tiene que girar a la izquierda y el diámetro asociado al giro se toma como negativo, por lo tanto $d_{giro} = -d(R, I_{i+1})$. En caso contrario, diferencia negativa, el giro es hacia la derecha y $d_{giro} = d(R, I_{i+1})$. Finalmente, se espera el tiempo de ciclo establecido antes de iniciar la siguiente iteración.

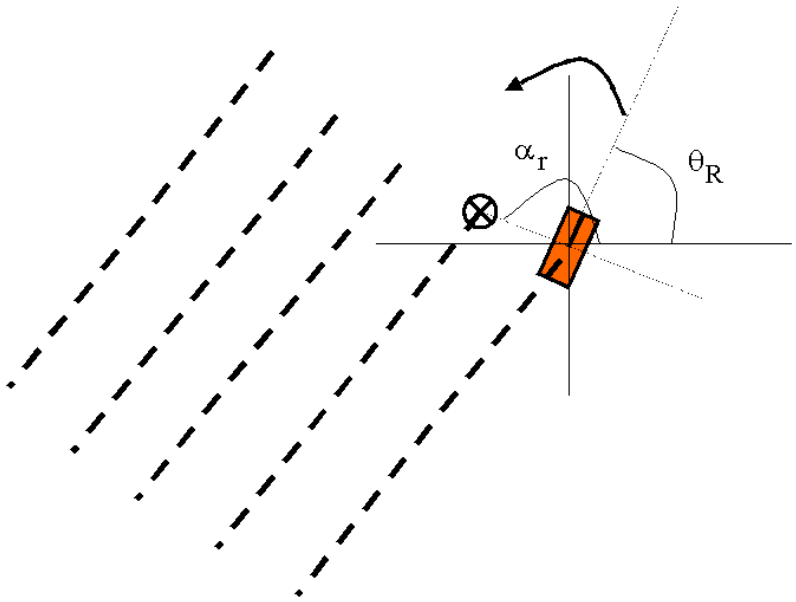


Figura 5.53 : Esquema para el cálculo del sentido del giro en el agente **Seguir Arco de Circunferencia**

Agente Recorrer Campo

El agente **Recorrer Campo** es el responsable de la navegación global de laboreo, es decir, de realizar la gestión de procesos necesaria que conduce a recorrer el campo de cultivo en toda su extensión (desde el punto de inicio de laboreo al punto final recorriendo todos los tramos de laboreo). La realización autónoma de la tarea agrícola correspondiente implica el seguimiento correcto las líneas de cultivo con las especificaciones de precisión facilitadas por el agricultor. Por ello, el agente **Recorrer Campo** dispone de un mapa digital geo-referenciado y un *Plan de Navegación de Laboreo*, que en la etapa actual se inyectan manualmente. No obstante, la

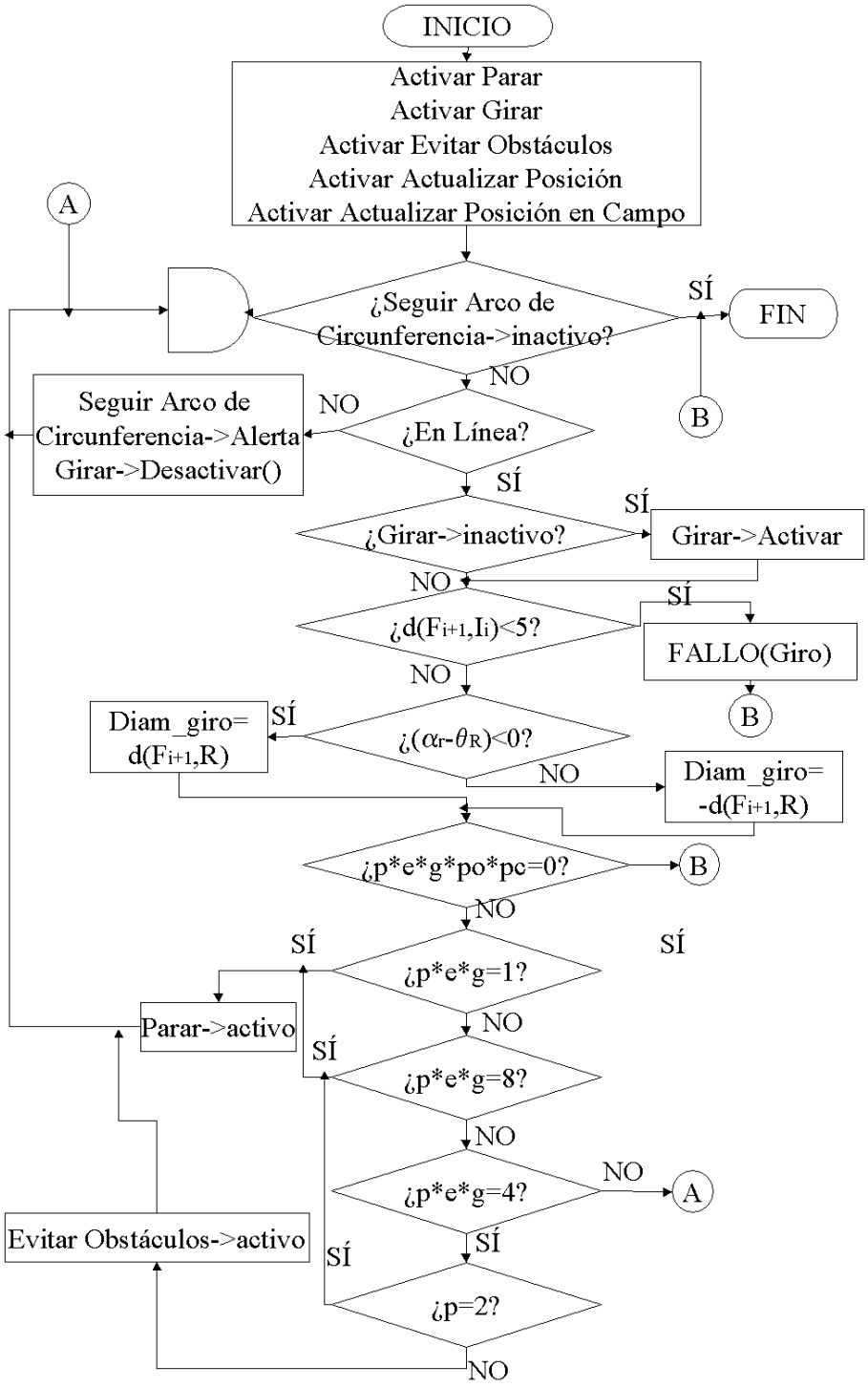


Figura 5.54 : Diagrama de flujo de información del agente **Seguir Arco de Circunferencia**

implementación del sistema está orientada a que en un futuro este plan sea generado por un planificador específico de navegación de laboreo [Stoll, 2003].

Los objetivos del agente **Recorrer Campo** son tres:

1. Determinar, partiendo del *Plan de Laboreo* y de la *Posición* del robot, en qué línea de cultivo se halla el tractor y cambiar adecuadamente las posiciones inicio de línea y fin de línea que la definen.

2. Detectar si el robot ha completado el *Plan de Navegación de Laboreo* asignado, llegando al destino de laboreo.

3. Detectar posibles fallos durante la ejecución.

Para llevar a cabo su tarea el agente **Recorrer Campo** envía señales de activación a tres agentes, **Seguir Recta** y **Seguir Arco de Circunferencia**, para el control de dos tramos específicos del camino, y **Actualizar Posición**, para mantener la *Posición* del robot, figura 5.56. Para iniciar la secuencia de procesos, el agente **Recorrer Campo** necesita el mapa digital georrefenciado del campo y el *Plan de Laboreo*. Las salidas del agente **Recorrer Campo** son las posiciones inicio de línea y fin de línea del tramo en el que está navegando el robot, $R = \overline{IF}$, la señal de FIN de ejecución, que indica que ha llegado a la posición destino de laboreo del campo satisfactoriamente, y la señal de FALLO, figura 5.55.

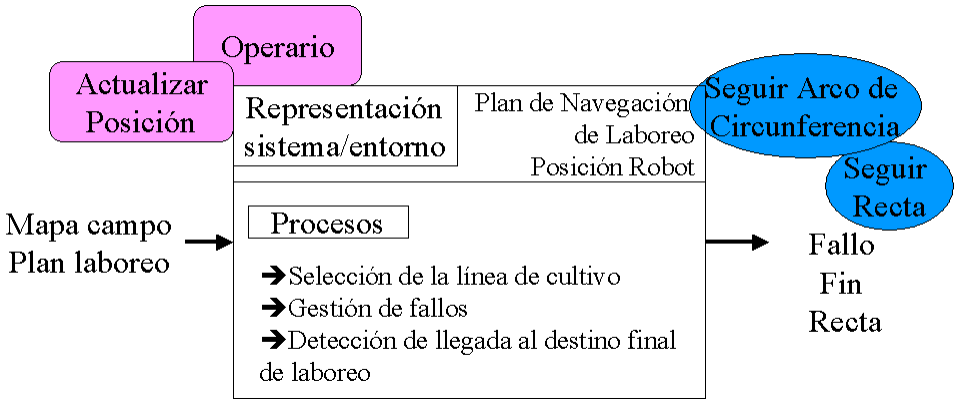


Figura 5.55 : Esquema de entradas, salidas, representación y procesos del agente **Recorrer Campo**

Una vez completadas las inicializaciones del plan de laboreo (leer el plan e inicializar las variables) y siempre que **Recorrer Campo** se mantenga en estado **activo** o **alerta** se inician los

procesos de forma iterativa. En primer lugar se comprueba en qué posición del plan se encuentra el robot, analizando si se ha alcanzado ya el punto inicio de línea de la recta de navegación actual, esto es, el punto I. Si aún no se ha alcanzado, se establece este punto como objetivo. Si ya se ha alcanzado se comprueba entonces si el tractor ha llegado al punto fin de línea F. Si se ha llegado al punto fin de línea se pasa al siguiente punto del *Plan de Navegación de Laboreo*. Si no se ha alcanzado, se establece el punto fin de línea de la recta actual F como objetivo.

Una vez determinado el punto de ejecución en que se encuentra el tractor, se comprueba si el robot ha alcanzado el objetivo, analizando la distancia que lo separa de él. Si esa distancia es mayor que el umbral D_{goal} establecido, se ejecuta el último proceso, esto es, la comprobación de errores en los agentes. Si la distancia que separa al robot del objetivo es menor que D_{goal}, hay que seleccionar un nuevo objetivo. Para ello se analiza si el objetivo al que se dirige el robot es el punto inicio de línea o el fin de línea. Si $G = I$ (el robot se dirige hacia el inicio de línea), entonces el punto que ha alcanzado es I, marcándose como alcanzado. De este modo en el proceso de comprobación del plan en la siguiente iteración, se verificará que el punto inicio de línea ya ha sido alcanzado. Si $G \neq I$ (el robot no se dirige al punto inicio de línea), se comprueba si se dirige a F (fin de línea); si es así, entonces $G = F$ y el punto que el robot ha alcanzado es F, se marca como alcanzado y se procede a la comprobación de fallos. En el caso en que $G \neq F$ ha ocurrido algún fallo, se envía un mensaje al operario y se finaliza el proceso.

El proceso de comprobación de fallos verifica en primer lugar que todos los agentes funcionan correctamente, si no es así se notifica el fallo y se finaliza el proceso. En caso de que los dos agentes de actuación **Seguir Recta** y **Seguir Arco de Circunferencia** estén activos se obliga al agente **Seguir Arco de Circunferencia** a pasar al estado de **alerta**, se espera el tiempo de ciclo correspondiente y se inicia la siguiente iteración.

Este capítulo describe la arquitectura híbrida multiagente AGRO-AMARA propuesta para la navegación de robots agrícolas autónomos, tanto para navegación global genérica como para navegación específica para laboreo. Tiene como principios básicos la reusabilidad de las habilidades, la facilidad de escalado y la existencia de un flujo bidireccional de

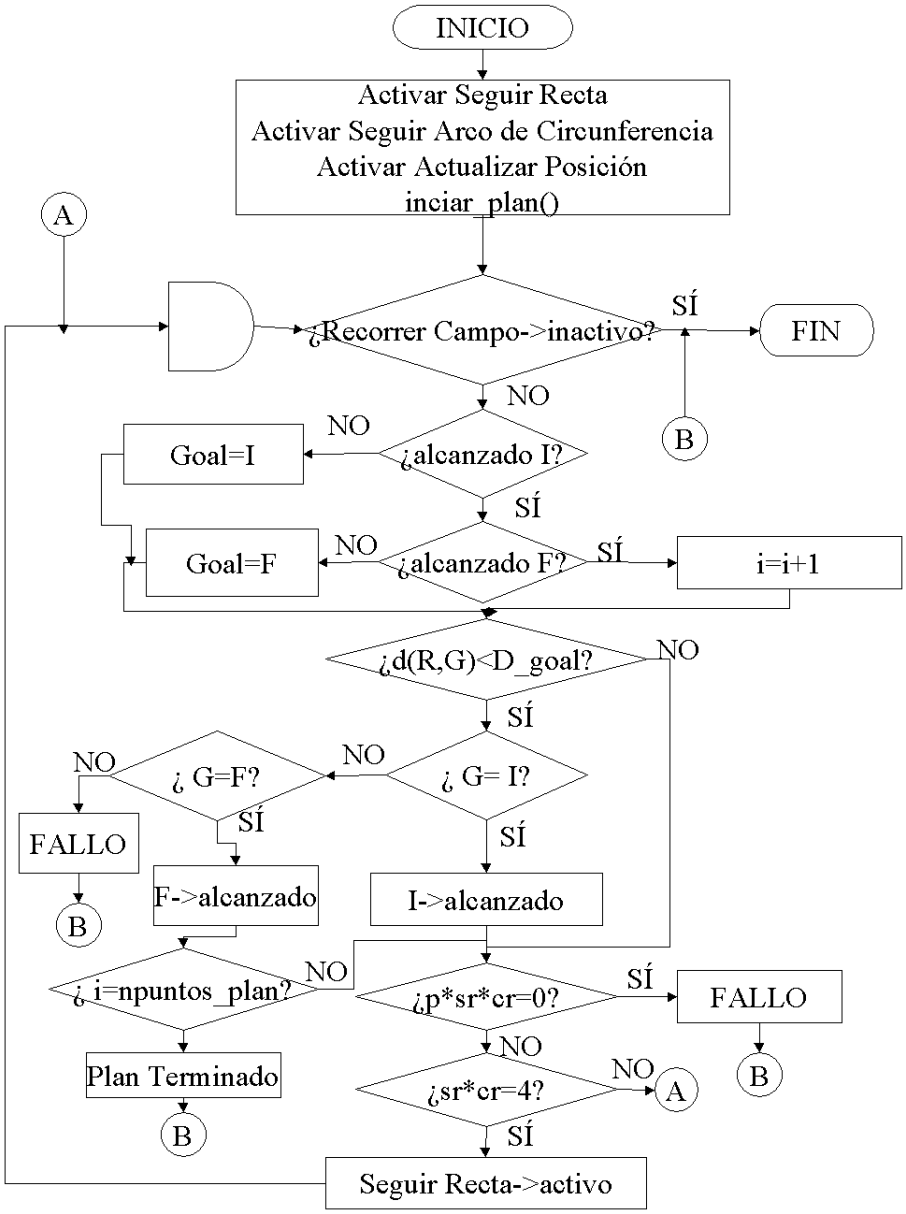


Figura 5.56 : Diagrama de flujo de información del agente **Recorrer Campo**

percepción y acción entre los agentes. La unidad básica de la arquitectura es el agente, entendido como *"proceso o conjunto de procesos dirigidos a conseguir o mantener un objetivo capaz de habilidades perceptivas, deliberativas y reactivas sin restricción alguna en su complejidad"*[García-Alegre y Recio, 1998]. Los agentes se organizan jerárquicamente en la arquitectura de modo que cada agente inicia la ejecución de aquellos cuya función necesita, permitiendo así la reutilización de las habilidades de los agentes y facilitando la integración de

nuevos agentes.

AGRO-AMARA divide los agentes en perceptivos y de actuación, en la línea de los trabajos de Arkin [Arkin, 1998]. Esta división favorece la reutilización de los procesos perceptivos y de acción, como se demuestra con los agentes **Actualizar Posición, Actualizar Mapa Local, Actualizar Obstáculos, Avanzar, Evitar Obstáculos** y **Parar**, empleados en diferentes situaciones de navegación. La activación selectiva de los agentes perceptivos permiten que la percepción en AGRO-AMARA esté orientada a la actuación, requisito considerado importante en una arquitectura para robots agrícolas, capítulo 3.

Las representaciones del sistema y entorno, elaboradas por los agentes perceptivos, se almacenan en una memoria compartida de pizarra. Esta memoria permite la coexistencia de estructuras representativas diferentes con distintos grados de abstracción para que cada agente emplee la información más adecuada a sus objetivos y procesos. En este sentido, la memoria compartida de AGRO-AMARA es similar al espacio perceptivo local de Shapira [Konolige et al., 1997]. También comparte con NHCA la consideración de que es necesario disponer de representaciones con resoluciones diferentes, aunque no fuerza la existencia de los mismos niveles de actuación y percepción. Esta característica también diferencia a AGRO-AMARA de su predecesora AMARA.

AGRO-AMARA permite la incorporación de la información sobre el sistema, la tarea y el entorno existente a priori. Este es el caso de los mapas georreferenciados y el plan de navegación de laboreo. La posibilidad de inyección inicial de información se consideró como un requisito conveniente en un arquitectura de control para robots agrícolas.

La selección de acción se articula en AGRO-AMARA definiendo cada agente con un contexto de activación propio. Tanto los objetivos, a través de las señales de activación y desactivación, como el entorno, a través del contexto, seleccionan el agente que toma el control en cada instante. El arbitraje está implícito en el diseño de los contextos de activación, de modo parecido al propuesto en la arquitectura de subsunción [Brooks, 1986] o de Mataric [Mataric, 1991].

La navegación en un robot agrícola es la tarea implementada con la arquitectura AGRO-

AMARA. Esta tarea de navegación puede descomponerse en dos partes bien diferenciadas, por un lado una navegación global genérica, común a cualquier otro robot autónomo. Su objetivo básico es asegurar el desplazamiento del robot de un punto a otro del entorno, evitando colisiones y optimizando la ruta, teniendo en cuenta las características propias de un robot agrícola (trabajo en exteriores, baja maniobrabilidad, etc...). Y por otro la navegación específica, orientada a realizar o facilitar la realización de una tarea agrícola, como puede ser la fumigación, fertilización o riego selectivo, la plantación o sembrado y la cosecha.

En la navegación de laboreo las trayectorias poseen unas restricciones más fuertes que en la navegación global de propósito general, como la que se halla implementada en el agente **Ir a Punto**. En primer lugar, el objetivo básico es diferente, pues en la navegación global se trata de alcanzar un objetivo, mientras que en la navegación de laboreo es fundamental la forma en que se debe recorrer esa trayectoria, ya que en su ejecución se realiza una labor agrícola concreta. A causa de estas diferencias se han encapsulado los agentes encargados de la navegación genérica y de laboreo por separado. La interfaz de usuario para la navegación genérica recibe el nombre de NAVEGADOR y la específica del laboreo NAVEGADOR DE LABOREO.

No obstante existen objetivos comunes en ambos tipos de navegación, como la seguridad ante imprevistos y el objetivo de llegar a un destino final. Aunque la navegación de laboreo utiliza estrategias diferentes a la navegación global motivadas por los requisitos diferentes, al compartir ciertos objetivos, algunos agentes de la navegación global se reutilizan en ambos escenarios. La arquitectura multiagente AGRO-AMARA contempla entre sus principios básicos el de la reutilización de agentes.

El agente **Ir a Punto** constituye el experto en la gestión de la navegación global genérica. Para lograr su cometido, -asegurar el desplazamiento del robot de un punto a otro del entorno, evitando colisiones y optimizando la ruta-, modula los agentes de navegación local: el agente **Avanzar** que realiza la navegación hacia el objetivo, el agente **Evitar Obstáculos**, el agente **Parar** que se encargan de la navegación segura y el agente **Planificar Caminos** que realiza la planificación de la ruta global. Tres agentes perceptivos elaboran y mantienen las estructuras perceptivas necesarias para los agentes de actuación. El agente perceptivo **Actualizar Posición**

mantiene actualizada la posición del robot, el agente **Actualizar Obstáculos** y el agente **Actualizar Mapa Local** elaboran la representación del entorno local como lista de obstáculos y un mapa local de rejilla, respectivamente.

El agente **Recorrer Campo**, gestiona y controla la ejecución de la navegación de laboreo. Tres son los objetivos en la navegación de laboreo: (1) Recorrer líneas rectas, (2) Seguir arcos de circunferencia y (3) Mantener la seguridad del sistema. Los agentes **Seguir Recta** y **Seguir Arco de Circunferencia** se encargan de la navegación en líneas rectas y del cambio de una recta a otra adyacente, respectivamente. En todo momento, es necesario disponer de información relativa a la *Posición* del robot, que es actualizada por el agente **Actualizar Posición**, también utilizado en la navegación global. Un nuevo agente, **Actualizar Posición en Campo**, mantiene la percepción de la situación relativa del robot en el campo, es decir si está o no en una línea de cultivo (real en el caso de frutales o virtual en cosecha de cereal). Tanto el agente **Seguir Recta** como el agente **Seguir Arco de Circunferencia** garantizan la seguridad en la navegación modulando los agentes **Parar** y **Evitar Obstáculos**. El agente **Seguir Recta** reutiliza la funcionalidad del agente **Avanzar**, modulándolo convenientemente para conseguir el seguimiento preciso de las trayectorias rectilíneas. El agente **Seguir Arco de Circunferencia** requiere las capacidades de un nuevo experto, el agente **Girar** que da lugar a trayectorias semicirculares.

AGRO-AMARA permite el crecimiento gradual de las competencias del robot. Esto se consigue añadiendo agentes con nuevas habilidades o que modulen de forma distinta los agentes ya existentes. El encapsulamiento de los procesos en agentes facilita este cometido, que resulta especialmente evidente en el caso de los agentes perceptivos.

El desarrollo de la arquitectura AGRO-AMARA según el paradigma cliente-servidor, de modo similar a como se hace en Saphira, facilita la implementación de la arquitectura en diferentes plataformas. Un ejemplo de ello es el hecho de que el mismo código empleado en el robot real DÉDALO se usa en el simulador AGRO-SIM, anexo B. En el capítulo 6 se describe la arquitectura software empleada.

La arquitectura AGRO-AMARA favorece el crecimiento de las habilidades del robot, tanto

en lo referente a la integración de nuevos sensores como al desarrollo de agentes nuevos, instalación de nuevos actuadores etc...

Capítulo 6

Experimentos en navegación autónoma en entornos agrícolas

Validar, según el diccionario de la RAE, es “dar fuerza o firmeza a algo, hacerlo válido”, tomando válido como “aceptable”. Lo que se pretende con la presentación y análisis de los resultados experimentales es precisamente eso, mostrar que la arquitectura propuesta en esta tesis, denominada, AGRO-AMARA permite conseguir los objetivos de navegación propuestos, para el tipo de robot, entorno y tareas abordados.

Validar una arquitectura de control para robots móviles es una tarea ardua, fundamentalmente porque tanto el robot como el entorno en el que se mueve son sistemas complejos cuyo comportamiento no es completamente predecible. Sin embargo la única manera de demostrar que una arquitectura de control es viable, consiste es implementarla en un robot real [Vázquez, 2002]. Validar la arquitectura con un robot real concreto y en un entorno especificado, demuestra que en esas condiciones es viable, pero no prueba que sea la mejor

arquitectura en cualquier otro entorno, con cualquier otro robot y cualquier otra tarea. Para asegurar que una arquitectura es la mejor en un determinado aspecto, habría que implementarla y demostrarla en plataformas muy diferentes, en entornos diversos y haciendo frente a un amplio abanico de tareas. Y posteriormente comparar esos resultados con aquellos obtenidos de aplicar en idénticas condiciones otras arquitecturas.

En este capítulo se pretende demostrar que la arquitectura y automatización propuestas constituyen una solución viable y eficaz para un tipo de condiciones robot-tarea-entorno propias de la agricultura. Para ello en la sección 6.1 se discuten algunos aspectos importantes de la implementación de la arquitectura AGRO-AMARA. Con robots de dimensiones medias a grandes como son los que se utilizan en agricultura, la experimentación es ardua, requiere dedicar mucho tiempo y se depende de factores externos entre los que se encuentran las condiciones meteorológicas. Por ello es de gran utilidad una etapa previa de depuración de estrategias en un simulador, descrito brevemente en este capítulo (una descripción más detallada se encuentra en el anexo B). A continuación, se detallan algunos aspectos importantes de la implementación de los agentes. Las secciones 6.2 y 6.3 describen parte de los experimentos concretos de navegación. La sección 6.2 se centra en la navegación global y la sección 6.3 en la navegación específica de laboreo. En ambos casos se presentan en primer lugar los resultados con el sistema completo, desarrollando a continuación, con más detalle, los aspectos más relevantes de los agentes.

6.1 Aspectos relevantes de la implementación de la arquitectura AGRO-AMARA

Como ya se comentó en el capítulo 4, el sistema está desarrollado siguiendo el paradigma de cliente-servidor. El servidor es el programa que se ejecuta en el procesador a bordo del robot y realiza el ciclo de control directo sobre los controladores de bajo nivel (dirección, freno y embrague), adquiere las señales de los sensores y envía datos a los clientes que los solicitan, recibiendo de ellos las consignas de control para la realización de una acción física por parte

de los actuadores. Dentro de este paradigma los procesos integrados en el modelo de agente, encargados de la navegación global y de la navegación de laboreo, son programas cliente que se conectan al servidor para solicitar los datos sensoriales y enviar las consignas de control adecuadas. Si se quiere desarrollar un simulador dentro de este esquema de manera que se pueda reutilizar el mismo código en el simulador y en un agente en el robot real, el simulador tendrá que actuar como un programa servidor, donde los datos, en lugar de proceder del robot proceden del simulador. Ciertos detalles de la implementación del simulador y de los programas cliente resultan fundamentales para entender cómo se ha materializado la arquitectura y cómo se obtienen los resultados experimentales, pues es precisamente la arquitectura quien guía el desarrollo de todos los procesos. A continuación se explican los aspectos que se han considerado más relevantes de la implementación del simulador y los distintos procesos de navegación global y de laboreo.

6.1.1 Simulador de entornos y vehículos dotados de sensores

Los simuladores constituyen una herramienta de gran utilidad en el diseño, experimentación y optimización de todo tipo de sistemas, pero son especialmente necesarios en aquellos ámbitos en los que las pruebas a realizar son costosas, difíciles o peligrosas [Sukthankar et al., 1996]. En el caso de los robots móviles el uso de simuladores se halla muy extendido, y la mayoría de los robots comerciales disponen de una herramienta propia de simulación. Sin embargo la utilización de simuladores sigue siendo objeto de controversia. Una de las críticas más extendida apunta al hecho de que los simuladores no consideran la problemática asociada a un robot real. Así, muchos simuladores asumen una percepción mucho más rica, precisa y completa que la que realmente el robot posee a través de su dotación sensorial. Por el contrario, los simuladores pueden plantear problemas que no existen en el mundo real, por ejemplo complicados cálculos geométricos en la búsqueda de vértices al simular un sensor láser. Otro argumento crítico con los simuladores se debe a que propician descomposiciones falsas de los problemas de control [Torrance, 1992].

Ahora bien, las ventajas de una utilización correcta de simuladores en el desarrollo e investigación de estrategias de percepción y navegación con robots móviles son múltiples, siempre que se tengan en cuenta sus limitaciones. Los simuladores se utilizan como banco de pruebas inicial, y permiten realizar una primera depuración de los algoritmos y estrategias que van a ser posteriormente implementadas en el robot real. Este hecho es de especial relevancia cuando se trata de pruebas que son:

- peligrosas por realizarse a gran velocidad o bien en entornos con presencia no prevista de operarios ,
- costosas, desde el punto de vista económico y de tiempo, como en el caso de robots submarinos, aéreos o terrestres

Los simuladores permiten enfrentarse a experimentos que dependen de configuraciones del entorno que no son muy comunes o que corresponden a fenómenos transitorios de corta duración: sería extremadamente tedioso esperar a que ocurriera cierto evento poco probable para poder realizar una prueba de comportamiento del robot [Sukthankar et al., 1996]. En la experiencia cotidiana de trabajo con robots, muchos de los problemas que se presentan en la experimentación no tienen tanto que ver con la investigación propiamente dicha, sino con el mal funcionamiento o fallos en piezas del robot (baterías descargadas, piezas rotas, holguras, perdida de señal en las comunicaciones). Por todo ello es muy útil disponer de un buen simulador que facilite el planteamiento y la resolución, a nivel virtual, de situaciones poco frecuentes y estrategias en versiones iniciales [Torrance, 1992].

Ahora bien, la utilidad de los simuladores para la investigación en robótica móvil depende fundamentalmente de cómo se plantee su implementación. Por ello, conviene tener muy claro cuales van a ser las limitaciones del mismo: (1) tipo de modelos que integra, (2) propiedades modeladas u obviadas y (3) aspectos críticos, donde pueden darse grandes diferencias entre los experimentos simulados y los reales. Algunas de las características que debe poseer un simulador para que sea útil y reutilizable, son:

- modular, pues facilita la integración de nuevos módulos

- escalable a otras tareas o entornos
- amigable o fácil de utilizar, de modo que no sea más difícil realizar pruebas con el simulador que con el robot real
- flexible y modulable, para introducir cambios en las variables de los módulos que lo componen.

Un aspecto esencial es la organización global de la programación en el simulador, de forma que el código tanto de los algoritmos como de las estrategias de control implementadas en el simulador puedan ser portadas con modificaciones mínimas, como procesos del robot real. La mayoría de los simuladores existentes y documentados que pretenden servir como soporte para el trabajo con robots reales cumplen muchas de estas características [Vaughan, 2001, Sukthankar et al., 1996]

Los simuladores son esenciales en las primeras etapas de desarrollo, pues permiten identificar y corregir errores en el diseño del sistema. Otra de sus propiedades relevantes de los mismos la constituye el hecho de que permiten el desarrollo de entornos virtuales que interactúan con el sistema. Esta interacción facilita el aprendizaje de estrategias y posibilita una experimentación amplia en el espacio de estados, difícil de lograr en un plazo breve de tiempo en el mundo real. El simulador reproduce el modo de operación de un sistema real. Por esta razón, en vez de construir modelos matemáticos extensos que requieren un conocimiento experto muy elaborado, el simulador permite modelar y analizar el comportamiento de un sistema real a aquellos operarios no-expertos en el modelo ni en la programación, pero que sin embargo son capaces de manejar y controlar el sistema en el mundo real.

En resumen, en el caso de los robots de exteriores la experimentación es más dura que con robots de interiores, de tamaño reducido y holonómicos. Los vehículos de exteriores, poseen mayores dimensiones, una mecánica compleja, un entorno poco estructurado y sin marcas, condiciones meteorológicas adversas y con una navegación en zonas alejadas de los laboratorios (donde se encuentran las herramientas de ajuste y verificación). En respuesta a esta problemática y una vez realizadas las primeras pruebas en el mundo real, surge la necesidad de disponer de un

simulador visual virtual tanto de los entornos de trabajo del vehículo como de las capacidades sensoriales y de actuación de éste, a fin de desarrollar y optimizar las estrategias de percepción y navegación requeridas para el cumplimiento de una determinada misión.

El simulador AGRO-SIM, ha sido imaginado y diseñado para reproducir el funcionamiento de robots móviles en entornos exteriores, en concreto del tractor DÉDALO en un entorno agrícola, integrando en él las prestaciones del programa DÉDALO-Servidor (descrito en el capítulo 4), residente en el procesador a bordo del robot DÉDALO. El simulador AGRO-SIM ha sido utilizado también con el tractor cortacésped ROJO en entornos y tareas de jardinería [García-Pérez y García-Alegre, 2001].

AGRO-SIM consta de tres módulos bien diferenciados: 1) el modelo cinemático del robot, 2) el modelo del entorno y 3) los modelos de los diferentes sensores. A continuación se comentan brevemente los aspectos más significativos de cada uno de estos módulos.

Modelo cinemático del robot.

El modelo implementado para emular el movimiento del robot es un modelo cinemático tipo coche, descrito con detalle en el anexo B . Este modelo cinemático sustituye y engloba a los agentes de más bajo nivel del robot DÉDALO: *Gira Volante* y *Pisa Pedal*.

El modelo cinemático se sintetiza en un conjunto de ecuaciones que relacionan la posición del vehículo (x_R, y_R, θ_R) con la velocidad del robot v_R, su longitud L y el ángulo de giro de las ruedas α_v. Las coordenadas del robot (x_R, y_R, θ_R) se expresan en un sistema de coordenadas con origen en la posición inicial de movimiento y orientación igual a la orientación inicial del robot.

Modelo del entorno

Un punto fundamental en el diseño de un simulador para un robot móvil es el modelo del entorno en el que el robot debe moverse para realizar las tareas. Sobre el modelo del entorno, en todos los simuladores, se realizan grandes simplificaciones, reduciéndolo a describir aquellas

características del entorno que se consideran fundamentales para la realización de su misión.

En la versión actual de este modelo sólo se han incluido dos propiedades del entorno real: altitudes del sensor de inclinación y objetos tanto estáticos como dinámicos. La altitud del terreno se modela mediante un conjunto de polígonos que definen líneas de nivel. Los objetos se modelan como polígonos con un número arbitrario de lados, con un movimiento definido por su centro de masas. La simulación de estos obstáculos imprevistos ya sean estáticos o móviles es importante, pues contribuye a proporcionar un mayor grado de realidad al mundo simulado. La descripción detallada del modelo del entorno se describe en detalle en el anexo B.

