

Una Arquitectura para Robots Móviles Pequeños

Danilo Bassi A. e Ignacio Garrido Z.

*Departamento de Ingeniería Informática
Universidad de Santiago de Chile
Santiago, CHILE
dbassi@ieee.org, igarrido@vtr.net*

Abstract

Este paper describe una arquitectura de control orientada a robots móviles pequeños, denominada SARA (*Small Robot Architecture*), que ha sido exitosamente implementada en un robot que recolecta latas. Se trata de una arquitectura modular orientada a sistematizar y facilitar la implementación del sistema de control y su posterior utilización.

Keywords: Robots Móviles, Arquitectura, Tareas.

1. INTRODUCCIÓN

1.1 Motivación

La utilización de robots móviles en el ámbito educativo y académico ha adquirido bastante auge en el último tiempo, debido al abaratamiento de la tecnología y a la búsqueda de formas novedosas y motivadoras de aprendizaje. Actualmente numerosos cursos en diversas universidades del mundo utilizan robots móviles como material de apoyo. [Asada et al. 00], [Pfeifer 97], y [Klassner & Anderson 01] describen algunas experiencias al respecto.

La mayoría de los robots utilizados en esta ámbito son sencillos, sin mucha capacidad de procesamiento, por lo que la aplicación de muchas de las metodologías y técnicas desarrolladas en el ámbito científico no son apropiadas para estos robots.

Surge por lo tanto la necesidad de herramientas metodológicas que apoyen dicha labor, y que sean adecuadas para robots con pocos recursos. Por un lado, es necesaria una organización del software de control del robot, y por otro, técnicas que faciliten su utilización en tareas específicas.

2.2 Arquitecturas de Control

El diseño e implementación del sistema de control de un robot no es una tarea trivial, y requiere de un

soporte metodológico apropiado para poder ser llevado cabo en forma eficiente y eficaz. Por ello, el diseño arquitecturas de control es uno de los temas centrales dentro del campo de los robots autónomos. Maes define una arquitectura de agente como:

“Una metodología particular para construir agentes, que especifica como el agente puede ser descompuesto en un conjunto de módulos y como dichos módulos deben interactuar entre sí. El conjunto completo de dichos módulos y sus interacciones deben proporcionar una respuesta acerca de cómo los datos sensoriales junto con el estado interno determinan las acciones y el futuro estado interno del agente. Una arquitectura debe estar acompañada de técnicas y algoritmos que soporten dicha metodología.” ([Maes 91], p115).

Kaelbling considera que una arquitectura de agente es:

“Una colección específica de módulos de software (o hardware), típicamente definida mediante cajas con flechas indicando el flujo de datos y control entre los módulos. Una visión más abstracta de una arquitectura es como una metodología general para diseñar descomposiciones modulares que permitan llevar a cabo tareas específicas.” ([Kaelbling 91], p86).

El último punto mencionado en esta última definición es de suma importancia: cada arquitectura esta orientada a cierto tipo de problemas específicos y además ha sido diseñada según criterios y prioridades particulares. Ello trae consigo el que existan una infinidad de arquitecturas muy diferentes entre sí, tanto en su estructura como en sus capacidades. Así, como ha señalado [Mataric 99], una arquitectura, además de establecer la estructura del sistema de control, impone restricciones en la forma en que el problema de control puede ser resuelto.

2. LA ARQUITECTURA SARA

2.1 Propósito

Una arquitectura esta determinada, principalmente, por el ámbito de aplicación y por el tipo de las tareas que

se desea puedan ser implementadas. En este sentido, la arquitectura SARA fue diseñada con dos objetivos en mente:

1. Debe estar orientada a robots que sirvan de apoyo a cursos de ingeniería o que sean utilizados en competencias robóticas, sin perjuicio de que pueda ser utilizada en otros ámbitos.
2. Debe permitir implementar tareas sencillas que involucren una secuencia de pasos discretos, como por ejemplo, recolectar latas o resolver un laberinto.

2.2 Descripción

En la Figura 1 se muestra la arquitectura desarrollada, y a continuación se describen sus componentes.

