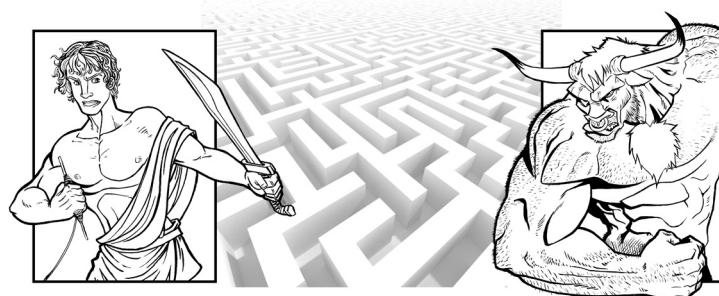




Laboratorio de Microcomputadoras [66.09]

Proyecto TCO



Primer Cuatrimestre 2013

Apellido	Nombre	Padrón	Mail
Carballeda	Ignacio L. J.	91646	carballeda.ignacio@gmail.com
Cerone	Sebastian	90259	sebascerone@gmail.com
Farace	Gisela	92457	gisela.farace@gmail.com

Índice

1. Motivación	3
2. Objetivo	3
3. Descripción general	4
Robot	4
Laberinto	5
Algoritmo	6
4. Resumen de funcionamiento	7
PWM y Motores	7
Funcionamiento de los encoders	7
Medición de distancia	7
Identificación del color del piso	8
Comunicación serie	9
5. Desarrollo del proyecto	10
Decisiones de diseño	10
Armado del robot	10
Problemas hallados	10
6. Conclusiones	11
7. Listado de componentes	12
Robot	12
8. Herramientas de software	13
9. Bibliografía	13
10. Apéndice	13
Esquemáticos (*)	13
Código fuente (*)	13

1. Motivación

Este desarrollo surge como un proyecto extracurricular del *Club de Robótica* de la *Facultad de Ingeniería de la Universidad de Buenos Aires* a realizarse dentro del ámbito del *Laboratorio Abierto (LABI)*.

Desde el año 2011, el *Club de Robótica* realiza una competencia anual de robótica. En un primer momento, empezó con una carrera de robots velocistas; que consiste en que un robot realice por sus propios medios, en el menor tiempo posible, un recorrido delimitado por una línea blanca sobre una superficie negra. Al año siguiente, se incorporó la competencia de "Mini-sumo"; en donde dos robots se enfrentan, cual luchadores de sumo, y deben empujar al oponente fuera del "tatami". Finalmente, este año, se incorporó la categoría "Laberinto", en donde un robot, en el menor tiempo posible, debe encontrar la salida de un laberinto.

Es por esta razón, que se optó por realizar un proyecto que sirva para aprender los temas de la asignatura *Laboratorio de microcomputadores* y pueda representar al *Club de Robótica* en la competencia anual.

Cabe aclarar, la elección del nombre *TCO*, está inspirado en el personaje de la mitología griega *Teseo*, quien logró salir de un laberinto luego de derrotar al Minotauro.

2. Objetivo

Por las razones planteadas anteriormente, el objetivo de este trabajo es el diseño, fabricación y programación de un robot capaz de salir de un laberinto por sus propios medios.

3. Descripción general

Robot

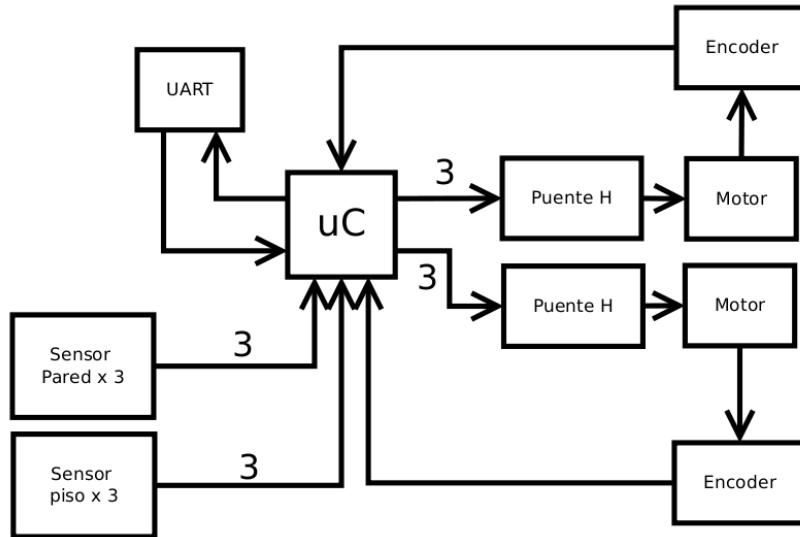


Figura 1: Diagrama de bloques del robot.