Los sensores

Exceptuando los sensores de posición para medir el ángulo de la dirección y el estado de los pedales de freno y embrague, el simulador incluye todos los sensores que se han integrado a bordo del tractor: sensor de nivel de batería, odometría, DGPS, brújula, sensor de choque, láser de barrido y sensores de inclinación. Los sensores virtuales del simulador, se han modelado con el objetivo de reproducir una señal sensorial lo más parecida posible a la proporcionada por el sensor real. En el caso del sensor odométrico y del láser de barrido, esto se ha conseguido modelando el comportamiento del sensor. Para el resto de los sensores se ha diseñado un algoritmo que devuelve una señal sensorial similar a la del sensor real. Se han tenido en cuenta las limitaciones de los modelos, a fin de pesar el grado de verdad de las conclusiones que se pueden extraer en simulación para una posterior extrapolación a un mundo real. En el anexo B se describen con detalle cada uno de los sensores virtuales implementados.

- **Sensor de nivel de batería.** Bajo las hipótesis de consumo constante y período de trabajo aproximado de 4 horas, se ha modelado el nivel de la batería como una función lineal decreciente en el tiempo.

- **Odometría.** El odómetro virtual implementado en el simulador AGRO-SIM, es un odómetro ideal, en el sentido de que la única fuente de imprecisión que incorpora respecto

a la posición real del robot es la resolución finita del sensor. Esta resolución, la establece el usuario en tiempo de ejecución del programa.

- **Brújula.** El valor de la orientación absoluta medida por la brújula se modela como el valor de la orientación obtenido del modelo cinemático del robot más un factor de ruido aleatorio, determinado por un valor de precisión y de resolución. El usuario puede definir tanto la precisión como la resolución de la brújula, en la ventana de configuración del sensor virtual en el interfaz del simulador con el investigador o desarrollador.

- **DGPS.** Con el simulador se ha tratado de generar una señal virtual que en cierto modo englobe la problemática y los errores de la señal proveniente de un GPS, sin necesidad de simular el modo de operación del sensor. Para ello se han clasificado los errores en tres tipos: 1) errores habituales, 2) esporádicos, y 3) de pérdida total de la señal.

- **Sensor de contacto.** En el simulador, el sensor de contacto se modela como una prolongación física del tractor capaz de detectar si limita o no con alguna región ocupada del espacio virtual. El modelo del sensor de choques es una línea recta frontal, perpendicular a la dirección de desplazamiento del tractor. Se verifica si cada punto de la línea se halla o no contenido en alguno de los obstáculos pertenecientes al mundo virtual. Si su intersección con los mismos es distinta de cero, el sensor de choque envía una señal de activación.

- **Sensor láser de barrido.** El modelo formulado para el sensor virtual láser en el simulador emula el principio de funcionamiento del sensor real. Para cada posición del sensor se definen las direcciones del barrido a través de 361 semirrectas orientadas, que parten del robot (x_R, y_R, θ_R) con una resolución angular de $0{,}5(grados)$, figura 6.1. Cada una de estas 361 semirrectas se discretiza, con incrementos de distancia de $10(cm.)$, y se recorre la misma comprobando si los puntos discretos tienen o no intersección con puntos de algún obstáculo del entorno simulado.

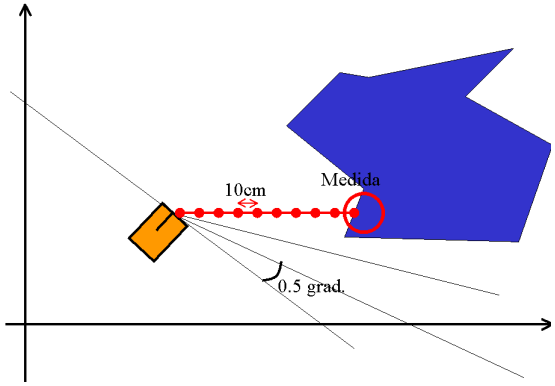


Figura 6.1 : Modelo del sensor láser virtual

- **Sensores de inclinación.** Para detectar las pendientes lateral y frontal se utilizan los valores de las alturas de cuatro puntos del robot, frontal, trasero, lateral derecho y lateral izquierdo. La diferencia entre las alturas de los puntos frontal y trasero determinará la pendiente frontal y la diferencia entre los laterales izquierdo y derecho la lateral.

Otras prestaciones del simulador AGRO-SIM

El simulador ofrece también otro tipo de funciones, especialmente pensadas para ayudar al desarrollo de estrategias de control. Estas funciones van dirigidas fundamentalmente a la oferta de un conjunto de opciones de almacenamiento y visualización de datos, anexo B: 1) visualización de los datos de los distintos sensores, 2) grabación de diferentes tipos de ficheros de datos y gráficos, y 3) actuación del simulador como un servidor equivalente al diseñado para los robots reales, capítulo 4.

En resumen, AGRO-SIM constituye una herramienta de gran utilidad en el diseño de estrategias de navegación para robots en entornos agrícolas. La característica más importante es que el código de dichas estrategias no varía, independientemente de que se ejecute en el simulador o en el robot real. Otra característica fundamental es la incorporación de obstáculos móviles que contribuyen a facilitar el diseño de las estrategias de navegación óptimas al disponer de una previsión o capacidad de predicción del comportamiento de los obstáculos

móviles en función de su historia.

6.1.2 Implementación de los programas cliente de navegación

Las dos aplicaciones de navegación desarrolladas: 1) navegación global, y 2) navegación de laboreo, se han implementado mediante un programa cliente completo e independiente que proporciona la interfaz con el usuario. Cada una de estas aplicaciones se conecta con el servidor, del robot real o del simulador, recibe los datos sensoriales que solicita y envía, a su vez, las consignas de control. Los clientes de navegación almacenan además todos los datos generados en cada sesión: datos sensoriales recibidos, consignas de control enviadas, activaciones y desactivaciones de los agentes, sus tiempos de ejecución, etc. Todos estos datos, registrados en ficheros, permiten un análisis posterior y representación en el laboratorio, a fin de detectar incidencias y ensayar algoritmos.

Un aspecto fundamental lo constituye la implementación de los agentes, pues cada agente se ha implementado como una hebra de ejecución distinta. Las percepciones y variables globales, como los valores de estado de los agentes, se almacenan en zonas de memoria compartidas (pizarra) por todas las hebras. La ejecución de las hebras puede iniciarse y detenerse en cualquier instante. Puesto que las hebras acceden independientemente a las variables compartidas hay que proteger esas variables mediante secciones críticas para evitar que se produzcan conflictos en el acceso a memoria.

6.2 Experimentos realizados en tareas de navegación global

El objetivo de la navegación global es el siguiente: dado un mapa digital geo-referenciado del entorno con unas zonas etiquetadas como prohibidas al paso de vehículos y dado un destino expresado como (x_D, y_D, θ_D), se trata de conseguir que el robot alcance el punto destino final sin colisionar con obstáculos, siguiendo una trayectoria óptima.

La aplicación visual responsable, recibe el nombre de Navegador. En la etapa actual,

esta aplicación, cuya ventana principal aparece en la figura 6.2, permite al usuario introducir manualmente la posición destino final con una orientación (directamente con el ratón o bien mediante un mensaje enviado desde un programa de visualización y representación de trayectorias), [Ribeiro et al., 2003]. Una vez establecida la conexión con el programa servidor del robot y determinados por el usuario el destino final y el mapa de zonas prohibidas de paso, el programa inicia la ejecución del agente **Ir a Punto** que junto con el agente **Planificar Caminos** inician la secuencia de procesos y verificaciones propia de la navegación global.

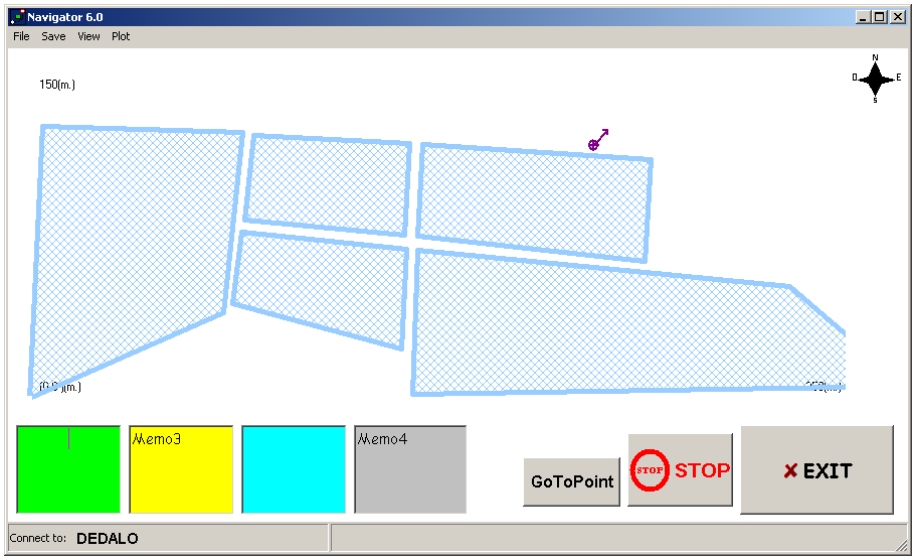


Figura 6.2 : Ventana principal de la aplicación Navegador para la navegación global

Experimentos en simulación

Las figuras 6.3 y 6.4 muestran una ruta autónoma en simulación. La gráfica 6.3 es una captura de pantalla del programa navegador y la 6.4 una captura del simulador. Se pueden observar la simulación de los errores en la posición del robot simulado comparando la trayectoria del navegador con la de la ventana del simulador que muestra la ruta de las posiciones calculadas usando el modelo cinemático.

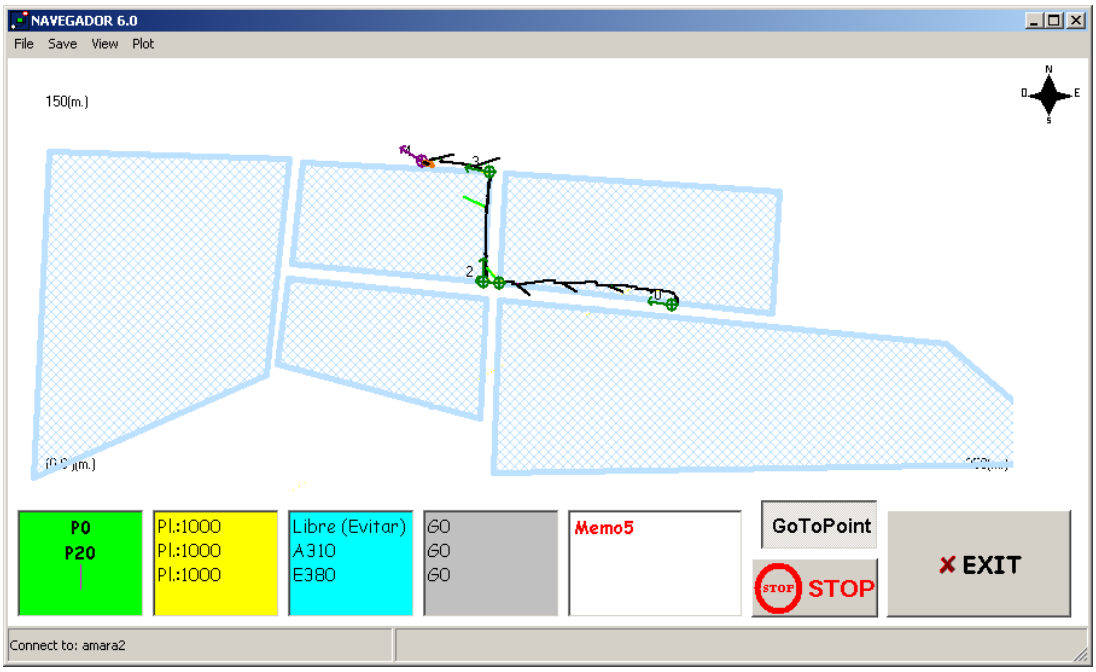


Figura 6.3 : Recorrido autónomo por la zona de pistas del IAI-CSIC. Interfaz del Navegador

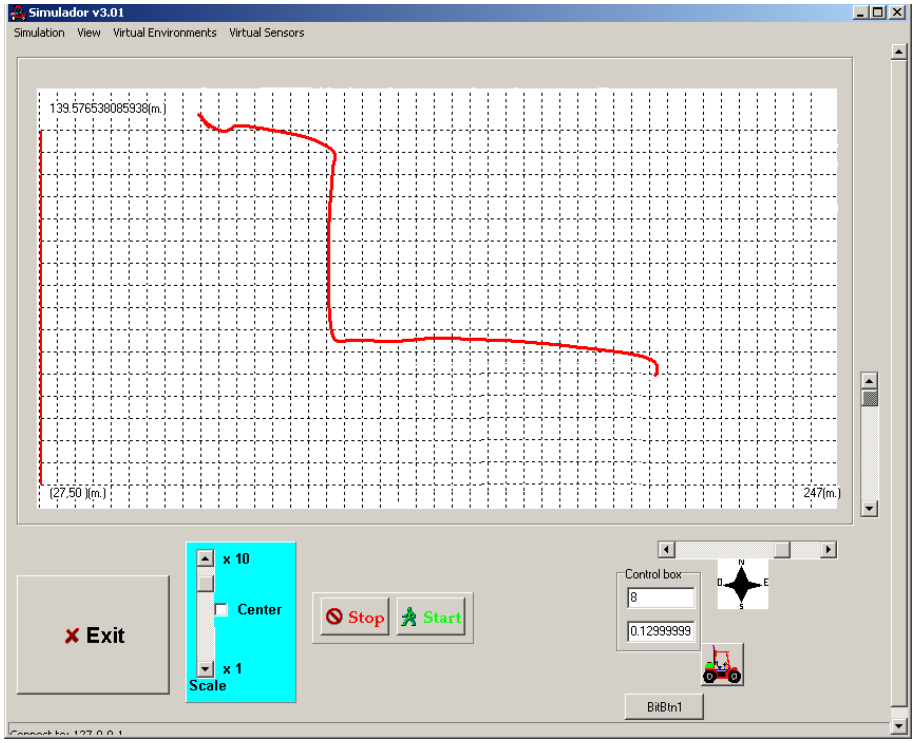


Figura 6.4 : Recorrido autónomo en simulación. Interfaz del simulador

Experimentos con el robot real DÉDALO

En los experimentos con el robot DÉDALO descritos a continuación se muestran diferentes situaciones y recorridos realizados en modo autónomo en la zona de pistas del IAI-CSIC. El recorrido de la figura 6.5 corresponde a un trayecto autónomo de corta longitud finalizado con éxito. La figura 6.5 es una captura de pantalla del programa Navegador que muestra el plano de las pistas del IAI-CSIC con las zonas prohibidas marcadas en rejilla azul. La trayectoria del robot está pintada en color negro, correspondiente al modo de posicionamiento 0 (DGPS y brújula).

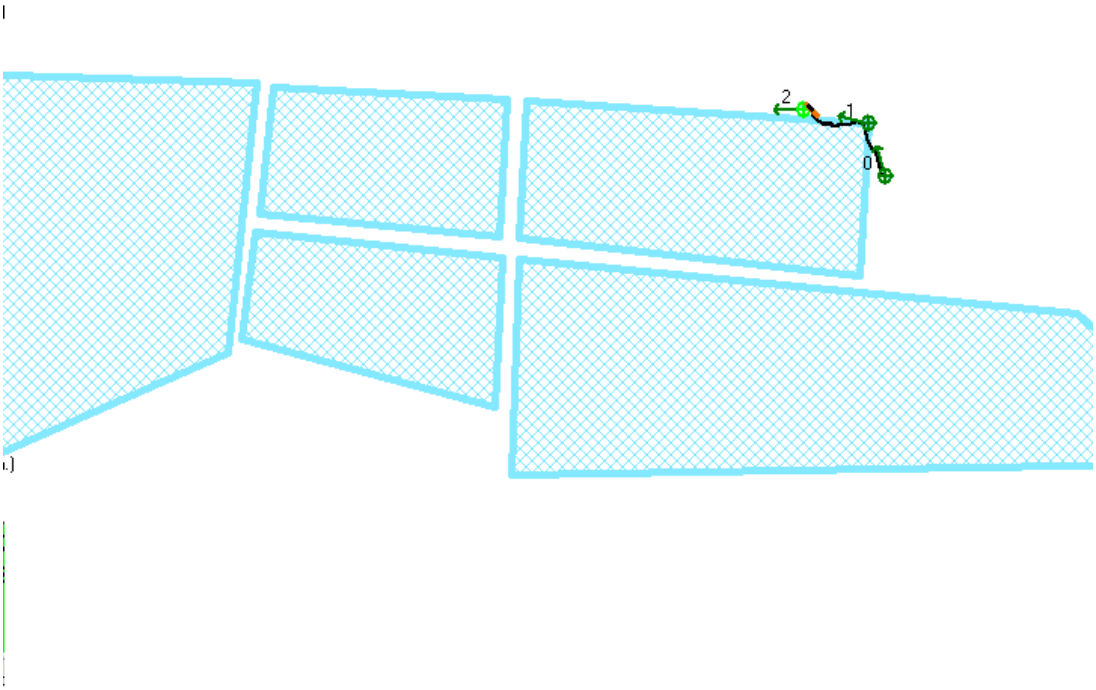


Figura 6.5 : Recorrido autónomo por las pistas

La trayectoria que aparece en la ventana del Navegador de la figura 6.6 muestra otro recorrido autónomo realizado por el robot DÉDALO en las pistas del IAI-CSIC. El plan propuesto por el agente **Planificar Caminos** para esta ruta consta de dos puntos intermedios, representados por círculos con flechas en verde. Puesto que las zonas prohibidas no corresponden con obstáculos, el robot puede salirse de las pistas y adentrarse en dichas zonas,

tal y como hace en el segundo tramo del recorrido, entre los puntos 2 y final. En esta zona el recorrido tiene oscilaciones, principalmente debido a que el terreno, tierra arada, dificulta la trayectoria perfectamente rectilínea. El robot llega a su destino final con un error en posición de 0,7 (m) y de orientación de 48 (grad.).

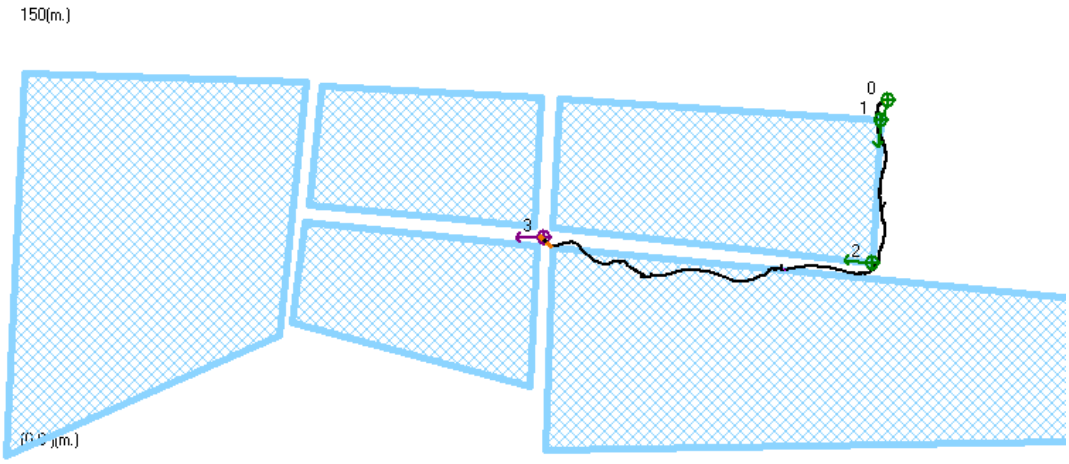


Figura 6.6 : Recorrido autónomo del robot DÉDALO por las pistas del IAI-CSIC

La figura 6.7 muestra un recorrido donde el robot se ha desviado debido a la presencia de un obstáculo en su camino. Se aprecia con claridad que el robot se desvía de la línea recta hacia la derecha al principio del recorrido para retomar finalmente la ruta recta hacia el objetivo final al que llega con un error de 0,8 (m) en posición y 3 (grad.) en orientación.

La figura 6.8 muestra un recorrido con obstáculos que ha terminado con una parada de seguridad ante un obstáculo muy próximo. El primero de los obstáculos fuerza al robot a evitarlo saliéndose de las pistas y navegando campo a través, puesto que en esa zona la inclinación del terreno y los obstáculos lo permiten y el plan es únicamente una “sugerencia” para el mejor trayecto. Sin embargo durante la navegación campo a través encuentra un obstáculo muy cercano que dispara el comportamiento **Evitar Obstáculo** deteniendo el robot. En la figura 6.8 puede verse el mapa local correspondiente al entorno del robot donde se observan celdillas ocupadas muy próximas a la posición del robot. También se presenta una captura de pantalla con la información de control, donde marca como activo el agente **Evitar Obstáculos** y valores

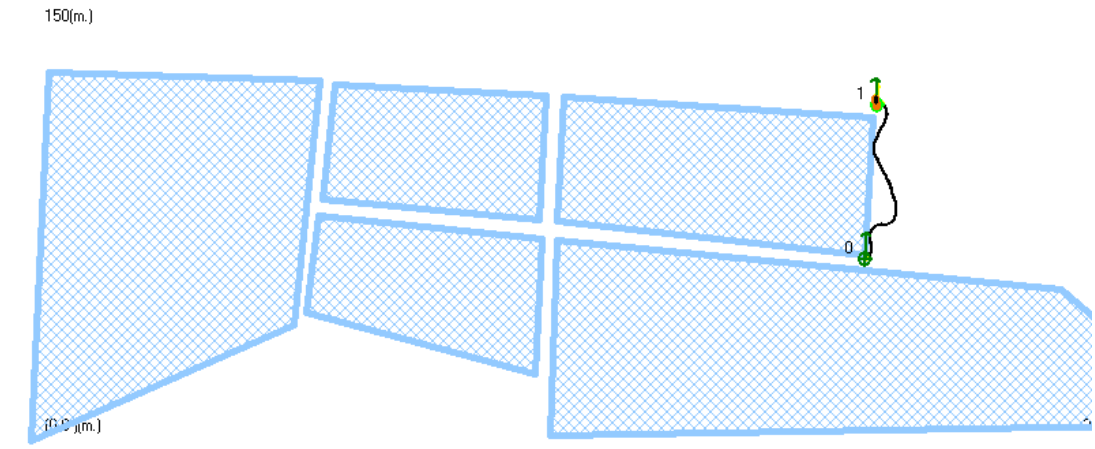


Figura 6.7 : Recorrido autónomo del robot DÉDALO por las pistas del IAI-CSIC. Actuación del agente **Evitar Obstáculos**

de salida del ángulo de volante 0 y el freno activado, como corresponde a la parada de seguridad.

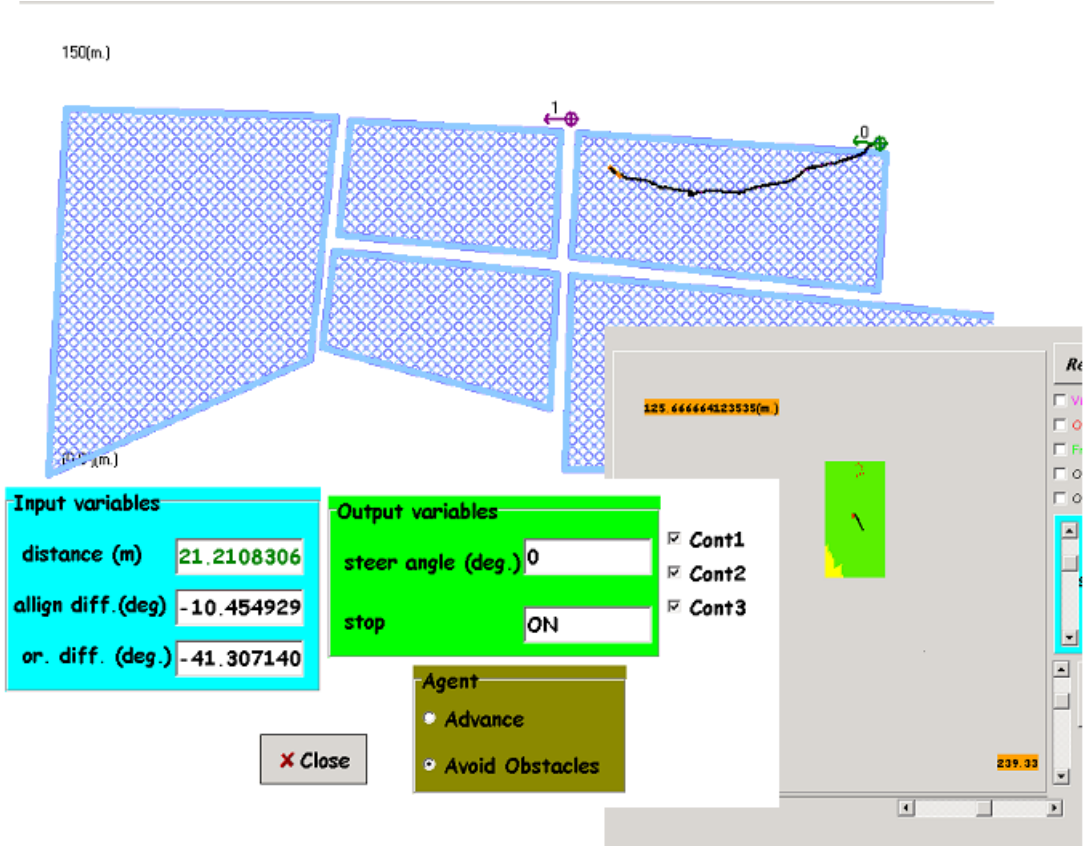


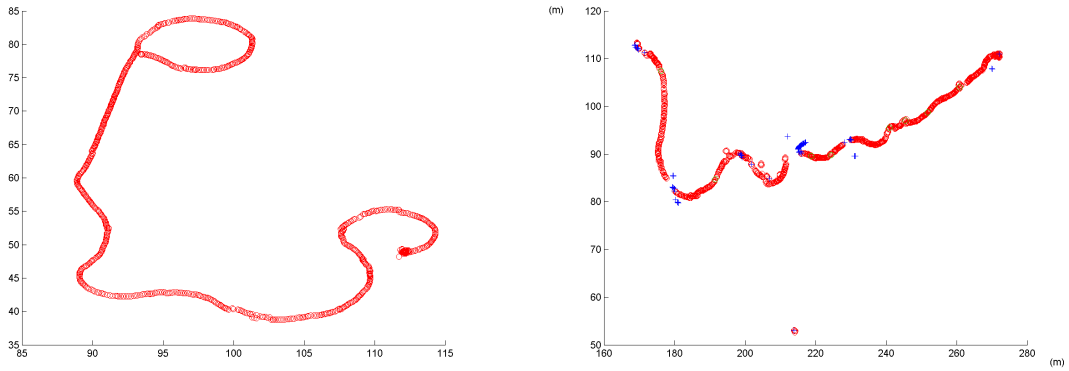
Figura 6.8 : Recorrido autónomo con parada de seguridad. Actuación del agente **Evitar Obstáculos**

6.2.1 Experimentación con el agente Actualizar Posición

En las pruebas realizadas para demostrar el funcionamiento de la arquitectura AGRO-AMARA en navegación global, apenas se aprecia el modo correcto de operar del agente **Actualizar Posición**. Por ello, en esta sección se exponen los resultados del funcionamiento de este agente en todos sus modos, describiendo los resultados que se han obtenido en diferentes pruebas. El objetivo fundamental es mostrar la robustez del agente frente a fallos en los diferentes sensores. En primer lugar están los resultados del agente, en modo de navegación autónoma, en una ruta real campo a través. Como se puede observar en la figura 6.9 (a) las posiciones calculadas por el agente generan una ruta continua que se corresponde con la ruta real efectuada. En este primer ejemplo, a pesar de tratarse de una ruta larga, unos 100 (m), el modo de posicionamiento ha sido el modo 0 correspondiente a utilizar el receptor DGPS y la orientación de la brújula.

Otro ejemplo de ruta real se muestra en la figura 6.9 (b) en la que aparecen tramos cortos donde se ha perdido la corrección diferencial del GPS, al pasar cerca de árboles. En esta gráfica además puede verse cómo el agente se recupera de pequeños errores, como los puntos que aparecen claramente fuera del camino en posiciones próximas a la $(215, 52)$. Los círculos rojos indican las posiciones calculadas con GPS y brújula, las cruces azules las posiciones calculadas con odometría y brújula. Los círculos verdes marcan las posiciones obtenidas únicamente con el GPS y las cruces negras las calculadas empleando sólo la odometría.

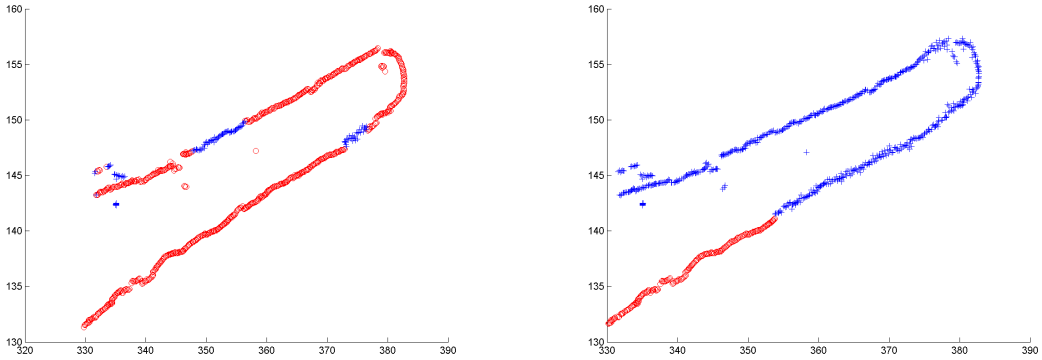
Sin embargo y afortunadamente los fallos en el GPS no son muy frecuentes y menos lo son aún los fallos en la brújula. Por este motivo y con el fin de mostrar el comportamiento del agente en todos los casos se han realizado una serie de experimentos en los cuales se han provocado fallos en los diferentes sensores. En primer lugar se ha suprimido la señal de la corrección diferencial en el GPS en algunos tramos del recorrido del robot. Como puede observarse en las gráficas de la figura 6.10 el posicionamiento es correcto, tanto en los tramos rectilíneos como en los tramos curvos. Incluso si el GPS tarda en recuperar la corrección diferencial, figura 6.10 (b), la navegación es adecuada. Esto es debido a que, el error en la odometría afecta



(a) (b)

Figura 6.9 : Localización del robot en navegación autónoma campo a través. Actuación del agente **Actualizar Posición**

fundamentalmente a la orientación, y mientras ésta se estime a través de la brújula digital, el posicionamiento por la odometría no resulta demasiado impreciso.



(a) (b)

Figura 6.10 : Recorrido autónomo del tractor con fallos provocados (ausencia de corrección diferencial) en el receptor GPS

En la gráfica que se presenta en la figura 6.11, se muestra la navegación global del tractor, con una señal errónea de la brújula, también provocada. Se aprecia que mientras posiciona con el GPS la ruta parece correcta, sin embargo el valor de la orientación a partir del GPS es bastante inestable y provoca graves fallos en el control del vehículo, ya que la orientación es determinante a la hora de calcular el ángulo de giro de las ruedas para alcanzar un objetivo. La situación más desfavorable se da en el tramo final, figura 6.11, cuando también falla el GPS

y la odometría, sin la ayuda de la brújula, envía una posición afectada por un error alto, lo que justifica la parada del robot a pocos metros del inicio. En este modo de funcionamiento del agente **Actualizar Posición**, posicionando únicamente con la odometría, se permite la navegación autónoma del robot únicamente 10 (m.). Si durante el tiempo que tarda en recorrer ese espacio no recupera la señal de los otros sensores el robot se detiene como medida de precaución.

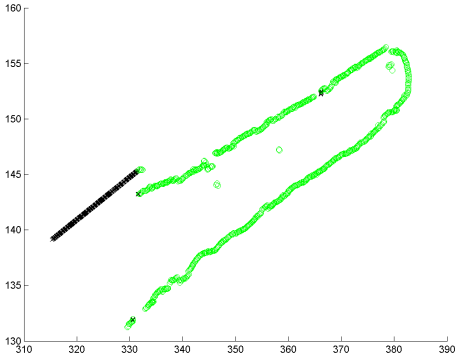


Figura 6.11 : Recorrido autónomo del robot en navegación global sin brújula

6.2.2 Experimentación con el agente Actualizar Obstáculos

Los experimentos de esta sección muestran el funcionamiento detallado del agente **Actualizar Obstáculos**. Su modo de operar está bajo la influencia directa del agente **Actualizar Mapa Local** de manera que el buen funcionamiento de este último agente está implícitamente incluido en esta sección. Los experimentos realizados tienen como objetivo validar el comportamiento de ambos agentes: la detección de obstáculos con sensor láser el agente **Actualizar Obstáculos**, y la capacidad del *Mapa Local* para incorporar información dinámica. Todos los experimentos mostrados en esta sección se han realizado con el tractor parado y algunos de ellos con el tractor comercial corta-césped ROJO, automatizado en el IAI-CSIC que dispone de la misma dotación sensorial que el tractor DÉDALO. El tractor ROJO puede considerarse en muchos aspectos el precursor del tractor DÉDALO [García-Pérez et al., 2000a].

Puesto que los experimentos se han efectuado con el tractor parado y ambos vehículos disponen de la misma dotación de sensores, se pueden analizar los resultados de los procesos integrados en estos dos agentes perceptivos, con independencia de la plataforma donde se efectuaron.

