INGENIERÍA ELECTRÓNICA

Diseño y prueba de un robot móvil con tres niveles de complejidad para la experimentación en robótica

Felipe Gómez*, Francisco Muñoz*, Beatriz E. Florián, Carlos A. Giraldo**, Eval B. Bacca-Cortes*§**

** Escuela de Ingeniería Eléctrica y Electrónica, Universidad del Valle, Cali, Colombia*
*** Escuela de Ingeniería de Sistemas, Universidad del Valle, Cali, Colombia*
§ e-mail: evbacca@univalle.edu.co

(Recibido: Mayo 28 de 2008 - Aceptado: Noviembre 12 de 2008)

Resumen

Un robot móvil (llamado UV-BOT), fue diseñado en la Universidad del Valle para la experimentación en robótica y puede ser usado por personas con o sin conocimiento en robótica. UV-BOT es un robot diferencial y para uso en interiores; cuenta con sensores de proximidad, detección de luz, localización y comunicaciones, y su velocidad es controlada mediante dos controladores PI, uno en cada rueda. El *firmware* del robot fue diseñado usando un micro-núcleo de tiempo real conocido como FreeRTOS, que soporta tres niveles de complejidad, los cuales poseen un conjunto de funcionalidades desde el punto de vista de percepción, movilidad, comunicación, programación y disponibilidad de demos. Estas funcionalidades son empleadas por la interfaz de usuario para programar el robot gráficamente (nivel básico), usando XML (nivel intermedio) o C (nivel avanzado). En su nivel avanzado, el robot soporta la programación orientada a comportamientos. Estos comportamientos fueron utilizados para realizar las pruebas de desempeño del robot móvil. Como resultado de estas pruebas, se encontró que el sistema de localización de UV-BOT muestra una incertidumbre promedio de 17 cm. Los comportamientos de retorno (por posición y luz) y evasión de obstáculos muestran errores promedio de 15 y 12 cm, respectivamente. De este modo, con UV-BOT se pueden implementar exitosamente tareas básicas en robótica cooperativa.

Palabras clave: Robots móviles, Programación de comportamientos, Robótica cooperativa, Educación en ingeniería.

ELECTRONICS ENGINEERING

Design and testing of a mobile robot with three levels of complexity for robotics experimentation

Abstract

A mobile robot (called UV-BOT) was designed at Universidad del Valle for robotics experimentation and can be employed for users with or without any previous knowledge on robotics. UV-BOT is a differential and for indoors-use-only robot; it has sensors for proximity, light detection, localization, and communications, and its speed is controlled by means of two PI controllers, one for each wheel. The robot's firmware was designed by using a real time micro-kernel, known as FreeRTOS, which supports three levels of complexity that have a set of functionalities from the standpoint of perception, mobility, communication, programming, and demos availability. These functionalities are employed by the user interface to program the robot graphically (basic level), using XML (intermediate level) or C (advanced level). At its advanced level, UV-BOT supports behavior-oriented programming. Behaviors were used to carry out the performance tests for the mobile robot. As an outcome of these tests, it was found that the localization system of UV-BOT exhibits an average error of 17 cm. Homing (by position and light) and obstacle- avoidance behaviors exhibit average errors of 15 and 17 cm, respectively. In this way, tasks in cooperative robotics can be implemented successfully.

Keywords: Mobile robots, Behavior programming, Cooperative robotics, Engineering education.

1. Introducción

La robótica móvil es un área de investigación y desarrollo de alta exigencia. Hace unos años, era un tema casi exclusivo de las universidades y grandes empresas. A lo largo de los últimos años, el tema ha estado migrando desde estas entidades al público en general, lo cual puede verse en el aumento de sitios comerciales donde se venden módulos de robots para armar, de diferentes grados de complejidad. Paralelo a esto, el empleo de la robótica móvil como herramienta en la educación de niños y jóvenes, tanto a nivel nacional como internacional, se ha incrementado notoriamente. Es el caso de programas de educación infantil, se pueden mencionar los siguientes: el programa de NASA *Mars Exploration Rover Mission: Classroom* (NASA, 2008) para la exploración de otros mundos, las escuelas de robótica de LEGO (LEGO, 2008), las experiencias de robótica en la Universidad de los Niños (EAFIT, 2007), los cursos de robótica en Maloka (MALOKA, 2006) y los semilleros de robótica de la Universidad del Valle (UNIVALLE, 2003).

Este impacto de la robótica móvil en la educación de niños y jóvenes se debe a la gran motivación que el área despierta y a las expectativas que inspira. Es innegable la naturaleza multidisciplinaria que la robótica posee. Esta naturaleza constituye la piedra angular para usar la robótica móvil como herramienta para la enseñanza de una amplia variedad de conceptos como programación, estructuras mecánicas, electrónica, ciencias básicas, comunicaciones, etc. En términos generales, la robótica móvil permite, a través de motivantes experiencias prácticas, iniciar científicamente a niños y jóvenes, lo cual actualmente es una necesidad ante el desinterés nacional y global de los jóvenes en estudiar profesiones en ingeniería (Ulloa, 2008; Londoño, 2008).

Hace ya cuatro años la Universidad del Valle, en convenio con la Biblioteca Departamental Jorge Garcés Borrero, desarrolla los semilleros de robótica, en los cuales aproximadamente 300 niños y jóvenes de diferentes edades se han iniciado científicamente a través de la robótica móvil. A lo largo de esta experiencia, se observó que una vez el módulo LEGO Mindstorms 2.0 (LEGO, 2008) ha cumplido las expectativas como juguete, los niños y jóvenes desean una plataforma con más sensores y posibilidades de programación.

Comercialmente, se encuentra una amplia gama de robots móviles orientados al entretenimiento, desarrollo de pequeñas aplicaciones e investigación. La Tabla 1 muestra un consolidado de las características de varias plataformas usadas comúnmente para educación e investigación. Estas plataformas fueron seleccionadas teniendo en cuenta como criterios la definición de las herramientas de programación y el soporte de documentos educativos. Sin embargo, las plataformas con las que normalmente se cuenta para educación, tienen una vida útil muy pequeña y pronto dejan de brindar la posibilidad de más experiencias para el niño o joven, lo cual precisamente se observó a lo largo de los semilleros de robótica. En contraste, las plataformas orientadas a la investigación poseen interfaces de programación más complicadas de manejar para alguien sin conocimientos previos en robótica y son realmente diseñadas para público más preparado en el tema. Entonces, en este trabajo se presenta el diseño y prueba de un robot móvil con capacidades de percepción, actuación y comunicaciones básicas, programable desde el PC a través de una interfaz amigable y que integra (a nivel de *firmware* e interfaz de usuario) diferentes niveles de complejidad para el aprendizaje de la robótica móvil. Estos niveles son: básico, para usuarios con ninguna experiencia en robótica; intermedio, para usuarios con experiencias previas en robótica; y avanzado, para usuarios con conocimientos previos en robótica.

La herramienta propuesta responde a algunas de las principales fortalezas de la robótica pedagógica como son la integración de distintas áreas del conocimiento, la operación con objetos manipulables y concretos, la apropiación de un lenguaje gráfico, la operación y el control de diferentes variables de manera sincrónica, el desarrollo de un pensamiento sistémico, la creación de entornos de aprendizaje, la construcción y evaluación de sus propias estrategias de adquisición del conocimiento bajo

Tabla 1. Resumen de características de plataformas para educación e investigación.

	PLATAFORMAS PARA EDUCACIÓN			PLATAFORMAS PARA INVESTIGACIÓN		
PLATAFORMA	**SR1** 	**LEGO Mindstorms RIS** 	**HEMISSON** 	**KHEPERA II** 	**AMIGOBOT** 	**P3-DX** 
Programación	BasicX	Herramienta propia, C y JAVA	BotStudio	Webots, C	C y JAVA	C y JAVA
Procesador	BasicX-24	Hitashi H8/3292 16 MHz	PIC16F877 20 MHz	Motorola 68331 25 MHz	Renesas SH7144	Hitashi H8S
Memoria	EEPROM (32 kbyte) RAM (400 bytes)	32 kbytes de RAM (externos)	–	Flash (512 kbytes) RAM (512 kbytes)	1 Mbyte de flash	1 Mbyte de flash
Locomoción	2 motores DC	2 motores DC	2 motores DC	2 motores DC con *encoders* incrementales	2 motores DC con *encoders* incrementales	2 motores DC con *encoders* incrementales
Sensores	2 de contacto 1 de inclinación 2 de luz 1 de infrarrojo 1 de ultrasonido 1 de temperatura	2 sensores de contacto, 1 sensor de luz 1 sensor de rotación	8 de luz infrarrojos, 6 de detección de obstáculos infrarrojos, 2 de detección de línea	8 de proximidad infrarrojos	8 de sonar	8 de sonar
Alimentación	6 baterías AA de 1.5 V	6 baterías AA de 1.5 V	1 batería de 9V	1 batería recargable de NiMH	1 batería de 12 V	1 batería de 12V
Comunicación	Puerto serial	Puerto infrarrojo	Puerto serial	Puerto serial	Puerto serial	Puerto serial
Dimensiones	–	–	12 cm de diámetro	70 mm de diámetro 30 mm de alto	33 cm de largo 28 cm de ancho 15 cm de alto	44 cm de largo 38 cm de ancho 22 cm de alto
Peso	–	–	200 g	80 g	3.6 kg	9 kg

una orientación pedagógica, el crecimiento personal, el aprendizaje del proceso científico y el modelado matemático (Esteban, 2000).

De esta forma, se ayuda a motivar el proceso de aprendizaje en los diversos niveles de la educación, desde temprana edad hasta la vida profesional, estimulando áreas del desarrollo como la cognición y el lenguaje, utilizando elementos que despiertan interés y motivación por aprender, y sin obligar a mecanizar conceptos abstractos sin que ellos sean analizados y observados para su aplicación práctica. De este modo, esta herramienta maneja un contexto concreto para su aprendizaje.

