

I. MEMORIA:

1. Introducción

De forma general, la robótica se define como:

El conjunto de conocimientos teóricos y prácticos que permiten concebir, realizar y automatizar sistemas basados en estructuras mecánicas poliarticuladas, dotados de un determinado grado de "inteligencia" y destinados a la producción industrial o al sustitución del hombre en muy diversas tareas. Un sistema robótico puede describirse, como "Aquel que es capaz de recibir información, de comprender su entorno a través del empleo de modelos, de formular y de ejecutar planes, y de controlar o supervisar su operación". La robótica es esencialmente pluridisciplinaria y se apoya en gran medida en los progresos de la microelectrónica y de la informática, así como en los de nuevas disciplinas tales como el reconocimiento de patrones y de inteligencia artificial.

Este proyecto se centra en el desarrollo de una plataforma para el desarrollo de algoritmos en robótica móvil. Siendo la robótica móvil un área de la robótica dedicada a la navegación en entornos tanto estructurados como no estructurados, es decir, el robot debe realizar sus tareas en un entorno dinámico, cambiante y en muchos casos desconocido.

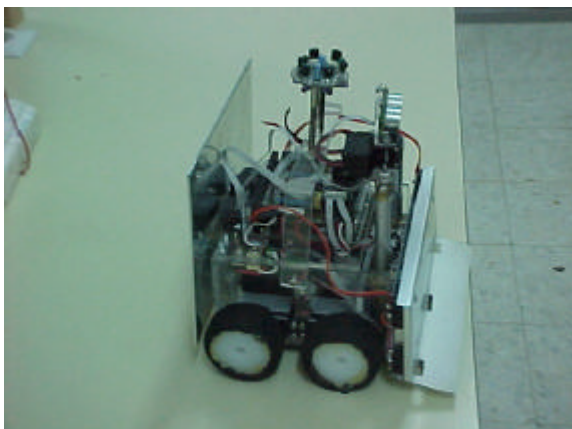
1.1. Experiencia en robótica móvil

Aunque la robótica no está contemplada como una especialización de esta carrera universitaria está muy ligada con ella. La robótica aúna disciplinas como mecánica, electrónica, comunicaciones e informática. Sin embargo aún no se considera una carrera por derecho propio porque su desarrollo es muy reciente y su uso raramente sobrepasa los laboratorios, salvo en el caso de brazos robóticos para montaje de automóviles y algunas aplicaciones de visión artificial. Sin embargo todo apunta a que antes de mediados este siglo su uso estará ampliamente extendido y bien merecerá su estudio como carrera universitaria.

En el caso de los autores de este proyecto la robótica ha estado íntimamente ligada a ellos durante los últimos tres años, constituyendo además de un hobby una pasión, que entre otros beneficios les reportó la posibilidad de poner en práctica la mayoría de los conocimientos adquiridos en sus estudios así como la adquisición de otros nuevos.

Durante este tiempo han construido un total de 9 robots móviles incluyendo el de este proyecto, algunos en colaboración con otros estudiantes y otros por separado. A continuación se describen los más significativos.

VULCANO



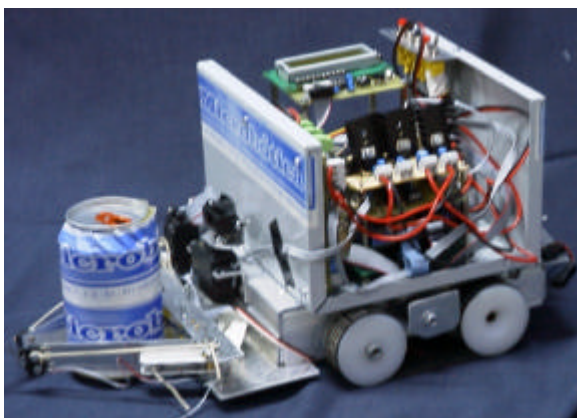
Autores: Jesus Donate (EUITT), Juan María Castro (EUITT), Iñaki Navarro (ETSIT), Marcos Larena (EUITT), Alvaro Gutierrez (ETSIT), Daniel Amor (ETSIT).