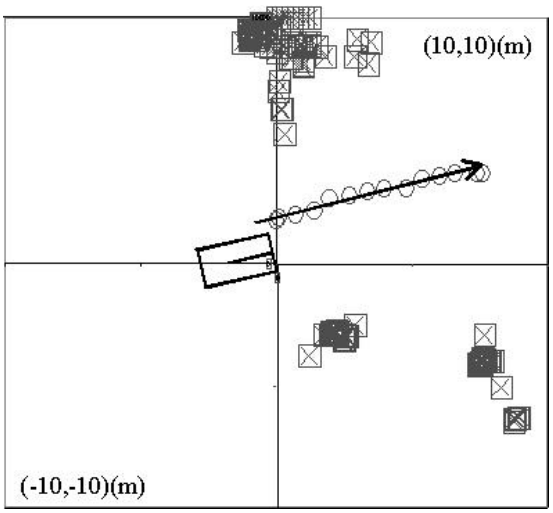
Los experimentos se han dividido en dos partes: en primer lugar aquellos experimentos donde se muestra el funcionamiento del agente en el seguimiento del tipo de obstáculos móviles más comunes en el entorno agrícola, los posibles operarios. En segundo lugar, se agrupan los experimentos realizados para comprobar los límites del agente en el cálculo de la velocidad de los obstáculos y en el grado de fiabilidad de los resultados.

Detección y seguimiento de obstáculos móviles

En esta experimentación se solicitó a los distintos peatones que caminasen, en cada experimento, siguiendo una trayectoria rectilínea a velocidad constante, en el intervalo de $0{,}15$ (m/s) a $0{,}7$ (m/s). En la figura 6.12 se muestra el seguimiento de una persona que se mueve a una velocidad aproximada de $0{,}25$ (m/s). En la columna de la izquierda se muestran las distintas posiciones del peatón calculadas mediante el algoritmo de detección de obstáculos móviles y que se han representado mediante círculos en un sistema de coordenadas centrado en el robot (superpuesto en el gráfico para mayor claridad). En la misma figura se representan los objetos estáticos presentes en la escena natural (arbustos y paredes) mediante cruces. Cada pareja color-símbolo corresponde a una etiqueta asignada por el algoritmo, es decir, significa un obstáculo distinto. Además para mayor claridad se ha resaltado la trayectoria seguida por el peatón mediante una flecha. En la columna de la derecha se muestran las imágenes correspondientes al instante inicial de la ejecución de cada experimento[1].

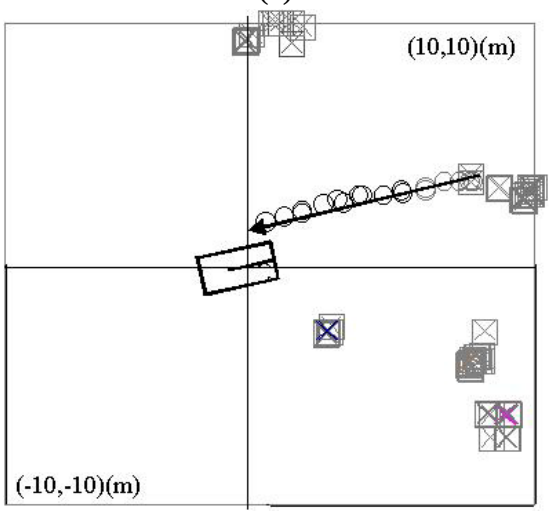
En los tres experimentos de la figura 6.12 se puede observar que las posiciones teóricas calculadas por el agente **Actualizar Obstáculos** se mantienen en la trayectoria rectilínea descrita por cada peatón en cada experimento. Aunque improbable, es posible que el robot se encuentre en escenarios donde haya más de un peatón. Los dos experimentos que se presentan a continuación, figuras 6.13 y 6.14, muestran el seguimiento simultáneo de dos peatones en

[1]Experimentos realizados con el robot corta-césped ROJO



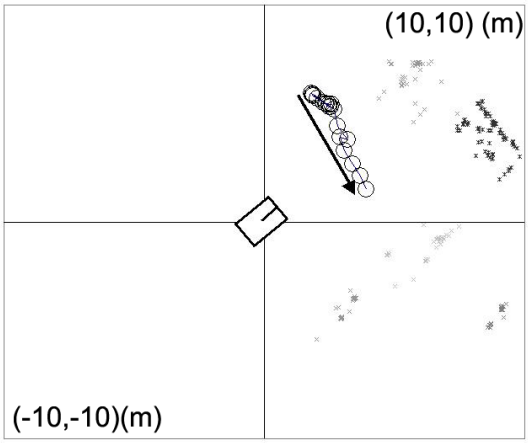
(a)

(b)



(c)

(d)

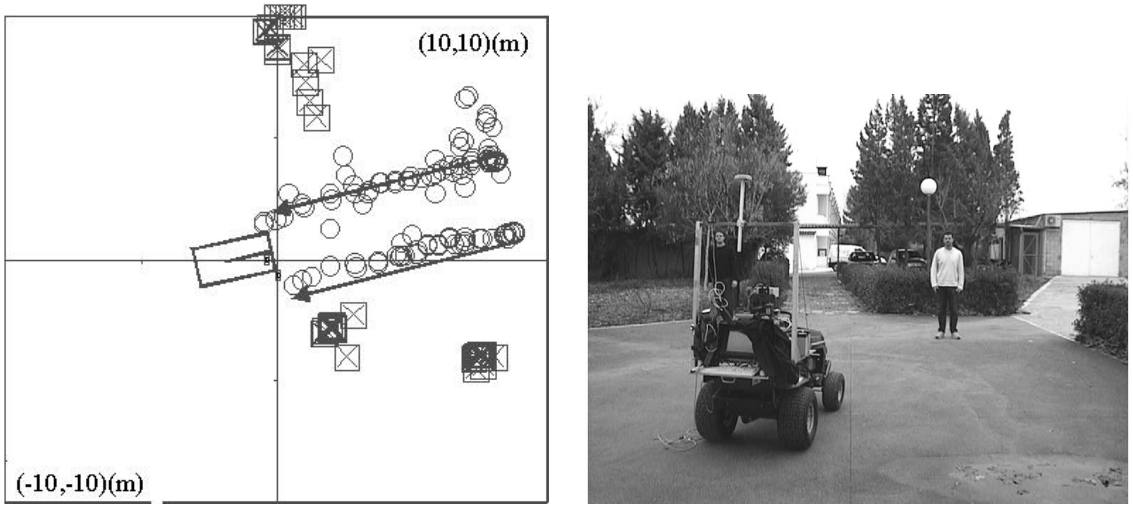


(e)

(f)

Figura 6.12 : Resultados del seguimiento de *Obstáculos*: un único peatón. La columna de la izquierda muestra las posiciones estimadas del obstáculo móvil en un mapa de 20×20 (m.) centrado en el robot. La columna de la derecha muestra la fotografía del instante inicial de cada experimento

movimiento, siguiendo trayectorias paralelas que se aproximan al vehículo en el caso de la figura 6.13 y trayectorias que intersecan en el caso de la figura 6.14.

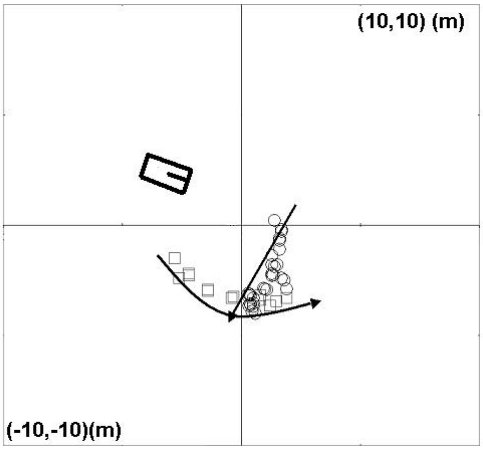


(a) (b)

Figura 6.13 : Detección y seguimiento simultáneo de dos *Obstáculos*: peatones caminando en rutas paralelas hacia el robot

En el experimento de la figura 6.13 también se aprecia cómo las posiciones de los obstáculos calculadas por el agente **Actualizar Obstáculos** se mantienen sobre la trayectoria rectilínea que han descrito los peatones. La mayor densidad de posiciones que aparece en torno a la localización inicial, se debe a que los peatones permanecieron inicialmente parados unos segundos, antes de comenzar el experimento.

El experimento de la figura 6.14 muestra una situación más compleja que el proceso implementado en el agente **Actualizar Obstáculos** ha resuelto favorablemente. El experimento es más complejo puesto que al cruzarse las trayectorias de los peatones existe riesgo de que el agente **Actualizar Obstáculos** identifique incorrectamente a cada uno de los peatones, puesto que uno ocluye al otro durante los breves instantes del cruce. El peatón identificado por las marcas cuadradas camina al doble de velocidad, $0{,}5$ (m/s), que el peatón identificado por marcas circulares. El agente **Actualizar Obstáculos** es capaz de discernir correctamente entre las trayectorias de los dos peatones, gracias a que ninguno de ellos cambia abruptamente su dirección de movimiento. Si alguno de ellos hubiera efectuado un cambio brusco en el curso de



(a)

(b)

Figura 6.14 : Detección y seguimiento de dos *Obstáculos*: peatones cuyas trayectorias intersectan frente al vehículo

su trayectoria, el agente hubiera sido incapaz de efectuar la identificación correctamente, puesto que ésta se basa en la predicción de la posición del objeto calculada mediante la velocidad estimada en el instante anterior. Ahora bien, mientras los cambios de velocidad o de dirección de los obstáculos son suaves la identificación se realiza correctamente.

Resultados del algoritmo de estimación de la velocidad de los obstáculos implementado en el agente Actualizar Obstáculos

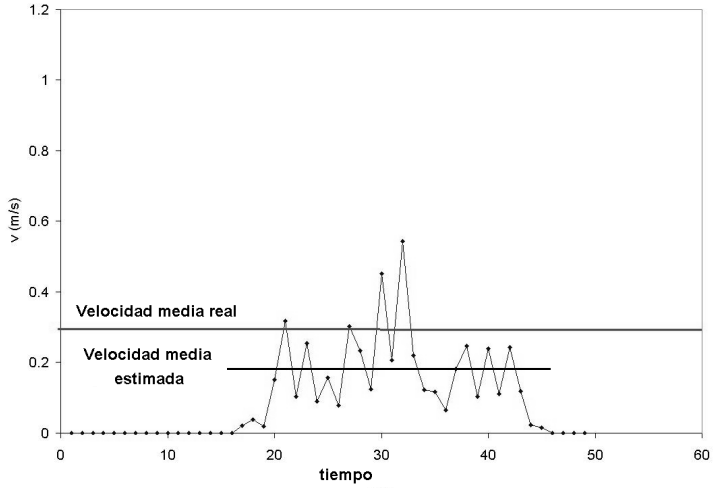
Para determinar los límites del algoritmo en el cálculo de la velocidad, se han realizado una serie de experimentos con un peatón moviéndose a velocidades en el intervalo de cero a $0{,}9$ (m/s). La velocidad “real” en estos experimentos se ha medido de forma artesanal, midiendo con un cronómetro el tiempo y la distancia recorrida por el peatón, que trata de ir a velocidad constante, con un metro. Evidentemente este valor es un valor aproximado; sin embargo esto no afecta a las conclusiones, puesto que el objetivo no es la determinación precisa de la velocidad de los objetos, sino una estimación adecuada que permita comparar los resultados y mejorar las estrategias de evitar obstáculos.

El primer experimento, presenta el cálculo de la velocidad de obstáculos estáticos, analizando los valores de velocidad calculados para uno de los arbustos del experimento descrito

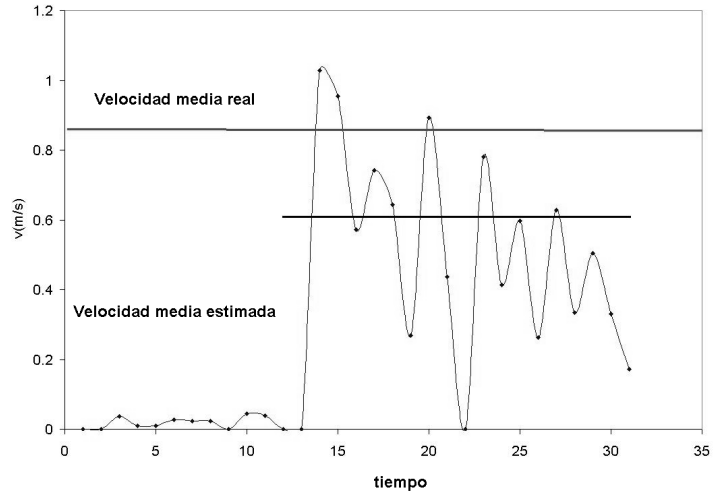
en la figura 6.12. El valor medio de la velocidad estimada del arbusto es de $v = 0{,}01$ (m/s), encontrándose que el $89\,\%$ de los valores calculados para la velocidad se hallan por debajo de $0{,}1$ (m/s), sobre un total de 45 valores calculados. Teniendo en cuenta la resolución del *Mapa Local* y la frecuencia de actualización del láser de barrido (que influye directamente en la frecuencia de actualización del *Mapa Local*), cualquier objeto con velocidad inferior a $0{,}04$ (m/s) se considerará estático. El segundo experimento analiza la velocidad de un peatón moviéndose a velocidades de $0{,}3$ y $0{,}8$ (m/s), figura 6.15. En ambas gráficas en el eje horizontal que representa el tiempo, se aprecian tres zonas claramente diferenciadas. La primera de ellas en la que la velocidad de la persona es 0 corresponde con la situación real del peatón que permanece de pie sin moverse en la posición inicial, antes de comenzar a caminar. La segunda zona corresponde a un periodo breve de aceleración, y va seguida por un tramo prolongado donde el peatón camina a velocidad constante. La zona final corresponde a la reducción de la velocidad y parada cuando el peatón se aproxima al destino final. En ambas gráficas pueden distinguirse las tres zonas, en las que se ha representado mediante línea punteada la evolución de la velocidad calculada por el agente **Actualizar Obstáculos** y el valor medio de la velocidad real calculada como la distancia recorrida dividida por el tiempo.

En la figura 6.16 se representa la trayectoria descrita por el peatón mediante flechas de longitud proporcional al módulo de la velocidad calculada por el agente **Actualizar Obstáculos** y orientación θ_{cent}.

La velocidad calculada por el agente **Actualizar Obstáculos** puede considerarse una buena estimación de la velocidad real de los obstáculos dentro de las limitaciones sensoriales y algorítmicas del sistema. Para objetos que se muevan a velocidades superiores a $1{,}5$ (m/s) aparecen problemas en la correspondencia de objetos entre dos imágenes consecutivas del *Mapa Local*. Sin embargo esta limitación no resulta ser un inconveniente demasiado grave en aplicaciones en agricultura puesto que en la mayoría de los casos tanto los vehículos como los operarios que se encuentran en el campo se mueven a velocidades bajas, por debajo de 1 (m/s), valores que se encuentran en el intervalo de detección correcta del agente.



(a)



(b)

Figura 6.15 : Valores de velocidad calculados por el agente **Actualizar Obstáculos** para un *Obstáculo*: peatón que se mueve en la realidad a $0{,}3$ (m/s), (a) y $0{,}9$ (m/s) (b)

6.2.3 Experimentación del agente Avanzar: maniobra de orientación

El proceso de control implementado en el agente **Avanzar** se ha probado intensivamente en el tractor DÉDALO, existiendo además experimentos realizados con el robot cortacésped ROJO [García-Pérez et al., 2003]. En primer lugar se realizaron experimentos en simulación para ajustar las funciones de pertenencia del controlador difuso y comprobar el funcionamiento apropiado del sistema. Posteriormente se probó el funcionamiento del controlador difuso en el robot real DÉDALO en la realización de distintos recorridos de forma autónoma.

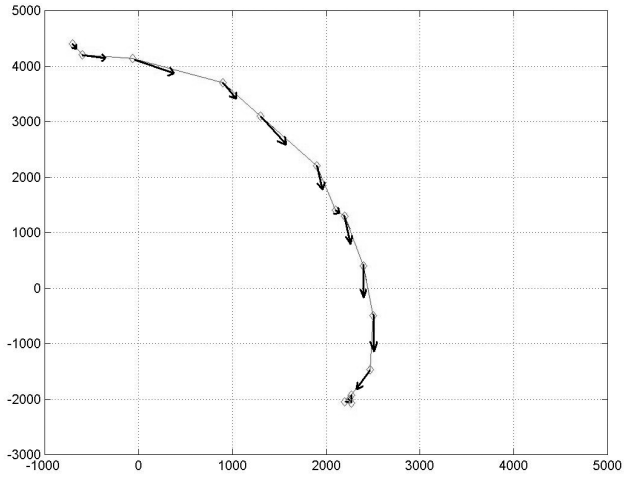


Figura 6.16 : Cálculo, hecho por el agente **Actualizar Obstáculos** de la velocidad de un *Obstáculo*: peatón a lo largo de su trayectoria. Las flechas representan la orientación del vector velocidad θ_{cent} y su longitud el módulo v_{cent}

Experimentos del agente Avanzar en simulación

Antes de iniciar los experimentos con el robot real se realizó un conjunto amplio de pruebas con el simulador. En cada experimento el usuario define una posición destino final, expresada como (x_d, y_d, θ_d). La *Posición* del robot, elaborada por el agente **Actualizar Posición**, se graba en cada iteración en un fichero para poder representarla y analizarla con mayor comodidad a posteriori. De todos los experimentos realizados, en la figura 6.17 se presentan cuatro casos con configuraciones robot-destino muy diferentes. En las gráficas, la trayectoria del robot se ha representado con diferentes símbolos a fin de diferenciar cada una de las tres zonas, definidas al diseñar el controlador difuso del agente **Avanzar**. La zona de "Aproximación" se representa con círculos azules, la de "Preparación" con asteriscos verdes y la de "Orientación" con aspas negras.

El caso (a) de la figura 6.17 constituye un ejemplo sencillo donde el robot llega al punto destino final con la orientación adecuada sin tener que realizar ningún tipo de maniobra. En la zona de "Aproximación" el robot gira para dirigirse hacia el objetivo, y en este caso al alinearse consigue orientarse adecuadamente, por lo que cuando llega a la zona de "Preparación" la

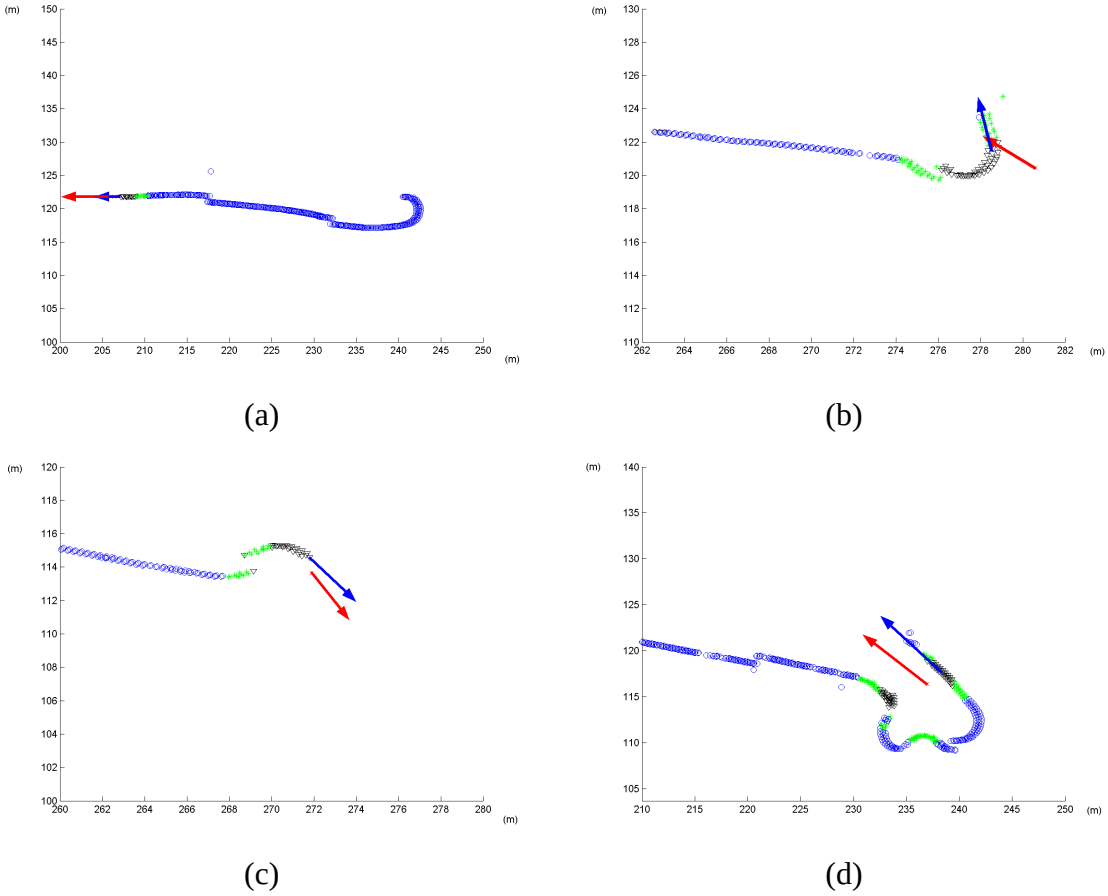


Figura 6.17 : Trayectorias del robot en simulación guiado por el agente **Avanzar**

diferencia entre la orientación del destino y la del robot es muy pequeña y el robot sigue, sin hacer maniobras. Esto mismo ocurre en la zona de “Orientación”. En los casos (b) y (d) cuando el robot entra en la zona de “Preparación” la diferencia entre la orientación del destino y la del robot es grande y el agente **Avanzar** dirige al robot alejándolo del destino. Al alejarse, permite corregir la orientación y llegar a las proximidades del destino con la orientación adecuada, como haría un conductor humano. La trayectoria mostrada en (c) muestra un caso más favorable que los dos anteriores. Al pasar de la zona de “Aproximación” a la de “Preparación” el robot se aleja del destino. De este modo en la zona final de “Orientación”, al girar para orientarse según la orientación objetivo, su trayectoria no se desvía del camino de llegada al destino.

Experimentos del agente Avanzar en el robot real DÉDALO

El agente **Avanzar** entra en operación siempre que se realice una navegación autónoma con el robot DÉDALO, de manera que su funcionamiento ha sido probado en un número muy amplio de casos. Para mayor claridad se muestran cuatro de ellos, de los cuales tres son ejemplos en los que se ha llegado al objetivo con una orientación satisfactoria. El cuarto muestra un caso de mal funcionamiento del agente **Avanzar** en el que el robot en un primer intento se acerca al objetivo incorrectamente y se vuelve a alejar para realizar de nuevo una aproximación, esta vez con éxito. Este cuarto ejemplo justifica la necesidad de detectar estas trayectorias no deseadas y evitarlas, puesto que en la mayoría de los casos, conocido el entorno, el tipo de robot y la tarea, es preferible que el robot se detenga cuando se encuentra aún alejado del destino y envíe un mensaje de PROBLEMA al operario, a que comience a dar vueltas alrededor del objetivo tratando de aproximarse cada vez más.

El ejemplo (a) de la figura 6.18 muestra un caso favorable para el robot, en el que apenas tiene que efectuar cambio alguno en su orientación para llegar al destino final, siendo 12 (grados) el error en la orientación del robot al alcanzar el destino. El ejemplo (b) muestra una parte de una trayectoria compleja, centrándola en la llegada a dos puntos objetivo intermedios en la ruta, en ambos casos se aprecia cómo el robot se aleja del objetivo en la zona de “Preparación” para que, al orientarse convenientemente hacia la orientación del destino en

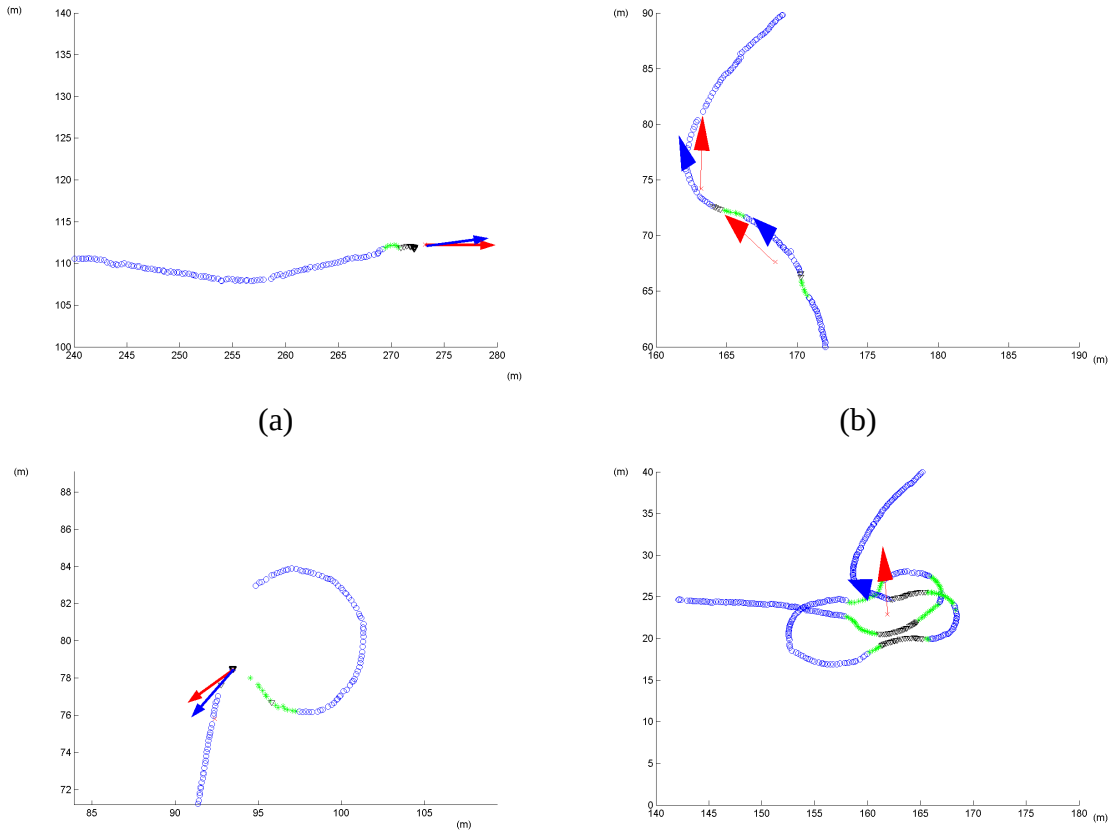


Figura 6.18 : Trayectorias reales descritas por el robot DÉDALO en el campo en navegación global autónoma

la zona de “Orientación”, no se desvíe del camino adecuado hacia el objetivo. En ambos casos conviene resaltar que al tratarse de puntos intermedios en una ruta larga se permite una tolerancia mayor que en el caso del destino final. La gráfica (c) muestra también un punto intermedio en una ruta larga. En este caso la configuración inicial del robot es bastante desfavorable para alcanzar el destino intermedio con la orientación requerida, sin embargo el robot consigue llegar correctamente al objetivo intermedio, prosiguiendo después su trayectoria hacia el destino final. La gráfica (d) muestra la trayectoria seguida por el robot cuando en el primer intento de aproximación al objetivo no lo logra y lo intenta una segunda vez con éxito. Como en los ejemplos anteriores, el robot al entrar en la zona de “Preparación” se aleja del destino para poder llegar con la orientación adecuada. En este caso al iniciar la zona de “Orientación” el robot no logra girar lo suficiente, lo cual ocurre a veces si una piedra introduce

una perturbación en las ruedas que retrasa la respuesta del controlador de la dirección. Al sobrepasar el destino final, la distancia robot-objetivo aumenta y entra de nuevo en la zona de "Preparación" y posteriormente en la de "Aproximación". De este modo se aleja; sin embargo en el segundo intento el robot ya dispone de espacio suficiente para girar y alcanza el objetivo con éxito, aunque el error en la orientación es mayor que en los otros casos ($error_{medio} = 20$ grados). Al ser este punto un objetivo intermedio en una ruta larga, el robot prosigue la navegación hacia el destino final.

El agente **Avanzar** dirige correctamente al robot DÉDALO en trayectorias complejas por el campo consiguiendo llegar al objetivo fijado con la orientación requerida, con un error medio de $error_{medio} = 20$ (grados).

6.2.4 Experimentos del agente Evitar Obstáculos en navegación autónoma

Igual que en los casos anteriores el agente **Evitar Obstáculos** se ha probado intensivamente primero con el simulador y posteriormente con el robot real DÉDALO. En ambos casos se muestra la capacidad de reacción del robot ante obstáculos estáticos, tratando de rodearlos y ante obstáculos dinámicos o muy próximos parándose.

Experimentos del agente Evitar Obstáculos en simulación

En los experimentos de la figura 6.19 se observa cómo el robot rodea diferentes obstáculos definidos en simulación por el usuario y representados en color azul. La columna de la derecha, corresponde a la pantalla del simulador donde se ha marcado en rojo la ruta seguida por el robot. La columna de la izquierda, corresponde a una captura de la pantalla, una vez alcanzado el objetivo, de la aplicación de navegación global.

Las gráficas (a) y (b) de la figura 6.19 muestran la trayectoria del robot evitando obstáculos que bloquean la parte central del camino tentativo panificado. En estos casos el robot rodea los obstáculos por la derecha, tal y como se define en las reglas del agente, capítulo 5. En la

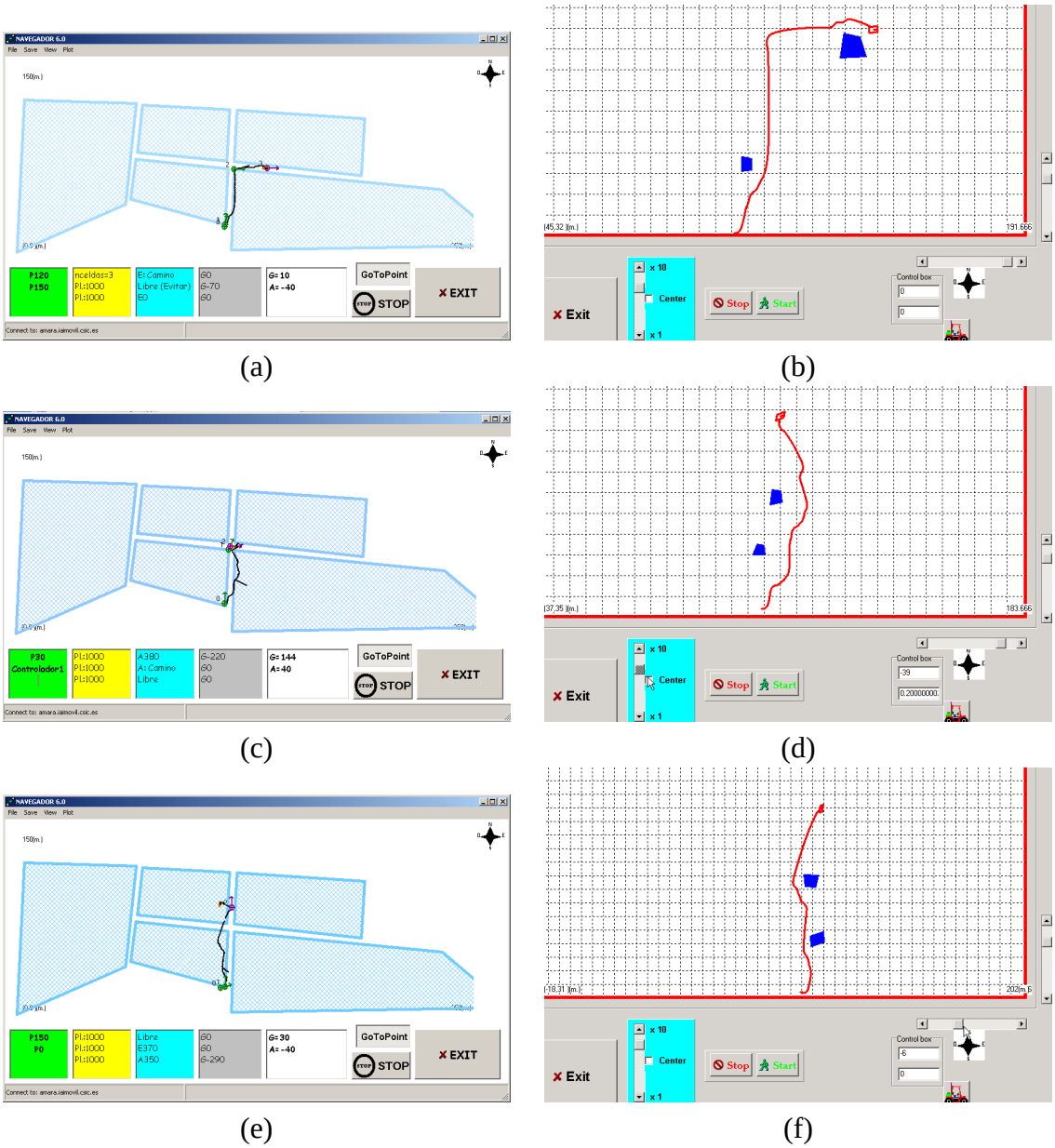


Figura 6.19 : Experimentos con el agente **Evitar Obstáculos** en simulación

la ruta.