Este artículo está estructurado de la siguiente manera. En primer lugar, se hace una descripción del hardware del robot, describiendo brevemente su estructura mecánica, su modelo cinemático y dinámico, sus sistemas de percepción, su actuación y sus comunicaciones. En segundo lugar, se discuten los requerimientos bajo los cuales se diseñaron el *firmware* y la interfaz de usuario de la herramienta. En tercer lugar, se describen tanto el *firmware* como la interfaz de usuario diseñados. Finalmente, se describen las pruebas y resultados de la herramienta, que están orientados a demostrar el cumplimiento de los requerimientos y la funcionalidad de la herramienta.

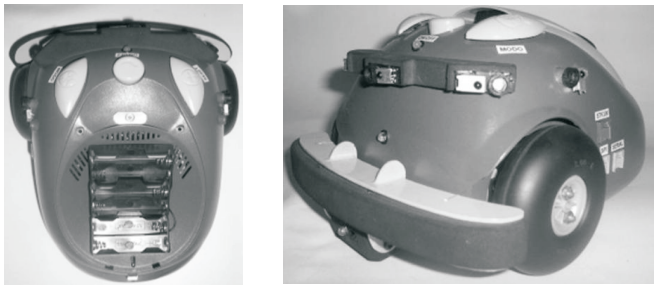
2. Descripción del hardware del robot

2.1 Estructuras mecánica y electrónica

Los dos robots móviles implementados presentan una forma ovalada que tiene 24.5 cm de largo, 18 cm de ancho y 13 cm de alto. El chasis está construido de un material plástico que, en la parte superior, tiene una tapa removible para alojar las baterías que alimentan el robot y tres botones:

encendido, reinicio y operación. El botón de encendido tiene forma circular y está en el centro; el botón de reinicio se encuentra a la derecha del botón de encendido y se usa para reiniciar una aplicación recientemente descargada. El botón de operación se encuentra a la izquierda del botón de encendido y se usa para determinar si se quiere descargar una nueva aplicación o ejecutar la que está descargada (Figura 1a). Al lado izquierdo, el robot posee tres conectores tipo RJ: uno para la descarga de programas al robot, uno para la actualización del *BootLoader* y otro que usa comunicaciones tipo SPI para futuras expansiones. La Figura 1b muestra el diagrama de bloques del hardware del robot. En esta figura se observan 5 componentes principales: el sistema de percepción (proximidad, luz y contacto), el sistema de actuación [servo-motores de DC con sus codificadores (*encoders*) incrementales], el sistema de comunicaciones (de luz infrarroja, RS232 y un pequeño alto-parlante para interacción con el usuario), la alimentación para todo el sistema y el microcontrolador.

a)



b)

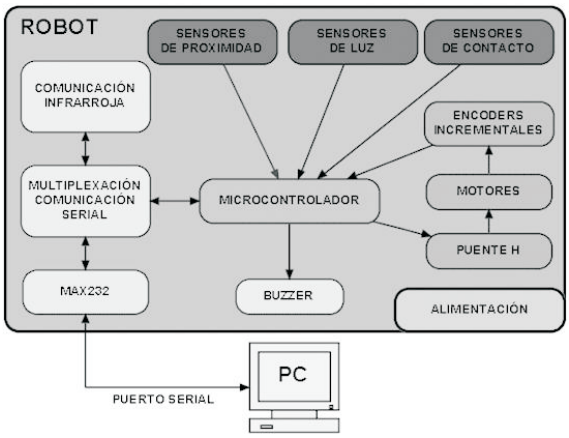


Figura 1. (a) Estructura mecánica del UV-BOT. (b) Diagrama de bloques de los componentes electrónicos del UV-BOT.

2.2 Sistema de percepción

UV-BOT posee un sistema de percepción compuesto por sensores de proximidad infrarrojos, sensores de contacto, sensores de intensidad de luz y odómetros en cada rueda. Estos sensores se seleccionaron teniendo en cuenta criterios como economía, mantenimiento de las dimensiones reducidas del robot, facilidad de ofrecer al niño o joven herramientas suficientes para realización de experiencias interesantes. De acuerdo con la revisión de las plataformas comerciales, estos sensores son los más comúnmente empleados.

Sensores de proximidad. La Figura 2a muestra la distribución de los sensores de proximidad. Dos de ellos se encuentran a cada lado del robot, otro en el frente y otros dos a 45º del sensor central. Para su implementación, se usaron dispositivos *Sharp* GP1UD28YK. Su rango de activación se encuentra alrededor de los 20 cm medidos desde el sensor.

Sensores de intensidad de luz. La Figura 2b muestra la distribución de los sensores de intensidad de luz. Dos de ellos se encuentran a cada lado del robot y otro en el frente. El sensor del frente está ubicado en un soporte que puede ser rotado hacia adelante o hacia el piso, de tal manera que puede usarse para implementar algoritmos sencillos de seguimiento de trayectorias con alto contraste. Estos sensores se implementaron usando fotorresistencias y su sensibilidad fue medida en el robot obteniéndose como resultado 0.353 cd / V.

Sensores de contacto. La Figura 2c muestra la distribución de los sensores de contacto. Todos ellos se ubicaron en el parachoques frontal del robot, ampliando el área de contacto.

Sistema de odometría. La Figura 3 muestra los planos coordenados global y del robot, los cuales son asumidos para el cálculo de la odometría. Los odómetros son codificadores incrementales ubicados en cada rueda del robot. Ellos brindan una resolución de 135 pulsos por giro de la rueda.

2.3 Modelos cinemático y dinámico

UV-BOT es un robot de tipo diferencial, con un soporte trasero y para ser usado en ambientes interiores. Según Ollero (2007), el modelo cinemático de un robot diferencial puede describirse mediante las expresiones:

$$v = \frac{v_d + v_i}{2} = \frac{(w_d + w_i)c}{2} \qquad (1)$$

$$w = \frac{v_d - v_i}{b} = \frac{(w_d - w_i)c}{b} \qquad (2)$$

donde c es el radio de las ruedas (3.81 cm para este caso), b (18 cm para este caso) es la separación de las ruedas, v_i es la velocidad lineal de la rueda izquierda, v_d es la velocidad lineal de la rueda derecha, w_i es la velocidad angular de la rueda izquierda, w_d es la velocidad angular de la rueda derecha, v es la velocidad lineal del robot y w es la velocidad angular del robot.

El sistema de locomoción del robot se basó en servo-motores modificados *Futaba* S3003, los cuales se basan en un motor de corriente directa (DC), cuyo modelo de primer orden es bien conocido (Ogata, 1996) y que se usó para establecer el modelo dinámico del robot.

a) b) c)

Figura 2. (a) Distribución de los sensores de proximidad. (b) Distribución de los sensores de intensidad de luz. (c) Distribución de los sensores de contacto.

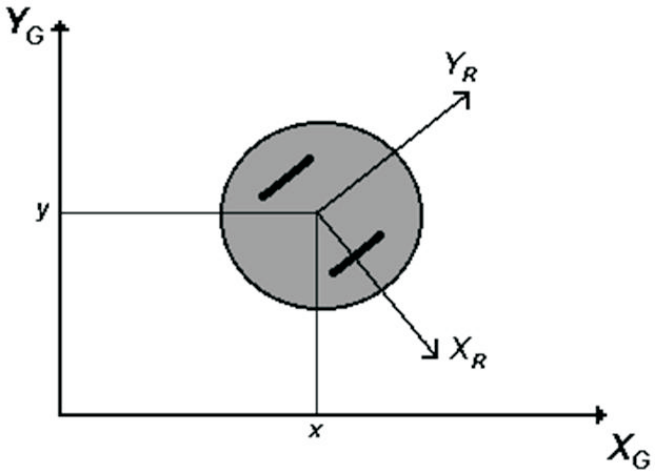


Figura 3. Planos coordenados para el cálculo de la odometría.

Después de realizar una identificación lineal usando la respuesta al escalón para cada uno de los motores, se obtuvieron los modelos expresados por las Ecs. (3) y (4).

$$V_D(s) = \frac{K}{\tau\, s+1} U_D(s) = \frac{0.1647}{0.161 s+1} U_D(s) \tag{3}$$

$$V_I(s) = \frac{K}{\tau\, s+1} U_I(s) = \frac{0.1694}{0.177 s\, +\, 1} U_I(s) \tag{4}$$

Estos modelos poseen tiempos de establecimiento de 0.644 y 0.651 s, respectivamente. Dado que los modelos son bastante similares, se unificó el modelo de la planta al promediar sus parámetros de primer orden, obteniendo las Ecs. (5) y (6), que corresponden al modelo continuo y discreto de la planta, respectivamente. El tiempo de muestreo usado fue de 250 ms.

$$V(s) = \frac{K}{\tau\, s+1} U(s) = \frac{0.17}{0.161 s+1} U(s) \tag{5}$$

$$P(z) = \frac{1.0559\, z}{z - 0.211655} \tag{6}$$

En las ecuaciones anteriores, V_D es la velocidad lineal de la rueda derecha, V_I es la velocidad lineal de la rueda izquierda, U_D es el voltaje aplicado al motor derecho, U_I es el voltaje aplicado al motor izquierdo, K es la ganancia de los modelos y τ es su constante de tiempo. Para mejorar el tiempo de establecimiento a 0.55 s y mejorar el error de estado estacionario, se diseño un control digital tipo PI usando el modelo de la planta mostrado en la Ec. (6). La función de transferencia del controlador digital se muestra en la Ec. (7) y la función de transferencia en bucla cerrada teniendo en cuenta el retenedor de orden cero se muestra en Ec. (8).

$$C(z) = 9.334\left(\frac{1.1553\, z - 0.1555}{z - 1}\right) \tag{7}$$

$$V(z) = \frac{0\ 3898\, z - 0.0773}{z^2 - 1.6018 z + 0.2893} \tag{8}$$

Las Figuras 4 y 5 muestran la respuesta del motor izquierdo y derecho a un escalón de velocidad de 17 cm / s. Las figuras muestran las respuestas en pruebas reales, a partir de las cuales se pudo medir un tiempo de establecimiento de 0.6 s y un sobrepaso del 3.2 %. La pequeña oscilación presente alrededor de los 5 s es causada por un cambio en las características del terreno, pero que fue útil con el fin de observar la capacidad de regulación del controlador.