Sistema de Control: FPGA

Usos: Sumo, recogida y selección de objetos

Inicialmente fue diseñado para combatir en SUMO^[1], uno de los juegos de robots más populares en los campeonatos universitarios. Ha sufrido varias remodelaciones y participado en varios campeonatos, actualmente está dedicado a la recolección y discriminación de objetos con la electrónica generada en este proyecto. Entre sus principales características se encuentra su pala con movimiento vertical, que actualmente monta unos dedos que le permiten atrapar objetos de pequeño tamaño. Tiene bastante fuerza motriz y cuenta con sensores para medir distancias mediante ultrasonidos e infrarrojos, sensores de suelo para distinguir entre colores claros y oscuros. Además se está pensando en la colocación de un lector/grabador de transponders para la selección de objetos.

TITAN



Autores: Iñaki Navarro (ETSIT), Alvaro Gutierrez (ETSIT), Jesus Donate (EUITT), Joaquín Yun (Informática UCM).

Sistema de Control: Control distribuido, dos 68HC11 y un PIC16F876

Usos: Recolector de Latas

Inicialmente fue concebido como robot luchador de sumo, pero tras una corta carrera fue remodelado para realizar recolección de latas esparcidas por un entorno cerrado, para ello contaba con varios microprocesadores trabajando en paralelo. Su desarrollo fue complejo y permitió a sus creadores indagar en la problemática del control distribuido. En la actualidad está pendiente de una remodelación electrónica y estructural.

BACO



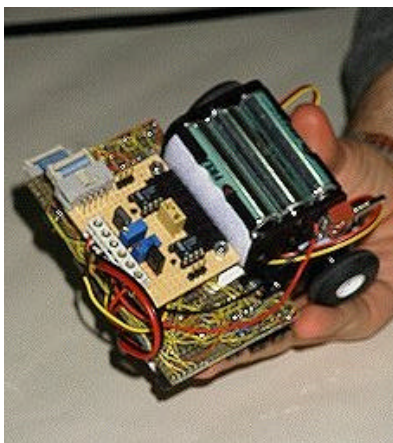
Autores: Juan María Castro (EUITT), Antonio Gil (EUITT), Alberto Sánchez (EUITT).

Sistema de Control: 68hc11

Usos: Robot rastreador, posee una radio que le permite enviar datos a un PC para que reconstruya su trayectoria. Puede ser tele comandado.

Uno de los trabajos más interesantes. Además de un excelente acabado, posee algunas peculiaridades como un sistema de control para el seguimiento de líneas con una estrategia basada en el seguimiento de líneas por su intensidad de reflejo, con capacidad para calibrarse sobre el terreno. Además cuenta con una fuente conmutada para elevar la tensión de los motores y acrecentar de este modo el rendimiento. Por otro lado cuenta con un enlace por radio que le permitía enviar los datos de su movimiento a un software específico, pudiendo este representar el camino seguido por el robot.

ODIN

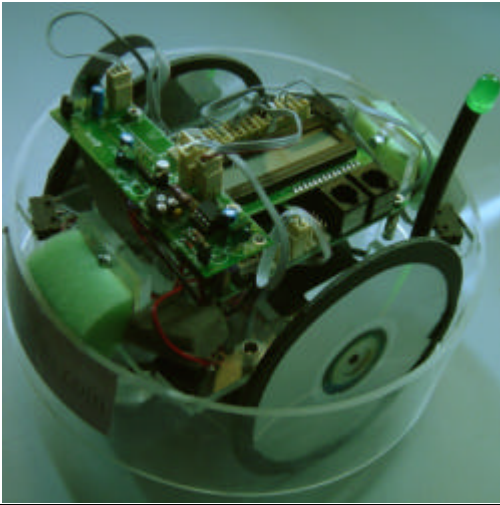


Autores: Juan María Castro (EUITT)