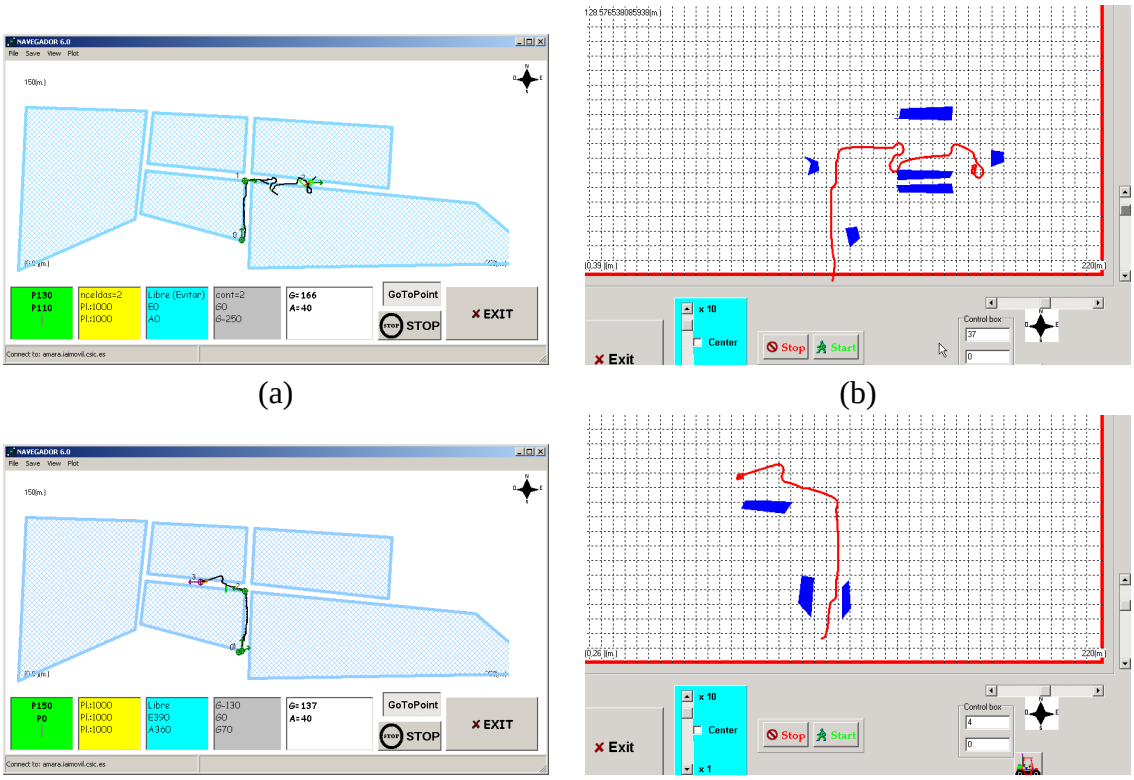
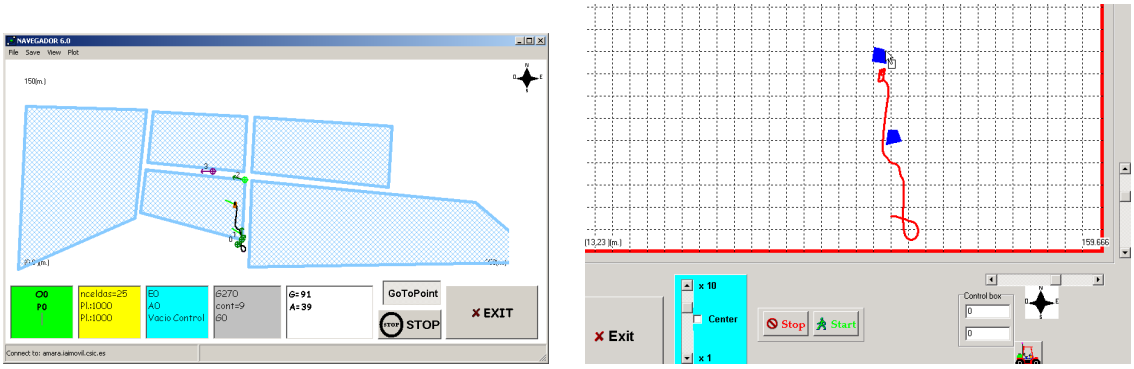


Figura 6.20 : Experimentos en simulación con el agente **Evitar Obstáculos**. Trayectorias descritas ante la aparición de múltiples obstáculos.

En los recorridos realizados con el simulador, figura 6.20, el robot esquiva eficazmente una serie de obstáculos y llega con éxito al punto destino con la orientación adecuada en todos los casos. En el recorrido que aparece en las gráficas (a) y (b) de la figura 6.20 puede observarse que el robot da una vuelta antes de detenerse en la posición destino final. Este comportamiento extraño, se debe al hecho de que el punto destino está muy próximo a un obstáculo y se produce una secuencia de procesos de activación y desactivación de los agentes **Avanzar** y **Evitar Obstáculos**, que explicaría esta situación. Finalmente las gráficas (c) y (d) ilustran el recorrido del robot al moverse entre dos obstáculos que forman un pasillo.

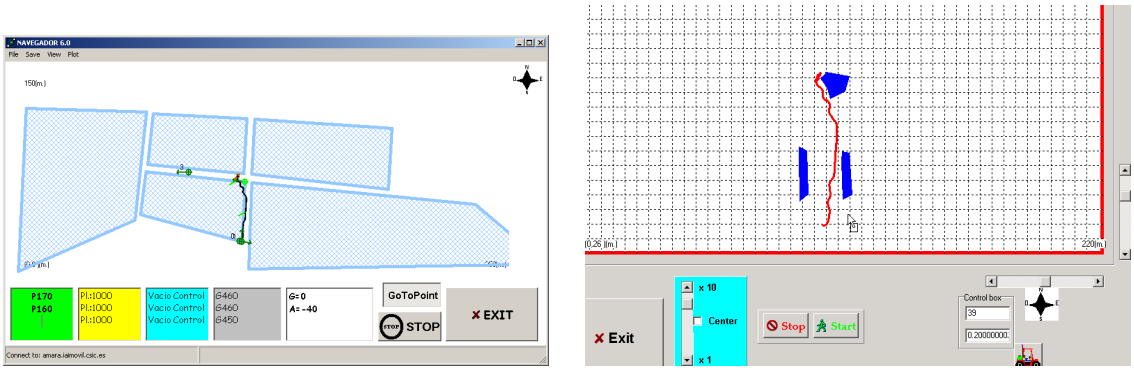
Para concluir los experimentos en simulación con el agente **Evitar Obstáculos**, se muestran dos ejemplos más, uno correspondiente a una parada de seguridad ante un obstáculo muy próximo que ha aparecido repentinamente antes de que el robot pueda maniobrar, figura 6.21. El

otro, figura 6.22, muestra un ejemplo en el cual el robot no ha sido capaz de rodear un obstáculo, aproximándose tanto a él que se activa la parada de seguridad y queda bloqueado. Este bloqueo es detectado por el agente **Ir a Punto** que envía una señal de alerta al operario humano para que resuelva la situación conflictiva.



(a) (b)

Figura 6.21 : Parada de seguridad realizada por el agente **Evitar Obstáculos**



(a) (b)

Figura 6.22 : Recorrido fallido. Parada ante un obstáculo muy próximo: agente **Evitar Obstáculos**

Experimentos del agente Evitar Obstáculos con el robot DÉDALO

En los experimentos presentados al principio de esta sección para mostrar el comportamiento global de navegación general no específica se mostraba en las gráficas de la figuras 6.7 y 6.8, recorridos con actuación del agente **Evitar Obstáculos**. En el primero de los casos, figura 6.7, se observa cómo el robot desvía su trayectoria tentativa hacia el objetivo al

existir un obstáculo imprevisto (por eso no aparece en el mapa local) a su izquierda, evitándolo correctamente. En el segundo caso, figura 6.8, el obstáculo está muy próximo al robot y el agente **Evitar Obstáculos** detiene al robot.

Un nuevo experimento muestra una captura de pantalla de la interfaz Navegador durante una ruta larga, figura . Durante esta ruta completa el robot ha evitado obstáculos en diversas ocasiones, estos obstáculos imprevistos han llevado a los agentes de navegación local: **Avanzar**, **Evitar Obstáculos** y **Parar**, a navegar fuera de las pistas, mostrando que el *Plan de Navegación* es únicamente un recurso. En concreto la zona resaltada con la elipse roja muestra la actuación de rodeando unas gradas instaladas en la zona de pistas del IAI-CSIC para un evento. Estas gradas se han dibujado aproximadamente en la captura de pantalla del navegador en color naranja.

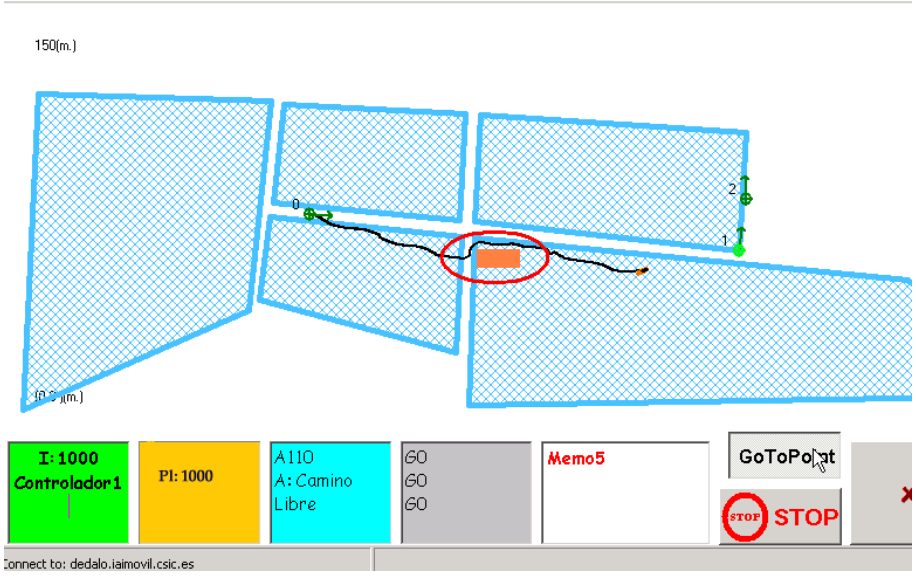


Figura 6.23 : Actuación del agente **Evitar Obstáculos** en una ruta real del tractor DÉDALO por las pistas del IAI-CSIC

6.3 Experimentos en navegación de laboreo autónomo

El objetivo de la navegación de laboreo es recorrer de forma autónoma el campo de labor en su totalidad ejecutando un conjunto de segmentos o trayectorias parciales, explicitadas en el

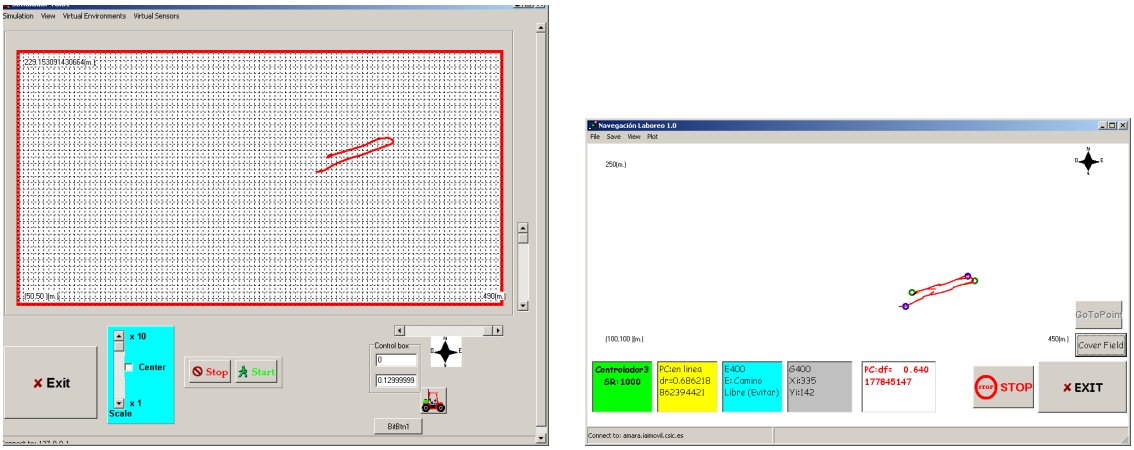
plan de laboreo presentado por el operario, partiendo de un mapa georreferenciado del campo y de una posición inicial del robot, preservando la seguridad del sistema y del entorno, capítulo 5.

La aplicación visual diseñada e implementada para ejecutar este tipo de navegación autónoma se denomina Navegación de Laboreo. La aplicación presenta un interfaz hombre-máquina amigable, que permite al usuario definir las localizaciones intermedias del *Plan de Laboreo* marcando con el ratón sobre el mapa georreferenciado del campus de Arganda del Rey donde se encuentra el IAI-CSIC. Esta aplicación actúa como un programa cliente que se conecta con el programa DÉDALO-servidor, ya sea el del tractor DÉDALO o del simulador, para solicitar el envío de datos sensoriales y activar la ejecución del agente **Recorrer Campo**, a partir del cual se desencadena la secuencia de procesos que constituyen la navegación de laboreo.

Experimentos de navegación de laboreo con el simulador AGRO-SIM

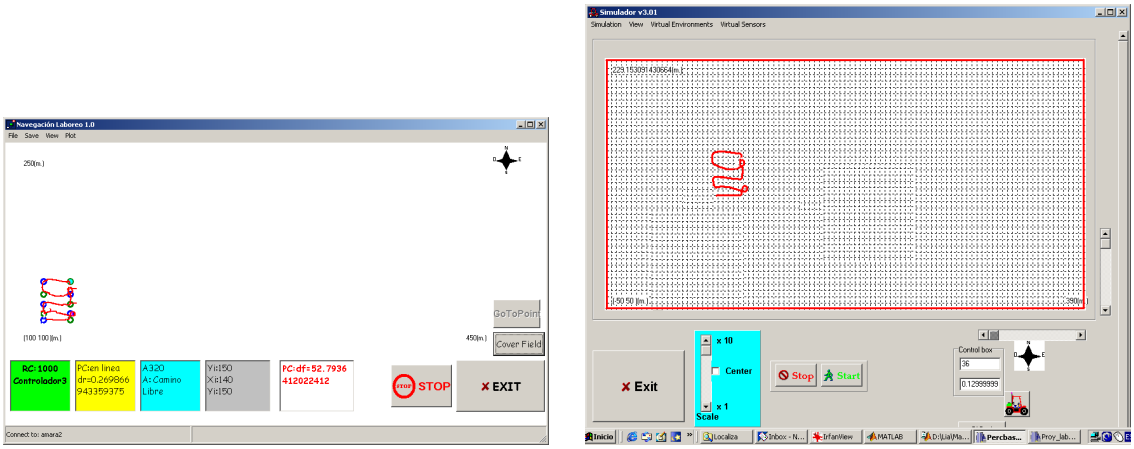
Los dos ejemplos de navegación de laboreo con el simulador corresponden a dos situaciones muy diferentes. El primer caso, figura 6.24, muestra el campo de pruebas del simulador, modelado a semejanza del campo de laboreo real del IAI-CSIC, que consiste en un tramo de terreno con dos hileras de olivos. El segundo caso, figura 6.25, es un campo ficticio que permite validar el comportamiento de navegación de laboreo en condiciones distintas a las de los campos que rodean al IAI-CSIC. Este campo ficticio correspondería a un campo con 5 hileras de cultivo. La separación entre hileras varía entre cada par para permitir la experimentación en simulación con variaciones en este parámetro. En ambos casos la gráfica de la izquierda, etiquetada como (b), corresponde a una captura de pantalla de la aplicación navegación de laboreo y la de la derecha, etiquetada como (a), a la pantalla del simulador.

En ambos casos se observa cómo el robot lleva a cabo el plan navegando en línea recta en las hileras y girando para cambiar de una hilera a otra. El recorrido simulado en el campo virtual semejante al del IAI-CSIC no presenta complicaciones pues el giro no es demasiado cerrado. En



(a) (b)

Figura 6.24 : Experimentos de navegación de laboreo en simulación en terrenos del IAI-CSIC



(a) (b)

Figura 6.25 : Experimento de navegación de laboreo en simulación en un campo ficticio

el caso del campo ficticio los giros son más cerrados y por ello el robot se ve forzado a realizar una maniobra en uno de ellos a fin de recorrer apropiadamente la hilera de olivos adyacente.

El experimento de la figura 6.26 muestra la parada del robot ante un obstáculo imprevisto próximo al robot. La gráfica (a) corresponde a una captura de pantalla del simulador que muestra el obstáculo en el instante de su creación en el simulador. La gráfica (b) es la pantalla del programa de navegación de laboreo cuando el robot se para al haber detectado el obstáculo y la (c) muestra la reanudación de la marcha del robot cuando se elimina el obstáculo del simulador. Puede observarse que el robot se ha parado a una distancia lejana del obstáculo, debido a que,

al aparecer en el simulador como “caído del cielo”, el agente **Actualizar Obstáculos** lo toma inicialmente por un obstáculo móvil y por lo tanto el agente **Evitar Obstáculos** opta por parar el robot.

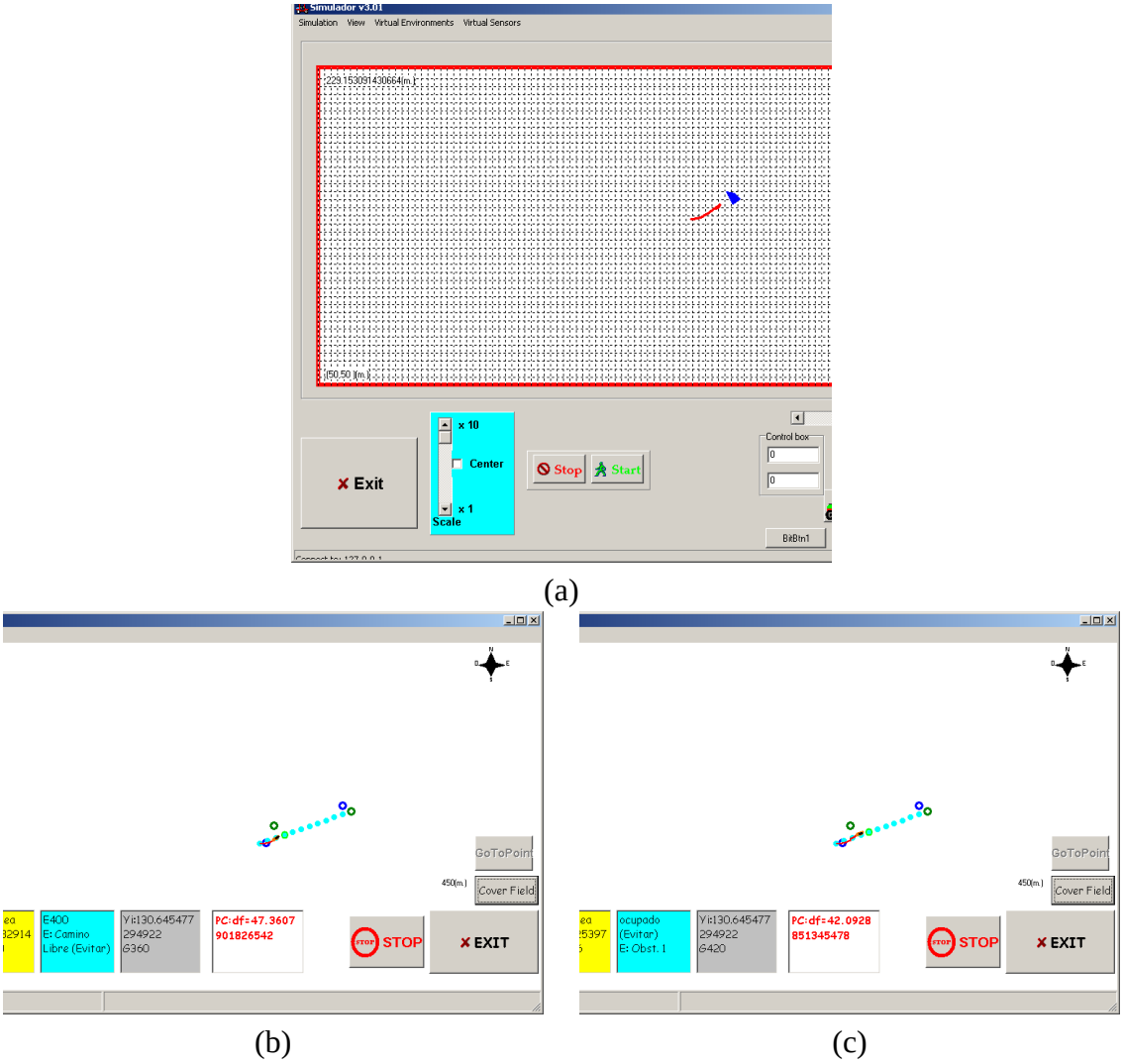


Figura 6.26 : Parada ante un *Obstáculo* durante la navegación de laboreo

Experimentos de navegación de laboreo con el robot DÉDALO

El experimento de la figura 6.27 muestra la captura de pantalla de la interfaz de Navegador Laboreo en un recorrido del tractor DÉDALO en el campo de olivos del IAI-CSIC. En la figura 6.27 puede verse la ruta completa compuesta de dos tramos de navegación en línea recta y uno

de navegación por el arco de circunferencia que une los dos trayectos rectilíneos.

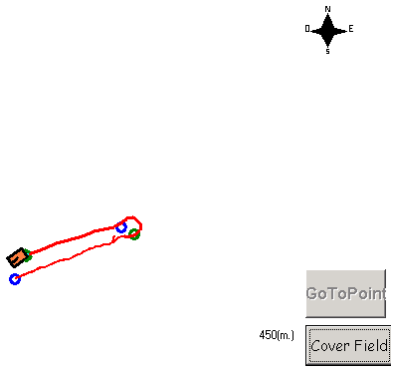


Figura 6.27 : Experimento de navegación real del robot DÉDALO en el campo de olivos del IAI-CSIC

6.3.1 Experimentos del agente Actualizar Posición en Campo

El agente **Actualizar Posición en Campo** mantiene actualizada la percepción *En Línea* que es la variable perceptiva que condiciona la activación de los agentes **Seguir Recta** y **Seguir Arco de Circunferencia**. Los experimentos presentados muestran el funcionamiento de este agente perceptivo, diferenciando las posiciones del robot en las cuales la percepción *En Línea* es cierta de aquellas en las que la percepción *En Línea* es falsa.

Los experimentos con el simulador muestran el valor de la percepción *En Línea* durante un trayecto complejo del robot. Se han representado mediante asteriscos azules aquellas posiciones para las cuales *En Línea* es cierta y con marcas rojas las posiciones en las que *En Línea* es falsa.

El recorrido de la figura 6.28 muestra un recorrido simulado por el campo virtual semejante al IAI-CSIC. Todo el tramo curvo de la trayectoria corresponde al valor falso de *En Línea*, salvo la parte final que enlaza con la segunda trayectoria rectilínea. Esto es debido a que, como se mostró en la descripción del agente en el capítulo 5, el valor falso de *En Línea*corresponde a posiciones en los alrededores de los puntos finales del camino. El hecho de que esto sea así no afecta negativamente a la navegación, pues el agente **Seguir Recta** retoma la navegación forzando al robot a moverse en la recta de referencia.

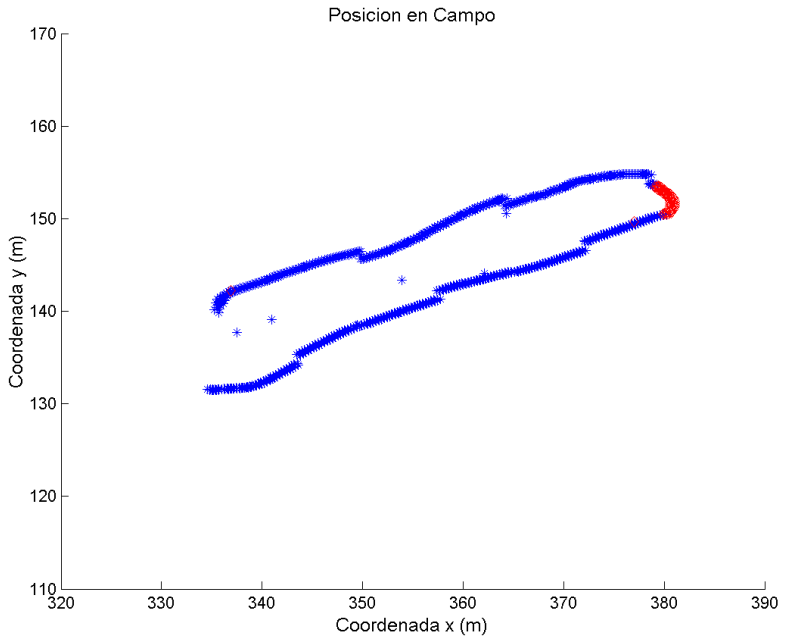


Figura 6.28 : Actuación del agente **Actualizar Posición en Campo**. En asteriscos azules las posiciones correspondientes a valores verdaderos de *En Línea*. Con marcas rojas las posiciones correspondientes a valores falsos de *En Línea*

6.3.2 Experimentos con el agente Seguir Recta

Con el fin de validar el comportamiento del agente **Seguir Recta**, se han analizado en detalle los tramos de navegación ejecutados por este agente, comparando los recorridos obtenidos con el simulador con los conseguidos con el robot real en el seguimiento de la trayectoria rectilínea tentativa.

Experimentos con el simulador del agente Seguir Recta

Con el simulador se han analizado las desviaciones de las posiciones del robot con respecto a las líneas rectas en el recorrido ficticio mostrado en experimentos de simulación anteriores. La figura 6.29 muestra el recorrido completo, donde las posiciones del robot se han señalado con aspas rojas y los puntos del plan con asteriscos azules, representando en ese mismo color la trayectoria rectilínea tentativa. Para analizar en detalle el seguimiento de las rectas se han revisado por separado las cinco rectas que forman el camino. Se ha calculado el error en los

cinco tramos, figura 6.30, como la distancia, según la perpendicular, entre cada posición del robot simulado a la trayectoria rectilínea tentativa. Los errores medios están por debajo de los $1{,}5$ (m) superándose este valor en casos puntuales. La gráfica (a) de la figura 6.30 corresponde al tramo R1 de la figura 6.29, siendo el tramo con menor error. En todos los casos se observa que los errores más grandes se presentan al principio y final de cada tramo rectilíneo. Esta tendencia se debe a que en el inicio el robot debe alinearse con la línea recta después del giro. Al final la situación es la opuesta, el robot comienza a girar para aproximarse y orientarse en la siguiente línea recta con el consiguiente aumento del error.

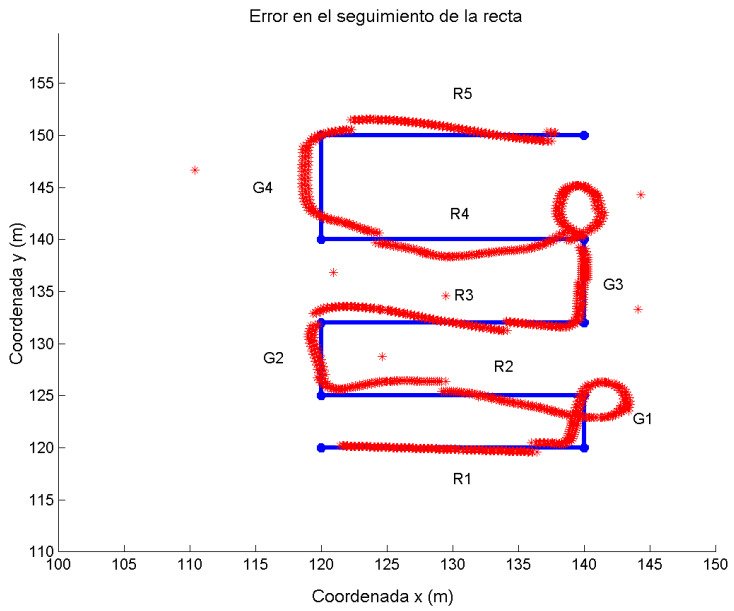


Figura 6.29 : Recorrido autónomo en un campo simulado. Comportamiento del agente **Seguir Recta**

La figura 6.31 muestra el recorrido efectuado en simulación en el campo virtual similar a los terrenos del IAI-CSIC, donde las posiciones del robot se marcan con aspas rojas y las posiciones del plan de laboreo con asteriscos azules unidos con rectas del mismo color que marcan la trayectoria rectilínea tentativa. Los dos tramos se han analizado por separado para estudiar el seguimiento de la trayectoria rectilínea, del mismo modo que en el caso anterior sobre el campo ficticio. En las gráficas (a) y (b) de la figura 6.32 puede verse cómo el error muestra la misma tendencia que en el experimento anterior, siendo más elevado en el primer

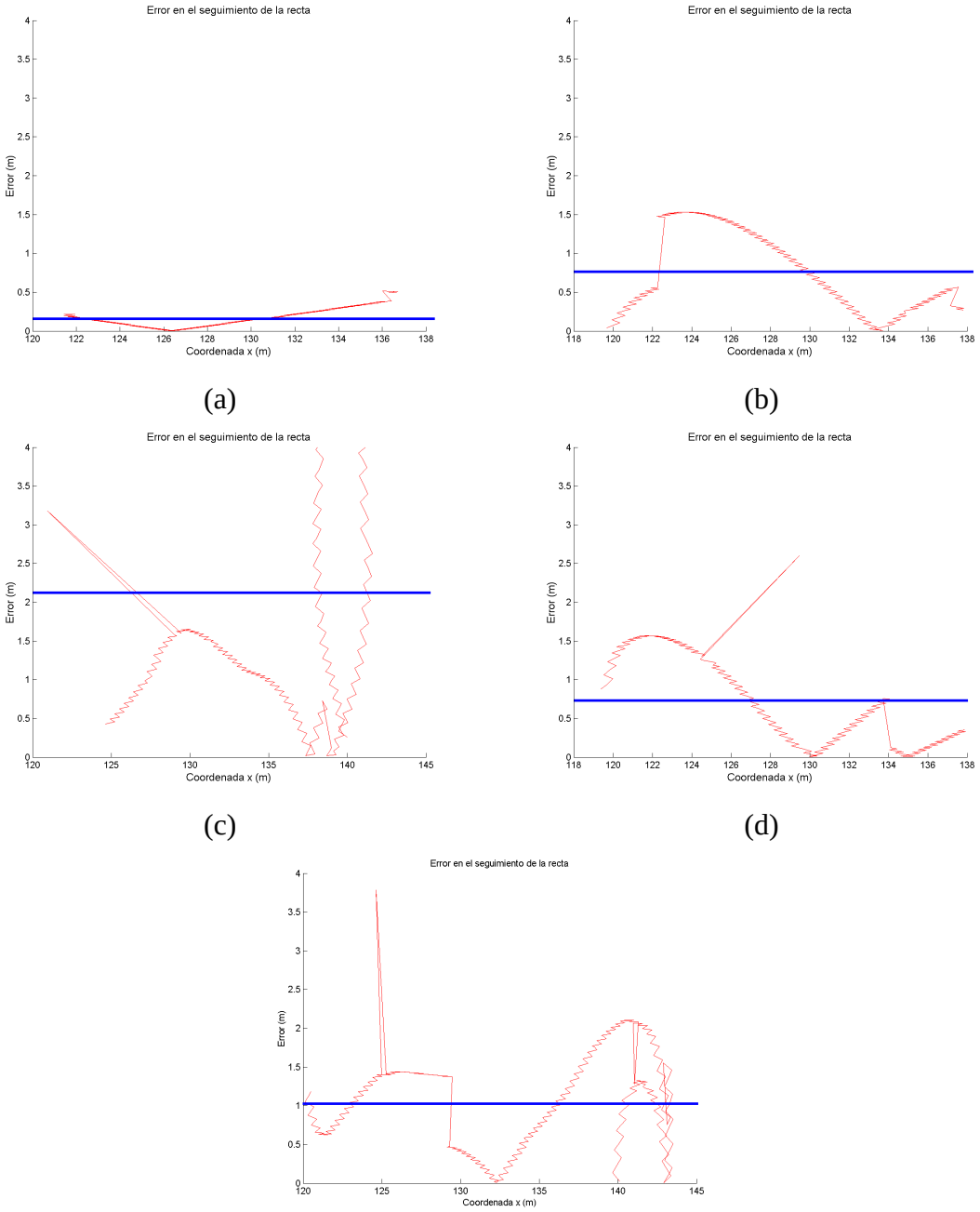


Figura 6.30 : Errores en el seguimiento de lineas rectas en simulación: agente **Seguir Recta**

tramo de ambos recorridos y disminuyendo posteriormente para mantenerse constante durante el resto de la trayectoria, con un valor por debajo de 1 (m).

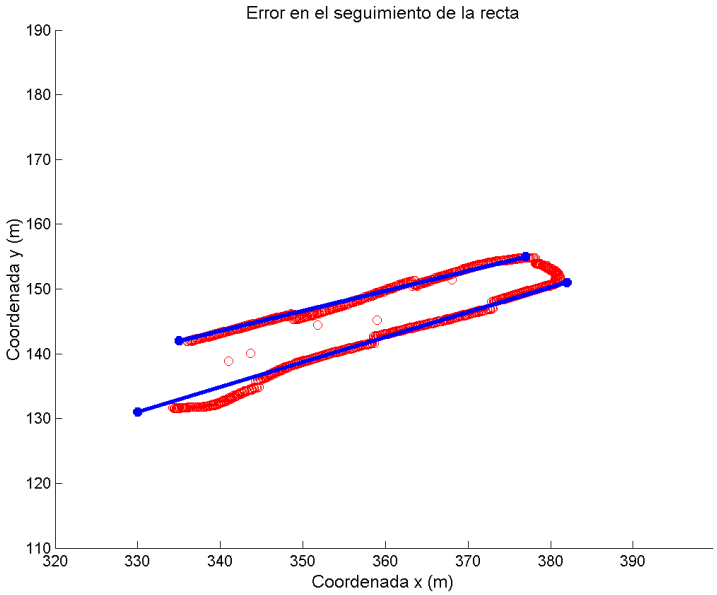
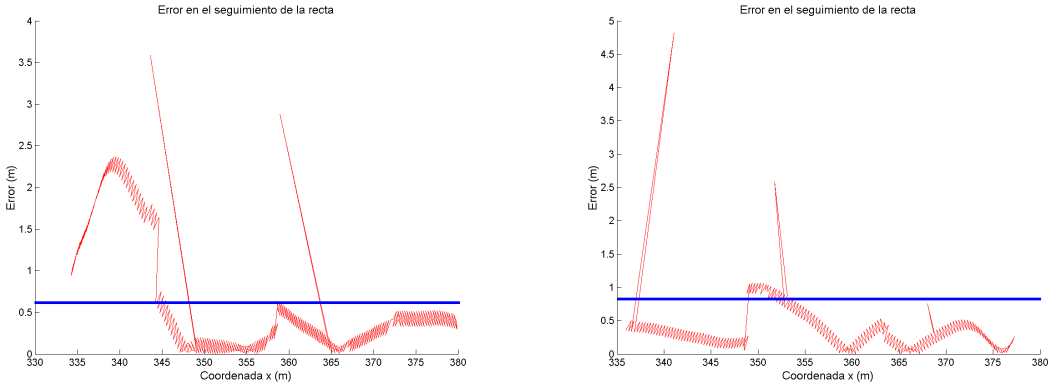


Figura 6.31 : Recorrido simulado en un campo virtual similar a los terrenos del IAI-CSIC. Comportamiento del agente **Seguir Recta**



(a) (b)

Figura 6.32 : Error en el seguimiento de una línea recta

Experimentos con el robot DÉDALO del agente Seguir Recta

En las gráficas de la figura 6.33 se muestran en el lado izquierdo de la imagen las capturas de pantalla con dos recorridos en línea recta del tractor DÉDALO por el campo de olivos del IAI-CSIC. En el lado derecho se ha representado el error en el seguimiento de la línea recta, calculado como la diferencia entre la posición del robot y la recta objetivo.

