Quaky-II: Robot Móvil para Interiores

Humberto Martínez Barberá, Miguel Ángel Zamora Izquierdo, Germán Villalba Madrid, Juan Pedro Cánovas Quiñonero

Dep. de Ingeniería de la Información y las Comunicaciones
Universidad de Murcia
30100 Murcia

humberto@um.es, mzamora@um.es, germanv@dif.um.es, juanpe@dif.um.es

Resumen

Este trabajo describe el robot autónomo móvil Quaky-II desarrollado por el grupo de Sistemas Inteligentes de la Universidad de Murcia. Este es un robot de interiores cuyo software está pensado para realizar tareas de exploración y realización de mapas en entornos no conocidos. Como característica importante se tiene la arquitectura hardware que permite de una forma sencilla ampliar o modificar las características de los sensores.

1. Introducción

El grupo de investigación ha desarrollado, construido y probado distintas plataformas, desde el muy simple MadCar hasta el robot Quaky-Ant [3]. El desarrollo de los distintos robots ha estado condicionado por diferentes factores, siendo los más importantes las dimensiones físicas de los laboratorios y pasillos adyacentes, y el coste total. Las dimensiones físicas limitaban el tamaño máximo a 50 cm de diámetro, y el coste estaba limitado por un presupuesto de 6.000€. Adicionalmente, se impusieron otras restricciones al desarrollo final:

- La plataforma debía ser suficientemente flexible para probar los distintos algoritmos y arquitecturas presentadas en esta tesis.
- La plataforma debía ser robusta y fácilmente actualizable, haciendo uso de distintos sensores (ultrasonidos e infrarrojos).
- La plataforma debía ser capaz de desarrollar satisfactoriamente tareas complejas, operando en entornos tanto interiores como exteriores no abruptos.

Como fruto de estos desarrollos y teniendo en cuenta estas consideraciones, se ha desarrollado el robot Quaky-II, del que se describe en las siguientes secciones tanto la arquitectura hardware como la software.

2. Plataforma Quaky-II

El robot *Quaky-II* (Figura 1) está pensado para navegación en entornos interiores de pequeñas habitaciones y pasillos. Sus pequeñas dimensiones (Tabla 1) permiten el paso por las puertas sin problemas. La filosofía de diseño ha sido la implementación de un robot de características similares al *Pioneer* de ActiveMedia Robotics o el *Magellan* de RWI, con una arquitectura abierta que nos permitiese añadir los sensores que estimásemos necesarios para la realización de nuestras pruebas, y al mismo tiempo a un coste inferior al citado diseño comercial.

Dimensiones	Equipamiento
Base: 40 x 40 cm	CPU Crusoe TMS400
Altura: 30 cm	Ethernet 802.11b
Peso: 15 Kg	Enlace vídeo 2.4GHz

Tabla 1. *Características del robot Quaky-II*

La planta del robot sigue un modelo diferencial con dos ruedas motrices y una rueda de movimiento libre (rueda loca). El robot se alimenta de una batería de 12V, y cuenta con otra auxiliar de 12V para el enlace inalámbrico de vídeo. Cada una de las ruedas motrices es impulsada por un motor de 12V y 15w (Tabla 2), con un lazo cerrado de control de velocidad para cada motor.

Sensores	Actuadores
16 sonars Polaroid	2 motores Maxon 12V
16 infrarrojos Sharp	2 reductoras Maxon 60:1
2 encoders HP	

Tabla 2. *Sensores y actuadores del robot Quaky-II*

El sistema sensorial del robot (Tabla 2) está formado por los siguientes sensores:

- Anillo de 16 sensores sonar Polaroid 600 disparados por los módulos 6500 de la misma marca.
- Anillo de 16 sensores infrarrojos Sharp GP2D02.
- Encoder diferencial en cada motor de 500 pulsos por vuelta.