Sistema de Control: Analógico

Usos: Seguidor de líneas

Este trabajo es particularmente interesante por tratarse de un robot totalmente analógico. Solamente sirve para seguimiento de líneas, pero lo hace de un modo impecable, utilizando para ello un total de 10 sensores de infrarrojos. Este robot pudo ganar el campeonato nacional de rastreadores de Barcelona, pero la suerte en semifinales no le acompañó, quedando en un honroso 4º puesto. Sin embargo fue la estrella durante todo el campeonato, por ser analógico, su suave movimiento y su reducido tamaño.



P

Autores: Juan María Castro (EUITT), Jesús Donate (EUITT), Marcos Lerena (EUITT), Iñaki Navarro (ETSIT), Alvaro Gutierrez (ETSIT), Daniel Amor (ETSIT)

Sistema de Control: PIC16F877

Usos: Aprendizaje y enseñanza.

Este robot fue diseñado para realizar talleres de robótica entre alumnos de universidades. Su fácil manejo y amplias posibilidades de expansión lo han hecho ideal para dar los primeros pasos en robótica móvil, e incluso para aprendizaje de electrónica digital y microcontroladores. Recibió el 1º premio internacional de diseño “IEEE Design Contest 2002” otorgado por *The Institute of Electrical and Electronics Engineers* (IEEE).

1.2. Robótica Reactiva

En sus orígenes la robótica buscaba conseguir máquinas inteligentes, capaces de desarrollar tareas complejas y de establecer modelos abstractos del entorno. Sin embargo, este entusiasmo y halagüeñas perspectivas fueron perdiéndose poco a poco, ya que la evolución de los resultados no estaba acorde con

lo esperado. Lo cierto es que el entusiasmo inicial no estaban en consonancia con la tecnología del momento, pues en la década de los 70 y 80 se construían robots en los laboratorios que utilizaban cámaras de video como elemento principal para la captación del mundo, pero el tratamiento de la imagen en robótica es algo bastante complejo, que en general requiere de procesos costosos en tiempo, aun con los procesadores actuales. Cuando los procesadores de la época habían calculado la siguiente acción del robot, el entorno podía haber cambiado sus condiciones, lo que obligaba al robot a recalcular sus acciones. Esto producía que los robots móviles recorriesen habitaciones en plazos de hasta 4 y 5 horas de tiempo.

En la década de los 80 surgió una nueva corriente dentro de la robótica denominada robótica reactiva y posteriormente teoría de agentes. Esta corriente fue encabezada por Brooks^[2]. En ella existe una ruptura con el clásico bucle PPA (percepción planificación y acción) usado hasta entonces.

Para Brooks era como si se estuviese empezando la casa por el tejado, ya que se intentaba descomponer el razonamiento humano en diversas partes más sencillas, que se utilizan en los denominados sistemas expertos, con la idea de que algún día la unión de estos elementos aislados de la inteligencia humana estuviesen lo suficientemente desarrollados para poder ser unidos en una máquina inteligente.

Sin embargo Brooks era de la opinión de que jamás se conseguiría de este modo. Sería más sencillo practicar intentando emular inteligencias mucho más simples, como los insectos, e ir avanzando en la escala evolutiva según se encontrasen resultados y se aumentasen los conocimientos sobre el pensamiento.

De este modo surgen las arquitecturas reactivas, que precisan de un tiempo de cálculo mucho menor que en el caso de los bucles PPA, incrementando en mucho la capacidad de reacción de los robots además de reducir la capacidad de cómputo necesaria, más acorde con la tecnología del momento. Se denominan reactivas porque el robot no tiene ningún modelo del entorno, sino que reacciona ante los estímulos de este. La descomposición de las tareas se realiza en tareas más simples que se ejecutan en paralelo, buscando ciertas características del entorno y actuando cuando estas aparecen. De este modo las distintas subtareas compiten por el control de los actuadores del robot.