El recorrido correspondiente a las gráficas a1 y b1 de la figura 6.33 muestra el buen seguimiento de la recta, pintada en la gráfica a1 en azul para mayor claridad. En este recorrido el error se mantiene por debajo del metro en la mayoría de los puntos y presenta un comportamiento muy similar al encontrado en los casos de simulación. Alrededor de la posición $x = 375$ se encuentran unos puntos con errores de casi 3 metros. Estos errores son debidos a una pequeña desviación del tractor DÉDALO en la ruta a causa de un terrón que provocó una perturbación importante en la dirección. Puede observarse cómo el tractor DÉDALO recupera correctamente la línea recta tras esta desviación.

El recorrido representado en las gráficas a2 y b2 presenta una desviación importante de la trayectoria rectilínea deseada alrededor de la mitad del recorrido. Esta desviación corresponde con un error más elevado de lo habitual en el posicionamiento que provocó una consigna de control inadecuada para la situación real del robot. Cabe destacar cómo DÉDALO recupera la trayectoria rectilínea una vez que se recuperan los valores habituales en el posicionamiento del robot. Durante el resto del recorrido los errores se mantienen en torno a 1 (m.), de acuerdo con las predicciones hechas en la simulación.

6.3.3 Experimentos con el agente Seguir Arco de Circunferencia

El agente **Seguir Arco de Circunferencia** toma el control de la navegación del robot cuando *En Línea* es falsa. Los experimentos siguientes muestran cómo efectúa este agente al cambio entre dos trayectorias rectilíneas.

Experimentos con el simulador del agente Seguir Arco de Circunferencia

Las figuras 6.34 y 6.35 muestran dos recorridos simulados, el primero de ellos sobre el campo virtual similar a los terrenos del IAI-CSIC y el segundo sobre un campo ficticio. La gráfica (a) de la figura 6.34 muestra la ruta completa realizada donde la parte correspondiente a la navegación controlada por el agente **Seguir Arco de Circunferencia** se ha dibujado con marcas rojas. La parte (b) ofrece este tramo más en detalle donde pueden observarse además los

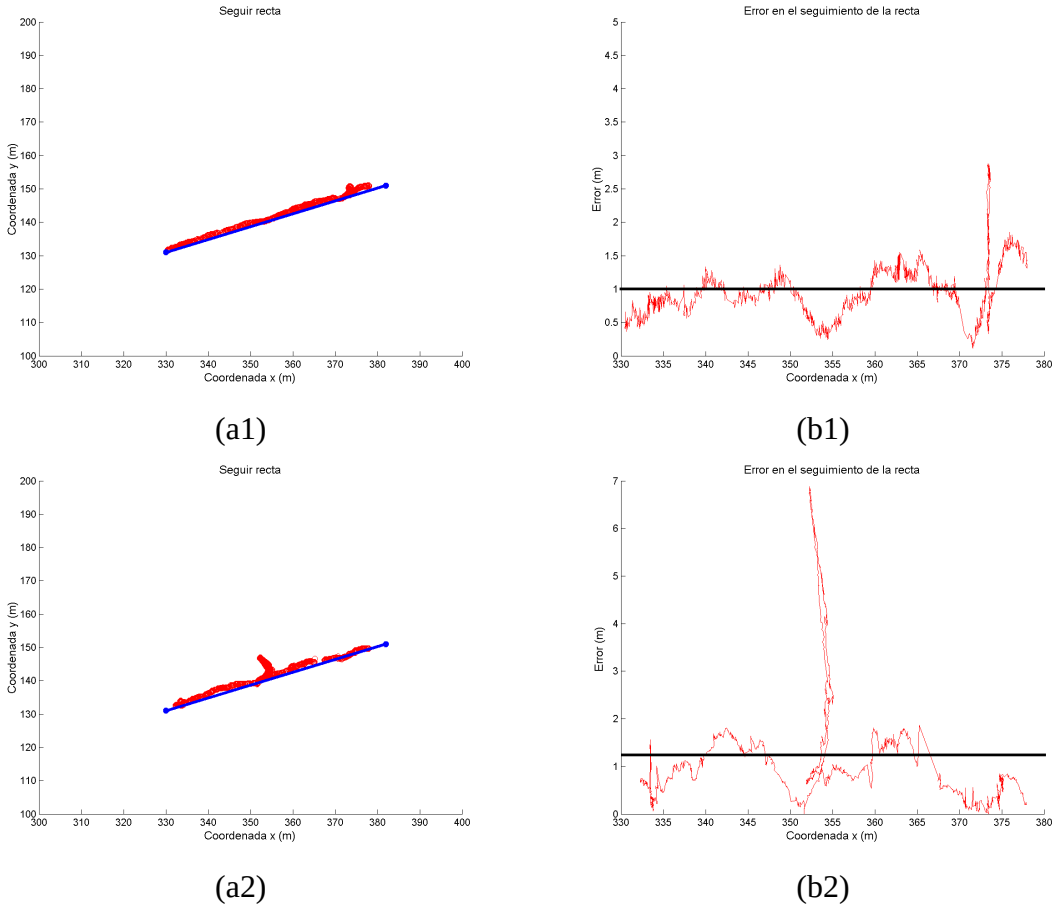
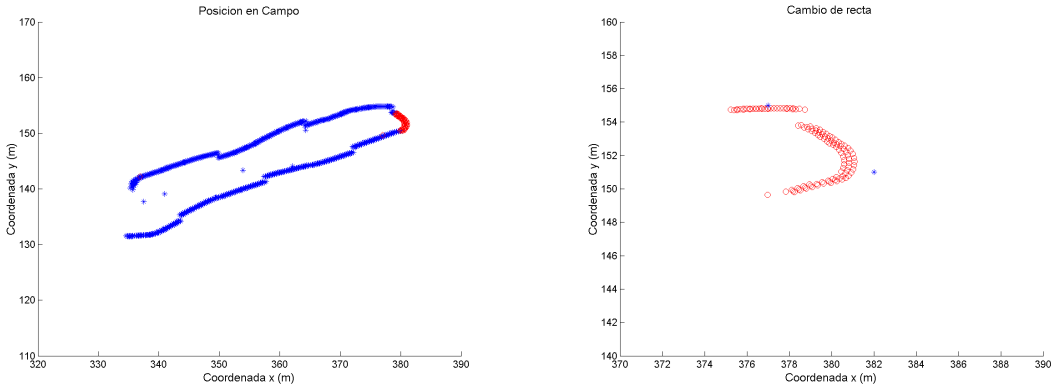


Figura 6.33 : Comportamiento del agente **Seguir Recta**. (a) Recorrido autónomo del robot DÉDALO por el campo de olivos del IAI-CSIC . (b) Error en el seguimiento de la línea recta

puntos final del primer tramo rectilíneo e inicial del segundo, indicados con asteriscos azules. Como puede observarse el giro comienza antes de llegar al primero de los puntos y se llega correctamente a la posición deseada.

La figura 6.35 muestra una ruta con cuatro tramos diferentes correspondientes al agente **Seguir Arco de Circunferencia**. Se ha variado la separación entre líneas de cultivo para así tener una experimentación más rica y comprobar el buen funcionamiento del agente en giros de diferente radio.



(a) (b)

Figura 6.34 : Actuación del agente **Seguir Arco de Circunferencia**. Campo simulado

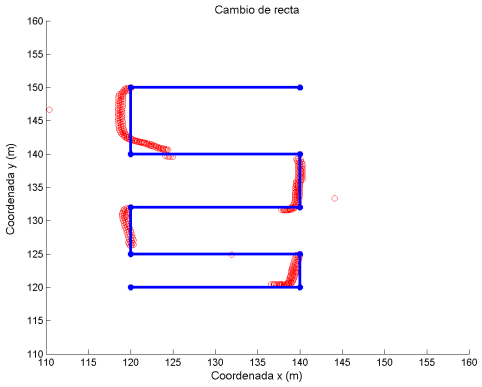


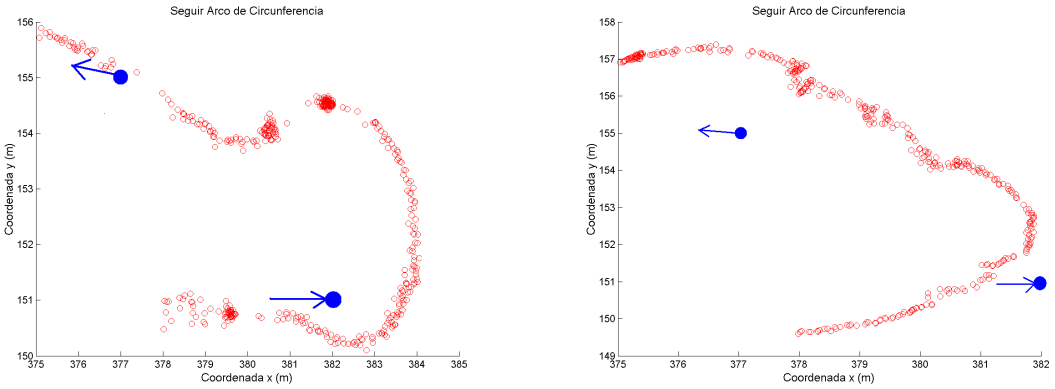
Figura 6.35 : Actuación del agente **Seguir Arco de Circunferencia**. Campo simulado con separación diferente entre líneas de cultivo

Experimentos con el robot DÉDALO del agente Seguir Arco de Circunferencia

En la figura 6.36 se muestran dos ejemplos del de navegación correspondiente al agente **Seguir Arco de Circunferencia** en un recorrido autónomo del robot DÉDALO en el campo de olivos del IAI-CSIC.

En las gráficas (a) y (b) de la figura 6.36 se aprecia cómo el robot DÉDALO comienza a girar, gobernado por el agente **Seguir Arco de Circunferencia** cuando se encuentra aproximadamente a 1 (m.) de la posición final de la primera recta objetivo, marcada con un círculo y flecha azul en la figura. Gira hasta que la diferencia entre su posición y el punto de inicio de la segunda recta, también marcado con un círculo azul y fecha, es menor que 1 (m.),

momento en el que retoma la navegación rectilínea.



(a) (b)

Figura 6.36 : Actuación del agente **Seguir Arco de Circunferencia**. Recorrido autónomo del robot DÉDALO en el campo de olivos del IAI-CSIC

En este capítulo se han presentado diferentes resultados experimentales tanto en simulación como con el robot DÉDALO que muestran el funcionamiento de la arquitectura propuesta en tareas de navegación generales y específicas de laboreo. Se muestran resultados tanto de rutas complejas de navegación como detalles del funcionamiento de los distintos agentes particulares.

Capítulo 7

Conclusiones y líneas de desarrollo futuro

Los objetivos de este trabajo han sido el diseño y desarrollo de una arquitectura de control y organización del conocimiento para la navegación autónoma de vehículos agrícolas en entornos dinámicos, que se ha denominado AGRO-AMARA. El trabajo aquí presentado ha tenido desde sus inicios el propósito de ir más allá de la mera propuesta de un modelo teórico de arquitectura por lo que ésta se ha validado experimentalmente implantándola en un sistema real complejo y por tanto afrontando todos los problemas y la carga de trabajo que esto conlleva. A tal fin se ha automatizado y sensorizado un tractor comercial, lo que ha permitido la validación de la arquitectura tanto en navegación global, en campo parcialmente estructurado, como en navegación de laboreo, en campo con cultivo estructurado.

7.1 Aportaciones

Las aportaciones más importantes del presente trabajo se aglutinan en dos grandes grupos. El primero, relacionado con las contribuciones a las áreas de automatización y control, se puede resumir en los siguientes puntos:

1. Se ha afrontado la automatización de un vehículo agrícola comercial dotado de un sistema de actuación hidráulico, integrando los componentes necesarios para el control de la dirección y de la tracción, capítulo 4. La complejidad del sistema y la existencia tanto de interacciones como de holguras entre las distintas partes del mismo, además de las perturbaciones ocasionadas por el desplazamiento en terrenos irregulares, han conducido al diseño de un sistema de control borroso para la dirección que alcanza sin oscilaciones, las precisiones requeridas en el giro. El sistema de control de dirección consta de un control proporcional borroso y un control por muestreo rápido que permite al sistema corregir errores pequeños. El controlador borroso de la dirección hidráulica constituye en sí mismo una novedad en Robótica y se ha comprobado su funcionamiento en el tractor con múltiples recorridos de navegación autónoma, obteniéndose excelentes resultados para cualquier variación del ángulo de giro.

2. Se ha incorporado al tractor una dotación sensorial adecuada a las tareas a realizar y al entorno que se debe percibir, requisito indispensable para afrontar una navegación autónoma, capítulo 4. Así, el sistema sensorial debe permitir por un lado conocer la posición del robot y por otro percibir los objetos del entorno que puedan afectar su integridad y modo de operación. Con objeto de obtener estos requisitos se han integrado, y posteriormente verificado, los siguientes sensores:

    - Sensores de localización global: receptor DGPS y brújula digital
    - Sistema odométrico de localización relativa, diseñado y construido en el IAI-CSIC dentro de esta tesis

- Sensores de percepción del entorno: láser de barrido, sensores de inclinación y sensor de contacto.

3. Se ha desarrollado un sistema software para la adquisición y el proceso de señales así como para la ejecución de los ciclos de control de los actuadores que se ejecuta en el procesador a bordo del robot, capítulo 4. Este procesador se comunica con cualquier estación de trabajo de la red inalámbrica del IAI-CSIC, vía Radio-Ethernet. Los datos son servidos por una aplicación, denominada DÉDALO-Servidor, diseñada y desarrollada dentro del paradigma cliente/servidor acorde con la implementación de la arquitectura propuesta. Este sistema permite el acceso remoto a los datos sensoriales y el envío remoto de comandos a los actuadores, facilitando por ejemplo la teleoperación del robot.

4. Motivado por las dificultades que tiene la experimentación con robots agrícolas en exteriores, se ha propuesto y desarrollado un simulador del tractor, anexo B. La experimentación en exteriores es ardua, requiere mucho tiempo y depende de múltiples factores ajenos al sistema, entre los que se encuentran las condiciones meteorológicas. El simulador implementado constituye una herramienta de gran utilidad en la depuración de procesos y en el análisis de situaciones poco frecuentes. La principal ventaja del mismo radica en el hecho de que la información de entradas y salidas sigue exactamente los mismos protocolos de comunicación que se han definido entre los agentes de la arquitectura de control del robot. Por lo tanto, todo el código de proceso y comunicación de los agentes simulados se puede transportar directamente al sistema de proceso del tractor para su experimentación inmediata.

Recapitulando se ha desarrollado una plataforma automatizada completamente operativa sobre un tractor comercial real, en un escenario donde hay pocos vehículos completamente automatizados pero que ofrece unos beneficios potenciales enormes, como se destacó en el capítulo 2.

El segundo grupo de contribuciones está relacionado con la arquitectura de control. En el presente trabajo se muestra cómo el análisis de las aplicaciones agrícolas parcialmente automatizadas existentes, de las características de las tareas que el tractor debe desempeñar, de las restricciones de los entornos de trabajo y de las arquitecturas de control existentes para robots autónomos, ha conducido a formular un conjunto de requisitos esenciales, capítulo 3. Estos requisitos modulan el desarrollo de la arquitectura de organización tanto del conocimiento como de los procesos de control que tiene como finalidad lograr la navegación autónoma segura de un robot móvil en un entorno agrícola, objetivo de esta tesis. Entre las características más relevantes que se han incorporado a la arquitectura propuesta, se encuentran:

- Capacidad de reacción alta para preservar la seguridad del sistema en un entorno dinámico alejado del operario
- Capacidad de deliberación a fin de optimizar la toma de decisiones, a partir la información disponible
- Representación del conocimiento y percepción orientadas a la tarea
- Inyección inicial al sistema de toda la información del contexto que disponga el experto
- Orientación de los procesos perceptivos hacia el estímulo que activa un agente

Por tanto, la arquitectura que se propone en esta tesis, AGRO-AMARA, es una arquitectura híbrida basada en agentes organizados jerárquicamente en lo que se refiere a la reutilización de habilidades. Se trata de una arquitectura diseñada específicamente para vehículos y labores agrícolas, lo que constituye un aporte de esta tesis. En esta arquitectura los agentes comparten información sobre la base de dos mecanismos, paso de mensajes y memoria compartida, en este caso concreto con la estructura de una memoria de pizarra. El diseño de arquitectura sigue una filosofía conservadora y oportunista en todas sus facetas que busca lograr los principios básicos de modularidad, reusabilidad de procesos, y de facilidad de escalado e implantación en diferentes plataformas.

Las principales características de la arquitectura y sus implicaciones en la consecución del objetivo final, se resumen en los siguientes puntos:

- La arquitectura AGRO-AMARA constituye un modelo distribuido de múltiples expertos cuyas habilidades perceptivas y de actuación están encapsuladas en los agentes.
- AGRO-AMARA establece una división de los agentes en dos tipos: perceptivos y de actuación.
- La generación de agentes exclusivamente perceptivos responde a la necesidad de elaborar un arquitectura con percepción distribuida y orientada a la tarea.
- Las percepciones y representaciones elaboradas por los agentes perceptivos, de utilidad a varios agentes, se almacenan en una memoria compartida o pizarra, de libre acceso.
- Los agentes puramente deliberativos, como **Planificar Caminos**, utilizan representaciones del sistema y entorno obtenidas con anterioridad por otros agentes o inyectadas por el operario, y generan un plan o representación de actuaciones que se almacena en la pizarra.
- La arquitectura lleva implícito un flujo de información bidireccional de percepción y acción, distribuida en sus múltiples agentes.
- Los agentes de nivel superior, **Ir a Punto** en navegación global y **Seguir Recta** y **Seguir Arco de Circunferencia** en navegación de laboreo, reutilizan las capacidades de otros agentes de bajo nivel **Avanzar**, **Evitar Obstáculos** y **Parar**, que disponen de comunicación directa con los procesos de control de los actuadores físicos, logrando comportamientos diferentes en función de la secuencia y la modulación que establezcan sobre los mismos.
- Los agentes tienen contextos de activación mutuamente exclusivos para evitar la activación simultánea de más de un agente.

La arquitectura AGRO-AMARA ha sido completamente validada con el tractor comercial automatizado DÉDALO, tanto en tareas de navegación global como en navegación de laboreo, capítulo 6. La arquitectura propuesta ha permitido la navegación autónoma del robot DÉDALO en un entorno dinámico abierto y en campos de cultivo estructurados, en un conjunto amplio de experimentos de navegación.

En resumen, el diseño modular de la arquitectura AGRO-AMARA permite su crecimiento gradual en la consecución de comportamientos más complejos, bien por la integración de otros subsistemas (cámara de visión, manipuladores, sistemas de fumigación con múltiples sectores, etc..) o bien por la variación de los objetivos y tareas.

La capacidad de reacción ante imprevistos y fallos en partes del sistema ha sido probada ante situaciones imprevistas y mediante la generación de fallos controlados. Las capacidades deliberativas de la arquitectura favorecen la optimización de la toma de decisiones a tiempo, dada la velocidad de operación de los procesadores. La percepción encapsulada que propone esta arquitectura, aumenta considerablemente la capacidad de respuesta del sistema, y la correcta consecución de los objetivos, al no pretender la reconstrucción del entorno sino dirigirse al estímulo requerido por los agentes de actuación.

La arquitectura ha sido diseñada para facilitar su futuro crecimiento de forma gradual, mediante la incorporación de dispositivos que aumenten y mejoren sus habilidades actuales.

7.2 Líneas de desarrollo futuro

En cuanto a las líneas por las que se va a continuar la investigación realizada en esta tesis destacan las siguientes.

De forma inmediata se procederá a incorporar una cámara de vídeo para la fusión de información de *Obstáculos* en la representación del *Mapa Local* ya existente. El sistema de visión permitirá el reconocimiento de estructuras para la ayuda al guiado del tractor tanto en campo abierto como en los campos de cultivo. Una lección aprendida durante el desarrollo de esta tesis es que la percepción constituye punto clave en la consecución de un comportamiento

autónomo en los robots de exteriores, de ahí que la incorporación de un sistema de visión en el robot DÉDALO introduzca mejoras en las prestaciones actuales de los agentes perceptivos: **Actualizar Mapa Local**, *Mapa Local*, **Actualizar Obstáculos** y **Actualizar Posición en Campo**.

Asimismo, a corto plazo se integrará un apero de laboreo consistente en una barra con múltiples sectores controlados por electroválvulas, que permitirá la aplicación selectiva de tratamientos. Esto exigirá la generación e integración de nuevos agentes en la arquitectura propuesta. Además de las mejoras en la percepción, mediante fusión sensorial, se apuntan como líneas de desarrollo futuro de la arquitectura: 1) la mejora en el seguimiento de líneas rectas, 2) la planificación automática del laboreo y 3) el arbitraje entre los agentes y la detección de situaciones anómalas.

Sin duda la navegación general y la de laboreo tienen que confluir en un único proceso que sea capaz de gestionar toda la navegación, desde la salida y entrada al hangar hasta las trayectorias específicas de un laboreo en campo de cultivo estructurado. De este modo se evolucionará hacia un robot agrícola con mayor grado de autonomía, con una interfaz de comunicación con el operario sencilla, intuitiva y eficaz que facilite la supervisión de las tareas agrícolas con una intervención mínima.

Esta tesis constituye un paso hacia la automatización progresiva de tareas agrícolas, aumentando los beneficios potenciales, tanto económicos, como medioambientales y de reducción de riesgos para los humanos.

Anexo A

Pseudocódigo de los agentes de la arquitectura AGRO-AMARA

A.1 Pseudocódigo de los agentes de navegación global general

A.1.1 Agentes perceptivos

Agente Actualizar Posición

```
WHILE(Actualizar Posición->activo)
t1=toma_tiempo()
IF(GPS_diff==true AND Num_satelites>4 AND Delta(d_gps)<d_th)
GPS=OK
ELSE
```

```
GPS=False
END IF
IF(Delta(Ang_brujula)<Ang_th)
Brujula=OK
ELSE
Brujula=FALSE
END IF
IF (d_modo3<10)
Odo=OK
ELSE
Odo=FALSE
END IF
IF((GPS=OK)AND(Brujula=OK))
modo_pos=0
X_R=X_gps
Y_R=Y_gps
Ang_R=Ang_brujula
ELSE IF((GPS=OK)AND(Brujula=FALSE))
modo_pos=1
X_R=X_gps
Y_R=Y_gps
Ang_R=Ang_gps
ELSE IF((GPS=false)AND(Brujula=OK)AND(Odo=OK))
modo_pos=2
X_R=X_R(t-1)+(d_odo(t)-d_odo(t-1))cos(Ang_brujula)
Y_R=Y_R(t-1)+(d_odo(t)-d_odo(t-1))sin(Ang_brujula)
Ang_R=Ang_brujula
ELSE IF((GPS=false)AND(Brujula=false)AND(Odo=OK))
```

```
modo_pos=3
X_R=X_R(t-1)+(X_odo(t)-X_odo(t-1))
Y_R=Y_R(t-1)+(Y_odo(t)-Y_odo(t-1))
Ang_R=Ang_R(t-1)+(Ang_odo(t)-Ang_odo(t-1))
d_modo3=d_modo3+(d_odo(t)-d_odo(t-1))
ELSE
modo=4
END
t2=toma_tiempo()
wait_for(200-(t2-t1))
END WHILE
```

Agente Actualizar Mapa Local

```
Actualizar Posición->Activar
inciar_mapalocal()
WHILE (agente Actualizar Mapa Local->activo)
t1=toma_tiempo()
IF(p=0)
FALLO
Terminar()
END IF
IF (Nuevo_Laser=TRUE)
incorpora_laser(medidas_laser,mapa_local)
incorpora_robot(posicion_robot,mapa_local)
IF(d(robot,mapa_local.final)<10)
reubica_mapa(centro,mapa_local)
END IF
END IF
```

```
t2=toma_tiempo()
wait_for(500-(t2-t1))
END WHILE
terminar_mapalocal()
Actualizar_Posicion->Desactivar
```

Agente Actualizar Obstáculos

```
Actualizar_Mapa_Local->Activar()
Actualizar_Posicion->Activar()
iniciar_obstaculos(obstaculos)
WHILE (Actualizar_Obstaculos->activo)
t1=toma_tiempo()
IF(po*m=0)
FALLO
Terminar()
END IF
actualizar_obstaculos(obstaculos)
t2=toma_tiempo()
wait_for(500-(t2-t1))
END WHILE
Actualizar_Mapa_Local->Desactivar()
Actualizar_Posicion->Desactivar()
terminar_obstaculos(obstaculos)
```

A.1.2 Agentes de actuación

Agente Parar

```
WHILE(Parar->alerta OR Parar->activo)
t1=toma_tiempo()
IF(choque=TRUE)
freno=ON
Parar->activo
ELSE
IF(pendiente_lateral>15)
freno=ON
Parar->activo
ELSE
IF(pendiente_frontal>15)
freno=ON
Parar->activo
ELSE
Parar->alerta
END IF
END IF
END IF
t2=toma_tiempo()
wait_for(500-(t2-t1))
END WHILE
```

Agente Evitar Obstáculos

```
Actualizar Posición->Activar()
Actualizar Obstáculos->Activar()
```

```
iniciar_evitar()
WHILE((Evitar Obstáculos->activo) OR (Evitar Obstáculos->alerta)
t1=toma_tiempo()
IF(choque=TRUE)
Evitar Obstáculos->alerta
ELSE
IF(pl<15)
Evitar Obstáculos->alerta
ELSE
IF(pl<15)
Evitar Obstáculos->alerta
ELSE
k=0
FOR(i=1;i<NumeroObstaculos;i++)
IF(d(Oi,R)<dSS)
Freno->ON
Evitar Obstáculos->activo
break;
ELSE
IF(d(Oi,R)<dS)
IF(v(Oi)>0)
Freno->ON
Evitar Obstáculos->activo
break;
ELSE
alfa_i=evitar(gamma_i)
Evitar Obstáculos->activo
END IF
```

```
ELSE
k=k+1
END IF
END IF
END FOR
IF (k=i)
Evitar Obstáculos->alerta
ELSE
alfa=(alfa_i/(i-k))
END IF
END IF
END IF
END IF
t2=toma_tiempo()
wait_for(500-(t2-t1)
END WHILE
Actualizar Posicion->terminar()
Actualizar Obstaculos->terminar()
terminar_evitar()
```

Agente Avanzar

```
Actualizar_Posicion->Activar()
Actualizar_Obstaculos->Activar()
iniciar_avanzar()
WHILE (Avanzar->activo OR Avanzar->alerta)
t1=toma_tiempo()
IF(choque=TRUE)
```

```
Avanzar->alerta
ELSE
IF(pf>15)OR(pl>15)
Avanzar->alerta
ELSE
FOR(i=1;i<numero_obstaculos;i++)
IF((O[i]->numero_celdas>3)AND((distancia(O[i],R))<distancia_seguridad))
Avanzar->alerta
BREAK
ELSE
Avanzar->activo
comandos=avanzar(G,R)
END IF
END FOR
END IF
END IF
t2=toma_tiempo()
wait_for(500-(t2-t1))
END WHILE
Actualizar_Posicion->Desactivar()
Actualizar_Obstaculos->Desactivar()
```

Agente Planificar Caminos

```
Actualizar Posicion->Activar()
WHILE(Planificar Caminos->activo)
t1=toma_tiempo()
IF(nuevo_plan=TRUE)
planificar(R,G)
```

```
END IF
t2=toma_tiempo()
wait_for(5000-(t2-t1))
END WHILE
Actualizar Posicion->Terminar()
```

Agente Ir a Punto

```
Parar->Activar()
Avanzar->Activar()
Evitar Obstáculos->Activar()
Planificar->Activar()
Actualizar Posición->Activar()
WHILE(Ir a Punto->activo)
toma_tiempo(t1)
IF(d(R,D)<d_FIN)
FINAL
Ir a Punto->inactivo()
ELSE
IF(nuevo_plan=TRUE)
i=1;
SubObjetivo=Plan[i]
t_transcurrido=0
nuevo_plan=FALSE
ELSE
IF(d(R,Objetivo)<d_OBJETIVO)
i=i+1
Objetivo=Plan[i]
t_transcurrido=0
```

```
ELSE
IF(t_transcurrido>t_esperado)
IF((Rx[t]=Rx[t-1])AND(Ry[t]=Ry[t-1]))
Send(FALLO)
Ir a Punto->inactivo()
ELSE
nuevo_plan=TRUE
END IF
ELSE
IF(po*pl*p*a*e=0)
Send(FALLO)
Ir a Punto->inactivo()
ELSE
SWICH(p*e*a)
CASE 1
Parar->Activo
CASE 8
Avanzar->Alerta
Evitar Obstaculos->Alerta
Parar->Activo
CASE 4
IF(p=2)
Avanzar->Alerta
Evitar Obstaculos->Alerta
Parar->Activo
ELSE
Avanzar->Alerta
Evitar Obstaculos->Activo
```

```
END IF
END SWICH
END IF
END IF
END IF
END IF
END IF
toma_tiempo(t2)
wait_for(1000-(t2-t1))
END WHILE
Parar->Desactivar()
Avanzar->Desactivar()
Evitar Obstáculos->Desactivar()
Planificar->Desactivar()
Actualizar Posición->Desactivar()
```

A.2 Pseudocódigo de los agentes de la navegación específica de laboreo

A.2.1 Agentes perceptivos

Agente Actualizar Posición en Campo

```
Actualizar_Posicion->Activar()
WHILE(Actualizar_Posicion_Campo->Activo)
toma_tiempo(t1)
Inicio de línea=PlanCampo[punto_ejecucion][0]
Fin de línea=PlanCampo[punto_ejecucion][1]
```

```
IF(distancia(R,Fin de línea)<d_F)
En Línea=FALSO
ELSE
IF(distancia(R,r)>d_r)
IF(En Línea=VERDADERO)
fallo_surco=VERDADERO
Terminar()
END IF
ELSE
En Línea=VERDADERO
END IF
END IF
toma_tiempo(t2)
wait_for(500-(t2-t1))
END WHILE
Actualizar_Posicion->Desactivar()
```

A.2.2 Agentes de actuación

Agente Girar

```
Actualizar_Posicion->Iniciar()
Actualizar_Obstaculos->Iniciar()
iniciar_girar()
WHILE((Girar->Alerta)OR(Girar->Activo))
t1=toma_tiempo()
IF(choque=TRUE)
Avanzar->alerta
```

```
ELSE
IF(pf>15)OR(pl>15)
Avanzar->alerta
ELSE
FOR(i=1;i<numero_obstaculos;i++)
IF((O[i]->numero_celdas>3)AND((distancia(O[i],R))<distancia_seguridad))
Girar->alerta
BREAK
ELSE
IF(radio_giro<5)
error_giro=TRUE
ELSE
Girar->activo
psi=atan((yg-yr)/(xg-xr))
IF((psi-theta_r)<0)
alfa_volante=girar(radio_giro)
ELSE
alfa_volante=girar(-radio_giro)
END IF
END IF
END FOR
END IF
END IF
t2=toma_tiempo()
wait_for(500-(t2-t1))
END WHILE
Actualizar_Posicion->Desactivar()
Actualizar_Obstaculos->Desactivar()
```

Agente Seguir Recta

```
Actualizar Posición->Activar()
Actualizar Posición en Campo->Activar()
Avanzar->Actualizar()
Evitar Obstáculos->Actualizar()
Parar->Actualizar()
WHILE((Seguir Recta->activo)OR(Seguir Recta->alerta))
t1=toma_tiempo()
IF (En Línea=TRUE)
IF (Avanzar->inactivo)
Avanzar->Activar()
END IF
IF(Fin de línea=Fin de línea_ant)nueva_recta=TRUE
ELSE nueva_recta=FALSE
END IF
IF (nueva_recta=TRUE)
Xr[0]=Xi
Yr[0]=Yi
beta=atan((Yf-Yi)/(Xf-Xi))
THETAr[0]=beta
FOR (k=1;k<d/5;k++)
Xr[k]=Xr[k-1]+(d/5(*cos(beta))
Yr[k]=Yr[k-1]+(d/5)*sin(beta))
THETAr[k]=beta
END FOR
Objetivo[0]=Xr[0]
Objetivo[1]=Yr[0]
```

```
Objetivo[3]=THETA[0]
punto_recta=0
END IF
IF(d(R,Objetivo)<d_OBJETIVO)
punto_recta++
Objetivo[0]=Xr[punto_recta]
Objetivo[1]=Yr[punto_recta]
Objetivo[3]=THETA[punto_recta]
END IF
IF(po*pc*p*a*e=0)
Send(FALLO)
Seguir Recta->inactivo()
ELSE
SWICH(p*e*a)
CASE 1
Parar->Activo
CASE 8
Avanzar->Alerta
Evitar Obstáculos->Alerta
Parar->Activo
CASE 4
IF(p=2)
Avanzar->Alerta
Evitar Obstáculos->Alerta
Parar->Activo
ELSE
Avanzar->Alerta
Evitar Obstáculos->Activo
```

```
END IF
END SWICH
END IF
END IF
t2=toma_tiempo()
wait_for(1000-(t2-t1))
END WHILE
Parar->Desactivar()
Avanzar->Desactivar()
Evitar Obstáculos->Desactivar()
Actualizar Posición en Campo->Desactivar()
Actualizar Posición->Desactivar()
```

Agente Seguir Arco de Circunferencia

```
Actualizar Posición->Activar()
Actualizar Posición en Campo->Activar()
Girar->Actualizar()
Evitar Obstáculos->Actualizar()
Parar->Actualizar()
WHILE((Seguir Arco de Circunferencia->activo)OR(Seguir Arco de Circunferencia->alerta))
t1=toma_tiempo()
IF (En Línea=FALSE)
IF (Girar->inactivo)
Girar->Activar()
END IF
IF(distancia(Fin de línea[i],Inicio de línea[i+1])<5)
Fallo_Giro=TRUE
Terminar()
ELSE
IF ((Alfa_r-Theta_r)<0)
```

```
diametro_giro=distancia(R,Inicio de línea[i+1])
ELSE
diametro_giro=-distancia(R,Inicio de línea[i+1])
END IF
IF(po*pc*p*g*e=0)
Send(FALLO)
Cambiar Recta->inactivo()
ELSE
SWICH(p*e*g)
CASE 1
Parar->Activo
CASE 8
Girar->Alerta
Evitar Obstaculos->Alerta
Parar->Activo
CASE 4
IF(p=2)
Girar->Alerta
Evitar Obstaculos->Alerta
Parar->Activo
ELSE
Girar->Alerta
Evitar Obstaculos->Activo
END IF
END SWICH
END IF
END IF
t2=toma_tiempo()
wait_for(1000-(t2-t1))
END WHILE
Parar->Desactivar()
Girar->Desactivar()
Evitar Obstáculos->Desactivar()
Actualizar Posición en Campo->Desactivar()
```

```
Actualizar Posición->Desactivar()
```

Agente Recorrer Campo

```
Seguir_Recta->Activar()
Cambiar_Recta->Activar()
Actualizar_Posición->Activar()
inciar_plan
WHILE((Recorrer_Campo->activo)OR(Recorrer_Campo->alerta))
t1=toma_tiempo()
IF(Inicio de línea->alcanzado=TRUE)
IF(Fin de línea->alcanzado=TRUE)
i=i+1
ELSE
Goal=Fin de línea
END IF
ELSE
Goal=Inicio de línea
END IF
IF(distancia(R,G)<D_goal)
IF(G=Inicio de línea)
Inicio de línea->alcanzado=TRUE
ELSE
IF(G=Fin de línea)
Fin de línea->alcanzado=TRUE
IF(i=npuntos_plan)
Plan_Terminado=TRUE
Terminar()
END IF
```

```
ELSE
Fallo()
Terminar()
END IF
END IF
END IF
IF(p*sr*cr=0)
Fallo()
Terminar()
ELSE
IF(sr*cr=4)
Seguir_Recta->activo
END IF
END IF
t2=toma_tiempo()
wait_for(1000-(t2-t1))
END WHILE
Seguir_Recta->Desactivar()
Seguir Arco de Circunferencia->Desactivar()
Actualizar_Posición->Desactivar()
```

Anexo B

El simulador AGROSIM: modelo del robot, entorno y sensores

B.1 Modelo de movimiento virtual

El modelo cinemático expresa las relaciones geométricas que describen el movimiento del vehículo con respecto a un sistema de referencia. Este modelo está compuesto por un conjunto de ecuaciones que relacionan la posición del vehículo (x_R, y_R, θ_R) con su longitud L, su velocidad v_R y el ángulo de las ruedas α_v. El modelo cinemático del tractor DÉDALO puede, en primera aproximación, reducirse al modelo de la bicicleta [Dudek y Jenkin, 2000], siendo esta aproximación válida para $\Delta t \downarrow\downarrow$.

