https://doi.org/10.17979/spudc.9788497498081.0821

Robótica móvil para el aprendizaje de conceptos de programación en tiempo real

C. Rodríguez[1], J.L. Guzmán[2], M. Berenguel[2], J.C. Moreno[2], F. Rodríguez[2], S. Dormido[1]
[1]Departamento de Informática y Automática, UNED, España, carlos@bec.uned.es
[2]Departamento de Informática, Universidad de Almería, ceiA3, CIESOL, España

Resumen

La enseñanza de asignaturas de programación en tiempo real puede resultar una difícil tarea debido a los múltiples conceptos teóricos que contiene. Este trabajo se centra en el uso de robots móviles tipo Lego Mindstorms NXT para la enseñanza de programación en tiempo real a estudiantes de grado. Como caso de estudio para mostrar la metodología de trabajo, se muestra el problema de control de estabilidad en tiempo real de un robot autobalanceado regulado mediante un controlador PI desarrollado en el lenguaje de programación Ada. En general, se obtuvieron críticas muy positivas en el cuestionario presentado a los estudiantes.

Palabras clave: Programación en tiempo real, robótica móvil, Lego Mindstorms NXT, Ada

1 INTRODUCCIÓN

La programación en tiempo real engloba muchos conceptos abstractos que pueden resultar difíciles de transmitir a los estudiantes. Por regla general, se suele utilizar un lenguaje de programación concreto como medio de aprendizaje, propiciando en algunos casos que el estudiante perciba la asignatura como una continuación de introducción a la programación. Sin embargo, el uso de sistemas físicos reales puede ayudar a afianzar los principales conceptos y a motivar a los alumnos en el desarrollo de este tipo de asignaturas [11, 12].

Una opción para motivar a los estudiantes y aclarar los conceptos puramente de tiempo real es la realización de un caso práctico que requiera de todos los contenidos de la asignatura. En este sentido, la robótica móvil resulta especialmente atractiva debido a la intuitiva realimentación que el estudiante obtiene de la aplicación de los algoritmos de tiempo real desarrollados.

Después de estudiar varias alternativas se optó por utilizar el paquete GNAT GPL para Lego Mindstorms NXT desarrollado por AdaCore. Dicho software es libre y permite explotar todas las características específicas de tiempo real del lenguaje de programación Ada en los robots móviles. Lego Mindstorms NXT (NXT) es un kit de iniciación a la robótica que permite a los estudiantes desarrollar aplicaciones de control en poco tiempo. Por esta razón, su uso es adecuado en asignaturas que duran unos pocos meses. Estas aplicaciones normalmente requieren unos plazos de ejecución temporales predefinidos, siendo deseable el uso de herramientas que integren características de tiempo real [2]. En la literatura, existen algunas aplicaciones de tiempo real de NXT orientadas principalmente a competición de robots [4, 6, 7]. En el ámbito docente, se han utilizado estos robots para la enseñanza del lenguaje de programación NXC para adquisición de datos y sistemas de control y tiempo real [5] y para el desarrollo de programas estructurados [8]. En este trabajo se ha optado por utilizar el lenguaje de programación Ada.

Ada es un lenguaje de programación en tiempo real ideado para su uso en sistemas empotrados de tiempo real. Dicho lenguaje integra concurrencia y características de tiempo real tales como tareas, paso síncrono de mensajes y uso de recursos compartidos. Por esta razón, es un lenguaje apropiado para asignaturas de programación en tiempo real. Junto con el perfil Ravenscar — un subconjunto de tareas de Ada destinado al desarrollo de aplicaciones con restricciones de tiempo real duras — hace que Ada sea el lenguaje ideal para trabajar con NXT [3].

Todo este conjunto de herramientas ha sido utilizado durante el segundo cuatrimestre del curso 2014/15 para el desarrollo de proyectos de control de trayectoria y estabilidad de robots móviles, obteniendo críticas muy positivas por parte de los estudiantes. Este trabajo está basado en [10] y presenta los resultados que obtuvieron un grupo de estudiantes de sistemas de tiempo real utilizando robots móviles. Como caso particular de estudio se muestra el control de estabilidad de un robot NXT autobalanceado.