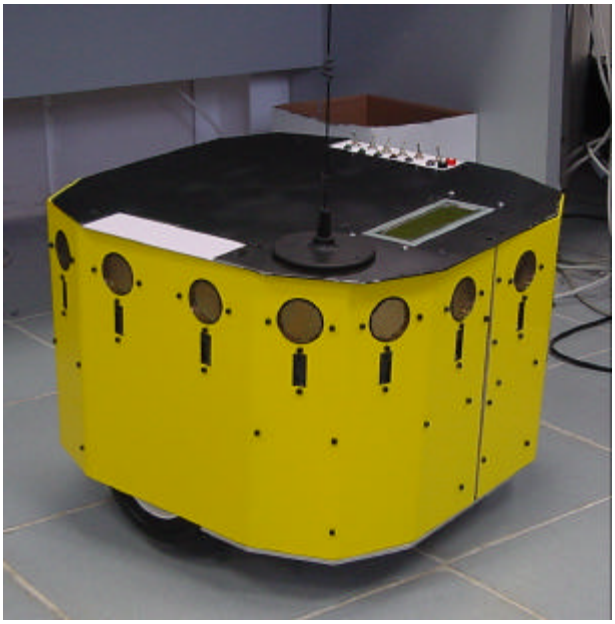

Figura 1. *El robot Quaky-II*

3. Arquitectura hardware Quaky-II

La arquitectura hardware del robot móvil está organizada modularmente. El procesador principal del sistema es una placa Single Board Computer (SBC) basada en un micro Crusoe TMS400 de Transmeta, donde se ejecuta una máquina virtual Java sobre un sistema operativo Linux. La SBC se encarga de la ejecución de software de alto nivel de control y navegación, así como de establecer la comunicación con el exterior por medio de una ethernet inalámbrica. A la tarjeta SBC le llega información de los sensores a través de los distintos puertos serie de que dispone.

Mientras que la tarjeta SBC se encarga del software de control de alto nivel, la adquisición de datos de los distintos sensores, y el control motor se realiza mediante una serie de módulos basados en microcontrolador, conectados mediante un bus I^2C [2]. Cada módulo contiene su propio microcontrolador que implementa la comunicación inter-modular, y lleva a cabo el procesamiento de señal necesario para sensores, o implementa los lazos de control motor según el caso. Los módulos conectados en el bus son:

- Un módulo maestro bus, para conectar el SBC con el bus I^2C .
- Ocho módulos para control de dos sensores de ultrasonidos y dos infrarrojos.
- Dos módulos para control motor y lectura de la posición odométrica.

El protocolo implementado sobre el bus I^2C ha sido desarrollado en configuración de maestro único, que coordina a los módulos esclavos. El módulo maestro, es el encargado de hacer de puente entre el bus de sensores I^2C y la placa principal de control SBC, mediante comunicación serie. El módulo maestro, transmite a los esclavos las órdenes de disparo de sensores, comandos motores y de configuración.

Básicamente el protocolo definido sobre I^2C consta de dos paquetes que denominamos PWRT y PANS. El paquete PWRT implementa un comando genérico para todas las acciones de los distintos módulos esclavos, donde se indica el comando a ejecutar por el módulo esclavo direccionado, así como los parámetros necesarios para ejecutar el comando. En el paquete PANS, el maestro requiere los datos procesados por el módulo esclavo direccionado, y su longitud varía por tanto según el módulo esclavo. Este protocolo está implementado de forma que si algún esclavo no responde, no deje bloqueado el bus y puedan seguir funcionando el resto de los nodos, con lo que el sistema es tolerante a fallos.

Los dos módulos destinados a controlar los motores del robot, implementan un lazo de control en velocidad cada uno, además de remitir la información del desplazamiento realizado (lectura del encoder acoplado al eje del motor). Dichos módulos implementan además la lectura de sensores de contacto con la posibilidad por tanto de incorporar al robot hasta 16 bumpers.

Los módulos de sensores de distancia pueden manejar dos módulos Polaroid 6500, que disparan dos transductores sonar de la misma marca, y dos sensores de infrarrojos Sharp GP2D02.