Para llevar a cabo este proyecto se fabricó un robot. El mismo está compuesto de las siguientes partes:

- Microcontrolador: Se eligió un *ATMEGA16* de *AVR*. Este microcontrolador, muy sencillo, tiene la cantidad adecuada de periféricos necesarios para la realización de este proyecto.
- Motores: Se optó por dos motores de la marca *pololu*. Éstos tienen encoders integrados, que facilitan un mejor control del desplazamiento.
- Puente H LMD18200: Este puente H simple, maneja cargas de hasta 3 Amperes y posee un flag de advertencia de alta temperatura.
- Sensores HC - SR04: Estos sensores de distancia por ultrasonido, sencillos de utilizar, y muy estables, detectan distancia en un rango de 2 cm a 450 cm, con un margen de error de 0,3 cm.
- Sensores CNY70: Estos sensores ópticos, se utilizan para diferenciar el color del piso entre blanco o negro. En este caso, se utilizarán para detectar si el robot llegó al final del laberinto.
- Baterías y fuente de alimentación: El robót cuenta con 3 baterías de Li-ion, 2 de 7,2 v y 1 de 3,7 v. Junto con dos fuentes DC/DC de *texas* proporcionando 12 v y 5 v.

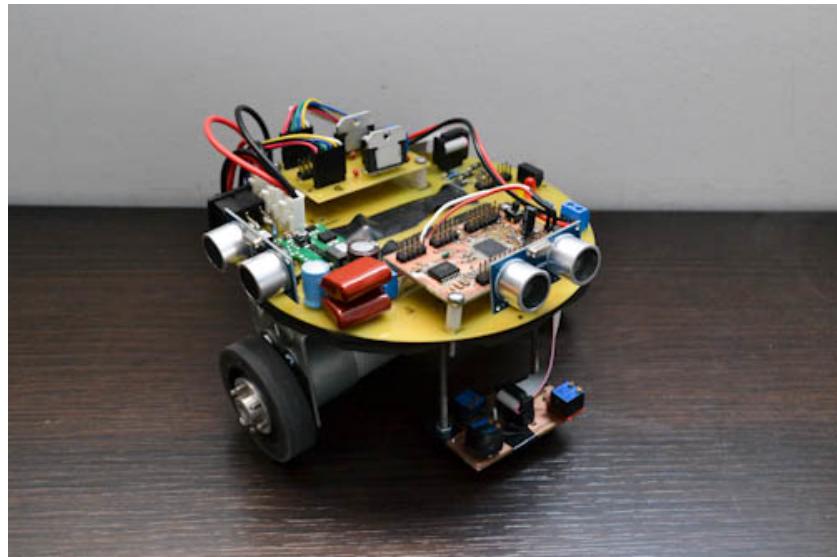


Figura 2: TCO

Laberinto

El laberinto se fabricó como un módulo modelizable como una matriz perteneciente a \mathbb{R}^{3x3} , donde sus paredes internas, definen *celdas* que se pueden interpretar como posiciones dentro de la matriz.

El mismo se construyó con paredes color blanco y piso color negro, dividiendo cada celda por una línea blanca, siendo que sobre este color el rebote de luz infrarroja resulta óptimo para los sensores del piso, y facilita la detección entre una celda y otra.