El resto del trabajo está organizado como se indica a continuación. La sección 2 presenta las características del kit robótico NXT y las herramientas de desarrollo. El problema de control de estabilidad de un robot autobalanceable se aborda en la

sección 3. Por último, en la sección 4 se exponen las principales conclusiones del trabajo.

2 MATERIALES Y MÉTODOS

2.1 MARCO DE ENSEÑANZA

Sistemas de tiempo real es una asignatura del tercer curso del grado de ingeniería informática de la Universidad de Almería cuya dotación es de 6 créditos ECTS. Este grado constituye una carrera multidisciplinar que permite a los graduados abordar problemas relacionados con la computación desde distintas esferas de conocimiento. Los principales objetivos de la asignatura de sistemas de tiempo real son:

- Presentar los conceptos de sistemas de tiempo real en el ámbito industrial.
- Estudiar planificación temporal de recursos en un marco multitarea.
- Desarrollar aplicaciones que incluyan el análisis, diseño y codificación en un entorno de tiempo real.
- Aplicar el conocimiento adquirido en sistemas de tiempo real físicos.

Para cubrir todos los objetivos expuestos se optó por utilizar la siguiente metodología. Durante el primer mes se imparten clases teóricas en las que se presentan los principales aspectos de los sistemas de tiempo real. Dichas clases son además complementadas con ejercicios individuales. En los siguientes meses se combinan las clases teóricas con clases prácticas de programación. En el último mes se forman equipos de trabajo para realizar un caso de estudio real en el que se evalúan las habilidades adquiridas por los estudiantes. En el curso 2014/15 se formaron tres grupos de trabajo (dos de dos estudiantes y uno de tres estudiantes) para la realización del supuesto práctico.

2.2 KIT ROBÓTICO NXT

El kit de iniciación NXT contiene un controlador programable, varios sensores y actuadores, y algunas piezas mecánicas tales como ruedas. Dicho controlador, el cual se muestra en la Figura 1, cuenta con un procesador ARMv3 de 32 bits con 64 KB de RAM y 256 KB de memoria Flash. Además, posee: tres puertos de salida bidireccionales para conectar actuadores tales como motores eléctricos; cuatro puertos de entrada que soportan sensores digitales o analógicos; y un coprocesador de 8 bits que realiza tareas de bajo nivel como la generación de señales PWM y la

Figura 1: Controlador programable de LEGO MINDSTORMS NXT

conversión analógica/digital. Es posible interactuar con el controlador en tiempo de ejecución haciendo uso de cuatro botones de goma y de una pantalla LCD de 100 x 64 píxeles. Para más información visite el sitio web de LEGO MINDSTORMS (`http://mindstorms.lego.com`).

A continuación se describen todos los pasos necesarios para ejecutar una aplicación Ada de tiempo real desarrollada en el NXT.

2.3 HERRAMIENTAS DE DESARROLLO

AdaCore codificó los drivers para NXT en Ada y portó la cadena de herramientas del compilador GNAT a la arquitectura ARM utilizando parte del proyecto Open Ravenscar Real-Time Kernel (ORK+) desarrollado por un equipo del Departamento de Ingeniería de Sistemas Telemáticas de la Universidad Politécnica de Madrid (DIT/UPM) [2].

Con dichas herramientas se puede ejecutar código Ada en el NXT siguiendo estos pasos:

1. Incluir las librerías con el driver NXT de Ada en el proyecto software importando (al menos) el paquete **NXT.AVR**.
2. Generar un fichero GNU *make* (**Makefile.inc**) para compilar y generar la imagen binaria.
3. Reemplazar el firmware original del NXT por la imagen binaria previamente generada de la aplicación Ada para su ejecución desde la memoria RAM. Para llevar a cabo este paso es necesario utilizar el programa de arranque por defecto (SAM-BA Boot Assistant) mientras el NXT está conectado a través del puerto USB.

A continuación se presentan los resultados obtenidos por un grupo de estudiantes que desa-

rrollaron una aplicación de tiempo real para controlar un robot NXT autobalanceable.