2.4 Sistema de comunicaciones

La Figura 6 muestra el sistema de comunicaciones infrarrojas que los robots poseen en el momento que están ejecutando alguna aplicación. El emisor es un LED infrarrojo que se encuentra en el frente del robot y el receptor es un módulo *Sharp* GP1UD28YK, el cual está ubicado en la parte trasera. Cuando el robot se encuentra en modo de programación, el conector RJ11 de uno de los costados es usado para la descarga del programa. En cualquiera de los dos casos se usa la misma UART del microcontrolador *ATMEGA32.*

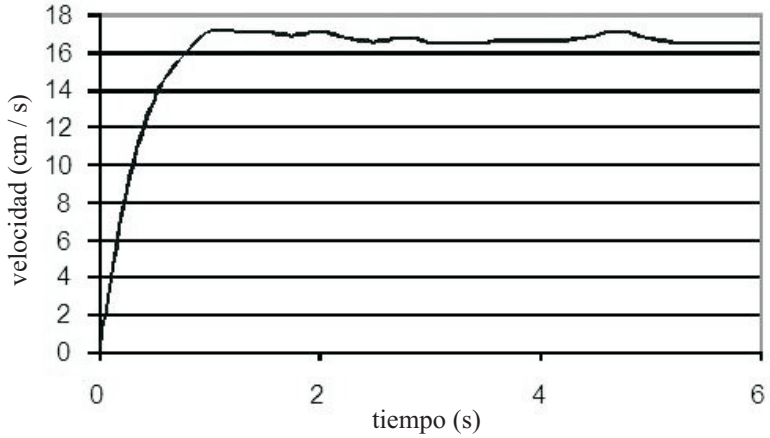


Figura 4. Respuesta real a un escalón de velocidad del motor derecho.

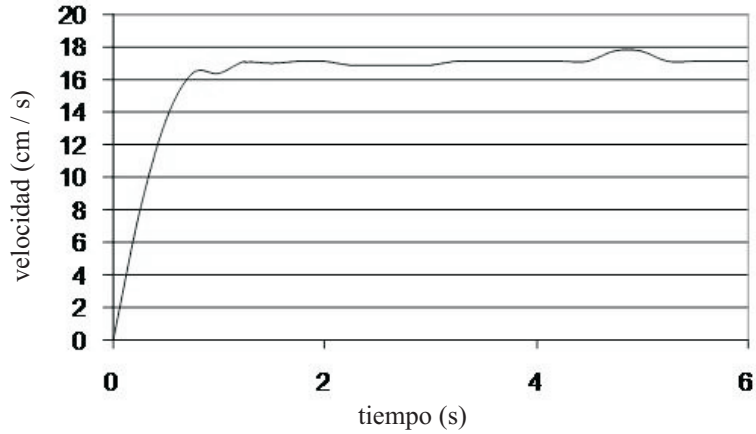


Figura 5. Respuesta real a un escalón de velocidad del motor izquierdo.

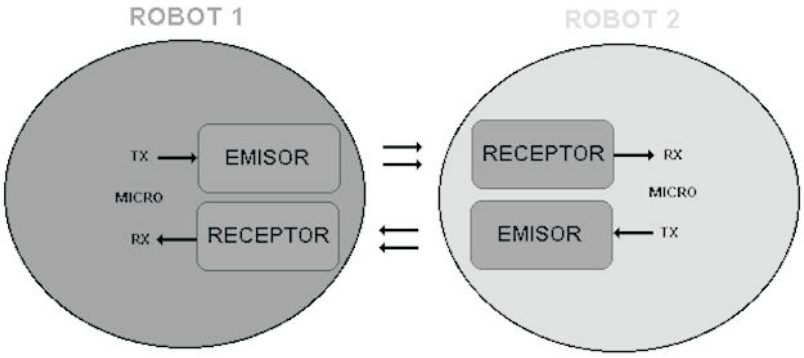


Figura 6. Comunicación infrarroja entre robots.

Dependiendo del modo de operación del robot, se selecciona como fuente de datos ya sea el conector RJ11 (programación) o el medio infrarrojo (ejecución de la aplicación).

3. Requerimientos para la experimentación en robótica

La herramienta propuesta fue diseñada e implementada con el fin de cumplir ciertos requerimientos, los cuales son producto de la experiencia de nuestro grupo de investigación en percepción y sistemas inteligentes en el desarrollo de los semilleros de robótica, y de la necesidad ya mencionada de brindar al usuario un esquema de trabajo de complejidad creciente que va desde el nivel básico hasta el nivel avanzado. A continuación, se detallan los requerimientos que el *firmware* y la interfaz de usuario tuvieron en cuenta para el desarrollo de la herramienta en 5 ejes fundamentales: movilidad, percepción, comunicaciones, programación y disponibilidad de demos.

3.1 Nivel de complejidad básico

El nivel básico está diseñado para que las funcionalidades de movilidad, percepción, comunicaciones y programación del robot sean invocadas sin mayor dificultad por el usuario, brindando una comprensión global de los estímulos y acciones del robot. La Tabla 2 muestra un resumen de sus características.

A este nivel, no se realizan procesos complejos de las señales de entrada o salida; se toma el flujo de información en bruto.

3.2 Nivel de complejidad intermedio

El nivel intermedio brinda la posibilidad de controlar al robot más delicadamente, permitiendo movimientos que tienen en cuenta una distancia o ángulo determinado. Se introduce el manejo del movimiento en un plano, comunicaciones con mensajes más complejos (tramas) y funciones de programación con operaciones lógicas. A este nivel, también se cuenta completamente con las funcionalidades del nivel básico. La Tabla 3 muestra un resumen de sus características. El objetivo de este nivel es comenzar a usar el robot como una herramienta capaz de realizar tareas que son un conglomerado de percepciones y movimientos simples introducidos en el nivel básico.

3.3 Nivel de complejidad avanzado

El nivel avanzado hace uso de todas las características anteriores y las usa para construir una interfaz de software orientada a la programación por comportamientos (Brooks, 1986). Se usan esquemas de percepción, hasta 7 diferentes comportamientos básicos que pueden ser fusionados (usando un árbitro colaborativo) y esquemas de actuación que modifican la velocidad lineal y angular del robot.

Tabla 2. Requerimientos del nivel de complejidad básico.

Tema	Funcionalidad
Movilidad	• Encendido / apagado de motores. • Variación de giro y velocidad de motores. • Giros básicos: aleatorio, adelante, derecha, izquierda y atrás por un tiempo dado.
Percepción	• Lecturas binarias (activo / no activo) de los sensores infrarrojos y de contacto. • Detección de umbrales de luz: iluminado u oscuridad para cada uno de los sensores de luz.
Comunicaciones e Interacción	• Tocar canciones y notas. • Enviar y recibir mensajes por IR.
Programación	• Definición de variables. • Operaciones aritméticas básicas. • Control de flujo de programas: while, for, if, repeat, etc.
Predefinidas	• Funciones para encontrar la luz u oscuridad. • Función para seguir objetos. • Función para evadir obstáculos.

Tabla 3. Requerimientos del nivel de complejidad intermedio.

Tema	Funcionalidad
Movilidad	• Avanzar y retroceder una distancia determinada. • Girar un ángulo determinado. • Avanzar, retroceder o girar hasta que un sensor perciba determinada condición.
Percepción	• Obtención de la posición XY del robot. • Obtención de la velocidad de desplazamiento del robot.
Comunicaciones e Interacción	• Envío y recepción de códigos de identificación de otros robots. • Determinación del momento adecuado para transmitir un mensaje.
Programación	• Operaciones lógicas (AND, OR, XOR). • Desplazamientos a la derecha o izquierda. • Operaciones aritméticas como potenciación y raíz cuadrada.
Predefinidas	• Desplazamiento del robot a una coordenada XY determinada.

Tabla 4. Requerimientos del nivel de complejidad avanzado.

Tema	Funcionalidad
Movilidad	• Modificación de la velocidad lineal y angular del robot. • Modificación de la velocidad lineal de las ruedas. • Modificación de las características de los controladores PI de las ruedas.
Percepción	• Representación vectorial de los sensores infrarrojos y de contacto. • Obtención del sector angular libre de obstáculos. • Obtención del sector angular con más o menos iluminación. • Obtención de la orientación relativa hacia un objetivo en el plano XY.
Comunicaciones e Interacción	• Idénticas a las del nivel intermedio.
Programación	• En este caso las estructuras de programación, definición de variables y operaciones sobre éstas son usadas teniendo en cuenta la sintaxis de ANSI C.
Predefinidas	• Comportamientos básicos de: evasión de obstáculos, adición de ruido, retorno por luz, retorno por posición, seguimiento de objetos, seguimiento de muros y escape. • Árbitro colaborativo para la fusión de las respuestas de los comportamientos. • Planificador de prioridades con el fin de determinar la importancia relativa de los comportamientos.

Además, se introduce el concepto de campos de potencial (Khatib, 1986), muy difundido en el ámbito de la navegación con robots móviles. El objetivo del nivel avanzado es usar al robot como una herramienta para solucionar un problema real. La Tabla 4 muestra estas características.