No se indagará más en esta arquitectura porque se desarrollará en el capítulo 3 de este proyecto. Solamente es necesario tener las siguientes ideas sobre la arquitectura reactiva:

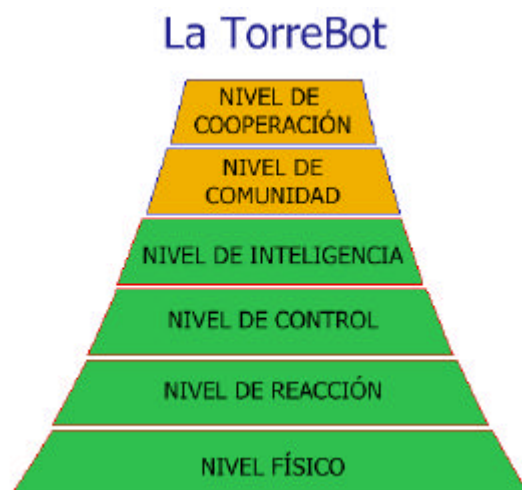
- La tarea se descompone en pequeñas subtareas llamadas comportamientos.
- Los comportamientos se procesan en paralelo
- Los comportamientos compiten por el control de los actuadores
- Existe un arbitraje que controla esta competencia
- En muchos casos los comportamientos se pueden implementar con facilidad como autómatas de estados finitos.

Con estas ideas presentes se entenderá el porqué de la elección de nuestro sistema para la consecución del proyecto.

1.3. La torrebot

Otro elemento importante que se debe introducir para comprender el desarrollo del proyecto es la Torrebot. Se trata de una serie de niveles cuyo cumplimiento por parte del robot lo sitúa en una escala de desarrollo. El concepto de la torrebot ha sido desarrollado por el grupo J&J*, actualmente Microbótica S.L.^[3] y se utiliza su propia definición de cada uno de los niveles:

Una de las clasificaciones que normalmente se utiliza para analizar a los microbots es la llamada Torre Bot. Se trata de un modelo en torre, donde cada uno de sus niveles representa un paso en la fabricación de un sistema microbótico.



Nivel Físico: Comprende la estructura física, las unidades motoras, y las etapas de potencia. Es posible encontrar desde sistemas sumamente sencillos basados en un único motor, hasta estructuras sumamente complejas que buscan emular las capacidades mecánicas de algunos insectos.

Nivel de Reacción: Está formado por el conjunto de sensores, así como los sistemas básicos para su manejo (circuitos de polarización). Estos transductores cubren un amplio margen de posibilidades, tal que podemos encontrar desde simples "bumpers", hasta micro cámaras digitales con sistemas de reconocimiento.

Un microbot que haya superado en cuanto a su construcción tanto el nivel físico como el de reacción, se denomina *microbot reactivo*.

* EL grupo J&J estaba formado por un grupo de estudiantes de la ETSIT (UPM) que desarrollaron diversos trabajos en robótica móvil durante sus estudios. En la actualidad forman la empresa Microbótica S.L. y son un referente y punto de apoyo para los estudiantes que dedican parte de su tiempo a la robótica.

Este tipo de unidades, trabajan cumpliendo la premisa, "acción-reacción". En estos casos, los sensores son los propios controladores de las unidades motoras, sin ningún tipo de control intermedio.

Nivel de Control: Incluye los circuitos más básicos que relacionan las salidas de los sensores con las restantes unidades. Partiendo de una simple lógica digital hasta potentes microcontroladores, se busca dotar al microbot de la capacidad para procesar la información obtenida por los sensores, así como actuar de una manera controlada sobre las unidades motoras.

Nivel de Inteligencia: Abarca el planificador a largo plazo; en este nivel, se introducen los objetivos del microbot que tienen relativa independencia de los sensores. Este es el más alto nivel de inteligencia que puede alcanzar un microbot como una unidad individual.