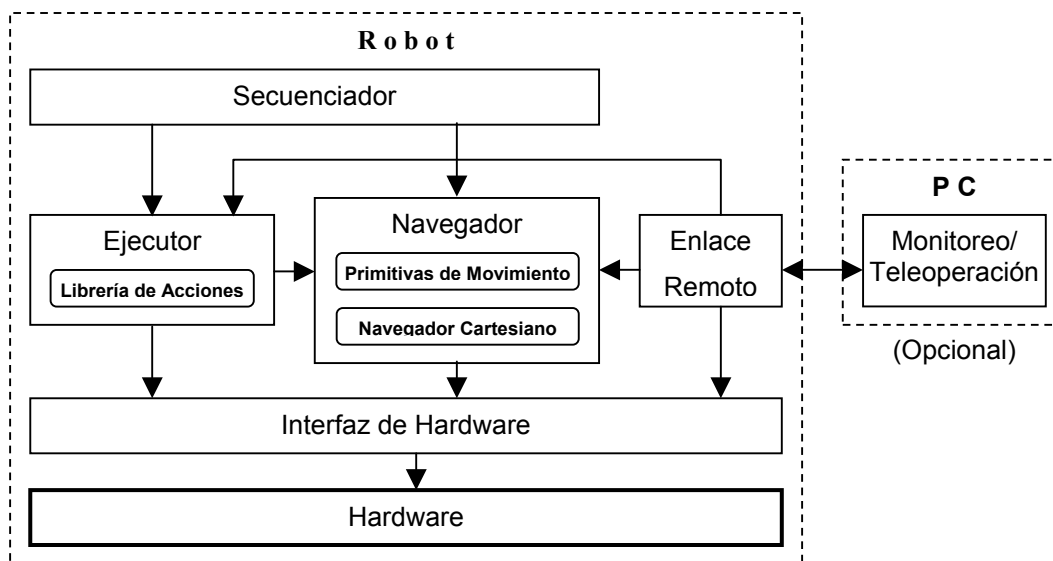


Figura 1. Arquitectura de SARA.

Interfaz de Hardware

Este módulo proporciona cierto grado abstracción del hardware, facilitando la reutilización del código en robots diferentes. También implementa rutinas de bajo nivel, como por ejemplo aquellas relacionadas con la interfaz de usuario del robot (control de teclado y display).

A pesar de que sería deseable que este módulo encapsulara todas las particularidades de un hardware específico, en la práctica esto no siempre es posible o conveniente, puesto que muchas veces para un módulo

es más sencillo y eficiente acceder directamente el hardware.

Navegador

Básicamente, el navegador proporciona funciones de locomoción. Esta compuesto por dos submodulos, uno de primitivas de movimiento y otro de navegación cartesiana.

El primer módulo, el de primitivas de movimiento, proporciona funciones de movimiento elementales, como fijar la velocidad de una rueda o avanzar una determinada cantidad de distancia en línea recta.

El segundo módulo, el navegador cartesiano, se basa en el módulo anterior para implementar un sistema basado en odometría que permite ubicar al robot en un punto del espacio dentro de un sistema de coordenadas. La función principal de este submódulo es aquella que permite llevar el robot a una determinada posición (x,y) esquivando obstáculos sencillos.

Ejecutor

Este módulo maneja una librería de acciones que se encarga de ejecutar, una a la vez, bajo las ordenes del Secuenciador. Una acción se define como una secuencia de pasos con un objetivo específico y significativo. Ejemplos de acciones son “coger objeto” y “seguir elemento luminoso”. Cada acción interpreta los sensores en forma particular y emite los comandos apropiados al sistema de navegación. Se considera que las acciones son atómicas, en el sentido de que o se ejecutan completamente o fallan.

Secuenciador

El secuenciador es el módulo que determina el comportamiento global del robot. Básicamente, consiste en la implementación de un StateChart [Harel 87], en el que los estados corresponden a las acciones mencionadas y los eventos a cambios en condiciones externas o internas. Dicha implementación es codificada en forma estática, es decir, no se modifica su estructura mientras el robot esta en ejecución.

Enlace

Este módulo se encarga de transmitir, probablemente mediante un enlace serial, las acciones del robot al sistema de Monitoreo/Teleoperación que corre en un computador. También se encarga de ejecutar los comandos enviados desde este último.

Monitoreo/Teleoperación

El módulo de Monitoreo/Teleoperación es un software que corre fuera del robot, en un computador, y que opcionalmente permite controlar el robot mediante un programa ejecutado en un computador, aprovechando así las grandes capacidades de procesamiento de este. Por otro lado, ya sea que el programa de control se ejecute dentro o fuera del robot, este módulo permite monitorear el funcionamiento del robot, facilitando la depuración del sistema y la realización de las pruebas.

Bootloader

Este es un pequeño programa que se ejecuta solo si el robot es iniciado en modo de descarga, es decir, cuando se desea cargar nuevamente el programa de

control. Desde el punto de vista de la arquitectura este módulo no es relevante, pero merece mención puesto que es un elemento de software importante para efectos prácticos.

3. RESULTADOS

3.1 Implementación

La arquitectura descrita fue implementada en el robot Gbot (Figura 2), de construcción propia, que posee sensores infrarrojos y un sistema trasero que le permite arrastrar latas. Utiliza un microcontrolador Atmel ATmega16, con 16 Kb. de memoria de programa y 1 Kb. de memoria ram.