2.4 RESULTADOS

Un robot autobalanceado es un vehículo de dos ruedas que se mantiene en posición vertical utilizando los motores eléctricos situados en su base. En la Figura 2 se muestra el robot NXT autobalanceado diseñado por un grupo de estudiantes de la asignatura de sistemas en tiempo real. Dicho robot está compuesto por un controlador principal, dos sensores y dos actuadores:

- Un *sensor de ultrasonidos* que se utiliza para avisar de posibles colisiones frontales.
- Un *sensor de luz* que proporciona una estimación del ángulo que forma el robot con la normal de la superficie. Dicha estimación solo puede ser considerada bajo ciertas condiciones específicas: la posición relativa del robot con respecto de la luz ambiental y la superficie de rodamiento deben ser apropiadas.
- Dos *motores eléctricos* que mueven las ruedas y permiten llevar a cabo el control de estabilidad. Ambos motores están físicamente conectados a través de una pieza mecánica y por tanto ambas ruedas se mueven a la misma velocidad.

La aplicación Ada desarrollada utiliza un paradigma de tiempo real con tres tareas paralelas:

1. La *tarea de control* se encarga de leer la entrada analógica que proporciona el sensor de luz, calcular la señal de control, y actualizar la velocidad de giro de las ruedas.
2. La *tarea de detección de colisiones* avisar de colisiones frontales a partir de la medida de distancia que obtiene del sensor de ultrasonidos.
3. La *tarea de la pantalla* muestra en la pantalla LCD información relativa a la ejecución del algoritmo de control y permite reconfigurar los parámetros del controlador utilizando los botones del NXT.

En la Figura 3 se muestra la red de Petri temporal utilizada para la planificación de tareas en la fase de diseño de la aplicación. En dicha figura se pueden apreciar las tres tareas concurrentes previamente descritas y la interacción entre las mismas utilizando paso de mensajes.

Figura 2: Robot NXT autobalanceable desarrollado por los alumnos

En cuanto a la tarea de control, se utilizó un algoritmo de realimentación de la señal controlada (ver Figura 4). En la figura, el ángulo de inclinación y ha de ser regulado cerca del punto de operación de referencia deseado r variando la potencia enviada a los motores u[1]. En la aplicación, la referencia se fija automáticamente al principio de la ejecución.

En este caso, se optó por el uso de un controlador PI para garantizar la estabilidad del sistema cuya función de transferencia en el dominio de Laplace viene dada por:

$$C(s) = K\left(1 + \frac{1}{T_i s}\right) \quad (1)$$

siendo K la ganancia proporcional y T_i el tiempo integral.

Además, se incluyeron otras dos características adicionales en el controlador para tratar con la saturación y la zona muerta de los motores:

[1]Se ha considerado que la misma señal de control es transmitida a ambos motores ya que están unidos físicamente