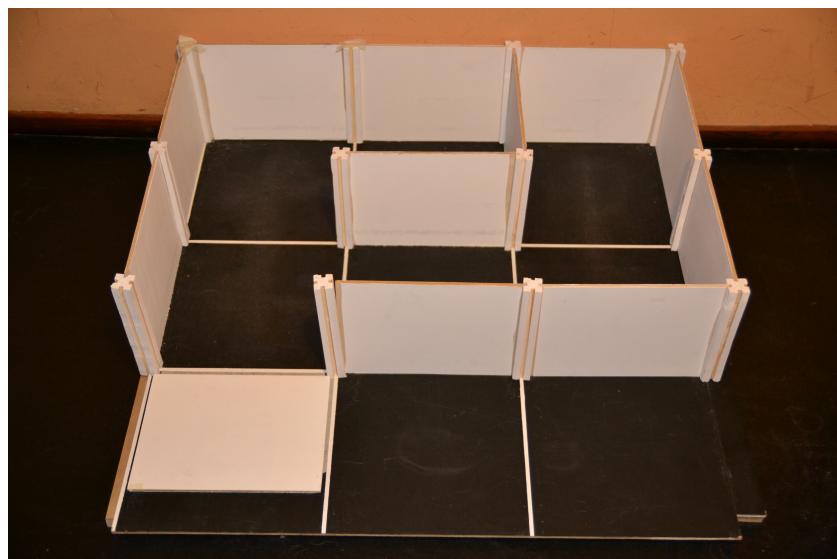


Figura 3: Laberinto

Algoritmo

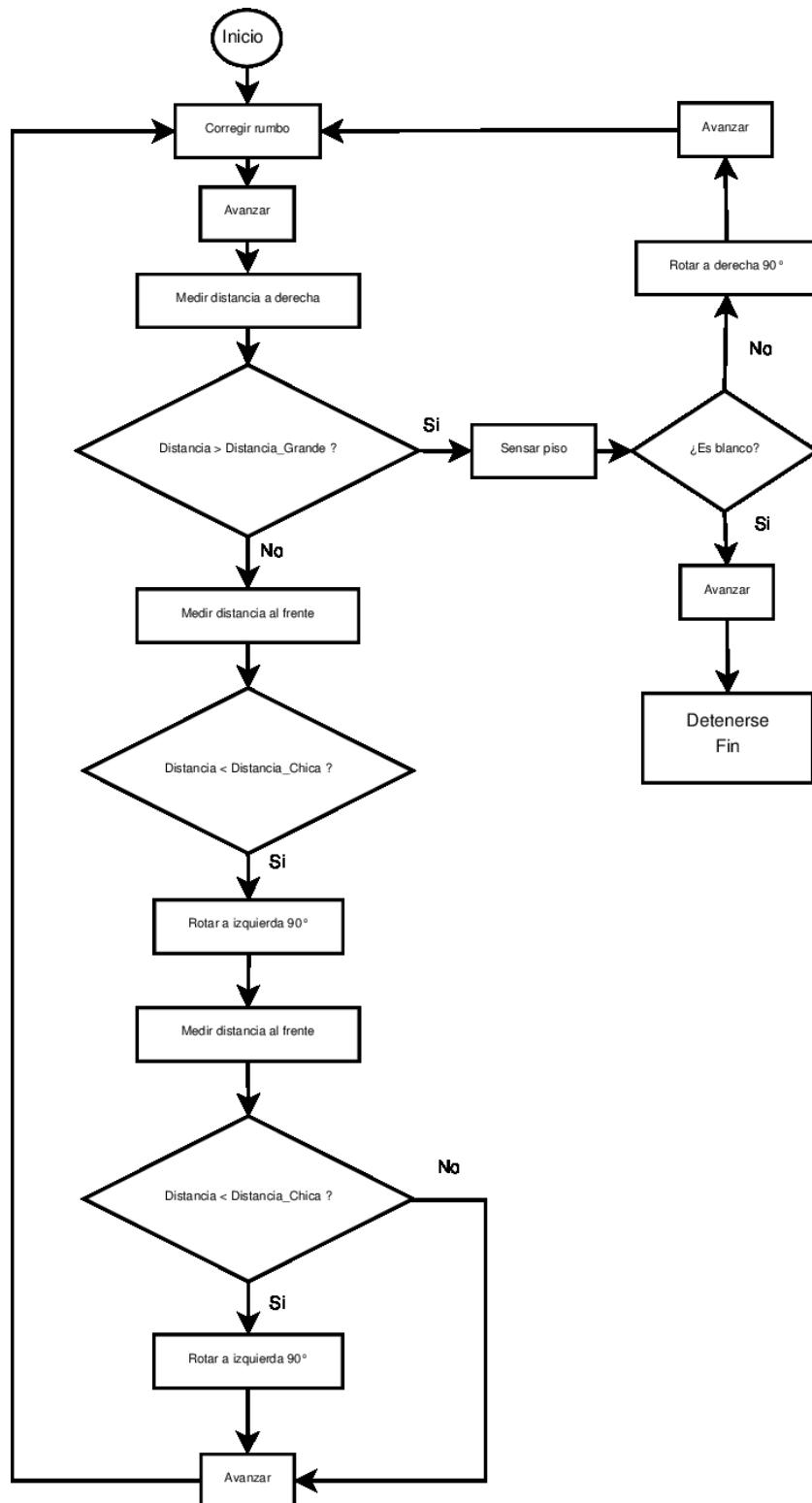


Figura 4: Laberinto

4. Resumen de funcionamiento

PWM y Motores

Para tener la libertad de poder cambiar la velocidad de los motores, se utilizó una señal PWM. Se configuró el *timer/contador1* en modo *Fast PWM*. De esta forma, el contador se incrementa de 0 a 256 cuentas. Cuando alcanza este último valor, se desborda y comienza otra vez desde cero. Luego, realizando dos comparaciones, se puede variar el ancho de pulso de dos señales que se usan para regular la potencia entregada a cada motor.

También se utilizaron dos puentes H *LMD18200* para poder frenar y cambiar el sentido de giro de cada motor por separado.