4. Descripción del *firmware*

La Figura 7 muestra la estructura del *firmware* de los robots móviles. Este fue desarrollado sobre un micro-núcleo de tiempo real de libre distribución llamado FreeRTOS (FreeRTOS, 2005), sobre el cual se implementaron los siguientes procesos:

• **Sensores IR**: para capturar y procesar la información de los sensores de proximidad infrarrojos.

• **PWM motor derecho**: para generar la señal de PWM necesaria para mover el motor derecho.

• **PWM motor izquierdo**: para generar la señal de PWM necesaria para mover el motor izquierdo.

• **Sensores de luz y contacto**: para capturar y procesar la información de los sensores de luz y contacto.

• **Comunicaciones**: para enviar y recibir información a través del sistema de comunicaciones infrarrojas.

• **Odometría y control de movimiento**: para determinar la posición y orientación del robot cada 250 ms y para determinar la velocidad de desplazamiento del robot con los controladores PI.

• **Eventos** (modificable por el usuario): para ubicar todos los eventos que pueden lanzar los sensores del robot, por ejemplo: detección de obstáculos en alguno de los sensores de proximidad o contacto, detección de ciertos niveles de iluminación u oscuridad, llegada de un mensaje por el sistema de comunicaciones, etc. A cada evento se le asocia una serie de instrucciones que el usuario especifica en las interfaces de programación.

• **Principal** (modificable por el usuario): para ubicar el programa que el usuario quiere ejecutar y que es concebido en alguna de las tres interfaces de programación.

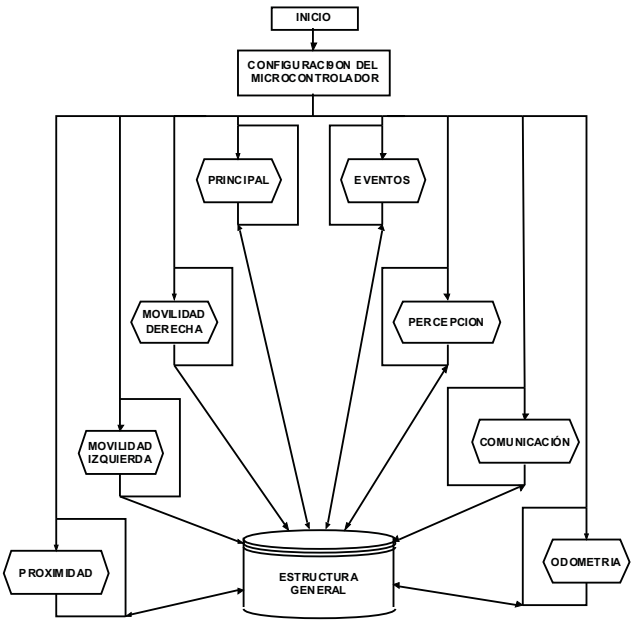


Figura 7. Esquema de tareas en el firmware del robot.

La Figura 7 muestra un bloque llamado e*structura general*, que es una estructura de datos en la cual todas las tareas leen y escriben datos. Este proceso de lectura y escritura es sincronizado por un mecanismo de semáforos que FreeRTOS ofrece con el fin de mantener la coherencia de la información. La estructura de datos alberga información como velocidades actuales de las ruedas derecha e izquierda, velocidad lineal del robot, velocidad angular del robot, valores de activación de los sensores infrarrojos y de contacto, valores actuales de los sensores de luz, valores de los controladores PI de las ruedas derecha e izquierda, dirección de giro de los motores, posición en el plano XY y orientación del robot. De esta manera, las tareas antes mencionadas son las únicas que interactúan con el hardware del robot, mientras que la interfaz de software del usuario sólo accede a la estructura de datos.

5. Descripción funcional básica de la interfaz de usuario

El aplicativo de software (ver diagrama de bloques en la Figura 8) permite la programación del robot en tres niveles de conocimiento progresivos: programación gráfica, programación basada en XML y programación en ANSI C. Como el programa final que se compila y descarga en el robot debe ser escrito en ANSI C, el aplicativo de software se encarga de hacer la traducción transparente entre los lenguajes utilizados en los dos niveles inferiores de conocimiento.

Para usuarios novatos se brinda una interfaz de programación gráfica. Para usuarios intermedios se brinda una interfaz de programación para código XML pues el lenguaje de programación del robot a nivel intermedio fue creado como un lenguaje basado en XML. Para usuarios avanzados se brinda una interfaz de programación para código ANSI C. El aplicativo permite al usuario generar y ver la traducción de código de sus programas desde el nivel de novatos hacia niveles superiores de aprendizaje.

Además de las tres interfaces diferentes de programación, el aplicativo de software presenta un módulo de entrenamiento mediante la descripción de problemas en forma de retos. El usuario debe superar estos problemas mediante la creación de un programa para cada reto. A continuación se describe funcionalmente cada interfaz desde el punto de vista de lo que encontrará el usuario al ejecutar el aplicativo.

5.1 Programación gráfica

En esta interfaz de programación, las rutinas, condiciones y funciones del robot se representan mediante íconos en forma de bloque, organizados

mediante pestañas por categorías. La Figura 9 muestra una ventana de esta interfaz. Al lado izquierdo se pueden escoger los bloques según su categoría. Los bloques se arrastran hasta el espacio de programación, donde se puede construir el programa mediante la unión y configuración de bloques. Al lado izquierdo también están los controles para la magnificación (*zoom*), para la navegación a lo largo del programa y para la eliminación de bloques del espacio de programación. En la parte central de la pantalla se arrastran los bloques para la construcción del programa, y en la parte inferior se configuran los datos de entrada de cada bloque previamente seleccionado.

5.2 Programación con XML

Se definió un lenguaje basado en XML para utilizar como lenguaje intermedio. Este lenguaje permite especificar todas las funciones y ciclos que el robot puede realizar. El lenguaje creado es extensible y las rutinas de traducción hacia los otros dos lenguajes del aplicativo (gráfico a XML, XML a ANSI C) también se expresan mediante transformaciones a partir de archivos XML. De este modo, todas las interfaces del aplicativo de software pueden seguir avanzando en conjunto ante cambios en la especificación del robot. La Figura 10 muestra una ventana de la interfaz de programación intermedia donde se brinda un editor de XML.

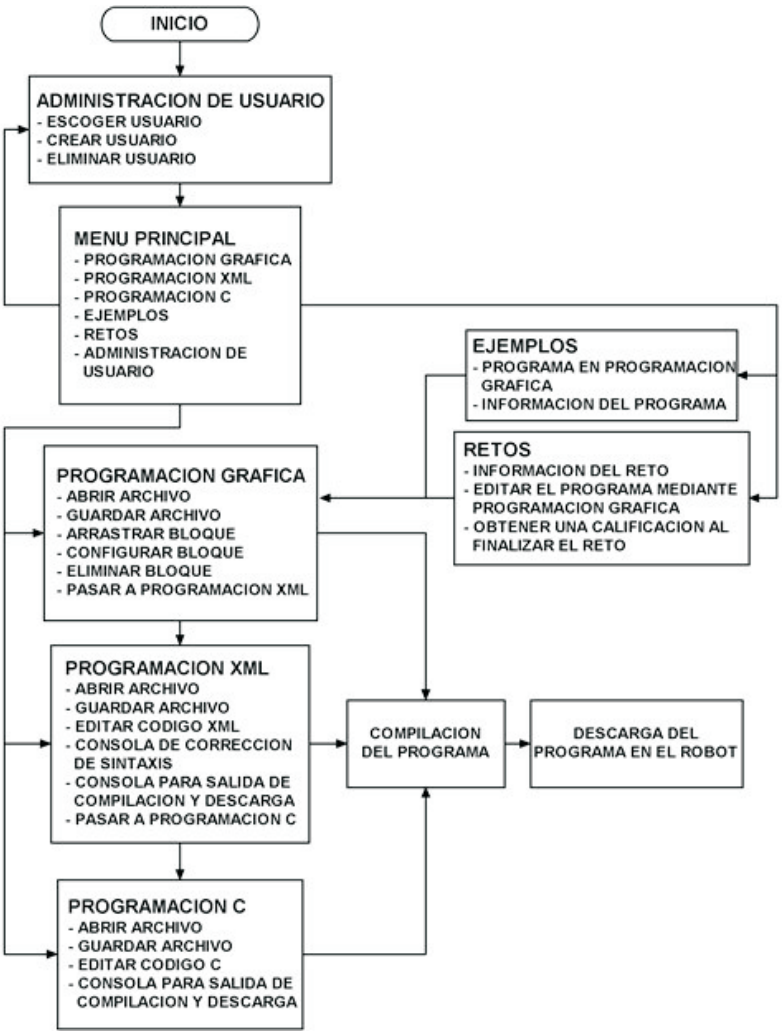


Figura 8. Diagrama de bloques básico del aplicativo.

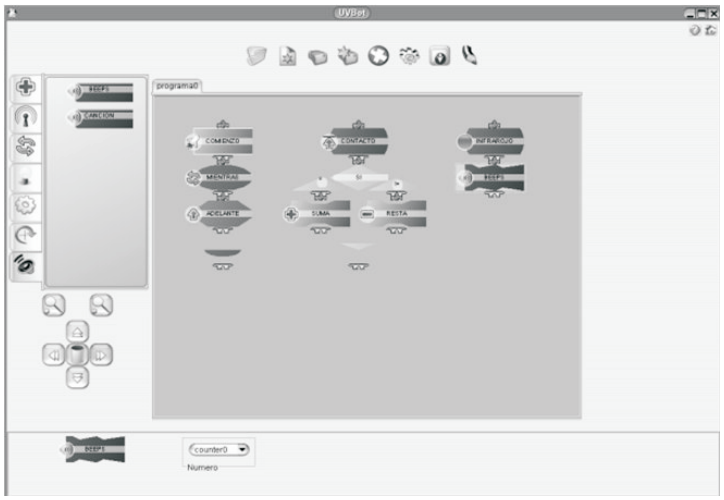


Figura 9. Interfaz para la programación gráfica (nivel básico).