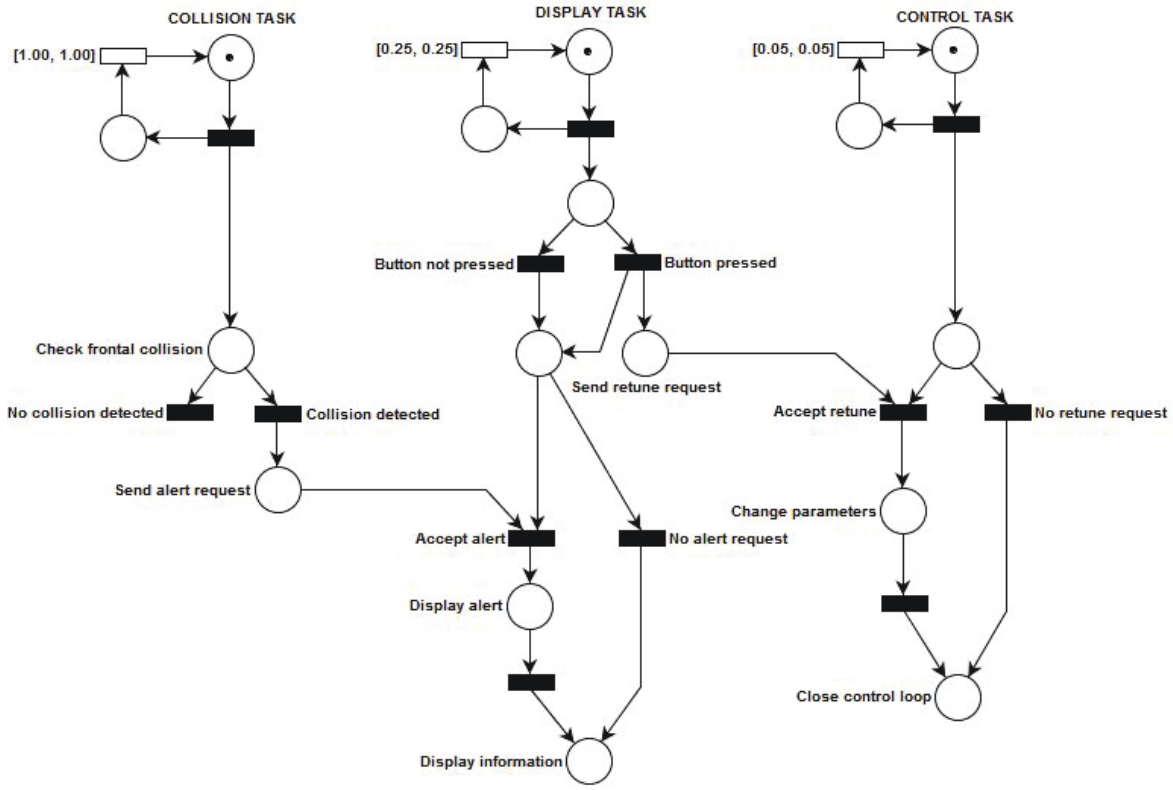

Figura 3: Red de Petri temporal utilizada para el diseño de la aplicación del robot NXT autobalanceado

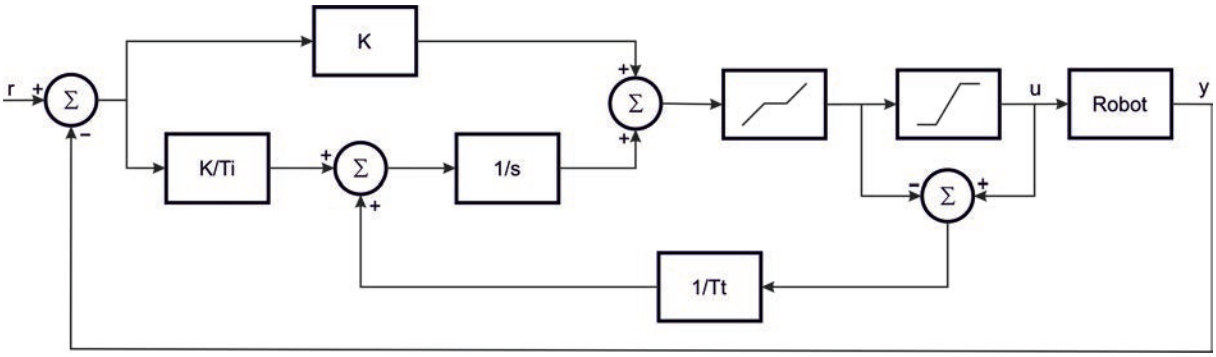

Figura 4: Diagrama de bloques del sistema de control del robot autobalanceado

1. Un *mecanismo de anti-windup* para evitar que el término integral acumule un error significativo durante la subida teniendo en cuenta que $u(t) \in [-100, 100]$. Dicho mecanismo se pondera en el controlador utilizando la variable T_t (ver Figura 4).

   siguiente expresión matemática:

   $$u(t) = \begin{cases} u(t) - \beta_1, & u(t) < 0 \\ 0, & u(t) = 0 \\ u(t) + \beta_2, & u(t) > 0 \end{cases} \quad (2)$$

   siendo $\beta_1 = 3.5$ y $\beta_2 = 5.0$ la señal de control mínima necesaria para mover las ruedas hacia atrás y hacia delante, respectivamente.