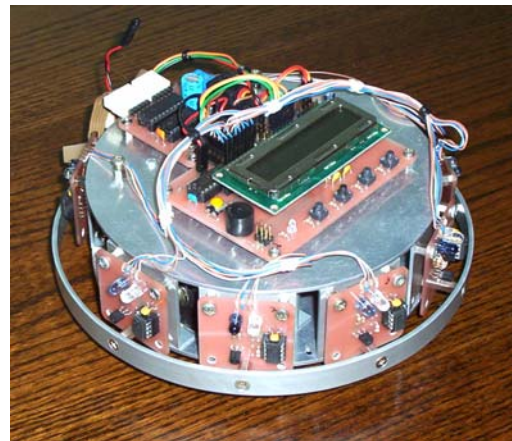


Figura 2. Robot Gbot.

La implementación se desarrollo en forma incremental, en el siguiente orden: Interfaz de Hardware, Navegador, Enlace, Ejecutor y Secuenciador. El modulo de Monitoreo/Telecontrol se desarrollo en forma paralela al resto del sistema, sirviendo de apoyo para las pruebas de cada modulo individual y del sistema completo.

3.2 Pruebas

El sistema fue validado mediante la implementación de una tarea de recolección de latas, en la que el robot debía primero encontrar un faro luminoso (evitando chocar con las latas) y luego buscar latas vacías y dejarlas junto al faro. El statechart de la tarea se muestra en la Figura 3.

La prueba fue realizada en un área cerrada de 1.5x1.2 metros, utilizando entre 5 y 10 latas. El experimento se repitió en diversas ocasiones, cambiando la distribución de las latas y la posición el faro, y en todas el robot logro su cometido.

Para una descripción detallada de este experimento y su modelamiento véase [Garrido 04].

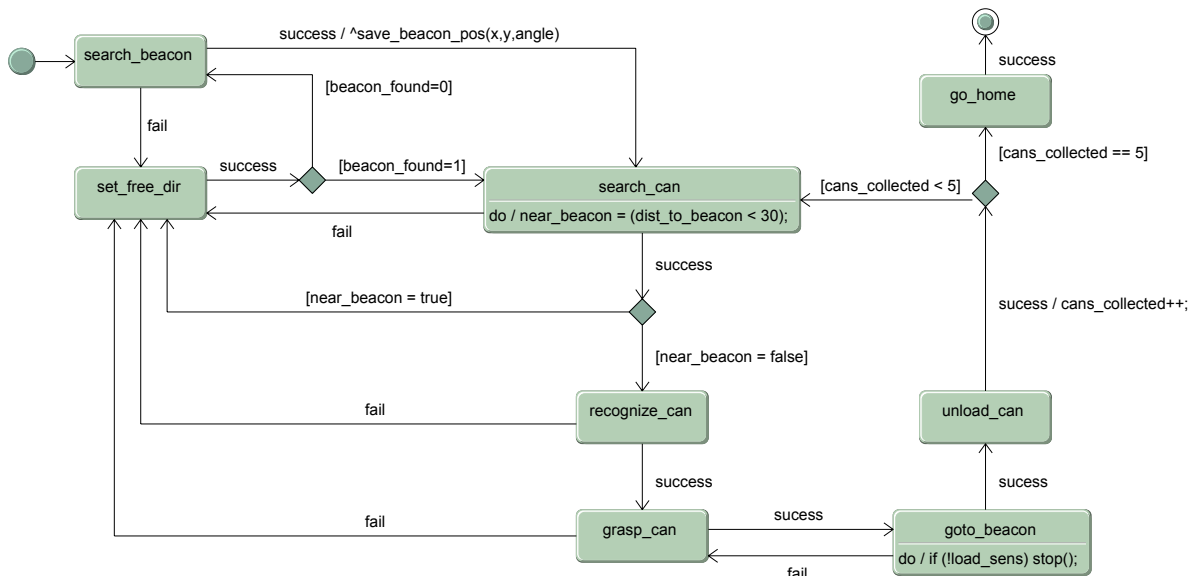


Figura 3. Statechart para la tarea de recolección de latas.

4. TRABAJOS RELACIONADOS

El control por comportamientos [Mataric 99] presenta cierta similitud con el enfoque desarrollado, puesto que tanto un comportamiento como un estado definen una unidad significativa de acción. Sin embargo, al modelar el sistema utilizando statecharts se define en forma explícita una secuencia temporal, mientras que en el control por comportamiento los comportamientos se ejecutan en forma concurrente. Además, un statechart define un plan en forma explícita, elemento rechazado en el control por comportamientos [Brooks 90].