El robot Quaky-II puede montar una cámara a color que se usa en tareas de exploración y teleoperación, para ofrecer feedback visual al operador Los parámetros de la cámara pueden configurarse vía un puerto RS232 desde la tarjeta SBC, y las imágenes son retransmitidas mediante un enlace inalámbrico de 2.4 GHz. Por último el robot está conectado con el exterior a través de una tarjeta ethernet inalámbrica IEEE 802.11b, que opera en la banda de 2.4GHz y alcanza una velocidad de hasta 11Mb/s.

4. Sistema de navegación

Para un robot móvil la habilidad de navegar es una de las más importantes. La navegación se puede definir como la combinación de tres habilidades fundamentales: la construcción de mapas, que es el proceso de realizar mapas a partir de las lecturas de los sensores en diferentes posiciones, la localización, que es el proceso de conocer la posición actual del robot a partir de las lecturas de los sensores y el mapa actual, y la generación de trayectorias, que es el proceso de obtener una trayectoria factible y segura desde una determinada posición a otra objetivo a partir del mapa actual.

La plataforma Quaky-II [3] ha sido dotada de distintos algoritmos y métodos para atacar el problema de la navegación. Inicialmente, el sistema asumía una odometría fiable para operaciones de corto tiempo y distancia. Teniendo en cuenta esta asunción, incorporaba un método de construcción de mapas y otro de generación de trayectorias. Para solventar el problema de la fuerte restricción de la odometría, se dotó al sistema de un método de localización basado en otro modelo de mapa. La combinación final de ambos métodos resultó funcionar bastante bien.

El primer modelo de navegación (Figura 2) usa un mapa de celdillas difusas y un algoritmo basado en el A* para generar las trayectorias. De forma distinta a otras soluciones, el robot construye los mapas mientras está en movimiento en vez de detenerse constantemente para actualizar el mapa. La idea es que el robot exhiba cierto grado de reactividad similar al utilizado por animales y personas. El proceso de navegación es como sigue. Cada vez que un sensor produce un valor nuevo el mapa se actualiza. Este paso no consume mucho tiempo y se realiza a una alta frecuencia (cada 300 ms aproximadamente). Simultáneamente el robot se va desplazando y cada cierto tiempo se genera una nueva trayectoria teniendo en cuenta el mapa actual. Este paso se realiza a una frecuencia menor (cada 1200 ms aproximadamente) debido a que la generación de trayectorias consume comparativamente más tiempo.

El método de representación de mapas está basado en los mapas de celdas difusa [5]. Con este método se usan y actualizan tres mapas: el de las celdas que están posiblemente vacías, que tienen un grado de pertenencia alto al conjunto difuso celda vacía, el de las celdas que están posiblemente ocupadas, que tienen un grado de pertenencia alto al conjunto difuso celda ocupada, y el de las celdas que son seguras para navegación, que se obtiene combinando los otros dos de forma que las celdas tienen simultáneamente un grado de pertenencia alto al conjunto difuso celda vacía y un grado de pertenencia bajo al conjunto difuso celda ocupada.

La forma más directa para generar trayectorias en un mapa de celdillas es considerar cada celda como un nodo de un grafo y posteriormente aplicar un A*. Este algoritmo permite el uso de información heurística cuando está disponible, resultando en una búsqueda eficiente. A partir del mapa de navegación segura se puede aplicar este método. Hay en la literatura aproximaciones para resolver este problema teniendo en cuenta el ruido de los sensores, el tamaño del robot, y recorridos de tamaño mínimo, pero no simultáneamente. En este trabajo se ha optado por una solución que encuentra caminos de tamaño mínimo a la vez que produce caminos seguros y alejados de obstáculos. La cota del coste mínimo de la función de evaluación del A* se calcula en dos pasos: en primer lugar el mapa difuso es dilatado localmente alrededor de la celda actual, para posteriormente aumentar exponencialmente la función de coste.