2. Una *estrategia para evitar la zona muerta* que garantice que una acción de control distinta de cero siempre genera movimiento en las ruedas. La estrategia considerada consiste en modificar la señal de control utilizando la

En futuras aplicaciones se pueden incluir en el controlador otras características que ayuden a mejorar la estabilidad del lazo de control tales como el término derivativo y filtros [1]. La sintonía de los parámetros del controlador se realizó de forma

heurística obteniendo los siguientes valores:

$$K = 22.5,$$
$$T_i = 8.2,$$
$$T_t = 10.0.$$

Por último cabe resaltar que la implementación final del controlador en el robot se realizó en el dominio del tiempo discreto utilizando un periodo de muestreo de 50 ms.

El lector interesado puede consultar un vídeo demostrativo de funcionamiento del robot propuesto en el siguiente enlace: `http://aer.ual.es/multimedia/videos/Docencia/Segway_Caso_practico.mp4`.

3 DISCUSIÓN

Los simuladores y laboratorios remotos tienen su lugar irreemplazable en el proceso educativo. Sin embargo, la realimentación física obtenida con el uso de dispositivos reales proporciona un valor añadido que en muchos casos supone un desafío motivacional para el estudiante.

La enseñanza de programación en tiempo real con el apoyo de la librería de Ada para LEGO MINDSTORMS NXT resulta muy últil para aclarar muchos de los conceptos teóricos de la asignatura. Desde un punto de vista de teoría de control, algunos fenómenos tales como la zona muerta de los motores o su saturación, que en ocasiones se desprecian al realizar simulaciones, tienen que ser resueltos por los estudiantes. Sin embargo, es necesario seguir trabajando en la librería de Ada ya que, por ejemplo, no existen buenas fuentes de documentación y el proceso de depuración de aplicaciones es complejo. A pesar de estas limitaciones, la experiencia resultó ser muy positiva a raíz de los comentarios recibidos por la mayoría de los estudiantes que cursaron la asignatura.

Por último, cabe destacar las ventajas y limitaciones que presenta el robot NXT autobalanceable que fue propuesto. Por un lado, es una solución económica ya que únicamente utiliza componentes del kit de iniciación. Sin embargo, la estimación del ángulo vertical a partir del sensor de luz es imprecisa ya que depende de la posición relativa del robot con respecto de la fuente de luz y de la homogeneidad de la superficie. Si es posible, es recomendable la adquisición de otros componentes tales como giróscopos para mejorar dicha estimación.

Para verificar la utilidad del uso de la metodología propuesta se presentó un cuestionario a los alumnos de la asignatura de sistemas de tiempo real

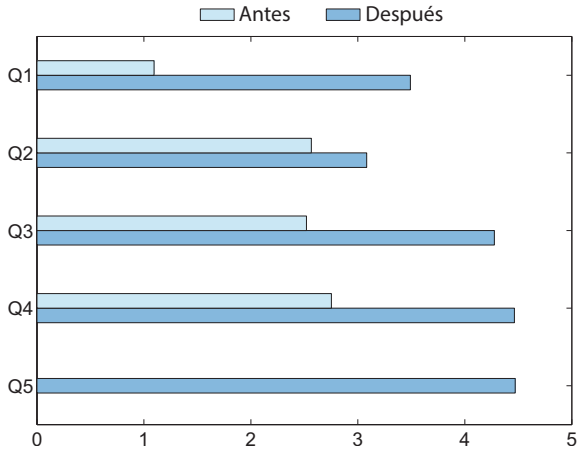

Figura 5: Resultados del cuestionario presentado a los estudiantes

(ver Tabla 1). En la Figura 5 se muestran los resultados medios obtenidos tras la realización del cuestionario por 7 estudiantes. En él se puede apreciar que los estudiantes consideraron muy positivamente la influencia del caso de estudio en el desarrollo de las competencias evaluadas.