Una relación más estrecha puede establecerse con arquitecturas como las descritas en [Bonasso et al. 97] y [Alami et al. 97]. Ambas definen básicamente tres niveles: uno reactivo o de habilidades básicas, un intermedio, que ejecuta secuencialmente conjuntos de dichas habilidades, y uno deliberativo, que establece objetivos a largo plazo. Así, ambas arquitecturas modelan el comportamiento de un robot descomponiéndolo en pasos discretos, como en SARA. Sin embargo, utilizan lenguajes formales para la definición del comportamiento, y no una representación explícita como en SARA. Por lo demás, estas arquitecturas están orientadas a sistemas complejos, por lo que su implementación en robots pequeños es poco factible.

En [Gini 96] se utilizan, aunque no de manera formal, estados para modelar una tarea en un robot construido para una competencia. También en [Arkin & MacKenzie 94] y [Kosecka & Christensen 95] se utilizan, con diversas variaciones, autómatas finitos

para modelar el comportamiento de un robot, pero no como parte integral de una arquitectura.

5. CONCLUSIÓN

La arquitectura SARA establece una organización clara y eficiente de los elementos de software del robot, proporcionando una metodología tanto para el desarrollo del sistema de control como para la posterior implementación de tareas particulares. Sus principales características son:

- **Baja complejidad**, por estar orientada a robots pequeños que posean pocos recursos computacionales y capacidad sensorial limitada.
- **Sencilla de utilizar**, por cuanto simplifica y sistematiza la implementación de tareas específicas.
- **Versátil**, puesto que permite implementar distintos tipos de tareas y favorece la reutilización de código.

La utilización de statecharts como elemento central de la arquitectura constituye un elemento novedoso, que proporciona una notación estándar poderosa y expresiva, adecuada para el modelamiento de comportamiento de un robot.

A pesar de que el enfoque desarrollado está orientado, en primera instancia, a robots sencillos, nada impide que sea escalable a sistemas más complejos; probablemente en tales sistemas serían útiles las capacidades de jerarquía y concurrencia de los statecharts, que no se exploraron en la implementación actual.

Referencias

- [Alami et al. 97] R. Alami, R. Chatila, S. Fleury, M. Ghallab, F. Ingrand, *An Architecture for Autonomy*, International Journal of Robotics Research, 1997.
- [Arkin & MacKenzie 94] R. Arkin, D. MacKenzie, *Temporal Coordination of Perceptual Algorithms for Mobile Robot Navigation*, IEEE Transactions on Robotics and Automation, Vol 10, No. 3, pp 276-286, Junio 1994.
- [Asada et al. 00] M. Asada, R. D'Andrea, A. Birk, H. Kitano, M. Veloso, *Robotics in edutainment*, Proceedings of the IEEE International Conference on Robotics and Automation 2000 (ICRA '00), Vol. 1, pp. 795 – 800, Abril 2000.
- [Bonasso et al. 97] R. Peter Bonasso, David Kortenkamp, David P. Miller, *Experiences with an Architecture for Intelligent, Reactive Agents*, Metrica, Inc. Robotics and Automation - NASA Johnson Space Center, 1997.
- [Brooks 90] Rodney A. Brooks, *Elephants Don't Play Chess*, Robotics and Autonomous Systems, Vol. 6, pp. 3-15, 1990..
- [Garrido 04] Ignacio Garrido, Danilo Bassi, *Utilización de Statecharts para el Modelamiento de Tareas de un Robot Móvil*, Anales de XV Congreso de la ACCA, Santiago de Chile, 2004.
- [Gini 96] Maria Gini, *Design and Building Autonomous Minirobots*, Department of Computing Science, University of Minnesota, 1996.
- [Harel 87] D. Harel, *Statecharts: A Visual Formalism for Complex Systems*, in Science of Computer Programming, Vol.8, pp. 231-274, Junio 1987.
- [Kaelbling 91] Leslie. P. Kaelbling, *A Situated Automata Approach to the Design of Embedded Agents*, SIGART Bulletin, Vol. 2, N° 4, pp 85–88, 1991.
- [Klassner & Anderson 03] F. Klassner, S. Anderson, *Lego MindStorms: Not Just for K-12 Anymore*, IEEE Robotics and Automation Magazine, June 2003.
- [Kosecka & Christensen 95] J. Kosecka, H. Christensen, *Discrete event systems for autonomous mobile agents*, Workshop on Intelligent Control, Julio 1993.
- [Maes 91] Pattie Maes, *The agent network architecture*, SIGART Bulletin, Vol 2, N°4, pp. 115–120, 1991.
- [Mataric 99] M. J Mataric, *Behavior-Based Robotics*, MIT Encyclopedia of Cognitive Sciences, pp. 74-77, 1999.
- [Pfeifer 97] Rolf Pfeifer, *Teaching Powerful Ideas with Autonomous Mobile Robots*, Journal of Computer Science Eduation, Vol. 7, No. 2, Universidad de Zurich, Suiza, 1997.