Por ello se ha modelado el tractor como un cuerpo rígido que se mueve en un plano con cuatro puntos de contacto con el suelo, uno por cada rueda. Las ruedas traseras son las de tracción y las delanteras las de dirección. El modelo relaciona las variables: α_v ángulo de giro

de las ruedas, L distancia entre los puntos de apoyo de las ruedas traseras y delanteras y v_R velocidad del vehículo con las coordenadas del robot en el plano (x_R, y_R, θ_R), relativas a un sistema de referencia, con el origen en el punto inicial de la trayectoria del robot y orientado según la orientación inicial de éste, figura B.1.

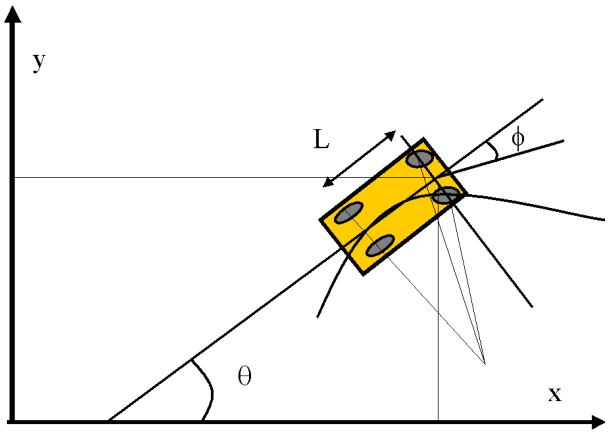


Figura B.1 : Esquema del modelo cinemático

Las restricciones impuestas al modelo son las siguientes: (1) no existe deslizamiento y (2) el vehículo se desplaza a baja velocidad. Los cambios de posición del vehículo vienen descritos por las expresiones:(B.1).

$$\begin{aligned} \dot{x_R} &= v_R \cdot \cos\alpha_v \cdot \cos\theta_R \\ \dot{y_R} &= v_R \cdot \cos\alpha_v \cdot \sin\theta_R \\ \dot{\theta_R} &= \frac{v_R}{L}\sin\alpha_v \end{aligned} \tag{B.1}$$

A pesar de que estas ecuaciones no son integrables [Laugier et al., 1999], si se asume que la velocidad v_R y el ángulo de giro del volante α_R permanecen constantes en cada ciclo de cálculo, se puede integrar la expresión (B.1). Así, integrando la ecuación en θ_R se obtiene (B.2).

$$\Delta\theta_R = (\frac{v_R}{L} \cdot \sin\alpha_v) \cdot \Delta t \tag{B.2}$$

Y considerando que $\theta_R = \theta_{R_{ant}} + \Delta\theta_R$, (B.3) son las expresiones para las coordenadas x_R e y_R.

$$\begin{aligned} \Delta x_R &= (v_R \cdot \Delta t \cdot \cos\alpha_v) \cdot \sin\theta_R \\ \Delta y_R &= (v_R \cdot \Delta t \cdot \cos\alpha_v) \cdot \cos\theta_R \end{aligned} \tag{B.3}$$

Las ecuaciones (B.2) y (B.3) pueden implementarse iterativamente:

1. Cálculo del cambio en la orientación del robot en función de la velocidad, el giro del volante y el intervalo de tiempo transcurrido, (B.2).
2. Cáculo de la orientación actual del robot: $\theta_R = \theta_{R_{ant}} + \Delta\theta_R$.
3. Cálculo de la coordenada x_R actual, (B.3).
4. Cálculo de la coordenada y_R actual, (B.3).

Este algoritmo es válido siempre que el intervalo de tiempo Δt sea lo suficientemente pequeño como para asegurar que, tanto α_v como v_R permanecen constantes.

■ **Limitaciones del modelo.** Al tratarse de un modelo cinemático sólo tenemos en cuenta consideraciones geométricas, modelando el vehículo como un sólido rígido con cuatro puntos de apoyo. Las condiciones no ideales de una dinámica real, hacen que factores como el deslizamiento o la descompensación por la distribución o cambios en la carga y componentes del vehículo modifiquen la trayectoria real. Estos factores al igual que la pendiente del terreno, no se contemplan en la versión actual del simulador ya que exigirían un modelo mucho más complejo y unos tiempos de desarrollo mucho más largos, que rebasan las expectativas impuestas a las prestaciones del simulador en el momento actual. Igualmente hay que tener en cuenta que el modelo empleado es únicamente una aproximación del modelo cinemático de los vehículos reales.

B.2 Generación de entornos virtuales

Un punto fundamental en el diseño de un simulador para un robot móvil es el modelo del entorno en el que el robot va a trabajar. Este modelo del entorno posee, en todos los simuladores, grandes simplificaciones quedando reducido a describir aquellas características del entorno que se consideran fundamentales para la realización de su misión. Así pues no tiene sentido considerar en el entorno las propiedades de color de los objetos cuando el robot carece de sensores para la detección del color. Todos los simuladores disponen de un modelo del mundo, ya sea un mapa de bits con objetos *Mobile Robot Simulator*, [de Lope Asiaín, 2002], *Stage*, [Vaughan, 2001], o *Botworld*, en este último los objetos incluyen también balizas de localización, o un entorno 3D complejo *SHIVA* [Sukthankar et al., 1996] y *MOBS* [Mobs, 2002].

En nuestro caso el modelo del entorno describe dos propiedades detectables con los sensores instalados a bordo del tractor real. Por un lado la altitud del terreno, modelada como un conjunto de polígonos que encierran una superficie con puntos de igual altitud. Y por otro la distancia a los objetos, modelados como polígonos de n lados, donde cada objeto lleva asociada una ecuación que describe su trayectoria temporal.

B.2.1 Definición de obstáculos

Una característica importante a modelar en el entorno de un vehículo real móvil, son los obstáculos dinámicos. Esto se debe a que la detección de su grado de dinamismo va a condicionar la estrategia de navegación del robot y ésta necesita un grado de anticipación amplio dado que los vehículos reales de tamaño medio y grande maniobran con dificultad.

El mundo se ha modelado mediante un conjunto de polígonos de n lados, sin límite en n, y una trayectoria asociada a cada uno de ellos. El movimiento de cada uno de los objetos se selecciona entre los 4 patrones definidos, tabla B.1. La dinámica de los obstáculos se simula, aplicando a su centro de masas (B.4) el modelo cinemático del tractor (B.1). El ángulo de giro y la velocidad instantánea son los parámetros que modulan los diferentes modos de movimiento.

(x_v, y_v) son las coordenadas de cada vértice del obstáculo y N_v el número de vértices.

$$x_{CM} = \frac{\sum_{i=1}^{N_v} x_{vi}}{N_v}$$
$$y_{CM} = \frac{\sum_{i=1}^{N_v} y_{vi}}{N_v} \tag{B.4}$$

Modo	Descripción	Velocidad	Ángulo de las ruedas
0	Estático	$v = 0$	$\alpha = 0$
1	Mov. rectilíneo uniforme	$v = v_0$	$\alpha = 0$
2	Mov. rectilíneo uniformemente acelerado	$v = v_0 + a \times t$	$\alpha = 0$
3	Mov. circular	$v = v_0$	$\alpha = \alpha_0$
4	Mov. pendular	$v = v_0 \times \sin(\omega t)$	$\alpha = 0$

Tabla B.1 : Trayectorias de obstáculos implementadas en AGRO-SIM

La posibilidad de incluir obstáculos móviles, facilita considerablemente el diseño de estrategias de navegación adecuadas, fundamentalmente en aquellas circunstancias en las que es necesario iniciar un cambio en la trayectoria con la suficiente anticipación como para que la maniobra de giro permita un posicionamiento correcto del robot DÉDALO. Conviene remarcar que el radio de giro del robot es limitado ($r_{min} = 5{,}0$ (m.)) y que el cambio de marcha adelante a marcha atrás se tiene que realizar de forma manual, previa parada del vehículo. Al estar parametrizada la definición de la dinámica de cada obstáculo, el usuario puede definir en tiempo de ejecución su modelo de movimiento. Además puede emplear una interfaz tipo joystick como el que se muestra en la figura B.2 para dirigir el obstáculo por la trayectoria que desee, facilitando de este modo la experimentación con obstáculos móviles que posean desplazamientos especiales.

Siguiendo la filosofía de definición de obstáculos propuesta, se podría añadir al simulador otros robots móviles, convirtiéndose así el simulador en un banco de análisis de interacciones y aprendizaje de estrategias de navegación en dominios con múltiples robots, de gran interés en tareas coordinación de la navegación colectiva de flotas de vehículos.

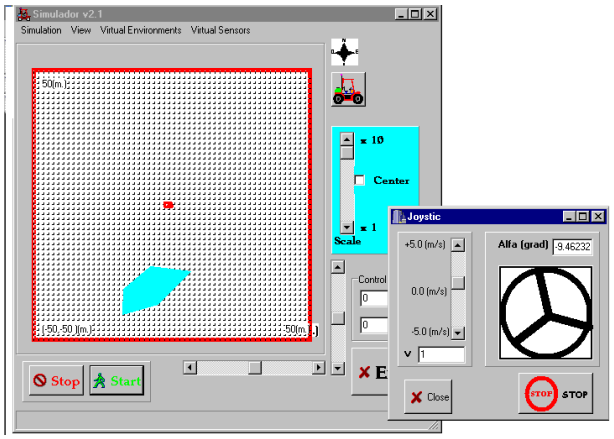


Figura B.2 : Obstáculo controlado por joystick

B.2.2 Definición de altitudes del terreno

Otra propiedad del entorno simulada es la altitud del terreno. Aquí las superficies de terreno con una misma altitud se han representado con el mismo color. Las líneas poligonales cerradas que encierran superficies de igual altura son las curvas de nivel que nunca se cortan. Para su representación se han escogido tres colores, rojo, verde y blanco. Las superficies en rojo representan altitudes mayores (con respecto al nivel del mar) que la referencia que se seleccionada que aparece reflejada en blanco. Las superficies verdes son de menor altitud con respecto de esta referencia anterior.

El usuario puede definir, en tiempo de ejecución, la altitud y la línea poligonal, con n vértices, que desee en cada región del mundo, figura B.3.

Las características buscadas en esta representación son: 1) facilidad de manipulación y 2) representación gráfica intuitiva, a fin de facilitar la labor de construcción de mapas por un usuario no experto.

B.3 Sensores virtuales

Los sensores son los dispositivos que permiten al robot la obtención de información del estado del entorno y de sí mismos. Son imprescindibles para guiar las estrategias de navegación

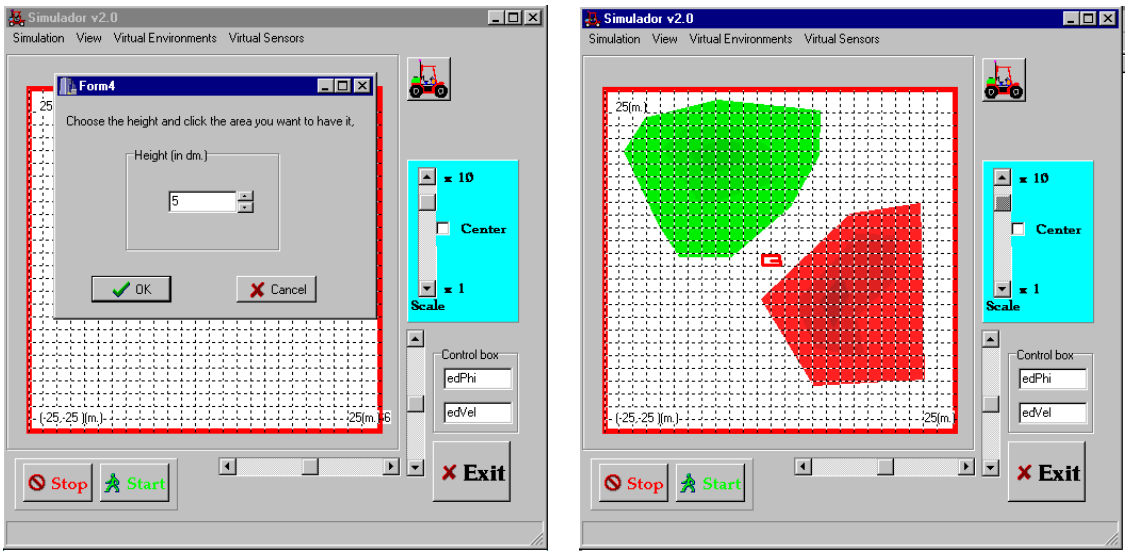


Figura B.3 : Definición topográfica del entorno

de un robot móvil en entornos que no están totalmente estructurados y siempre que exista incertidumbre, tanto en el modelado de los sistemas como en el control de los mismos.

En lo que respecta al posicionamiento relativo del robot, fundamental para el establecimiento de relaciones con el resto de los objetos del mundo, es posible encontrar en la literatura un conjunto amplio que carece de los mismos, *Botworld* [Teo y Benson, 1995] y *Mobile Robot Simulator* [de Lope Asiaín, 2002]. Sin embargo la mayoría de los simuladores incluyen sensores odométricos (*Stage*, *SHIVA* y *MOBS*) [Vaughan, 2001, Sukthankar et al., 1996, Mobs, 2002], brújula (*Stage*) o incluso tacómetro (*SHIVA*) para su localización relativa. Para la percepción del entorno circundante, los más difundidos son los sensores de ultrasonidos, para robots móviles de interiores, o bien un sensor genérico de largo alcance *Mobile Robot Simulator*. Únicamente el simulador *SHIVA* incluye un sensor de corto-medio-largo alcance que simula un láser de barrido SICK. Algunos disponen de sensores virtuales infrarrojos de corto alcance como (*KEPHERA*) [Khepera, 2002] y *Mobile Robot Simulator*. Los simuladores más sofisticados incluyen cámaras virtuales monocromas (*Stage*, *KEPHERA* y *MOBS*) o incluso sensores para detección de color *SHIVA*. Sin embargo, la documentación que aportan es escasa y proporcionan muy pocos detalles de la implementación, del tipo de errores que simulan y de las limitaciones del simulador. Ahora bien, este último

punto es fundamental, ya que que las limitaciones del modelo deben estar estar claramente explicitadas a fin de pesar el grado de verdad de las conclusiones que se pueden extraer de la simulación con los mismos para una extrapolación adecuada al mundo real. Por esto se realiza una descripción una exhaustiva de los sensores virtuales del robot, así como de las limitaciones de los distintos modelos implementados.

B.3.1 Sensor del nivel de la batería

En DÉDALO el sensor del nivel de carga de la batería proporciona información sobre la cantidad de energía disponible en el robot. El sensor de batería virtual proporciona al robot virtual la medida de la energía disponible para realizar su misión.

- **Modelo del sensor.** En la hipótesis de un consumo constante y de funcionamiento de aproximadamente 4 horas, se ha modelado el nivel de la batería como una función lineal decreciente en el tiempo (B.5). El nivel máximo corresponde al $100\,\%$ y despues de 4 horas de funcionamiento, el nivel disminuye hasta el $87\,\%$; nivel para el cuál la energía no es suficiente para asegurar el buen funcionamiento del sensor de barrido láser.

$$N_{bat} = -0{,}00091 \times t + 100 \tag{B.5}$$

- **Limitaciones del modelo.** Se ha partido de un consumo ideal constante lineal que no se corresponde con la realidad ya que existen picos de consumo, como en el caso del encendido del láser o de la activación de las válvulas.

- **Ejemplo de funcionamiento** En la figura B.4 se muestra la representación gráfica utilizada en la pantalla de datos del simulador para visualizar el nivel de energía disponible, después de 15 minutos de funcionamiento del simulador.

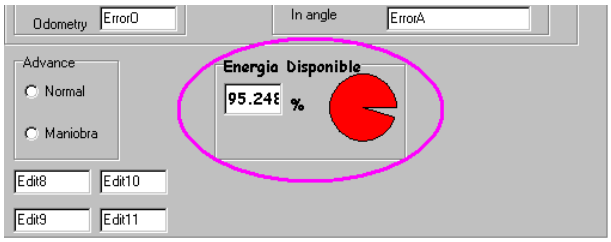


Figura B.4 : Diagrama de la energía disponible

B.3.2 Sensores de posicionamiento relativo: Odómetros

Los odómetros constituyen uno de los tipos de sensores utilizados en la medida de la posición relativa de un vehículo. Estos sistemas disponen de un sensor que calcula la distancia recorrida por el vehículo contando el número de vueltas por segundo de un motor o de una rueda. Son sensores que requieren poco proceso, por ello prácticamente todos los robots móviles los incorporan como estimadores de su posición relativa a la posición inicial.

En el robot DÉDALO el sistema odométrico está compuesto por sensores detectores de metal que detectan el paso de un disco con tornillos que gira con la rueda trasera. Una descripción detallada del sistema está en el capítulo 4. El odómetro virtual implementado en el simulador es un odómetro ideal, en el sentido de que la única fuente de imprecisión que incorpora respecto a la posición real del robot es la resolución finita del sensor. Esta resolución puede ser establecida por el usuario en tiempo de ejecución del programa, figura B.5. Para mejorar el funcionamiento de este sensor virtual habría que modelar detalladamente las posibles fuentes de error e integrarlas en el sensor simulado.

- **Modelo del sensor.** La construcción del sensor virtual parte de las ecuaciones que rigen la estimación de la posición por un odómetro [Borenstein y Feng, 1998]. El principio básico es un conteo del número de vueltas que da cada una de las ruedas, izquierda y derecha. A partir de ellos, se estima tanto la distancia recorrida por el robot como el giro efectuado (B.6).

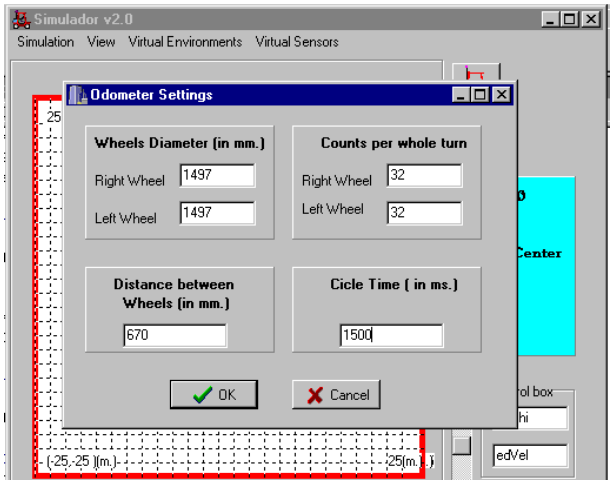


Figura B.5 : Pantalla de configuración del odómetro virtual

$$\begin{aligned} \Delta d_i &= \frac{\Delta_i d_L + \Delta_i d_R}{2} \\ \Delta \theta_i &= \frac{\Delta_i d_R - \Delta_i d_L}{b} \end{aligned} \tag{B.6}$$

donde b es la distancia entre los puntos de apoyo de las ruedas derecha e izquierda y $\Delta_i d_{L/R}$ viene definida por la expresión (B.7).

$$\Delta_i d_{R/L} = N_{i_{R/L}} \frac{\pi d_{R/L}}{n_{R/L}} \tag{B.7}$$

siendo $n_{R/L}$ la resolución del odómetro o número máximo de pasos por vuelta, $d_{R/L}$ el diámetro de la rueda y $N_{i_{R/L}}$ el número de pasos contados en el instante i.

De las ecuaciones anteriores se deduce el algoritmo que define el cambio en las coordenadas del vehículo (B.8) [Borenstein y Feng, 1998]:

$$\begin{aligned} \theta_i &= \theta_{i-1} + \Delta\theta \\ x_i &= x_{i-1} + \Delta s_i \cos\theta_i \\ y_i &= y_{i-1} + \Delta s_i \sin\theta_i \end{aligned} \tag{B.8}$$

El camino para la simulación es en cierto modo opuesto al del sensor real, puesto que la información de la que disponemos (proporcionada por el modelo cinemático del robot en la sección B.1) es de la posición exacta del robot y no del número de vueltas que da cada una de las ruedas.

En esta simulación se recorre de modo inverso el proceso que se lleva a cabo con el robot real, contar vueltas y calcular posición. En simulación se dispone de la posición del robot, sección B.1 y de la resolución del sensor. A partir de ambas se calcula el número de vueltas que hipotéticamente habría contado el odómetro y que se asume como el valor proporcionado por los odómetros. Así:

1. A partir de las coordenadas del robot, proporcionadas por el modelo cinemático, sección B.1, se obtiene la distancia recorrida y el giro efectuado (de modo inmediato) respecto al instante anterior.

$$\Delta s = \sqrt{(\Delta x)^2 + (\Delta y)^2}$$

2. Se calculan los desplazamientos correspondientes a las ruedas derecha e izquierda, respectivamente, para la distancia recorrida y el ángulo girado, obtenidos en el punto anterior. A partir de las ecuaciones que modelan la odometría se obtiene (B.9).

$$\Delta d_{i_L} = |\Delta s - \frac{b}{2}\Delta\theta|$$
$$\Delta d_{i_R} = \Delta s + \frac{b}{2}\Delta\theta \tag{B.9}$$

3. En este momento es cuando se tiene en cuenta la resolución del odómetro (B.10).

$$\Delta_i d_{R/L} = N_{i_{R/L}} \frac{\pi d_{R/L}}{n_{R/L}} \tag{B.10}$$

Al considerar la resolución se obliga a que se verifique (B.11).

$$N_{i_{R/L}} \epsilon \mathbf{N} \tag{B.11}$$

y esto se consigue calculando $N_{i_{R/L}}$ como el entero más próximo a (B.12).

$$N_{i_{R/L}} = \Delta d_{i_{R/L}} \frac{n_{R/L}}{\pi d_{R/L}} \tag{B.12}$$

4. Finalmente se calculan las coordenadas del robot $(x_{i_{odo}}, y_{i_{odo}}, \theta_{i_{odo}})$, que ya no serán las obtenidas geométricamente sino las estimadas por el odómetro virtual, siguiendo las ecuaciones del modelo odométrico (B.6).

■ **Ejemplo de funcionamiento.** En la figura B.6 se muestra en la pantalla de visualización del simulador las trayectorias real y "percibida" por los odómetros virtuales. Los valores de configuración en este caso corresponden a una resolución sensorial de ± 47 (mm).

B.3.3 Brújula

La brújula proporciona la orientación del vehículo respecto al norte magnético de la Tierra. Se trata pues de un sensor de orientación absoluto. Puesto que la medida de la brújula

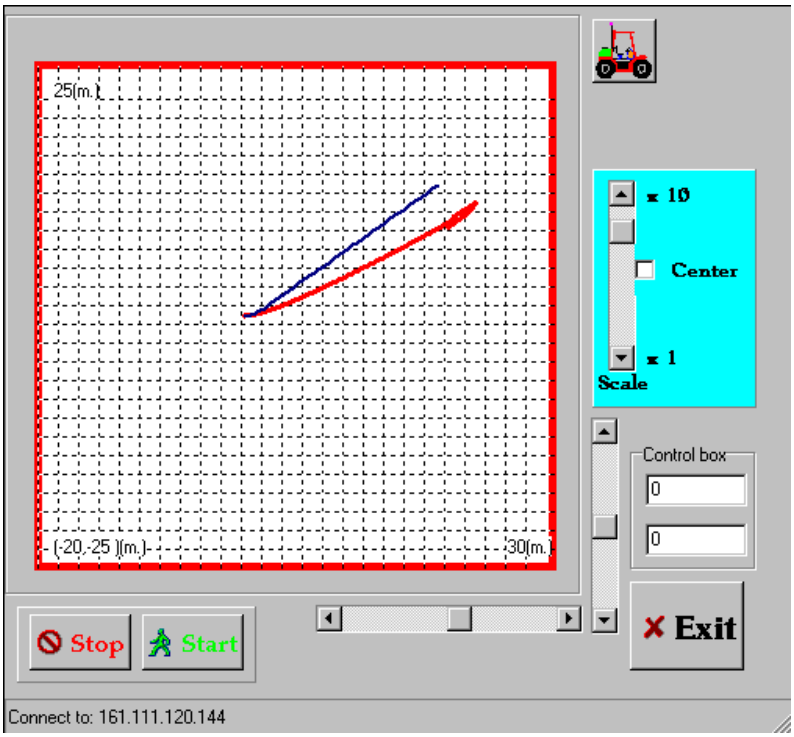


Figura B.6 : Diferencias entre la trayectoria calculada por el modelo cinemático, en rojo, y la calculada por los odómetros virtuales, en azul.

está basada en la detección del campo magnético de la Tierra, en aquellas zonas donde existan interferencias magnéticas la brújula proporcionará una medida errónea. Este último efecto no se ha contemplado en la brújula simulada.

- **Modelo del sensor.** La brújula se ha modelado, figura B.7 a partir de la ecuación (B.13). Puesto que las características de la brújula real son las indicadas en la tabla B.2 [KVH, 2001], el parámetro $Error$ en la ecuación (B.13) toma valores en el intervalo $Error \in [-0{,}5, +0{,}5]\ (deg.)$.

Error (deg.)	Resolución (deg.)
$\pm 0{,}5$	$0{,}1$

Tabla B.2 : Características de la brújula

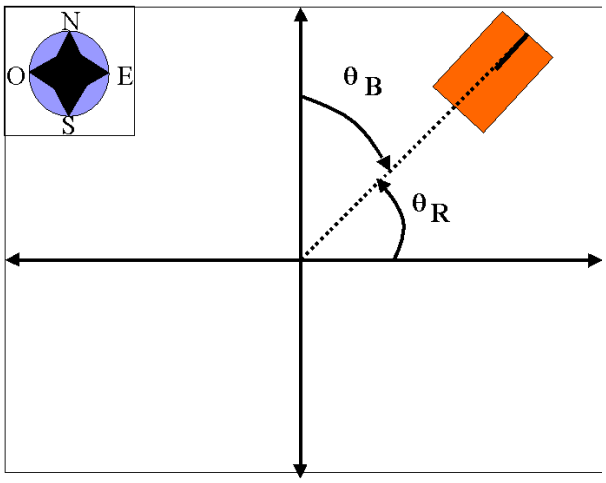


Figura B.7 : Relación entre la orientación del robot y la medida de la brújula

$$\theta_B = \theta_R + F(E),$$
$$\text{donde} F(E) = F_{aleatoria}, F(E) \in [-Error, +Error] \tag{B.13}$$

En esta simulación de la brújula, el usuario puede definir tanto su precisión como su resolución mediante la ventana de configuración del sensor virtual, figura B.8

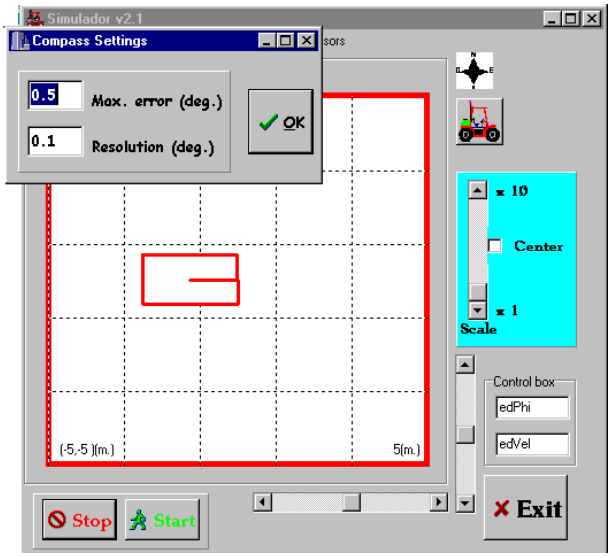


Figura B.8 : Pantalla de configuración de la brújula virtual

- **Limitaciones.** La principal limitación en el modelo de este sensor estriba en no considerar los posibles errores debidas a las interferencias electromagnéticas presentes en el entorno de trabajo.

- **Ejemplo de funcionamiento.** La medida proporcionada por la brújula virtual se muestra en la figura B.9 tanto en representación gráfica como numérica. Se visualiza tanto el valor de la orientación proporcionado por este sensor como el de la orientación obtenida a partir del modelo cinemático.

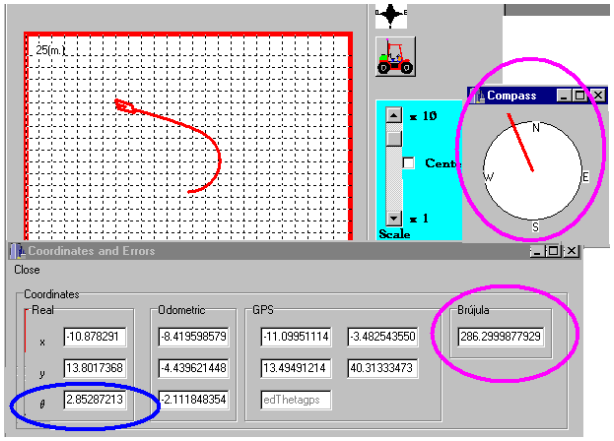


Figura B.9 : Ejemplo de funcionamiento de la brújula virtual

En el ejemplo de la figura B.9 puede verse como la orientación del robot, obtenida del modelo cinemático $\theta_R = 0{,}45729(rad) = 26{,}2(grad.)$ corresponde a la orientación proporcionada por la brújula virtual $\theta_B = 63{,}3(grad.)$ tras una rotación de, $\theta_B = 90 - \theta_R$.

B.3.4 Receptor DGPS

El sensor DGPS es un sensor de posicionamiento absoluto que proporciona su localización en coordenadas angulares terrestres, esto es longitud, latitud y altitud. Son múltiples las causas de error de esta señal. Entre ellas se encuentran los errores multicamino, errores debidos a perturbaciones eléctricas, a oclusiones de la señal por obstáculos entre emisor-receptor y a la visibilidad de sólo una parte de los satélites de la constelación (capítulo 4). Con el simulador se

ha tratado de generar una señal virtual que en cierto modo recoja la problemática y los errores de la señal proveniente de un GPS, sin necesidad de simular el modo de operación del sensor.