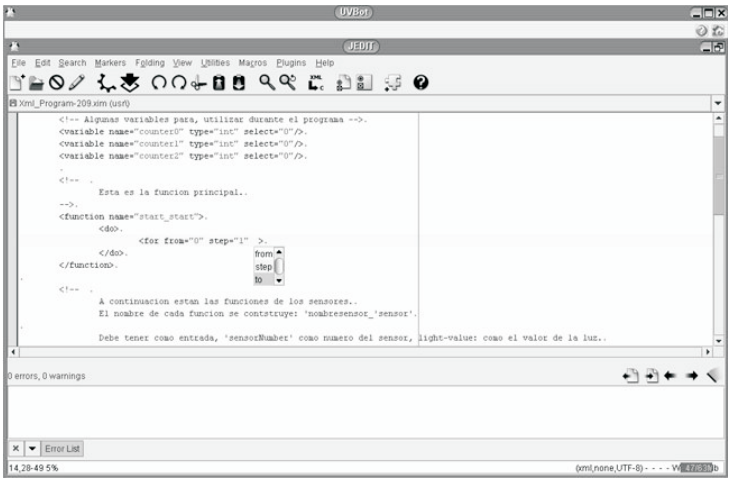


Figura 10. Interfaz para la programación en XML (nivel intermedio).

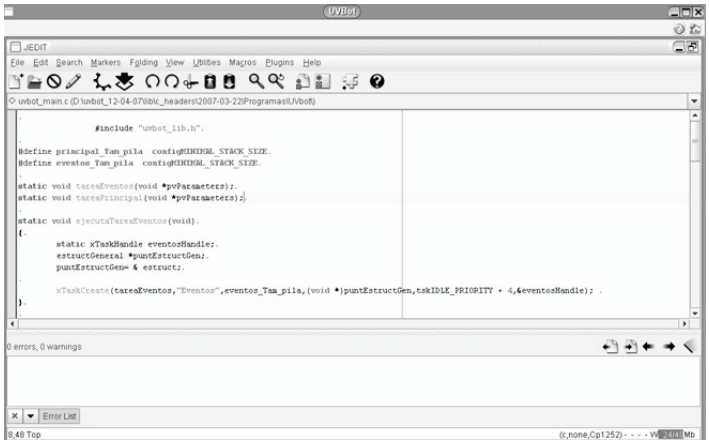


Figura 11. Interfaz para la programación en ANSI C (nivel avanzado).

En esta interfaz se puede programar el robot usando el lenguaje creado (basado en XML). El editor contiene una consola para la corrección de sintaxis y otra consola donde se muestra la salida de la compilación y descarga.

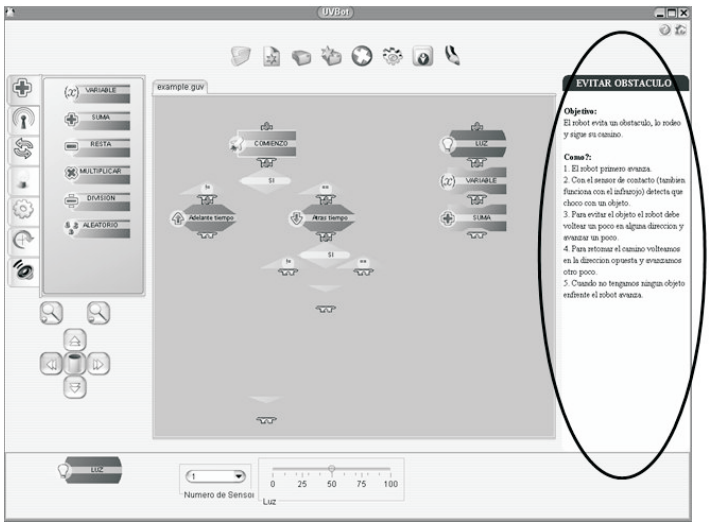
5.3 Programación en ANSI C

El nivel de programación más avanzado que se brinda es la programación directa en ANSI C del robot. La Figura 11 muestra la interfaz de este nivel. Esta interfaz de programación brinda un editor de código ANSI C. El editor contiene una consola donde se muestra la salida de la compilación y descarga. Además, siempre se muestra una plantilla con la cual el usuario sólo debe agregar el código C del programa principal y la activación o no de los eventos asociados a los sensores.

5.4 Retos y manejo de usuarios

Los requerimientos del reto a superar se presentan en el módulo de retos. El usuario debe utilizar la interfaz de programación gráfica para construir el programa que supere el reto. La Figura 12a muestra un ejemplo de problema-reto donde se le pide al usuario realizar un programa para que el robot sea capaz de evitar un obstáculo (las instrucciones se muestran en el lado derecho). El usuario ha realizado el programa correspondiente y debe ahora contestar las preguntas que permiten al aplicativo de software evaluar si se ha superado el reto. El aplicativo también incluye el registro de usuarios, el mantenimiento de los usuarios que usan el aplicativo y la realización de un proceso de LOGIN para entrar al aplicativo y reanudar los retos que el usuario tiene almacenados.

a)



b)

c)

Figura 12. (a) Interfaz para el desarrollo de retos. (b) y (c) Interfaz para el proceso de LOGIN y registro de usuarios.

Las Figuras 12b y 12c muestran las interfaces de LOGIN y creación de un registro en el aplicativo.

6. Pruebas y resultados

Considerando las secciones anteriores, se observa que existen muchos módulos que constituyen la herramienta propuesta. Con el fin de probar la funcionalidad de la herramienta, se implementaron pequeñas aplicaciones en el nivel de complejidad avanzado, lo cual implica usar las funcionalidades de los niveles inferiores. Para cada una de las aplicaciones se usó la interfaz de software tanto en su nivel básico como en el avanzado. Este último nivel se usó para la programación orientada a comportamientos.

Las aplicaciones seleccionadas son bien conocidas en el ámbito de la robótica móvil: evasión de obstáculos, retorno (*homing*) de posición, retorno (*homing*) de luz y una carrera de relevos como aplicación básica de robótica cooperativa.

6.1 Sistema de localización

La determinación de la incertidumbre del sistema de localización es una prueba interesante que es previa a las pruebas con las aplicaciones basadas en comportamientos. En tal prueba, el robot describe un cuadrado de 1.5 m x 1.5 m, girando hacia la derecha y luego hacia la izquierda. Esto se muestra en las Figuras 13a y 13b. La prueba descrita fue realizada 5 veces para cada lado. Las gráficas de dispersión se muestran en las Figuras 13c y 13d, tanto para el giro hacia la izquierda como hacia la derecha.

Para el recorrido antihorario (hacia la izquierda) se midió una incertidumbre promedio en X de ± 13.89 cm y en Y de ± 13.8 cm, usando una distribución t-Student con un margen de confianza de 97.5 %. Para el recorrido horario (hacia la derecha) se midió una incertidumbre promedio en X de ± 17.48 cm y en Y de ± 17.36 cm, usando una distribución t-Student con un margen de confianza de 97.5 %.

6.2 Evasión de obstáculos

La evasión de obstáculos es una característica muy deseada en un robot móvil, que en nuestro caso se logra a través de un comportamiento que usa esquemas de percepción basados en una representación vectorial de los sensores infrarrojos (Bacca et al., 2003). Con este comportamiento se obtiene una percepción vectorial de los obstáculos que rodean al robot. La respuesta del comportamiento es entonces convertida en comandos motores (Bacca et al., 2003) a través de un árbitro que usa un esquema de prioridades definidas en el momento de la programación.

Las respuestas de los comportamientos son vectores normalizados, cuya dirección señala la orientación deseada del robot y cuya magnitud indica el porcentaje de modificación de las velocidades lineales de cada rueda. Las coordenadas X / Y del vector deseado se obtienen mediante las siguientes expresiones:

$$X = \sum_{n=0}^{n=7} p_n \, x_n \tag{9}$$

$$Y = \sum_{n=0}^{n=7} p_n \, y_n \tag{10}$$

donde p_n es la prioridad asociada al comportamiento n, y x_n y y_n son las respuestas de cada comportamiento. Estas ecuaciones muestran cómo el árbitro colaborativo obtiene la respuesta que es enviada realmente a los motores, siempre teniendo en cuenta el aporte de cada comportamiento activo. El conjunto de comportamientos que la herramienta emplea se muestra en la Figura 14. De este conjunto, sólo se usa la evasión de obstáculos.

La Figura 15 muestra dos pruebas típicas para observar el comportamiento del robot ante los obstáculos del entorno.