Nivel de Comunidad Se trata de la puesta en funcionamiento de más de un microbot dentro de un mismo entorno de forma simultánea y sin que ninguno de ellos tenga conocimientos explícitos de la existencia de otros en su mismo entorno. A estos recintos se los denomina granjas.

Los centros de investigación utilizan las granjas como entornos de observación de los microbots. Dichos establecimientos pueden contar con sistemas sofisticados que permitan a un operario monitorizar el comportamiento de la comunidad así como alterar las condiciones externas del sistema (agregar obstáculos, cambiar la temperatura, etc...).

Nivel de Cooperación Comprende los sistemas donde a partir de un nivel de comunidad, se planifican o programan los microbots para que tengan conocimiento de la existencia de otros, tal que posean la capacidad de cooperar para el buen desarrollo de una tarea.

Microbótica,
Noviembre 1996.

1.4. Objetivos del proyecto

Se trata de construir una plataforma para desarrollos de robótica móvil reactiva hasta el nivel comunidad. Ambos conceptos, robot reactivo y nivel comunidad han sido introducidos en los puntos anteriores.

Se atenderá a las siguientes condiciones para el diseño:

1. Fácilmente implementable

Esto conlleva varias consideraciones a la hora de diseñar el hardware:

- i. Componentes asequibles en precio y disponibilidad
- ii. Encapsulados de paso no demasiado fino, facilidad para soldadura manual

- iii. PCB de 2 caras y distancias entre pistas no demasiado pequeñas¹.

Si se cumplen estas premisas solamente será necesario tener conocimientos básicos para implementar el sistema de desarrollo y además su precio será bajo. Así cualquier estudiante de formación técnica podrá beneficiarse de este trabajo.

2. Reprogramable en Circuito

Propiedad indispensable para un sistema de desarrollo, pues se realizarán multitud de pruebas y desarrollos sobre el sistema. Por tanto no será válida una pieza que sea reprogramable pero su número de ciclos de escritura sea pequeño, al igual que tampoco lo será una pieza que necesite ser extraída del sistema para ser borrada o reprogramada.

3. Herramientas de desarrollo libres ó de muy bajo coste

Normalmente se usan microcontroladores para los desarrollos en robótica, pero en muchos casos las herramientas para programarlos (los programadores) y los compiladores cruzados (compiladores C normalmente) resultan caros para el bolsillo de los estudiantes. Por el contrario es posible encontrar programadores para algunos μ C de muy bajo coste y de hardware libre, por lo que es posible fabricarlos. Del mismo modo las herramientas para trabajar con ellos pueden ser de software libre e incluso abierto.

4. Escalable

El sistema debe ser fácilmente ampliable. De este modo se podrán aumentar sus capacidades cuando el usuario lo exprima al máximo. Por otro lado, al ser escalable permitirá el desarrollo por parte de terceros, pudiendo aumentar sus capacidades mediante el trabajo de otras personas, en definitiva, no reinventar la rueda. Como por ejemplo realizar un desarrollo basado en un μ C potente que aproveche las características de la FPGA para controlar sensores.

1.5. Arquitecturas de control propuestas

En una primera fase del proyecto se propusieron distintas arquitecturas hardware en las que basar el diseño, todas ellas salvo una estaban basadas en microcontroladores. Hasta ahora todos los diseños de robótica realizados por los autores estaban basados en ellos (salvo Odin), y de hecho es bastante raro encontrar diseños de otro tipo. Se presentan las barajadas y por que fueron desechadas,

¹ Condiciones indispensables para su fabricación de forma artesanal.

finalmente se muestra la propuesta que finalmente ha sido implementada y sus razones.

mC PIC16F877

Microcontrolador de 8 bits, bastante rápido, barato y muy asequible. Cumple todos los requisitos propuestos en los objetivos, además es un μ C conocido para los autores, pues ha sido utilizado en otros robots del grupo. No posee ningún compilador C libre pero si uno de bajo coste.