■ **Modelo del sensor.** Analizando la señal del detector diferencial DGPS LR3100 [B.Díaz y Ribeiro, 2000], se han clasificado los errores en tres tipos a fin de ser simulados e integrados en el sensor DGPS virtual:

1. **Errores habituales**. Son aquellos que contaminan la señal en cada instante de tiempo. En el receptor LR3100 estos errores vienen definidos por:

$$Error_{h,latitud} \in [-1{,}49, +1{,}49](m.)$$
$$Error_{h,longitud} \in [-0{,}75, +0{,}75](m.)$$

2. **Errores esporádicos**. Son errores muy superiores a los habituales y aparecen con muy poca frecuencia:

$$Error_{e,latitud} \in [-11{,}5, +11{,}5](m.)$$
$$Error_{e,longitud} \in [-5{,}7, +5{,}7](m.)$$

3. **Errores de pérdida total de la señal**. La pérdida total de la señal ocurre en el DGPS con cierta frecuencia y usualmente durante muy breves periodos de tiempo, debido fundamentalmente a oclusiones. Este caso se ha incluido en el sensor virtual, de modo que aleatoriamente y con poca frecuencia el sensor virtual presenta un valor de $Longitud = 0{,}0$, $Latitud = 0{,}0$. Al cabo de un breve intervalo de tiempo, pseudo-aleatorio, se

recupera la señal con errores del tipo 1 y 2.

Una vez que se han integrado los errores se procede al cálculo de las coordenadas (x_{gps}, y_{gps}) en metros, pero el DGPS proporciona la localización del punto con respecto a las coordenadas angulares absolutas de la Tierra. Ello implica una transformación de las coordenadas, que asigna al punto $(0, 0)$ las coordenadas angulares de un edificio del IAI en el campus del CSIC en Arganda del Rey:

$$Longitud_0 = -3{,}48244(deg.)$$

$$Latitud_0 = 40{,}31321(deg.)$$

A partir de las ecuaciones de transformación propuestas en [B.Díaz y Ribeiro, 2000], (B.14), se convierten las coordenadas (x_{gps}, y_{gps}) en $(Longitud_{gps}, Latitud_{gps})$.

$$\Delta\phi = \frac{180}{\pi} \times \frac{y_{gps}}{R},$$

$$\Delta\lambda = \frac{180}{\pi} \times \frac{x_{gps}}{R \times \cos(\Delta\phi)}$$

$$\text{donde:} R = 6378137(m)$$

$$Latitud_{gps} = Latitud_0 + \Delta\phi$$

$$Longitud_{gps} = Longitud_0 + \Delta\lambda \tag{B.14}$$

El modelo de sensor virtual DGPS se sintetiza en el código siguiente:

```
void VirtualGPS(xr,yr){
gps_off=false;
randomize();
if(k==95){ // Errores de tipo 2
```

```
xgps=xr+((2.0/(float)RAND_MAX)*(float)rand()-1)*5.7;
ygps=yr+((2.0/(float)RAND_MAX)*(float)rand()-1)*11.5;
k=0;
}
else { // Errores de tipo 1
xgps=xr+((2.0/(float)RAND_MAX)*(float)rand()-1)*0.75;
ygps=yr+((2.0/(float)RAND_MAX)*(float)rand()-1)*1.49;
k=k++;
}
// Caso 3
if((((float)(rand())/(float)RAND_MAX)>0.95)||(gps_off==true)){
    LatitudGPS=0.0;
    LongitudGPS=0.0;
    gps_off=true;
    if(((float)rand()/(float)RAND_MAX<0.5))gps_off=false;
    }
  else{
   // Paso de a diferencias en Lon/Lat
DeltaLatitud=(180/PI)*ygps/R;
if(cos(DeltaLatitud*180/PI)==0)DeltaLongitud=0;
else  DeltaLongitud=180/PI*xgps/(R*cos(DeltaLatitud));
// 3) Calculo de la Lon/lat
Longitud_gps=Longitud_0+DeltaLongitud;
Latitud_gps=Latitud_0+DeltaLatitud;
}
```

Sin embargo para visualizar la señal de posición del DGPS es necesario invertir de nuevo el proceso de conversión a fin de obtener de nuevo coordenadas locales,(B.15)

$$\Delta\phi = |Latitud_{gps} - Latitud_0|$$

$$\Delta\lambda = |Longitud_{gps} - Longitud_0|$$

$$\Delta y = \Delta\phi \times R \times \frac{180}{\pi}$$

$$\Delta x = \Delta\lambda \times R \times \cos(\Delta\phi) \times \frac{180}{\pi}$$

$$x_{gps} = \begin{cases} -\Delta x & \text{si} Longitud_{gps} < Longitud_0, \\ \Delta x & \text{en otro caso} \end{cases}$$

$$y_{gps} = \begin{cases} -\Delta y & \text{si} Latitud_{gps} < Latitud_0, \\ \Delta y & \text{en otro caso} \end{cases} \tag{B.15}$$

■ **Limitaciones.** El DGPS virtual no emula el funcionamiento de un sensor DGPS, por lo tanto posee algunas limitaciones, aunque la medida que proporciona integra muchas de las características de una señal GPS. Ninguno de los errores que contaminan una señal GPS real, relacionados con la configuración de los satélites, zonas de sombra, errores multi-camino, se han incluido en la simulación del sensor virtual utilizado en la actualidad.

■ **Ejemplo de funcionamiento.** En la figura B.10 se muestra la señal proveniente del GPS virtual, trazo en verde, frente a las medidas de localización procedentes del modelo cinemático (consideradas como la posición real del robot) que aparecen con trazo rojo y las de los odómetros, en trazo azul. Se puede observar que los errores de tipo 1, errores habituales, son bajos y son los responsables de las pequeñas oscilaciones de la medida GPS respecto de la posición real. En determinados momentos el error aumenta, error de tipo 2, aunque únicamente en dos ocasiones en este experimento simulado. También pueden observarse los errores de tipo 3 en los que el sensor virtual "pierde" la señal. En estos casos las coordenadas (x_{gps}, y_{gps}) toman un valor $(0{,}0, 0{,}0)$ y por ello aparecen unos trazos bidireccionales al origen.

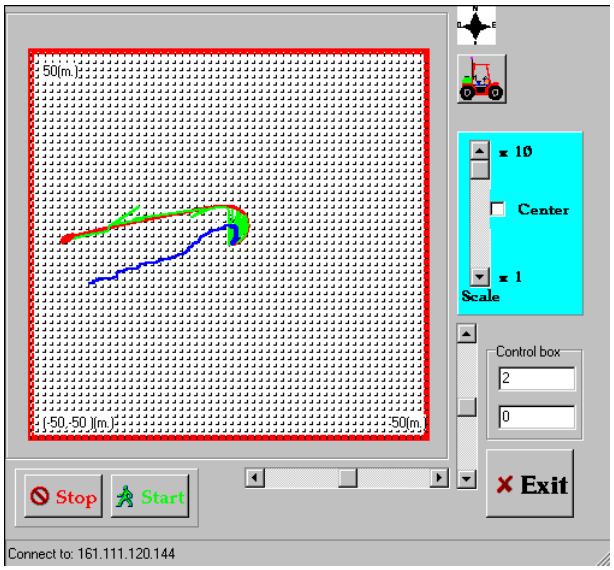


Figura B.10 : Trayectorias de un experimento simulado, en rojo la trayectoria real, en azul la proporcionada por la odometría virtual y en verde la que se obtiene del GPS virtual

B.3.5 Sensor de contacto

El sensor de contacto de los robots ROJO y DÉDALO es un parachoques de goma capaz de detectar las deformaciones generadas por la colisión con un obstáculo. El fundamento físico subyacente consiste en las desviaciones sufridas por el haz de luz conducido por fibra óptica ante la deformación ocasionada en el mismo por un choque. En el simulador, el sensor de contacto se modela como una prolongación física del tractor capaz de detectar si limita o no con alguna región ocupada del espacio virtual.

- **Modelo del sensor.** El modelo del sensor de choques es una línea recta frontal, perpendicular a la dirección de desplazamiento del tractor. Se verifica si cada punto de la línea se halla o no contenido en alguno de los obstáculo pertenecientes al mundo virtual. Si su intersección con los mismos es distinta de cero, el sensor de choques envía una señal de activación.

El sensor de contacto se encuentra en la línea recta que une los puntos 2 y 3, figura B.11. Las ecuaciones que definen los puntos 1,2 y 3 son (B.16), donde b es la distancia entre los puntos de apoyo de las ruedas derecha e izquierda y G la anchura del parachoques que protege el sensor.

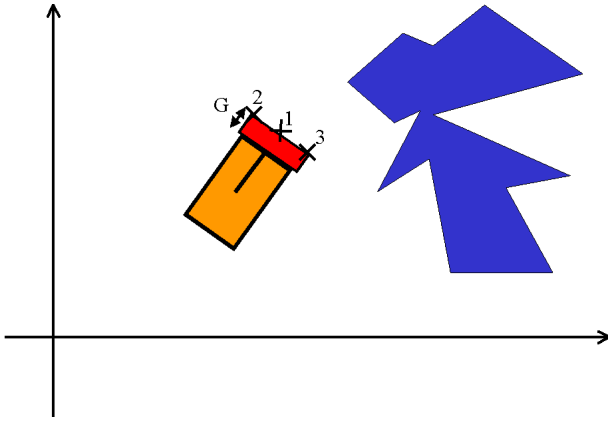


Figura B.11 : Esquema del sensor virtual de contacto

$$\begin{aligned}
&\text{Punto 1,}(x_R + G \cdot \cos(\theta_R), y_R + G \cdot \sin(\theta_R)) \\
&\text{Punto 2,}(x_1 + \frac{b}{2} \cdot \sin(\theta_R), y_1 + \frac{b}{2} \cdot \cos(\theta_R)) \\
&\text{Punto 3,}(x_1 - \frac{b}{2} \cdot \sin(\theta_R), y_1 - \frac{b}{2} \cdot \cos(\theta_R))
\end{aligned} \tag{B.16}$$

y la ecuación de la línea recta que une el punto 2 con el punto 3, viene dada por (B.17).

$$\begin{aligned}
&\text{si } \theta_R \neq 0, \frac{\pi}{2} \\
&y = (\frac{x - x_R}{\tan(\theta_R)} + y_R + G \cdot (\sin(\theta_R) - b \cdot \cos^2(\theta_R)) + \frac{b}{2} \cdot (\cos(\theta_R) - \frac{b}{\cos(\theta_R)}) \\
&\text{si } \theta_R = 0 \\
&x = x_R + G \\
&\text{si } \theta_R = \frac{\pi}{2} \\
&y = y_R + G
\end{aligned} \tag{B.17}$$

El algoritmo que define el comportamiento del sensor virtual de contacto es el siguiente:

```
choque BumperVirtual(x_R,y_R,theta_R){
```

```
Delta_x=2;
x=x_2;
y=y_2;
while((y<y_3)||(choque==true)){
color=Look_up_color(x,y);
if(color==true)choque=true;
else choque=false;
x=x+Delta_x;
y=Recta_bumper(x);
}
}
```

- **Limitaciones.** La primera de las limitaciones del sensor se debe a la falta de continuidad. Al considerar los puntos de la recta del sensor de contacto es imprescindible discretizar. Esta discretización se ha realizado con un incremento $\Delta x = 2(cm.)$. Por esta razón, aunque la probabilidad es muy baja, el sensor de contacto simulado puede no detectar obstáculos que presenten un perfil muy afilado.

  Otra limitación viene del hecho de no modelar los errores que aparecen en el sensor de choques real, como falsas detecciones o ruido. Esto se debe a la alta precisión del sensor de contacto real. No constituye una limitación relevante ya que la probabilidad de aparición es, como en el caso anterior, extremadamente baja.

- **Ejemplo de funcionamiento.** En la siguiente gráfica se presenta, figura B.12, en color marrón la línea recta que define los límites del sensor de contacto.

B.3.6 Láser

El sensor láser instalado en DÉDALO es el modelo LMS210 de SICK, capítulo 4. Es un láser de barrido horizontal de 180 grados de un plano. Proporciona directamente la distancia al

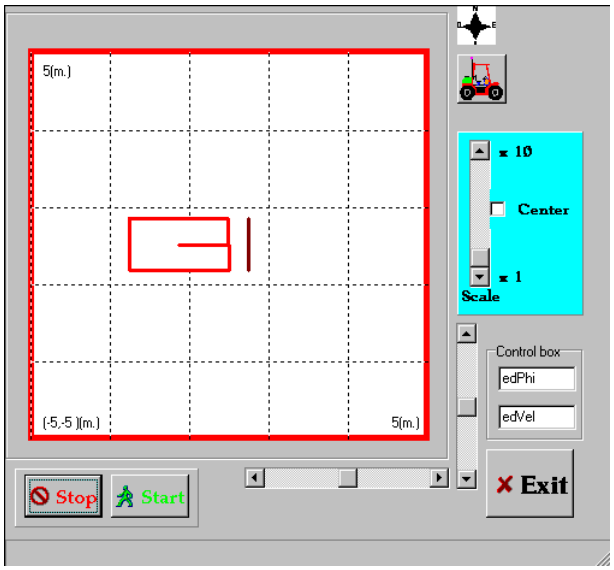


Figura B.12 : Funcionamiento del sensor virtual de contacto

objeto más próximo con una resolución de $0{,}5(grad.)$.

- **Modelo del sensor.** El modelo formulado para el sensor virtual, emula el principio de funcionamiento del sensor real. Para cada posición del sensor se definen las direcciones del barrido a través de 361 semirrectas orientadas, que parten del robot (x_R, y_R, θ_R) cada $0{,}5(grados)$, figura B.13. Para cada una de estas 361 semirrectas se definen incrementos de distancia de $10(cm.)$. Se van recorriendo estos puntos discretos y se comprueba si tienen o no intersección con puntos de algún obstáculo. Si es un punto obstáculo la distancia recorrida desde el sensor hasta la intersección con el obstáculo corresponde con la medida del láser para el ángulo de la pendiente de la recta. Si el punto está vacío se continúa recorriendo la recta hasta que se llega a la distancia máxima, en ese caso la medida es esa distancia que indica que no se ha encontrado ningún objeto para ese ángulo.

```
void LaserVirtual(x_R,y_R,theta_R){
angulo=0;
for(i=0;i<361;i++){
x=0;
delta_x=10;
```

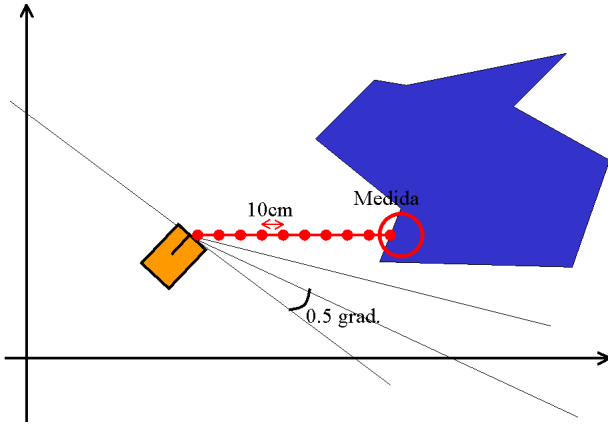


Figura B.13 : Modelo del sensor virtual láser

```
angulo=angulo+0.5;
distancia=0.0;
while((color!=ocupado)OR(distancia<d_maxima)){
y=Recta(angulo,x);
color=Look_up_color(x,y);
distancia=sqrt(x*x+y*y);
x=x+delta_x;
}
MedidaLaser[i]=distancia;
}
}
```

- **Limitaciones.** Igual que en el caso del sensor de contacto virtual la primera de las limitaciones se debe a la discretización de las medidas, indispensable a efectos de velocidad de cómputo. Esta discretización puede dar lugar a medidas ligeramente superiores a las que corresponderían a la realidad. La segunda de las limitaciones surge del hecho de que el modelo de sensor simulado no contempla los errores debidos a la reflectancia de los materiales, ni los errores multirrebote, ni otros tipos de errores menos comunes.

■ **Ejemplo de funcionamiento.** En la figura B.14 se muestra un obstáculo, enfrentado al vehículo, en un mundo virtual. En otra ventana se presenta la representación gráfica de las medidas de distancia a obstáculos, obtenidas por el sensor láser virtual. En esta gráfica la distancia entre semicírculo es de $1(m.)$; las medidas que aparecen sobre el semicírculo de $10(m.)$ corresponden a valores por encima de la máxima distancia detectable. Esto se traduce en que no existe obstáculo en esa dirección.

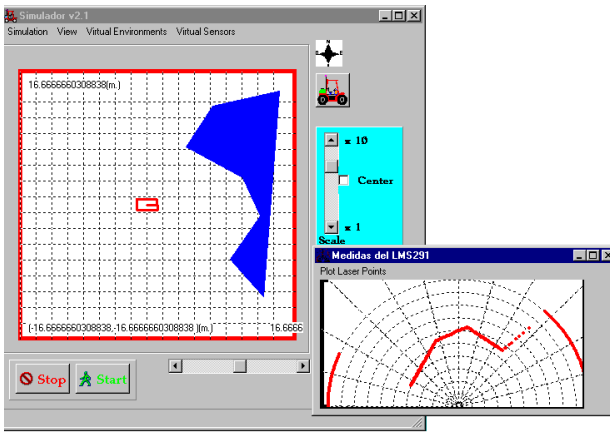


Figura B.14 : Ejemplo de representación de medidas tomadas con el sensor virtual láser.

B.3.7 Sensores de inclinación

El tractor DÉDALO tiene instalados dos sensores de inclinación, uno para medir la inclinación lateral y otro para medir la inclinación frontal del terreno. Esta información es fundamental para generar una parada del tractor en caso extremo con el fin de evitar que vuelque.

■ **Modelo del sensor.** Las variables medidas por los sensores de inclinación reales, han sido definidas en el simulador de la siguiente forma:

- La pendiente lateral es la diferencia de alturas entre los puntos 1 y 2 del tractor, figura B.15 dividida entre la distancia entre los puntos de apoyo de las ruedas izquierda y derecha (B.18).

- La pendiente frontal es la diferencia de alturas de los puntos 3 y 4 entre la distancia entre las ruedas traseras y delanteras L (B.19).

$$Pendiente_{lateral} = \frac{h(P1) - h(P2)}{b} \tag{B.18}$$

$$Pendiente_{frontal} = \frac{h(P3) - h(P4)}{L} \tag{B.19}$$

Para detectar las pendientes lateral y frontal se emplean los valores de la altura, definidos en el modelo del entorno, correspondientes a los puntos del tractor frontal, trasero, lateral derecho y lateral izquierdo. A partir de las diferencias entre las alturas de los puntos frontal y trasero (3 y 4 en la figura B.15) se calcula la pendiente frontal. Y usando las diferencias entre los puntos laterales (1 y 2 en la figura B.15) la pendiente lateral.

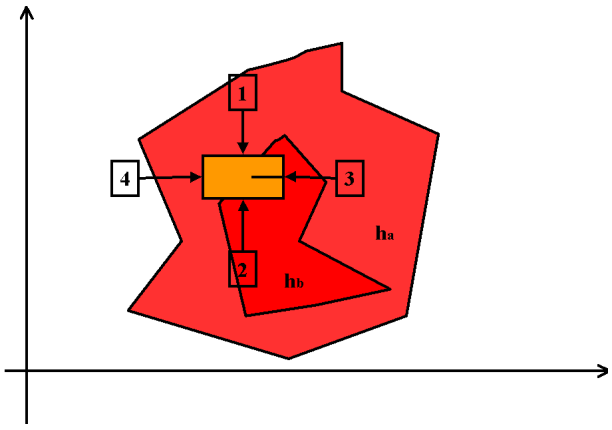


Figura B.15 : Sensores de inclinación virtuales. La diferencia de alturas se representa mediante diferentes colores. La pendiente frontal se calcula a partir de las diferencias entre los puntos 3 y 4 y la lateral entre los puntos 1 y 2

- **Limitaciones.** El modelo de los inclinómetros es ideal en el sentido de que no considera ningún tipo de errores o ruido. Por un lado la forma de definir las altitudes del terreno, que sólo permite saltos discretos en las alturas, da lugar a modelos de inclinación poco realistas. Por otro, dado que el modelo del vehículo es puramente cinemático en este simulador, aunque el robot virtual pueda detectar variaciones de amplitud en los sensores de inclinación virtuales, estos

cambios no se van a incorporar a su modelo de movimiento, pero sin embargo sí se pueden utilizar a nivel de entradas en agentes más complejos que simulen estrategias de navegación próximas a las que debe realizar el vehículo en el mundo real.

- **Ejemplo de funcionamiento.** La ventana de visualización de los sensores de inclinación se visualiza en la figura B.16. Presenta el valor numérico de la inclinación lateral y frontal en grados. En la representación gráfica que se muestra, en forma de triángulo, el ángulo de la base con la hipotenusa corresponde a la inclinación de los sensores de inclinación. La ventana superior de la medida de la pendiente corresponde a la pendiente frontal y la inferior a la lateral.

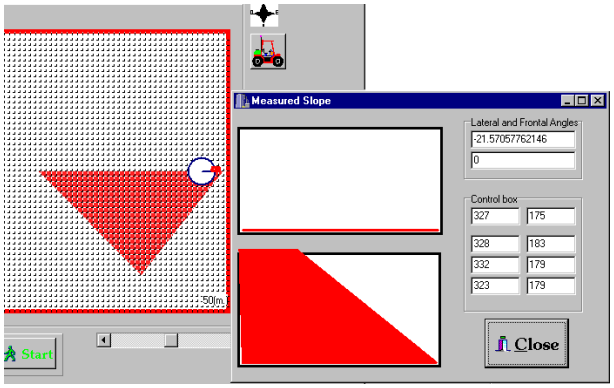


Figura B.16 : Ventanas de visualización de datos de los sensores virtuales de inclinación

B.4 Otras prestaciones del simulador AGRO-SIM

B.4.1 Visualización de datos

La visualización de los diferentes datos, tanto sensoriales como de control, permite seguir la evolución temporal de las distintas sensaciones y acciones de control ejecutadas. Disponer de la información del estado del robot es imprescindible para depurar un programa de control correctamente en el menor tiempo posible. En el simulador los datos sensoriales pueden visualizarse de dos formas: bien numéricamente, mediante la opción *Sensor Data* del

menú *View*, figura B.17 o bien gráficamente. A la visualización gráfica de los datos de los sensores puede accederse por dos vías: 1) de forma directa, seleccionando la opción *Control Console* del menú *View*, que presenta en pantalla todos los formularios para la visualización gráfica de los datos sensoriales, figura B.18; o 2) accediendo a los datos de cada sensor independientemente mediante las diferentes opciones del menú *Virtual Sensors*. Los datos de los sensores de posición pueden verse mediante la visualización de las trayectorias percibidas bajo el menú *Trajectory*.

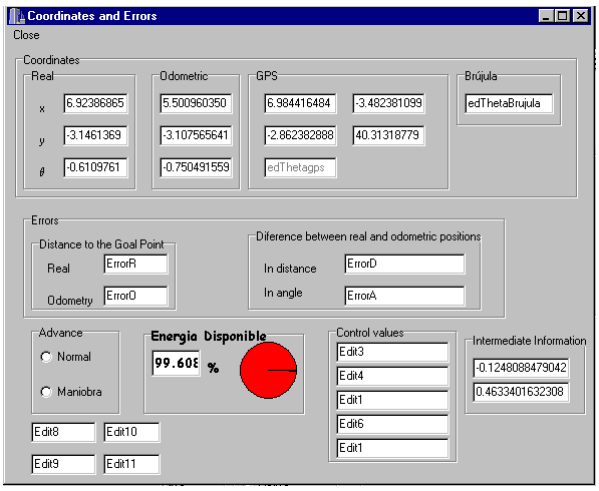


Figura B.17 : Visualización de distintos datos numéricos proporcionados por los sensores virtuales.

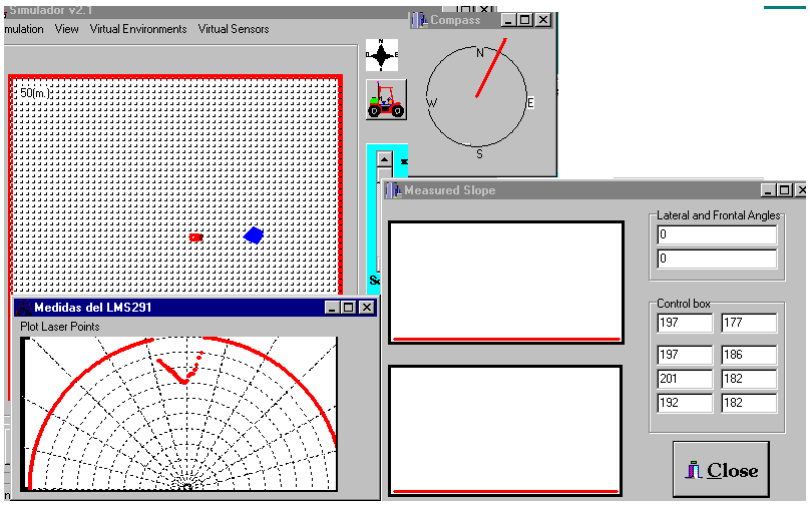


Figura B.18 : Visualización de la representación gráfica de los datos de varios sensores virtuales

B.4.2 Generación de ficheros de datos

También es posible almacenar en ficheros diferentes tipos de datos. Por un lado se puede guardar en un fichero los datos de la trayectoria del robot o bien en mapas de bits imágenes asociadas a las distintas ventanas mostradas por el simulador, durante una simulación. Esta funcionalidad es de gran ayuda para el montaje de vídeos correspondientes con la evolución temporal de la ventana de simulación. Es posible también, almacenar las configuraciones de obstáculos o modelos del mundo para poder utilizarlas, como punto de partida en otras simulaciones. Esto mismo sucede con los planos o ficheros de datos de las altitudes del terreno. Las imágenes del barrido del sensor láser virtual pueden igualmente almacenarse en un fichero, independientemente o bien junto con las posiciones del vehículo y el instante de tiempo en que fueron obtenidas.

B.4.3 Servidor de datos virtuales

De todas las características del simulador, la más relevante es la que permite al simulador para actuar del mismo modo que el programa DÉDALO-Servidor, encargado de la adquisición y servicio de los datos sensorial y del control del robot al más bajo nivel. El simulador puede actuar exactamente igual que el programa, como servidor de datos sensoriales virtuales y puede aceptar consignas de control para el modelo cinemático, equivalente al robot real, con el mismo interfaz que el programa DÉDALO-Servidor. Esto quiere decir que cualquier programa cliente puede conectarse indistintamente y de modo equivalente o bien al robot real o bien al simulador. Esto permite que **el mismo código** de control que se ejecuta en el simulador puede ser probado exactamente en el robot y viceversa, sin tener que cambiar una sola líneas del programa.

Bibliografía

[Albus, 1991] Albus, J. S. (1991). Outline for a Theory of Intelligence. *IEEE Transactions on Systems, Man and Cybernetics*, 21(3):473–509.

[Arbib, 1989] Arbib, M. A. (1989). Schemas and Neural Netwoks for Sixth Generation Computing. *Journal of Parallel and Distributed Computing*, 6:185–216.

[Arbib y Cobas, 1990] Arbib, M. A. y Cobas, A. (1990). Schemas for Prey-Catching in Frog and Toad. En *From Animals to Animats: Proceedings of the first International Conference on Simulation of Adaptive Behavior*, págs. 142–151.

[Arbib y Liaw, 1995] Arbib, M. A. y Liaw, J.-S. (1995). Sensorimotor transformation in the worlds of frogs and robots. *Artificial Intelligence*, (72):53–79.

[Arias-Paz, 2000] Arias-Paz, M. (2000). *Tractores*. CIE Dosaat.

[Arkin, 1987] Arkin, R. C. (1987). Motor Schema-Based Mobile Robot Navigation. *International Journal of Robotics Research*, 8(4):92–112.

[Arkin, 1989] Arkin, R. C. (1989). Towards the Unification of Navigational Planning and Reactive Control. Informe técnico.

[Arkin, 1992] Arkin, R. C. (1992). Homeostatic Control for a Mobile robot: Dynamic Replanning in Hazardous Environment. *Journal of Robotic Systems*, 9(2):197–214.

[Arkin, 1995] Arkin, R. C. (1995). Just what is a robot architecture anyway ? Turing equivalency versus organizing principles. En *AAAI Spring Symposium: Lessons Learned from Implemented Software Architectures for Physical Agents*.

[Arkin, 1998] Arkin, R. C. (1998). *Behaviour-based robotics*. Intelligent robots and autonomous agents. MIT Press, 2a edición.

[Arkin et al., 2000] Arkin, R. C., Ali, K., Weitzenfeld, A., y Cervantes-Pérez, F. (2000). Behavioral models of the praying mantis as a basis for roboti behavior. *Robotics and Autonomous Systems*, 32(1):39–60.

[Arkin y Balch, 1997] Arkin, R. C. y Balch, T. R. (1997). AuRA: Principles and Practice in Review. *Journal of Experimental and Theoretical Artificial Intelligence (JETAI)*, 9(2/3):175–188.

[Arkin et al., 2003] Arkin, R. C., Fujita, M., Takagi, T., y Hasegawa, R. (2003). An ethological and emotional basis for human-robot interaction. *Robotics and Autonomous Systems*, 42:191–201.

[Ashtech, 1996] Ashtech (1996). GPS G12 Board: Interface guide and operating manual. Manual técnico.

[Astrand y Baerveldt, 2002] Astrand, B. y Baerveldt, A.-J. (2002). An Agricultural Mobile Robot with Vision-Based Perception for Mechanical Weed Control. *Autonomous Robots*, 13:21–35.

[Babuska y Verbruggen, 1997] Babuska, R. y Verbruggen, H. B. (1997). *Fuzzy modeling and model-based control for nonlinear systems*, págs. 49–75. En [Jamshidi et al., 1997].

[Baerveldt, 2002] Baerveldt, A.-J. (2002). Guest Editorial: Agricultural Robotics. *Autonomous Robots*, 13:5–7.

[Barrientos et al., 1997] Barrientos, A., Peñin, L., Balaguer, C., y Aracil, R. (1997). *Fundamentos de robotica*. McGraw Hill.

[B.Díaz y Ribeiro, 2000] B.Díaz y Ribeiro, A. (2000). Estudio en estático de la señal de los receptores DGPS LR3100 y LR2100. Informe técnico 10/00b, Instituto de Automática Industrial. Departamento de Sistemas.

[Beeline, 2002] Beeline (2002). Beeline. Agriculture -Broadacre and Row Crop Farming. http://www.beelinenavigator.com/agriculturebody.html (online).

[Beer, 1990] Beer, R. D. (1990). *Intelligence as Adaptive Behavior. An Experiment in Computational Neuroethology*, volumen 6 de *Perspectives in Artificial Intelligence*. Academic Press, Inc.

[Berenji y Khedkar, 1992] Berenji, H. R. y Khedkar, P. (1992). Learning and tuning of fuzzy logic controllers through reinforcements. *IEEE Trans. Neural Networks*, 3(5):724–740.

[Billingsley y Schoenfish, 1995] Billingsley, J. y Schoenfish, M. (1995). Vision and mechatronics applications at the NCEA. En *Proceedings of the fourth IARP workshop on Robotics in Agriculture and the Food Industry*, págs. 101–107, Toulouse,(Francia).

[Bonasso et al., 1997] Bonasso, R. P., Firby, R. J., Gat, E., Kortenkamp, D., Miller, D. P., y Slack, M. G. (1997). Experiences with an Architecture for Intelligent, Reactive Agents. *Journal of Experimental and Theoretical Artificial Intelligence*, 9(2/3):237–256.

[Borenstein y Feng, 1998] Borenstein, J. y Feng, L. (1998). Measurement and correction of systematic odometry errors in mobile robots. *IEEE Transactions on Robotics and Automation*, 12(6):869–880.

[Borenstein y Koren, 1991] Borenstein, J. y Koren, Y. (1991). The Vector Field Histogram-Fast Obstacle Avoidance for Mobile Robots. *IEEE Journal of Robotics and Automation*, 7(3):278–288.

[BRAIN, 2002] BRAIN (2002). Urgent Developments. http://www.brain.go.jp/Urgent/iam_eurgpro.htm (online).

[Brooks, 1991] Brooks, R. (1991). New Approaches to Robotics. *Science*, 235:1227–1232.

[Brooks, 1986] Brooks, R. A. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2(1):14–23.

[Brooks, 1987] Brooks, R. A. (1987). A hardware retargetable distributed layered architecture for mobile robot control. En *Proceedings of the 1987 International Conference on Robotics and Automation*, págs. 106–110, Raleigh-NC. Computer Society Press.

[Cañas y García-Alegre, 1999] Cañas, J. y García-Alegre, M. (1999). Modulated Agents for Autonomous Robot Piloting. En *Proceedings of the 8 Conf. Española para la Inteligencia Artificial (CAEPIA'99)*, págs. 98–106, Murcia (España).

[Cañas, 2003] Cañas, J. M. (2003). *Jerarquía dinámica de esquemas para la generación de comportamiento autónomo*. Tesis doctoral, Escuela Técnica Superior Ingenieros de Telecomunicaciones. Universidad Politécnica de Madrid.

[Cañas y García-Pérez, 2002] Cañas, J. M. y García-Pérez, L. (2002). Construcción de mapas dinámicos. En *Proceedings of the 3th spanish workshop on physical agentes (WAF02)*, págs. 43–54, Murcia (España).

[Caladín et al., 2001] Caladín, L. G., Tornero, J., Montserrat, I., y López, C. G. (2001). Sistema de posicionamiento por visión en líneas de cultivo. En *Primer Congreso Nacional de Ingeniería para la Agricultura y el Medio Rural*, págs. 535–541, Valencia (España).

[Chateau et al., 2000] Chateau, T., Debain, C., Collange, F., Rassoudaine, L., y Alizon, J. (2000). Automatic guidance of agricultural vehicles using a laser sensor. *Computers and Electroncis in Agriculture*, 28:243–257.

[Claas, 2002] Claas (2002). Claas Laser Pilot. http://www.claas.com/sp/home_index.htm (online).