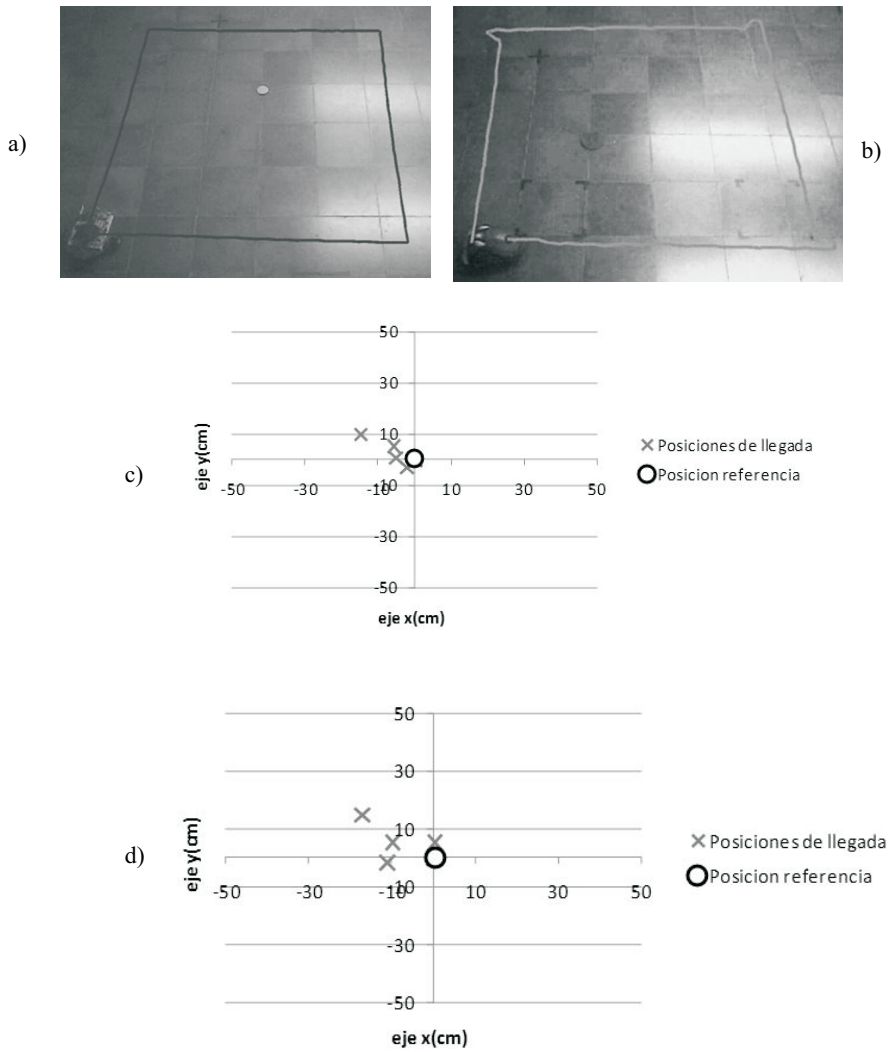


Figura 13. Pruebas del sistema de localización. (a) Recorrido cuadrado hacia la izquierda. (b) Recorrido cuadrado hacia la derecha. (c) Curva de dispersión sentido antihorario. (d) Curva de dispersión sentido horario.

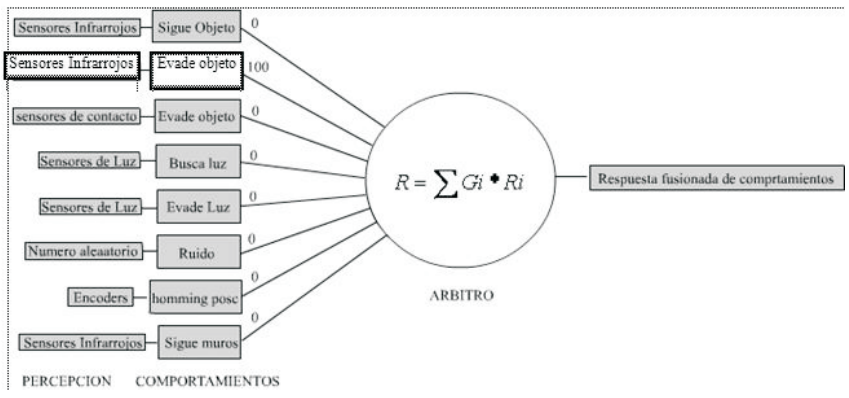
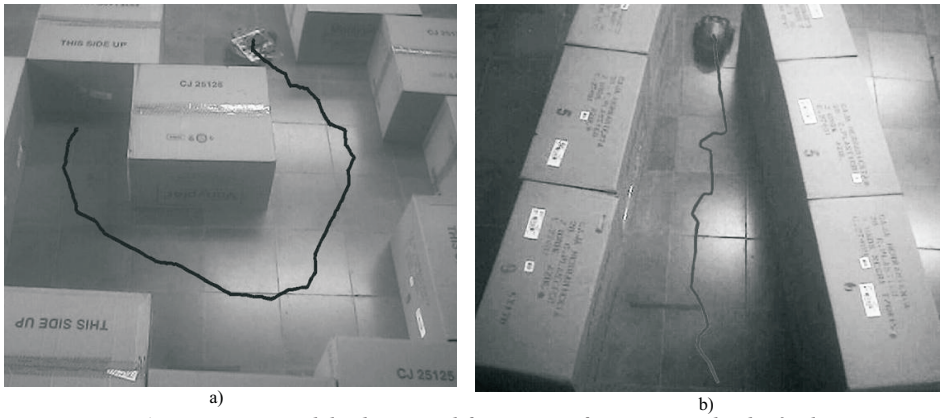


Figura 14. Conjunto de comportamientos de los robots.



a) b)

Figura 15. Trayectorias del robot ante diferentes configuraciones de obstáculos.

Ambas configuraciones de obstáculos no son conocidas previamente por el robot. La configuración mostrada en la Figura 15a es un laberinto típico y la configuración de la Figura 15b es un corredor que se estrecha a medida que avanza el robot. Esta configuración se usa para observar la respuesta del robot ante corredores amplios y estrechos. Las trayectorias en ambas pruebas fueron filmadas y procesadas con el fin de extraer su evolución en el tiempo, usando un programa de procesamiento digital de imágenes (Bacca et al., 2006).

6.3 Retorno por posición y por luz con evasión de obstáculos

El retorno (*homing*) es un comportamiento típico en robótica móvil cuyo objetivo es orientar al robot hacia una región de interés. Gracias a que los robots móviles soportan la programación orientada a comportamientos, el comportamiento de retorno puede ser combinado con la evasión de obstáculos, de tal manera que la combinación de ambos resulta en un comportamiento emergente más complejo (Arkin, 1989). En nuestro caso, la región de interés se define en dos formas: usando una posición en el plano coordenado global y empleando un estímulo como es la luz. Ambas formas se usaron en las pruebas que se muestran a continuación.

6.3.1 Retorno por posición y evasión de obstáculos

La Figura 16 muestra el conjunto de comportamientos activos para el retorno por posición con evasión de obstáculos. El comportamiento de retorno por posición tiene como objetivo la posición XY (100 cm, 0 cm) en el plano global, y cuando el robot se acerca alrededor de una zona con ± 5 cm, el robot se detiene. La prueba fue realizada 5 veces. Las trayectorias obtenidas al procesar digitalmente la imagen se muestran en la Figura 17a y la nube de puntos de las posiciones reales de llegada se muestra en la Figura 17b. De esta figura, se obtuvo una incertidumbre promedio de 12.8 cm en el eje X y 15.32 cm en el eje Y usando una distribución t-Student con un margen de confianza del 97.5 %.

6.3.2 Retorno por luz y evasión de obstáculos

Al cambiar la definición del área de interés usando un estímulo análogo como es la luz, el componente que cambia en el conjunto de comportamientos de la Figura 16 es el comportamiento de retorno. Es decir, se desactiva el comportamiento 7 y se activa el 4, de arriba a abajo. En este caso, el robot no conoce previamente la posición del estímulo; éste se guía usando sus fotoceldas, obteniendo un vector que se orienta hacia el lugar con más intensidad lumínica, y se debe garantizar que el robot pueda tener línea de vista hacia la fuente de luz, esto es, los obstáculos no pueden ser muy altos. El robot tampoco conoce la distribución de los obstáculos. Ambos comportamientos básicos se mezclan para obtener el resultado de la Figura 18a. La prueba fue realizada 5 veces. La Figura 18a muestra las trayectorias obtenidas al procesar digitalmente los vídeos de las pruebas.

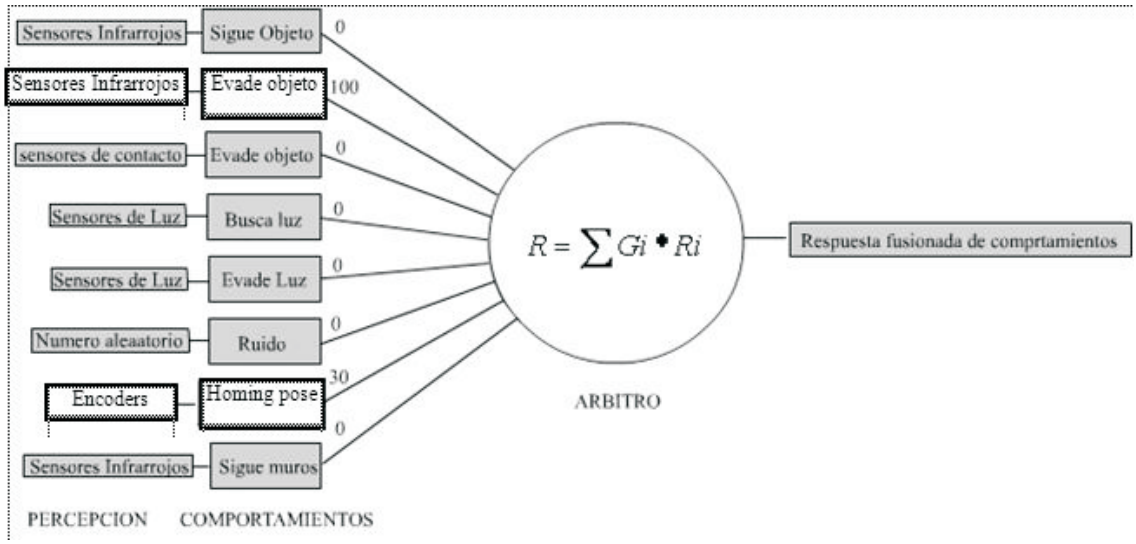
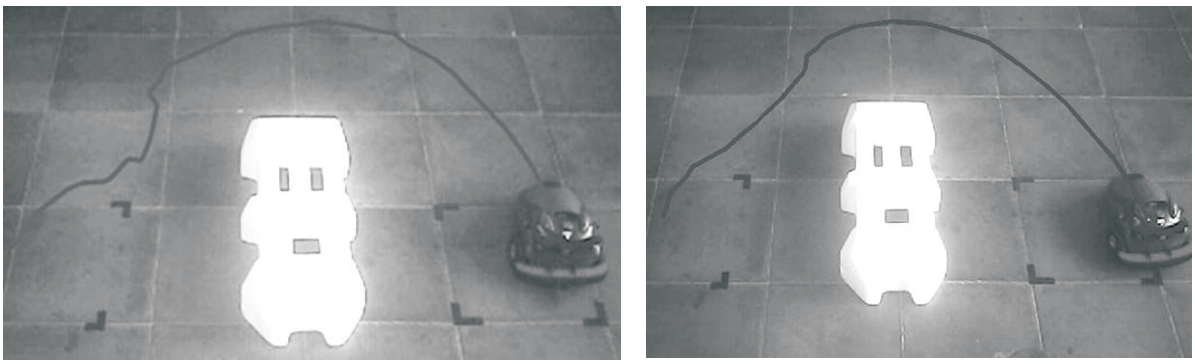
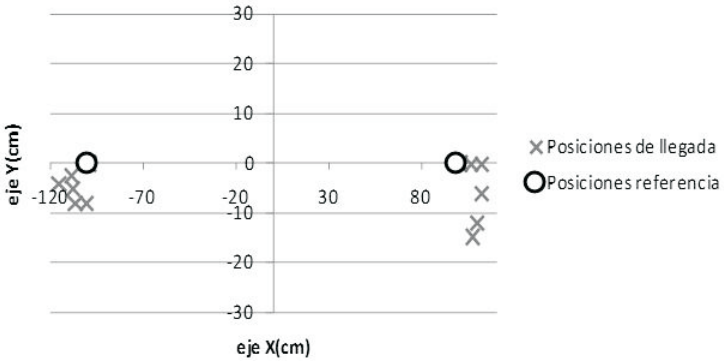