Agradecimientos

Este trabajo ha sido parcialmente financiado por los siguientes proyectos: DPI2014-55932-C2-1-R y DPI2014-56364-C2-1-R del Ministerio de Ciencia e Innovación y fondos FEDER.

Referencias

[1] K.J. Åström and T. Hägglund. *Advanced PID Control*. ISA Press, Research Triangle Park, NC, 2006.

[2] Peter J. Bradley, Juan A. de la Puente, and Juan Zamorano. Ada user guide for LEGO MINDSTORMS NXT. *Ada User Journal*, 32(3):194–203, 2011.

[3] Alan Burns, Brian Dobbing, and Tullio Vardanega. Guide for the use of the Ada Ravenscar Profile in high integrity systems. *ACM SIGAda Ada Letters*, 24(2):1–74, 2004.

[4] Dictino Chaos, Jesús Chacón, José A. López-Orozco, and Sebastián Dormido. Virtual and remote robotic laboratory using EJS, MATLAB and LabVIEW. *Sensors*, 13(2):2595–2612, 2013.

[5] Ana Cruz-Martín, Juan A. Fernández-Madrigal, Cipriano Galindo, Javier A. González-Jiménez, Charbel Stockmans-Daou, and José L. Blanco-Claraco. A LEGO Mindtorms NXT approach for teaching at Data Acquisition, Control Systems

Tabla 1: Cuestionario presentado a los estudiantes

Q1	Valore de 1 a 5 su conocimiento acerca del lenguaje de programación Ada
Q2	Valore de 1 a 5 su conocimiento acerca de sistemas de control
Q3	Valore de 1 a 5 su conocimiento acerca de planificación de tareas
Q4	Valore de 1 a 5 su conocimiento acerca de sistemas robóticos
Q5	Valore de 1 a 5 la influencia del caso de estudio en el desarrollo de las competencias previas

Engineering and Real-time Systems undergraduate courses. *Computers & Education*, 59(3):974–988, 2012.

[6] Jesús M. Gómez-de-Gabriel, Anthony Mandow, Jesús Fernández-Lozano, and Alfonso J. García-Cerezo. Using LEGO NXT mobile robots with LabVIEW for undergraduate courses on Mechatronics. *IEEE Transactions on Education*, 54(1):41–48, 2011.

[7] Raffaele Grandi, Riccardo Falconi, and Claudio Melchiorri. Robotic competitions: Teaching robotics and real-time programming with LEGO mindstorms. In *Preprints of the 19th IFAC World Congress*, Cape Town, Agosto 2014.

[8] Wojciech Grega and Adam Pilat. Real-time control teaching using LEGO MINDSTORMS NXT Robot. In *Proceedings of the International Multiconference on Computer Science and Information Technology 2008*, páginas 625 – 628, Wisla, Octubre 2008.

[9] Torsten K. Iversen, KÅre J. Kristoffersen, Kim G. Larsen, Morten Laursen, Rune G. Madsen, Steffen K. Mortensen, Paul Pettersson, and Chris B. Thomasen. Mode-checking real-time control programs: Verifying LEGO MINDSTORMS systems using UPPAAL. In *Proceedings of the 12th Euromicro Conference on Real-Time Systems 2000*, páginas 147 – 155, Stockholm, Junio 2000.

[10] Carlos Rodríguez, José Luis Guzmán, Manuel Berenguel, Sebastián Dormido. Teaching real-time programming using mobile robots. In *11th IFAC Symposium on Advances in Control Education 2016*, Bratislava, Junio 2016.

[11] Payman Shakouri, Olga Duran, Andrzej Ordys, and Gordana Collier. Teaching fuzzy logic control based on a robotic implementation. In *10th IFAC Symposium on Advances in Control Education 2013*, páginas 192 – 197, Sheffield, Agosto 2013.

[12] Jaroslav Sobota, Roman Pišl, Pavel Balda, and Miloš Schlegel. Raspberry Pi and Arduino boards in control education. In *10th IFAC Symposium on Advances in Control Education 2013*, páginas 7 – 12, Sheffield, Agosto 2013.