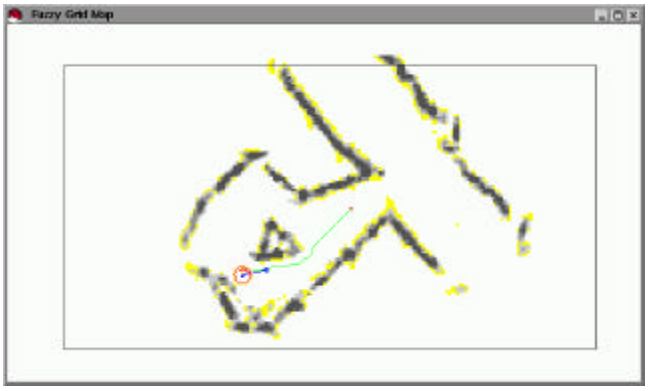

Figura 2. *Mapa de celdas difusas y trayectoria.*

El primer modelo de navegación asumía una odometría fiable para corto tiempo y distancia. Esto es claramente una restricción importante en entornos grandes o en operaciones que requieran mucho tiempo. Para solventar esta limitación, el método anterior se complementa con un método de localización basado en mapas de segmentos difusos (Figura 3). Los mapas globales de celdillas no son válidos para localización, mientras que los mapas locales de celdillas requieren una referencia precisa de un compás para obtener resultados fiables. Por otro lado, el método anteriormente descrito, si se provee con una localización fiable produce caminos seguros de una forma rápida. Así, el segundo modelo de navegación está basado en el primero con la adición de una técnica de localización que no necesita un modelo a priori del entorno. El nuevo modelo funciona como sigue. El mapa de celdas difusas y el de generación de trayectorias se ejecutan de igual forma que en el caso anterior, pero en lugar de utilizar la posición obtenida por odometría usan la posición corregida por el método de localización. Simultáneamente, un mapa local de segmentos difusos se actualiza cada vez que se llena un buffer se ultrasonidos (cada 1200 ms aproximadamente). Este mapa se compara (cada 2500 ms aproximadamente) con un mapa global de segmentos difusos para obtener las diferencias y calcular el error actual de la odometría, para posteriormente corregir la posición. Ambos mapas se construyen sin tener ningún conocimiento previo del entorno.

El método del mapa de segmentos difusos [1] es una representación geométrica formada únicamente por segmentos de línea. Manejando cuidadosamente la incertidumbre relativa a la posición del robot y el ruido de los sensores, el método intenta solucionar los problemas típicamente asociados a los modelos geométricos. En este caso, el mapa ampliamente reduce la falta de estabilidad disminuyendo la precisión de los ajustes de las líneas (haciéndolas más gruesas), manteniendo la incertidumbre durante todo el proceso. El modelo original genera los segmentos de línea considerando cada sensor individualmente a lo largo del tiempo. Las medidas de los sensores son preprocesadas para eliminar errores claramente detectables y agrupar medidas correlativas. Finalmente estos grupos son aproximado por una recta para así generar un segmento difuso. Esta aproximación genera mapas aceptables, pero no tiene la habilidad de combinar medidas de distintos sensores al nivel sensorial. Esto hace que el proceso de generación de los mapas sea lento, porque se necesita bastante tiempo para obtener buenas medidas que puedan ser ajustadas por una recta. Se ha desarrollado un nuevo método de generación de segmentos para solventar este problema. Así, este método construye y mantiene de forma heurística un buffer circular que almacena las últimas medidas de todos los sensores. Cada vez que un nuevo conjunto de medidas está disponible (un ciclo de disparo de los sensores), las que son menores de una determinada longitud se almacenan secuencialmente en el buffer sobreescribiendo los valores menos recientes. Este buffer se utiliza para extraer las agrupaciones de puntos que posteriormente se ajustarán con una recta para dar lugar a un segmento difuso.