Figura 16. Conjunto de comportamientos para el retorno por posición y evasión de obstáculos.

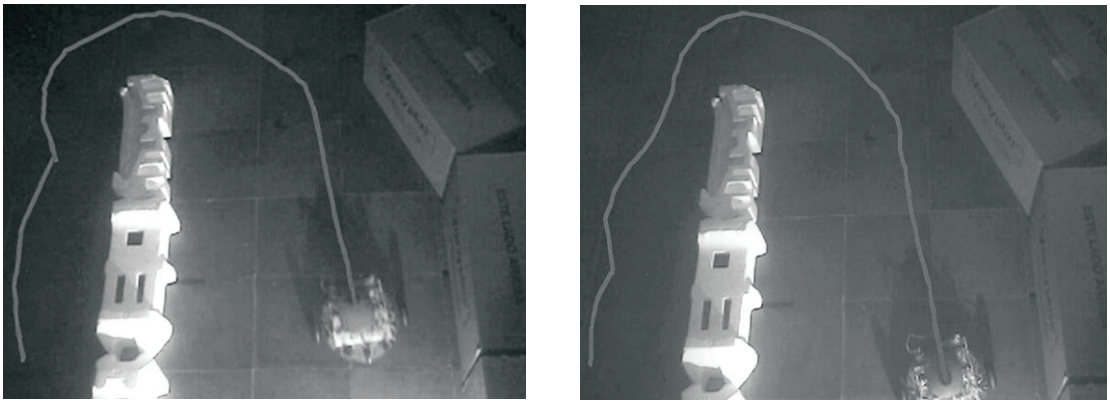


a)

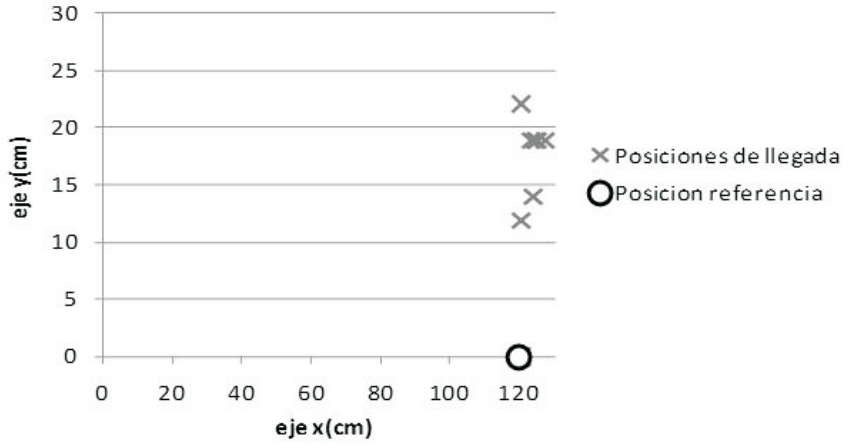


b)

Figura 17. (a) Trayectorias obtenidas para el retorno por posición con evasión de obstáculos. (b) Nube de puntos de las posiciones reales de llegada.



a)



b)

Figura 18. (a) Trayectorias de la prueba de retorno por luz y evasión de obstáculos. (b) Nube de puntos de las posiciones XY reales alcanzadas en la prueba.

Con el fin de establecer la incertidumbre para esta prueba, el estímulo fue ubicado en la posición (120 cm, 0 cm) del plano global. Como criterio de parada, se estableció la lectura del 98 % de la capacidad máxima de medición de iluminación de las fotoceldas del robot. La Figura 18b muestra la nube de puntos XY que muestran los lugares donde el robot detuvo su movimiento, los cuales se encuentran en un rango entre 12 cm a 22 cm de la fuente de luz, teniendo una incertidumbre de 17.3 cm (usando una distribución t-Student con un margen de confianza del 97.5 %.

6.4 Cooperación básica

La última prueba realizada está orientada a demostrar esta funcionalidad de la herramienta, con el fin de resaltar la capacidad de comunicación entre robots y poder con esto desarrollar tareas sencillas de cooperación. La prueba consistió en la implementación de una carrera de relevos. El robot inicia su recorrido siguiendo una trayectoria ovalada, al dar una vuelta se encuentra con el otro robot el cual aguarda el envío de un mensaje a través del sistema de comunicaciones infrarrojas y sólo en este instante inicia el mismo recorrido y así sucesivamente.

La Figura 19 muestra una secuencia de imágenes en la cual se puede observar lo anterior. Esta prueba es uno de los retos que los niños y jóvenes realizan en el semillero de robótica de la Universidad del Valle y la Biblioteca Departamental.

7. Discusión de resultados

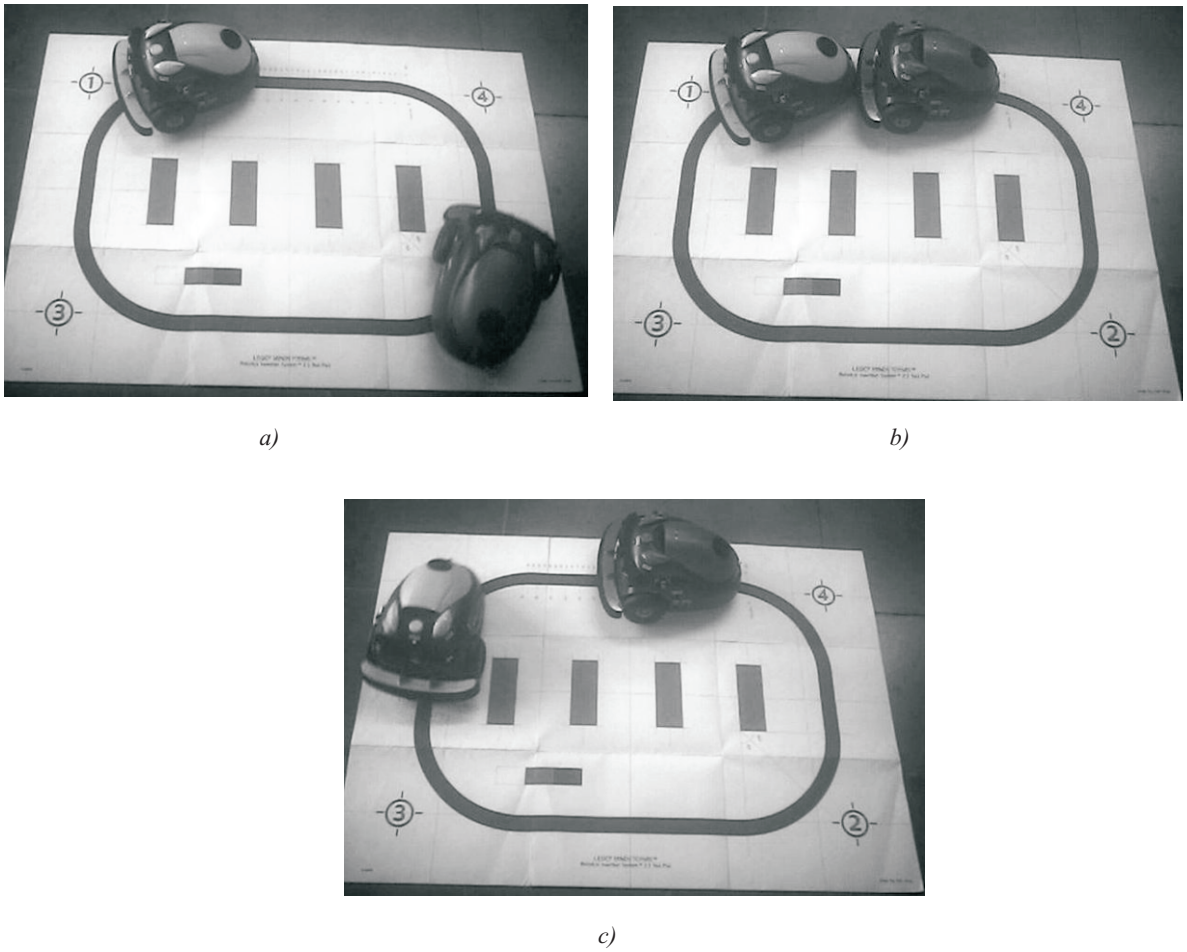
El desarrollo de las pruebas del sistema de localización proporciona un círculo de incertidumbre que en el peor de los casos tiene un radio de 17 cm. Este valor no es ajeno a la bien conocida inexactitud de un sistema de localización relativo basado en codificadores incrementales (Borenstein et al., 1996), pero gracias a la flexibilidad de la plataforma, esto permite el planteamiento de desarrollos a corto plazo para la corrección de errores sistemáticos de odometría (Borenstein et al., 1996) y la adición de un compás digital para la compensación de los errores de odometría usando filtros de Kalman (Ivanjko et al., 2004).