[Crowley, 1985] Crowley, J. L. (1985). Navigation for an Intelligent Mobile Robot. *IEEE Journal of Robotics and Automation*, RA-1(1):31–41.

[de Chile, 2002] de Chile, U. (2002). Controladores: P, I, D, PI, PID. http://cipres.cec.uchile.ch/ iq57a/controladores.htm (online).

[de Lope Asiaín, 2002] de Lope Asiaín, J. (2002). Mobile Robot Simulator. http://jdlope.tripod.com/mrs.html (online).

[Devy et al., 1995] Devy, M., Chatila, R., Fillatreau, P., Lacroix, S., y Nashashibi, F. (1995). On autonomous navigation in a natural environment. *Robotics and Autonomous systems*, (16):5–16.

[Dickmanns, 1993] Dickmanns, E. D. (1993). Vehicles capable of dynamic vision: a new breed of technical beings? *Artifical Intelligenge*, 103:49–06.

[Driankov y Palm, 1998] Driankov, D. y Palm, R. (1998). *Advances in Fuzzy Control*. Physica-Verlag.

[Dudek y Jenkin, 2000] Dudek, G. y Jenkin, M. (2000). *Computational Principles of Mobile Robotics*. Cambridge Universiy Press.

[Edan y Miles, 1994] Edan, Y. y Miles, G. E. (1994). System Engineering of Agricultural Robot Design. *IEEE Transactions on Systems, Man and Cybernetics*, 24(8):1259–1265.

[Edan et al., 2000] Edan, Y., Rogozin, D., Flash, T., y Miles, G. E. (2000). Robotic Melon Harvesting. *IEEE Transactions on Robotics and Automation*, 16(6):831–834.

[Everett, 1995] Everett (1995). *Sensors for Mobile Robots*. John Wiley and Sons.

[Fikes y Nilsson, 1971] Fikes, R. E. y Nilsson, N. J. (1971). STRIPS: A new Approach to the Application of Theorem Proving to Problem Solving. *Artificial Intelligence*, 2:189–208.

[Firby, 1987] Firby, R. J. (1987). An Investigation into Reactive Planning in Complex Domains. En *Proc. of the National Conf. on Artificial Intelligence AAAI*, págs. 202–206, Seattle (USA).

[Firby, 1992] Firby, R. J. (1992). Building Symbolic Primitives with Continuous Control Routines. En *Proc. of the First Int. Conf. on AI Planning Systems*, págs. 62–29, College Park, (USA).

[Firby et al., 1995] Firby, R. J., Kahn, R. E., Prokopowicz, P.Ñ., y Swain, M. J. (1995). An Arquitecture for Vision and Action. En *Proc. of the Int. Joint Conf. on Artificial Intelligence IJCAI*, págs. 72–79.

[Fong y Thorpe, 2001] Fong, T. y Thorpe, C. (2001). Vehicle Teleoperation Interfaces. *Autonomous Robots*, (11):9–18.

[Fox et al., 1997] Fox, D., Burgard, W., y Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robotics and Automation Magazine*, págs. 23–33.

[Fraichard y Garnier, 2001] Fraichard, T. y Garnier, P. (2001). Fuzzy control to drive car-like vehicles. *Robotics and Autonomous Systems*, 34(1):1–22.

[Franklin y Firby, 1997] Franklin, D. y Firby, R. J. (1997). Integrating Range and Object Data for Robot Navigation. En Johnson, W. L. y Hayes-Roth, B., editores, *Proceedings of the First International Conference on Autonomous Agents (Agents'97)*, págs. 185–192, New York. ACM Press.

[García-Alegre y Guinea, 1992] García-Alegre, M. C. y Guinea, D. (1992). Building and arquitecture for a farming robot. En *Bio-Robotics 97. International Workshop on Robotics and Automated Machinery for bio-Productions*, págs. 255–260, Gandia (España).

[García-Alegre et al., 1993] García-Alegre, M. C., Ángela Ribeiro, Gasós, J., y Salido, J. (1993). Optimization of fuzzy behavior-based robots navigation in partially known

industrial environments. En R.Langari, J.Yen, y J.Painter, editores, *Industrial Fuzzy Control Intelligent Systems*. IEEE Press, New York.

[García-Alegre y Recio, 1998] García-Alegre, M. C. y Recio, F. (1998). Basic Visual and Motor Agents for Increasingly Complex Behavior Generation on a Mobile Robot. *Autonomous Robots*, 5:1–10.

[García-Pérez et al., 2000a] García-Pérez, L., Cañas, J. M., García-Alegre, M. C., Yáñez, P., y Guinea, D. (2000a). Fuzzy Control of an electropneumatic actuator. En *Proceedings of the X Congreso Español sobre tecnologías y lógica fuzzy STYLF2000*, págs. 233–978, Sevilla (España).

[García-Pérez y García-Alegre, 2001] García-Pérez, L. y García-Alegre, M. (2001). A Simulation Environment to Test Fuzzy Navigation Strategies. En *10th IEEE International Conference on Fuzzy Systems*, págs. 87–98, Melbourne (Australia).

[García-Pérez et al., 2003] García-Pérez, L., García-Alegre, M., Ángela Ribeiro, y Guinea, D. (2003). Fuzzy control for an approaching orienting maneuver with a car-like vehicle in outdoor environments. En *Proceedings of the spanish Workshop on physical agents*, Alicante (España).

[García-Pérez et al., 2000b] García-Pérez, L., Garcia-Alegre, M. C., Marchant, J., y Hague, T. (2000b). Fuzzy decision system for threshod selection to cluster cauliflower plant blobs from field visual images. En *Proceedings of the World Multiconference on Systemics, Cybernetics and Informatics*, volumen 1, págs. 23–68, Orlando, Florida (USA).

[Garcia-Alegre et al., 1995] Garcia-Alegre, M., P.Bustos, y D.Guinea (1995). Complex behaviour generation on autonomous robots: A case study. En *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, págs. 1729–1734, Vancouver (Canadá).

[Gasós et al., 1991] Gasós, J., García-Alegre, M. C., y García, R. (1991). Fuzzy strategies for

the navigation of autonomous mobile robots. En *Proceedings of the International Fuzzy Engineering Symposium*, págs. 1024–1034, Yokohama (Japón).

[Gasós et al., 1990] Gasós, J., Zuliani, P. D. F., García-Alegre, M. C., y Rosa, R. G. (1990). Environment for the development of fuzzy controllers. En *Proceedings of the IASTED, International Conference on Artificial Intelligence Aplications and Neural Networks. AINN90*, págs. 121–124, Zurich (Suiza).

[Gat, 1992] Gat, E. (1992). Integrating Planning and Reacting in a Heterogeneous Asyncronous Architecture for Controlling Real-World Mobile Robots. En *Proc. of Tenth National Conf. on Artificial Intelligence (AAAI92)*, págs. 809–815.

[Geyer-Schulz, 1998] Geyer-Schulz, A. (1998). *Fuzzy Genetic Algorithms*, págs. 403–409. Kluwer.

[Guinea et al., 1995] Guinea, D., Sánchez, G., Bustos, P., y García-Alegre, M. (1995). A Distributed Architecture for Active Perception in Autonomous Robots. En *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, págs. 1740–1745, Vancouver (Canadá).

[Gutiérrez, 2003] Gutiérrez, D. (2003). Control Fuzzy de un sistema de actuación hidráulico en un tractor agrícola. Proyecto fin de carrera. Universidad Pontificia de Comillas. Escuela Técnia Superior de Ingeniería (ICAI).

[Hagras et al., 1999] Hagras, H., Callacham, V., Colley, M., y Carr-West, M. (1999). A behaviour based hierarchical fuzzy control architecture for agricultural autonomous mobile robots. En *Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation*, págs. 172–177, Viena (Austria).

[Hagras et al., 2002] Hagras, H., Colley, M., Callaghan, V., y Carr-West, M. (2002). Online Learning and Adaptation of Autonomous Mobile Robots for Sustainable Agriculture. *Autonomous Robots*, 13:37–52.

[Hague et al., 1999a] Hague, T., Marchant, J. A., y Tillett, N. D. (1999a). Autonomous Robot Navigation for Precision Horticulture. En *Proceedings of the IEEE International Conference on Robotics and Automation*, págs. 21–29, Alburquerque (USA).

[Hague et al., 1999b] Hague, T., Marchant, J. A., y Tillett, N. D. (1999b). A system for plant scale husbandry. En *Proceedings of the first European Conference on Precision Agriculture*, págs. 635–642, Warwick (Reino Unido).

[Hayes-Roth, 1985] Hayes-Roth, B. (1985). A Blackboard Architecture for Control. *Artificial Intelligence*, 26:251–321.

[Henten et al., 2002] Henten, E. J. V., Hemming, J., Tuijl, B. A. J., Kornet, J. G., Meuleman, J., Bontsema, J., y Os, E. A. V. (2002). An Autonomous Robot for Harvesting Cucumbers in Greenhouses. *Autonomous Robots*, 13:241–258.

[Hergalite, 1996] Hergalite (1996). Hergalite Safety Systems. Instalation and Maintenance Manual. Manual técnico.

[Hirota y (eds.), 1995] Hirota, K. y (eds.), M. S. (1995). *Industrial Applications of Fuzzy Technology in the World*. World Scientific.

[Hofmann-Wellenhof et al., 1992] Hofmann-Wellenhof, B., Lichtenegger, H., y Collins, J. (1992). *GPS. Theory and Practice*. Springer Verlag, Wien- New York, 2a edición.

[Iida et al., 1998] Iida, M., Umeda, M., y Suguri, M. (1998). Automatic follow-up vehicle system for agriculture. En *ASAE paper no.:983112*, Orlando, Florida, USA.

[Institute, 2002] Institute, S. R. (2002). Automated Mushroom Harvesting. http://www.sri.bbsrc.ac.uk/science/rag/mushroom.htm (online).

[Integrinautics, 2002] Integrinautics (2002). AutoFarm. http://www.integrinautics.com/uses/index.html (online).

[J. Mínguez, 2002] J. Mínguez, L. Montano, J. S.-V. (2002). Reactive Navigation for Non-holonomic Robots using the Ego Kinematic Space. En *Proceedings of the International Conference on Robotics and Automation ICRA02*.

[Jahns, 1975] Jahns, G. (1975). Safety demands for agricultural vehicles with automatic steering. En *ASAE paper no.:75-1028*, Davis, California (USA).

[Jahns, 1983] Jahns, G. (1983). Automatic Guidance in Agriculture. A Review. En *ASAE paper no.:83-404*, Weyburn, Saskatchewan (Canadá).

[Jahns, 1997] Jahns, G. (1997). Automatic Guidance of Agricultural Field Machinery. En *Proceedings of the Joint International Conference on Agricultural Enginnering and Technology Exhibition*, págs. 70–79, Bangladehs (India).

[Jamshidi et al., 1997] Jamshidi, M., Titli, A., Zadeh, L., y Boverie, S., editores (1997). *Applications of fuzzy logic: Towards Hih Machine Intelligence Quotient Systems*. Prentice Hall.

[Jang et al., 1997] Jang, J. S. R., Sun, C. T., y Mizutani, E. (1997). *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*. Prentice Hall.

[Jiménez, 1998] Jiménez, A. (1998). *Sistema de reconocimiento y localización de objetos cuasi-esféricos por telemetría láser. Aplicación a la detección automática de frutos para el Agribot*. Tesis doctoral, Universidad Complutense de Madrid. Facultad de C.C. Físicas.

[Jiménez et al., 1999] Jiménez, A. R., Jain, A. K., Ceres, R., y Pons, J. L. (1999). Automatic fruit recognition: a survey and new results using Range/Attenuation images. *Pattern Recognition*, 32:1719–1736.

[Kaplan, 1996] Kaplan, E. D. (1996). *Understanding GPS. Principles and Applications*. Mobile Communications. Artech House, Boston- London.

[Kassler, 2001] Kassler, M. (2001). Agricultural Automation in the new Millennium. *Computers and Electronics in Agriculture*, 30:237–240.

[Keicher y Seufert, 2000] Keicher, R. y Seufert, H. (2000). Automatic guidance for agricultural vehicles in Europe. *Computers and Electroncis in Agriculture*, 25:169–194.

[Khepera, 2002] Khepera (2002). Principles of simulation with Webots 3. http://www.cyberbotics.com/webots/principles.html (online).

[Klir y Folger, 1988] Klir, J. y Folger, T. (1988). *Fuzzy sets uncertainty and information*. Prentice Hall.

[Konolige et al., 1997] Konolige, K., Myers, K. L., Ruspini, E. H., y Saffiotti, A. (1997). The Saphira Architecture: A Design for Autonomy. *Journal of experimental & theoretical artificial intelligence: JETAI*, 9(1):215–235.

[Kortenkamp et al., 1998a] Kortenkamp, D., Bonasso, R. P., y Murphy, R., editores (1998a). *Artificial Intelligence and Mobile Robots*, capítulo 10, págs. 243–275. MIT Press.

[Kortenkamp et al., 1998b] Kortenkamp, D., Bonasso, R. P., y Murphy, R. R. (1998b). *Artificial Intelligence and Mobile Robots*. MIT Press, 1a edición.

[Kropff et al., 1997] Kropff, M., Teng, P., Aggarwal, P., Bouma, J., Bouman, B., Jones, J., y van Laar, H., editores (1997). *Applications of Systems Approaches at the Field Level*. Kluwer Academic.

[KVH, 2001] KVH (2001). Technical Manual. KVH C100 Compass Engine. Manual técnico.

[LASE, 2002a] LASE (2002a). Chrysanthemum cutting sticking robot. http://mama.agr.okoyama-u.ac.jp/lase/chrys.html (online).

[LASE, 2002b] LASE (2002b). Cucumber harvesting robot. http://mama.agr.okoyama-u.ac.jp/lase/cucum.html (online).

[LASE, 2002c] LASE (2002c). Laboratory of Agriculture Systems Engineering (LASE). http://mama.agr.okoyama-u.ac.jp/lase (online).

[LASE, 2002d] LASE (2002d). Tomato and cherry tomato harvesting robot. http://mama.agr.okoyama-u.ac.jp/lase/tomato.html (online).

[Laugier et al., 1999] Laugier, C. T., Fraichard, P., Garnier, I., Paromtchik, y Scheuer, A. (1999). Sensor-Based Control Architecture for a Car-Like Vehicle. *Autonomous Robots*, 6:165–185.

[Lee, 1990] Lee, C. C. (1990). Fuzzy logic in control sytems: Fuzzy logic controller, (Parts I and II). *IEEE Trans. on Systems, Man and Cybernetics*, 20(2):404–435.

[Lindgren et al., 2002] Lindgren, D. R., Hague, T., Smith, P. J. P., y Marchant, J. A. (2002). Relating Torque and Slip in an Odometric Model for an Autonomous Agricultural Vehicle. *Autonomous Robots*, 13:73–86.

[MacFarland, 1990] MacFarland, D. (1990). *The Biology of Behavior -Criteria for Success in Animals and Robots*, volumen 144 de *NATO ASI Series*. Editor Luc Steels.

[MacFarland y Bosser, 1993] MacFarland, D. y Bosser, T. (1993). *Intelligent Behavior in Animals and Robots*. MIT Press.

[Maes, 1989] Maes, P. (1989). The dynamics of action selection. En *Proceedings of the International Joint Conference on Artificial Intelligence*, Detroit. Morgan Kaufmann.

[Maes, 1990a] Maes, P. (1990a). How to do the Right Thing. *Connection Science Journal, Special Issue on Hybrid Systems*, 1:291–323.

[Maes, 1990b] Maes, P. (1990b). Situated agents can have goals. *International Journal on Robotics and Autonomous Systems*, 6(1):49–70.

[Malki et al., 1994] Malki, H. E., Li, H., y Chen, G. (1994). New Design and Stability Analysis of Fuzzy Proportional-Derivative Control Systems. *IEEE Transactions on Fuzzy Systems*, 2(4):245–254.

[Mamdani y Assilian, 1975] Mamdani, E. y Assilian, M. (1975). An experiment in linguistica synthesis with a fuzzy logic controller. *Int. Journal Man-Machine Studies*, 7:1–13.

[Mamdani, 1974] Mamdani, E. H. (1974). Applications of fuzzy algorithms for simple dynamic plant. *Proceedings of the IEEE*, 121(12):1585–1588.

[Mamdani, 1993] Mamdani, E. H. (1993). Twenty years of Fuzzy Control: Experiences Gained and Lessons Learnt. En *Proc. of the IEEE Fuzzy Conference and Neural Networks*, págs. 339–344, San Francisco (USA).

[Mandow et al., 1996] Mandow, A., Gómez-de Gabriel, J. M., Martínez, J. L., Muñoz, V., Ollero, A., y García-Cerezo, A. (1996). The Autonomous Mobile Robot AURORA for Greenhouse Operation. *IEEE Robotics and Automation Magazine*, págs. 19–28.

[Manikonda et al., 1995] Manikonda, V., Hendler, J. A., y Krishnaprasad, P. S. (1995). Formalizing Behavior-based Planning for Nonholonomic Robots. En *IJCAI*, págs. 142–149.

[Mann et al., 1599] Mann, G. K., Hu, B.-G., y Gosine, R. G. (1599). Analysis of Direct Action Fuzzy PID Controller Structures. *IEEE Transactions on Systems, Man and Cybernetics- Part B: Cybernetics*, 29(3):371– 388.

[Margaliot y Langholz, 2000] Margaliot, M. y Langholz, G. (2000). *New approaches to fuzzy modeling and control. Design and analysis*, volumen 38 de *Machine Perception Artificial Intelligence*. World Scientific.

[Mataric, 1991] Mataric, M. J. (1991). Navigation with a rat brain: a neurobiologically-inspired model for robot spatial representation. En Meyer, J.-A. y Wilson, S., editores, *From Animals to Animats*, págs. 169–175. MIT Press.

[Mataric, 1992a] Mataric, M. J. (1992a). Behavior-Based Control: Main Properties and Implications. En *Proceedings of the IEEE Conference on Robotics and Automation, Workshop on Architectures for Intelligent Control Systems*, págs. 46–54, Nice (Francia).

[Mataric, 1992b] Mataric, M. J. (1992b). Integration of Representation Into Goal-Driven Behavior-Based Robots. *IEEE Transactions on Robotics and Automation*, 8(3):304–312.

[Mataric, 1994] Mataric, M. J. (1994). *Interaction and Intelligent Behavior*. Tesis doctoral, Massachussetts Institute of Technology.

[Mataric, 2001] Mataric, M. J. (2001). Learning in beahavior-based multi-robot systems: policies, models and other agents. *Cognitive Systems Reseach*, 2(1):81–93.

[Matellán y Borrajo, 1998] Matellán, V. y Borrajo, D. (1998). ABC^2: an architecture for intelligent autonomous systems. En *Proceedings of the 3rd IFAC Symposium on Intelligent Autonomous Vehicles IAV'98*, págs. 57–61, Madrid.

[Meystel, 1987] Meystel, A. (1987). Plannig/Control architectures for master dependet autonomous systems with nonhomogeneus knowledge representation. En *Proceedings of the IEEE International Symposium on Intelligent Control*, págs. 31–41, Philadelphia (USA).

[Meystel, 2003] Meystel, A. M. (2003). Multiresolutional Hierarchical Decision Support Systems. *IEEE Transactions on Systems, Man and Cybernetics- Part C: Applications and Reviews*, 33(1):86–101.

[Minsky, 1985] Minsky, M. (1985). *The society of mind*. Touchstone.

[Miura et al., 1999] Miura, J., Uozumi, H., y Shirai, Y. (1999). Mobile Robot Motion Planning Considering the Motion Uncertainty of Moving Obstacles. En *Proc. 1999 IEEE Int. Conf. on Systems, Man and Cybernetics*, volumen 4, págs. 692–698, Tokyo (Japón).

[Mobs, 2002]

Mobs (2002). MOBS- Mobile Robot Simulator. http://www.ee.uwa.edu.au/ braunl/mobs/ (online).

[Monserrat, 1998] Monserrat, J. (1998). *La percepción visual*. Biblioteca Nueva, Madrid, 1a edición.

[Montgomery et al., 1995] Montgomery, J., Fagg, A., y Bekey, G. (1995). The USC AFV-I. A behavior-based entry in the International Aerial Robotics Competition. *Expert Intelligent Systems and their applications*, págs. 16–22.

[Mudi y Pal, 1999] Mudi, R. K. y Pal, N. R. (1999). A Robust Self-Tuning Scheme for PI- and PD-Type Fuzzy Controllers. *IEEE Transactions on Fuzzy Systems*, 7(0):2–16.

[Murphy, 2000] Murphy, R. R. (2000). *Introduction to AI Robotics*. Intelligent Robots and Autonomous Agents. MIT Press, 1a edición.

[Murrieta-Cid et al., 2002] Murrieta-Cid, R., Parra, C., y Devy, M. (2002). Visual Navigation in Natural Environments: From Range and Color Data to a Landmark-Based Model. *Autonomous Robots*, 13(1, pages =).

[Nehmzow y Owen, 2000] Nehmzow, U. y Owen, C. (2000). Robot navigation in the real world: Experiments with Manchester's FortyTwo in unmodified, large environments. *Robotics and Autonomous systems*, 33:223–242.

[Noguchi y Terano, 1997] Noguchi, N. y Terano, H. (1997). Path planning of an agricultural mobile robot by neural network and genetic algorithm. *Computers and Electronics in Agriculture*, 18:187–204.

[Ogata, 1990] Ogata, K. (1990). *Modern Control Engineering*. Prentice Hall.

[Ollero et al., 1997] Ollero, A., García-Cerezo, A., Martínez, J. L., y Mandow, A. (1997). Fuzzy tracking methods for mobile robots. En Jamshidi, M., Titli, A., Zadeh, L., y Boverie, S., editores, *Applications of fuzzy logic: Towards high machine intelligence quotient systems*. Prentice-Hall, New Jersey.

[Ollis, 1997] Ollis, M. (1997). *Perception Algorithms for a Harvesting Robot*. Tesis doctoral, The Robotics Institute. Canergie Mellon University, Pittsbrugh, PA 15213.

[Parker, 1998] Parker, L. (1998). ALLIANCE: An architecture for fault-tolerant multi-robot cooperation. *IEEE Transactions on Robotics and Automation*, 14(2):220–240.

[Paromtchik et al., 1997] Paromtchik, I., Garnier, P., y Laugier, C. (1997). Autonomous maneuvers of a nonholonomic vehicle. En *Int. Symp. on Experimental Robotics*, Barcelona (España).

[Payton et al., 1990a] Payton, D. W., Rosenblatt, J. K., y Keirsey, D. M. (1990a). Plan guided reaction. *IEEE Transactions on Systems Man and Cybernetics*, 20(6):1370–1382.

[Payton et al., 1990b] Payton, D. W., Rosenblatt, J. K., y Keirsey, D. M. (1990b). Plan guided reaction. *IEEE Transactions on Systems, Man and Cybernetics*, 20(6):1370–1382.

[Pilarski et al., 2002] Pilarski, T., Happold, M., Pangels, H., Ollis, M., Fitzpatrick, K., y Stentz, A. (2002). The Demeter System for Automated Harvesting. *Autonomous Robots*, 13:9–20.

[Pozo-Ruz, 2001] Pozo-Ruz, A. (2001). *Sistema sensorial para control y localización de vehículos en exteriores*. Tesis doctoral, Universidad de Málaga.

[Pozo-Ruz et al., 2000] Pozo-Ruz, A., Ribeiro, A., García-Alegre, M. C., García-Pérez, L., Guinea, D., y Sandoval, F. (2000). Localización de vehículos. Fusión de medidas GPS y odometría. *Mundo Electronico*, 309(1):46–54.

[Prassler et al., 2000] Prassler, E., Scholz, J., y Elfes, A. (2000). Tracking multiple Moving Objects for Real-Time Robot Navigation. *Autonomous Robots*, (8):105–116.

[Quinn y Shute, 1996] Quinn, B. y Shute, D. (1996). *Windows Sockets Network Programming*. Adison Wesley.

[Ribeiro, 1991] Ribeiro, A. (1991). Inteligencia artificial distribuida, agentes benévolos. *Mundo Electrónico*, 214:50–59.

[Ribeiro et al., 2003] Ribeiro, A., García-Pérez, L., García-Alegre, M. C., y Guinea, D. (2003). A friendly man-machine visualisation agent for remote control of an autonomous tractor GPS guided. En *Proceedings of the Europeran Conference on Precision Agriculture (4ECPA)*, págs. 541–542, Berlin (Alemania).

[Rodríguez et al., 1994] Rodríguez, F. J., Mazo, M., Urueña, J., y Sotelo, M. (1994). Guiado de robots móviles por carreteras. *EUROFACH Electrónica*, págs. 60–74.

[Rorabaugh, 1995] Rorabaugh, B. (1995). *Mechanical devices for the electronics experimenter*. Library of Congress Cataloging-in-Publication Data. TAB Books.

[Rosenblatt, 1997] Rosenblatt, J. (1997). The Distributed Architecture for Mobile Navigation. *Journal of Experimental and Theoretical Artificial Intelligence*, 9(2/3):339–360.

[Rosenblatt, 1995] Rosenblatt, J. K. (1995). DAMN: A Distributed Architecture for Mobile Navigation. En *Proc. of the AAAI Spring Symp. on Lessons Learned from Implememted Software Architectures for Physical Agents*, Stanford, CA.

[Rosenblatt, 2000] Rosenblatt, J. K. (2000). Optimal selection of Uncertain Actions by Maximazing Expected Utility. *Autonomous Robots*, 9:17–25.

[Rosenblatt y Payton, 1989] Rosenblatt, J. K. y Payton, D. W. (1989). A fine-grained alternative to the subsumption architecture for mobile robot control. En *Proceedings of IEEE International Joint Conference on Neural Networks*, volumen 2, págs. 317–323, Washington D.C. (USA).

[Saffiotti et al., 1993] Saffiotti, A., Ruspini, E. H., y Konolige, K. (1993). Blending Reactivity and Goal-Directedmess in a Fuzzy Controller. En *Proceedings of Fuzzy-IEEE Conference*, págs. 134–139, San Francisco (USA).

[Sanchiz et al., 1998] Sanchiz, J. M., Pla, F., y Marchant, J. A. (1998). An approach to the vision tasks involved in an autonomous crop protection vehicle. *Enginnering Applications of Artificial Intelligence*, 11:175–187.

[Saridis, 1985] Saridis, G. (1985). Foundations of the theory of intelligent controls. En *Proceedings of the IEEE Workshop on Intelligent Control*, págs. 23–28, New York (USA).

[Scarlett, 2001] Scarlett, A. J. (2001). Integrated control of agricultural tractors and implements: a review of potencial opportunities relating to cultivation and crop establishment machinery. *Computers and Electroncis in Agriculture*, 30:167–191.

[Scheuer y Fraichard, 1996] Scheuer, A. y Fraichard, T. (1996). Planning Continuous-Curvature Paths for Car-Like Robots. En *Proceedings of the International Conference on Intelligent Robots and Systems*, volumen 3, págs. 1304–1311, Osaka, Japón.

[Simmons, 1994] Simmons, R. (1994). Structured control for autonomous robots. *IEEE Transactions on Robotics and Automation*, 10(1):34–43.

[Simmons, 1996] Simmons, R. (1996). The curvature-velocity method for local obstacle avoidance. En *Proceedings of the 1996 International Conference on Robotics and Automation*, Minneapolis-MN.

[Simmons y Apfelbaum, 1998] Simmons, R. y Apfelbaum, D. (1998). A Task escription Language for Robot Control. En *Proceedings of Conference on Intelligent Robot and Systems (IROS)*, Victoria (Canadá).

[Simmons et al., 1997] Simmons, R., Goodwin, R., Zita Haigh, K., Koening, S., y O'Sullivan, J. (1997). A layered architecture for office delivery robots. En *Proceedings of the ACM International Conference on Autonomous Agents*, págs. 245–252, Marina del Rey (USA).

[Slaughter et al., 2000] Slaughter, D., Giles, D. K., Lamm, R. D., y Lee, W. S. (2000). Robotic Weed Control Systems for California Row Crops. En *Proceedings of the European Conference on Agricultural Enginering (EurAgEng)*, Warwick, Reino Unido.

[Spiegel y Abellanas, 1988] Spiegel, M. R. y Abellanas, L. (1988). *Fórmulas y tablas de la matemática aplicada*. Schaum. McGraw-Hill, Madrid.

[Stentz et al., 2002] Stentz, A., Dima, C., Wellington, C., Herman, H., y Stager, D. (2002). A System for Semi-Autonomous Tractor Operations. *Autonomous Robots*, 13:83–104.

[Stocker, 1998] Stocker, C. (1998). The search for life in Mars: The role of rovers. *Journal of geophysical research*, 103:28557–24575.

[Stoll, 2003] Stoll, A. (2003). Automatic operation planning for GPS-guided machinery. En *Proceedings of the Europeran Conference on Precision Agriculture (4ECPA)*, págs. 657–664, Berlin, Alemania.

[Sttoytchev y Arkin, 2001] Sttoytchev, A. y Arkin, R. C. (2001). Combining Deliberation, Reactivity and Motivation in the Contex of a Behavior-Based pobot Architecture. En *Proc. of the IEEE International Symposium on Computational Intelligence in Robotics and Automation*, Bariff (Canadá).

[Sugeno, 1987] Sugeno, M. (1987). An Introductory Survey of Fuzzy Control. *Information Science*, 36:59–83.

[Sugeno y Nguyen, 1994] Sugeno, M. y Nguyen, H. T. (1994). *Fuzzy modeling and control: selected works of M. Sugeno*. CRC Press, 1a edición.

[Sukthankar et al., 1996] Sukthankar, R., Hancock, J., Pomerleau, D., y Thorpe, C. (1996). A simulation and design system for tactical driving algorithms. En *Proceedings of AI, Simulation and Planning in High Autonomy Systems*.

[Takagi y Sugeno, 1985] Takagi, T. y Sugeno, M. (1985). Fuzzy identification of systems and its applications to modeling and control. *IEEE Trans. System, Man and Cybernetics*, 15(1):116–132.

[Technologies, 1998] Technologies, L. (1998). *Lucent Technologies, IEEE 802.11 WaveLan PC Card User's Guide*.

[Tecuci, 1998] Tecuci, G. (1998). *Building Intelligent Agents. (An Apprenticeship Multistrategy learning theory, methodology, tool and case study*. Academic Press, San Diego, USA.

[Teo y Benson, 1995] Teo, P. C.-S. y Benson, S. (1995). *Botworld 3.2*.

[Terano et al., 1994] Terano, T., Asai, K., y Sugeno, M. (1994). *Applied Fuzzy Systems*. AP Profesional.

[Thuilot et al., 2002] Thuilot, B., Cariou, C., Martinet, P., y Berducat, M. (2002). Automatic Guidance of a Farm Tractor Relying on a single CP-DGPS. *Autonomous Robots*, 13:53–71.

[Tillet et al., 1993] Tillet, N. D., Marchant, J. A., y Hague, T. (1993). Autonomous Plant Scale Crop Protection. En *Proceedings of the AGENG 97*, Madrid (España).

[Tinbergen, 1950] Tinbergen, N. (1950). The hierarchical organization of nervous mechanims underlying instinctive behaviour. *Symposia of the Society for Experimental Biology*, 4:305–312.

[Torii, 2000] Torii, T. (2000). Research in autonomous agriculture vehicles in Japón. *Computers and Electroncis in Agriculture*, 25:133–153.

[Torrance, 1992] Torrance, M. (1992). The case for a realistic mobile robot simulator. En *Working Notes of the AAAI Fall Symposium on Applications of Artificial Intelligence to RealWorld Autonomous Mobile Robots*.

[Trevelyan, 1997] Trevelyan, J. (1997). Simplifying robotics- A challenge for research. *Robotdcs and Autonomous Systems*, 21:207–220.

[Trimble, 2002] Trimble (2002). AgGPS AutoPilot System. Technical Notes. http://www.trimble.com/aggps_autopilot.html (online).

[Tyrrel, 1993] Tyrrel, T. (1993). *Computational Mechanisms for Action Selection*. Tesis doctoral, University of Edimburgh.

[Vaughan, 2001] Vaughan, R. T. (2001). *Stage: a multiple robot simulator Version 0.8c User Manual*.

[Vogel y Angermann, 1977] Vogel, G. y Angermann, H. (1977). *Atlas de biología*. Ediciones Omega.

[Volpe et al., 2001] Volpe, R., Nesnas, I., Estlin, T., Mutz, D., Petras, R., y Das, H. (2001). The claraty architecture for robotic autonomy. En *Proceedings of the 2001 IEEE Aerospace Conference*, Big Sky, Montana (USA).

[Vázquez, 2002] Vázquez, C. (2002). *Arquitectura fractal de especialistas para robots móviles autónomos*. Tesis doctoral, Universidade de Santiago de Compostela.

[Wang, 1994] Wang, L. X. (1994). *Adaptive Fuzzy Systems and Control*. Prentice Hall.

[Whittaker et al., 1998] Whittaker, W., Bapna, D., Macmone, M. W., y Rollins, E. (1998). Atacama Desert Treck: A Planetary Analog Field Experiment. Informe técnico, Carnegie Mellow University.

[Wilson, 2000] Wilson, J. (2000). Guidance of agricultural vehicles -a historical perspective. *Computers and Electroncis in Agriculture*, 25:3–9.

[Yager y Filev, 1991] Yager, R. y Filev, D. (1991). *Essentials of fuzzy modeling and control*. John Wiley and Sons Inc.

[Yen y Pfluger, 1995] Yen, J. y Pfluger, N. (1995). A Fuzzy Logic Based Extension to Payton and Rosenblatts Command Fusion Method for Mobile Robot Navigation. *IEEE Transactions on Bystems, Man and Cybernetics*, 25(6):971–978.

[Zadeh, 1973] Zadeh, L. A. (1973). Outline of a new approach to the analysis of complex systems and decision proceses. *IEEE Trans. on Systems, Man and Cybernetics*, 1:28–44.