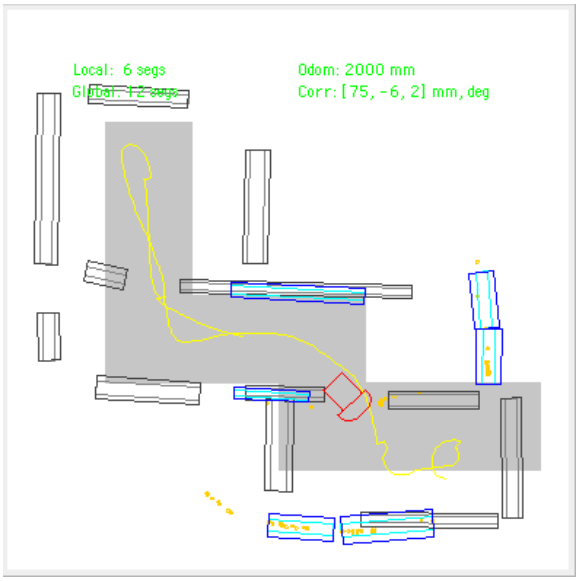

Figura 3. *Mapa de segmentos difusos.*

5. Arquitectura de Control

La tarea que ha de realizar el robot es navegar desde una determinada posición inicial a una objetivo, y después regresar a la inicial, en un entorno completamente desconocido similar a cualquier entorno interior de oficina (con diferentes salas y pasillos). Las coordenadas de las dos posiciones se conocen de antemano. Adicionalmente, el robot no debe colisionar ni quedarse paralizado. Para cumplir el objetivo el sistema ha sido descompuesto, extrayendo sus elementos más importantes [3][4], en los siguientes módulos (Figura 4):

- Reactive Control. Es el agente que se encarga de la navegación reactiva. Implementa una serie de comportamientos para evitar colisiones, seguir un camino determinado, alinearse a una pared, o escapar de obstáculos en forma de U.
- Planning. Es el agente deliberativo que se encarga de la planificación de alto nivel y de la supervisión del objetivo.
- Navigation. Es el agente deliberativo que se encarga de construir un modelo del entorno, localizar el robot sobre él, y generar una posible camino para alcanzar el objetivo.
- Sensor Processing and Actuator Control. Son los agentes encargados de procesar los datos sensoriales y controlar los motores. El primero accede a los señales directas de los sensores, las filtra, las procesa, y pone la información relevante en la pizarra. Es en este agente donde se ejecutan los algoritmos de fusión sensorial. El segundo lee los comandos motores de la pizarra y envía las correspondientes acciones de control a los motores.

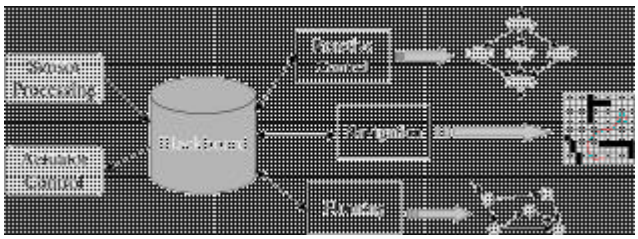

Figura 4. *Módulos de la arquitectura de control.*

El agente *Reactive Control* (Figura 5) es el más importante y crucial porque implementa los comportamientos básicos reactivos, que tienen la responsabilidad de mantener la integridad física del robot por medio de la evitación de colisiones. Casi todos los comportamientos se definen mediante bases de reglas difusas, obtenidas a partir de experiencia previa o de técnicas de modelado difuso. Se han identificado y desarrollado los siguientes comportamientos reactivos:

- Avoid-left. Evita un obstáculo situado a su izquierda.
- Avoid-front. Evita un obstáculo situado a su frente.
- Avoid-right. Evita un obstáculo situado a su derecha.
- Align-left. Mantiene el robot a una distancia determinada de la pared izquierda.
- Align-right. Mantiene el robot a una distancia determinada de la pared derecha.
- Move-to-goal. Dirige el robot en la dirección del objetivo actual.
- Escape. Gira el robot aleatoriamente para escapar de un pared en forma de U.