Las plataformas móviles están en capacidad de ejecutar aplicaciones que usan la programación basada en comportamientos. Los resultados de las pruebas de evasión de obstáculos presentadas en la Figura 15a y 15b muestran al robot usando sensores simples para recorrer un laberinto o un pasillo. Sin embargo, nótese que hay ciertas oscilaciones en la trayectoria detectada, lo cual responde a que el espacio libre de obstáculos con el cual se encuentra el robot es muy parecido al rango de alcance de sus sensores. Esto se puede ver con más fuerza en la Figura 15b cuando el pasillo se estrecha paulatinamente. El robot actualmente es programado bajo una perspectiva reactiva, lo cual no le permite tener una percepción global de su entorno, sólo local y sin historia. Sin embargo, al introducir la navegación con campos de potencial, la plataforma se convierte en una herramienta bastante útil para el aprendizaje de este concepto a diferentes niveles de conocimiento en robótica.

Lo interesante de la programación orientada a comportamientos es poder obtener un comportamiento emergente más complejo que los comportamientos básicos que lo conforman.

Este es el caso del retorno con evasión de obstáculos, prueba que se llevó a cabo de dos maneras: la primera usando el sistema de odometría, donde se tienen dos comportamientos con diferentes grados de importancia, 100 % de importancia para la evasión de obstáculos y 30 % para orientar al robot hacia la zona de interés (punto 100, 0) como se observa en la Figura 17a. La Figura 17b contiene dos de las 5 pruebas realizadas, las cuales estuvieron dentro del margen de incertidumbre del sistema de localización (círculo de 17 cm de radio). En la Figura 17b se observa que inicialmente el robot se orienta hacia su destino, pero modifica su trayectoria al encontrarse con el obstáculo. La segunda forma de implementación fue usando estímulos externos; en este caso, luz que fue capturada por sus fotoceldas. Estas fotoceldas fueron usadas para establecer un marco coordenado de intensidades de luz y luego un vector cuyo ángulo determinaba la orientación hacia donde el robot debía dirigirse. La Figura 18a muestra un comportamiento inicial parecido al descrito anteriormente, con una modificación de la trayectoria para evadir el obstáculo y luego acercarse a la zona de interés. Ya que en esta prueba es difícil alcanzar por completo la fuente del estímulo, dada la simplicidad de los sensores, el robot terminó alrededor de la fuente con un círculo de incertidumbre de radio igual a 17 cm, distancia a la cual, dependiendo del ángulo de aproximación, los sensores del robot entraban en saturación. Nótese adicionalmente que el conjunto de comportamientos con que cuenta el robot está siendo parametrizado por las prioridades de cada uno de ellos. Esto significa que la plataforma está preparada para modular su comportamiento emergente, dependiendo de la adecuada activación y valor de las prioridades asociadas a cada comportamiento.

Finalmente, los resultados mostrados permiten observar que la plataforma posee una muy buena flexibilidad de programación y que está concebida teniendo en cuenta conceptos para la navegación de robots móviles acordes con el estado del arte actual. Esto resulta muy útil para los objetivos educativos bajo los cuales fue concebido el sistema inicialmente, ya que permite la introducción a los siguientes conceptos: control de movimiento, percepción básica usando la



a) *b)*

c)

Figura 19. Secuencia de imágenes cooperación básica. (a) Robot próximo a llegar. (b) Transmisión del mensaje. (c) El otro robot inicia el recorrido.

información original de los sensores, percepción avanzada (Arkin, 1989) usando el concepto de esquemas (al convertir las lecturas de los sensores a un espacio vectorial), y programación de soluciones a problemas concretos (usando las plataformas móviles, su *firmware* y su aplicativo con sus tres niveles de complejidad).

8. Conclusiones

Se presenta un sistema de experimentación en robótica compuesto por dos plataformas móviles y un aplicativo para la programación de los robots usando tres niveles de complejidad. Los robots están compuestos por 5 sensores infrarrojos de proximidad, 3 de contacto, 3 fotoceldas, un sistema de movilidad diferencial, un sistema de localización relativa usando codificadores incrementales con una incertidumbre de 17 cm, una velocidad lineal máxima de 19 cm / s, un sistema de control de velocidad PI para cada una de las ruedas y un *firmware* diseñado usando un aplicativo de tiempo real (FreeRTOS) capaz de soportar las funcionalidades para la programación del robot con tres niveles de complejidad. El aplicativo es un software que se comunica con el robot usando un enlace serial y que permite la programación gráfica de aplicaciones para un nivel básico de conocimientos de robótica, la programación usando XML para un nivel intermedio, la programación en ANSI C para un nivel avanzado, la introducción a la herramienta

usando retos, el registro de usuarios y el manejo de los mismos con el fin que cada usuario mantenga la evolución de sus retos.

UV-BOT es un sistema de experimentación que reúne las ventajas de dos mundos, uno en el cual se tienen robots de entretenimiento (cuya vida útil es corta en el momento de ser usados como herramientas de aprendizaje en robótica) y otro que tiende hacia el otro extremo, al tener plataformas con un nivel de complejidad más elevado y orientadas más que todo a investigación. UV-BOT es una herramienta con la cual el usuario puede realizar un acercamiento a la robótica móvil sin tener conocimientos previos de ésta, para luego gradualmente incrementar la complejidad de los conceptos empleados y de los retos propuestos.

9. Referencias bibliográficas

Arkin, R. C. (1989). Motor schema-based mobile robot navigation. *The International Journal of Robotics Research* 8 (4), 92-112.

ATMEL. (2005). *8-bit AVR microcontroller with 32 Kbytes in-system programmable flash (ATmega32, ATmega32L).* http://www.atmel.com/dyn/resources/prod_documents/doc2503.pdf

Bacca, E. B., Caicedo, E. F., & Santos-Victor, J. A. (2003). *A behavior-based homing strategy implemented on a layered control architecture*. In Proceedings of the 13th International Symposium on Measurement and Control in Robotics (ISMCR03), p. 149 – 154.

Bacca, E. B., & Caicedo, E. F. (2006). Control del empuje de una barra usando dos robots móviles y visión artificial. *Ingeniería (Universidad Distrital Francisco José de Caldas)* 11 (2), 70-79.

Borenstein, J., & Feng, L. (1996). Measurement and correction of systematic odometry errors in mobile robots. *IEEE Transactions on Robotics and Automation* 12 (6), 869-880.

Brooks, R. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation* 2 (1), 14-23.

EAFIT (Escuela de Administración, Finanzas y Tecnología). (2007). *Universidad de los Niños*. http://www.eafit.edu.co/eafitcn/ninos/semilleros.html

Esteban, M. (2000). *El diseño de entornos de aprendizaje constructivista*. Adaptación de un artículo de D. Jonassen, publicado en C.H.Reigeluth: el diseño de la instrucción, Madrid, Aula XXI Santillana. http://www.um.es/ead/red/6/documento6.pdf

FreeRTOS. (2005). *RTOS implementation for AVR microcontrollers*. http://www.freertos.org

Gómez, F., & Muñoz, F. (2007). *Diseño e implementación de un robot móvil diferencial*. Trabajo de grado, Escuela de Ingeniería Eléctrica y Electrónica, Universidad del Valle, Cali, Colombia.

Ivanjko, E., & Petrovic, I. (2004). *Extended Kalman filter based mobile robot pose tracking using occupancy grid maps*. In Proceedings of the 12th IEEE Mediterranean Electrotechnical Conference (MELECON 2004), Vol. 1, p.311-314.

Jones, J. L., Flynn, A. M., & Seiger, B. A.(1999). *Mobile robots: inspiration to implementation.* Second edition, Natick, Massachusetts: A.K. Peters.

LEGO (2008), Mindstorms RIS 2.0 / NXT, http://mindstorms.lego.com

Londoño, J. G. (2008). *Escasez de ingenieros en Colombia.* Revista Portafolio, Redacción de Economía y Negocios, Bogotá, Colombia.

Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research* 5 (1), 90-98.

MALOKA. (2006). *Semana colombiana de robótica en Maloka del 16 al 18 de agosto.* http://www.estereofonica.com/article.php?sid=2210

NASA (National Aeronautics and Space Administration). (2008). *The robotics alliance project.* http://robotics.nasa.gov/index.php

NASA JPL (National Aeronautics and Space Administration Jet Propulsion Laboratory). (2008). *Mars exploration rover mission: classroom*. http://marsrovers.nasa.gov/classroom/students.html

Ogata, K. (1996). *Sistemas de control en tiempo discreto.* Segunda edición, Pearson Prentice-Hall.

Ollero, A. (2007). *Robótica; manipuladores y robots móviles.* Barcelona, España: Marcombo S.A.

Ulloa, G. (2008). *¿Qué pasa con la ingeniería en Colombia ?*. Tecno-Tips, Universidad ICESI, Cali, Colombia. http://www.icesi.edu.co/blogs/tecnotips/2008/02/15/%c2%bfque-pasa-con-la-ingenieria-en-colombia/

UNIVALLE (Universidad del Valle). (2003). *Semillero de robótica.* http://www.valledelcauca.gov.co/publicaciones.php?id=1089