No fue elegido precisamente porque ya se había creado una plataforma para este procesador (el robot π).

mC 68hc12

Potente μ C de 16 bits de la firma Motorola. Posee muchas cualidades, mucha memoria interna, es rápido y con un gran repertorio de instrucciones entre las que se incluyen operaciones orientadas a lógica borrosa. Además posee un compilador C ANSI GNU.

No fue elegido porque no cumple con uno de los objetivos del proyecto, ya que su encapsulado es QFTP de paso 0.6mm. Esto lo hace casi imposible de soldar a mano.

mC Dragon Ball

Procesador de 32 bits de la firma Motorola. Es muy potente y cuenta con muchos recursos hardware. La memoria no es un problema ya que la direcciona externamente. Posee compiladores GNU para lenguajes C, C++ y Java, además del Sistema Operativo μ C-Linux (S.O. Linux para procesadores sin MMU).

También se desechó por el mismo problema que el anterior, además era difícil de conseguir en pequeñas cantidades.

mC H8

Procesador de 16 bits del fabricante Hitachy. Posee un potente Kernel CISC con compilador C GNU. Además posee bastante memoria interna Flash y RAM, así como recursos hardware (timers, PWM, ADC, DAC...). Otra de sus ventajas es que existe en un encapsulado DIP 64, lo que permitiría su uso.

Tampoco se escogió porque el fabricante solamente asegura 100 ciclos de grabación de la Flash de programa, a partir de la 100 el tiempo de grabación se incrementa exponencialmente. Esto provocaría grabaciones muy lentas, no deseables, ya que corregir una línea de código puede resultar muy tedioso.

FPGA 10K10

Esta es una apuesta bastante arriesgada, pues apenas existen robots que utilicen una FPGA como núcleo del sistema. Sin embargo cumple con todos los requisitos de los objetivos, pues los entornos de desarrollo son gratuitos y se pueden

programar en entorno gráfico, VHDL, AHDL y Verilog. Se elige esta opción por cumplir con todos los objetivos y ser la opción más exótica.

Por otro lado posee un punto fuerte en nuestro sistema. La plataforma está diseñada para un robot móvil reactivo, recordando las características del control reactivo:

- La tarea se descompone en pequeñas subtareas llamadas comportamientos o conductas.
- Los comportamientos se procesan en paralelo
- Los comportamientos compiten por el control de los actuadores
- Existe un arbitraje que controla esta competencia
- En muchos casos los comportamientos se pueden implementar con facilidad como autómatas de estados finitos.

Que los comportamientos se ejecuten en paralelo implica que en un sistema con un solo μC implica que las tareas serán secuenciales, por tanto para implementar un control reactivo correctamente sería necesario contar con un sistema operativo multitarea de tiempo real. Esto último no es estrictamente cierto, pero como se verá en detalle en el tema 3 es muy deseable y facilita enormemente el desarrollo.

Sin embargo la implementación en una FPGA es inherentemente un proceso paralelo. Además las conductas pueden implementarse como autómatas de estados finitos, y eso es algo muy sencillo de implementar en una FPGA en código VHDL.

Como colofón y no menos importante, el código generado tiene una portabilidad muy grande, se pueden generar bloques de código para controlar cualquier elemento como sensores y motores y según el número que tengamos de cada uno poner más o menos. En el caso de un μC puede que al utilizar dos funciones en el mismo código, que por separado funcionaban perfectamente, no hagan por no tener el procesador tiempo o recursos.

Por estos y otros motivos que se destacarán a lo largo de los siguientes temas se ha elegido este peculiar sistema de control.

1.6 Bibliografía

¹ <http://aess.upc.es> y www.depeca.alcala.es/alcabot

² “*Intelligence without representation*” Rodney A. Brooks.*
www.ai.mit.edu/people/brooks/brooks.html

³ www.microbotica.es

* Incluido en el CD