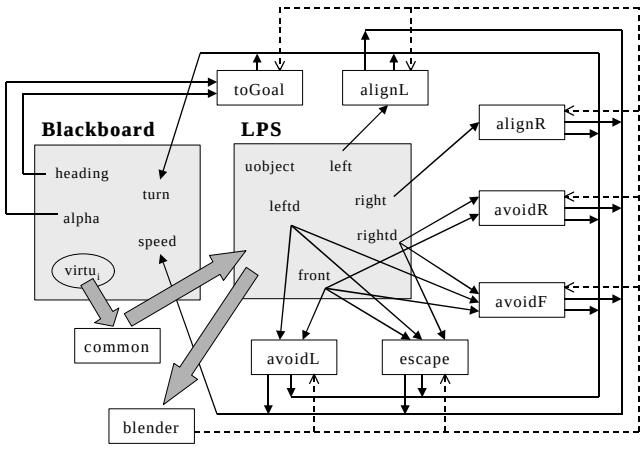

Figura 5. *Módulo de control reactivo.*

El agente *Navigation* (Figura 6) es el encargado de construir un mapa del entorno, calcular un camino a la posición objetivo, y mantener localizado el robot. Se utiliza la estructura de navegación descrita en la sección anterior. El mapa se actualiza continuamente con las medidas de los sensores. El agente *Planning* (Figura 6) se encarga de

supervisar la tarea que se está ejecutando, cambiando el objetivo cuando se ha completado un bordo y deteniendo la ejecución cuando se completa la tarea global. Para implementar este planificados simple se utiliza una estructura de autómata finito. Los distintos estados del autómata tienen asociados coordenadas objetivo, que son usadas por el agente Navigation para producir un rumbo que será posteriormente seguido por el robot hasta cumplir el objetivo del estado.

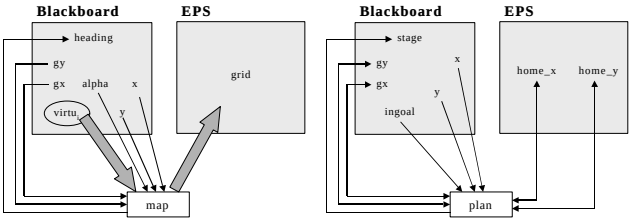

Figura 6. *Módulos de navegación y planificación.*

6. Conclusiones

Teniendo en cuenta el coste, el rendimiento y la capacidad de ampliación, el robot Quaky-II se compara favorablemente con distintos robots comerciales. Las principales características de la plataforma se pueden resumir en:

- Exhibe una arquitectura abierta. Dos puntos clave son el uso del bus I^2C para la interconexión de los distintos módulos y la definición de un protocolo abierto genérico.
- Una consecuencia del diseño de la arquitectura es la robustez y capacidad de ampliación (todos los módulos se acceden de la misma forma independientemente de su naturaleza).
- La plataforma no está ligada a ninguna arquitectura de control o sistema de desarrollo, aunque ha servido de base para el desarrollo de una.
- El robot opera suave y correctamente en entornos interiores y es capaz también de operar en exteriores no abruptos. En ambos casos, no exhibe una alta velocidad máxima (en torno a los 0.45 m/s) ya que esta no era fundamental en el diseño. En contraste, exhibe una buena maniobrabilidad y precisión.

7. Agradecimientos

Para la realización de este trabajo se ha contado con la financiación parcial del proyecto CICYT TIC2001-0245-C02-01.

Referencias

[1] Gasós, J. (2000). *Integrating Linguistic Descriptions and Sensor Observations for the Navigation of Autonomous Robots*, in Fuzzy Logic Techniques for Autonomous vehicle Navigation (D. Driankov and A. Saffiotti eds.), Springer-Verlag, Germany

[2] Lekei, D. (1997). *Using a PIC16C5X as a Smart I2C Peripheral*, AN541, Microchip Technology, Chandler, USA

[3] Martínez, H. (2001) *Una Arquitectura Distribuida para el Control de Robots Autónomos Móviles: un Enfoque Aplicado a la Construcción de la Plataforma Quaky-Ant*, Tesis Doctoral, Departamento de Ingeniería de la Información y las Comunicaciones, Universidad de Murcia

[4] Martínez, H.; Zamora, M.A.; & Gómez, A. (2002) *A Framework for Defining and Learning Fuzzy Behaviours for Autonomous Mobile Robots*, International Journal of Intelligent Systems, 17(1):1-20

[5] Oriolo, G.; Ulivi, G. & Venditelli, M. (1998). *Real-Time Map Building and Navigation for Autonomous Mobile Robots in Unknown Environments*, IEEE Trans. on Systems, Man and Cybernetics-Part B: Cybernetics, 3(28):316-333