Los motores del robot son de la marca *Pololu*, los mismos vienen equipados con reducciones 29 : 1 y encoders en tablatura con una resolución de 64 pasos por vuelta de motor. Gracias a la reducción incorporada en los motores se obtiene una buena relación velocidad/torque sin las complicaciones mecánicas que conlleva utilizar un sistema de engranajes por separado. Por otra parte, los encoders dotan al robot de una resolución de $29 \times 64 = 1856$ cuentas por revolución de rueda.

Los motores requieren una tensión de 12V y pueden llegar a consumir una corriente de 3A cuando están bloqueados, por esta razón se incorporó una fuente DC/DC de 12V 6A *PTN78020* marca *Texas Instruments*.

Para controlar los motores, se realizaron funciones que usan en conjunto las cuentas obtenidas de los encoders, variaciones de la señal PWM y combinaciones en los pines de los puentes H. Estas funciones se encuentran en el archivo *motores.c*.

Funcionamiento de los encoders

Ambos motores tienen un sensor de efecto Hall en el extremo; estos, gracias a la circuitería integrada, emiten un pulso cuadrado por cada vez que se da el efecto Hall en ellos.

Sabiendo que por cada revolución de motor se producen 64 pulsos cuadrados y teniendo en cuenta la reducción de 29:1 en los motores, se puede determinar con precisión la cantidad de giro en cada rueda. Para que el microcontrolador pueda llevar la cuenta de los encoders, se conectó la salida de cada uno a los pines de interrupción externa INT1 e INT2. De esta forma; cada vez que hay un flanco ascendente en alguna de las interrupciones, se incrementa una variable.

Medición de distancia

Para medir la distancia del robot a las paredes se utilizaron tres módulos *HC-SR04*, los mismos funcionan por ultrasonido.

Se eligieron estos sensores por varias razones:

- Su rango de 2 centímetros a 4 metros.
- Su resolución de 3 milímetros.

- Simplicidad y costo reducido.

Se desconoce la circuitería interna del mismo, por lo que será tratado como a una caja negra con el cual se puede interactuar mediante sus pines de Entrada/Salida.

El HC-SR04 además de sus pines para alimentación (VCC y GND), tiene otros dos pines: Trigger y Echo. Para obtener una medición, el módulo sólo requiere un pulso de $10\mu S$ en el pin Trigger, luego el módulo pone en alto el pin de Echo con un tiempo proporcional a la distancia medida.

En el código fuente de este proyecto, se configuró una interrupción para que se active con cada desbordamiento del *timer/contador1*. Esta interrupción en cada iteración incrementa en 1 la variable **interrupciones_timer_1**. También se configuran los pines Echo y Trigger como entrada y salida respectivamente.

Para obtener una medición, primero se pone en alto el pin de Trigger por un tiempo mayor a $10\mu S$, luego se espera con un ciclo “while” a obtener un “uno” en pin de Echo; acto seguido se toma el valor actual del contador/timer1 y se guarda en la variable **cuentas_inicial**. A continuación, se espera a que el pin de Echo vuelva a cero con otro ciclo “while” y se guarda el valor del registro del contador/timer1 en otra variable llamada **cuentas_final**.

Luego, teniendo en cuenta que el timer1 fué configurado para que desborde a las 256 cuentas; para obtener el número total de cuentas se utiliza el siguiente código:

```
1 cuentas = (256 - cuentas_inicial) + 256 * interrupciones_timer_1 +
    cuentas_final;
```

Finalmente se realiza una conversión para relacionar las cuentas con una distancia aproximada.

Identificación del color del piso

Con el objetivo de detectar la celda blanca, se utilizaron tres sensores detectores de color en la parte baja del mismo, que le indican al robot que salió correctamente del laberinto. Los sensores utilizados son los *CNY70*, compuestos por un emisor (led infrarrojo) y un receptor (transistor TBJ NPN).

Estos sensores pueden ser encendidos y apagados a voluntad por el microcontrolador.

Cuando el robot está situado en una celda blanca (salida), la luz emitida por el led del sensor se refleja en la superficie y excita la base del transistor, permitiendo de esta forma un flujo de corriente de emisor a colector.

Se ubicó un resistor entre el emisor del transistor y la referencia del circuito (GND), para detectar los cambios de tensión.

En la hoja de datos del *ATMEGA16* se especifica que la máxima tensión admisible para un cero lógico (con el pin configurado como entrada), es de $0,2 \times VCC$, en nuestro caso $VCC = 5V \Rightarrow V_{IL_{max}} = 1V$. Por lo tanto una tensión que no supere 1 Volt, será considerada

como un cero lógico. Por otro lado, $V_{IH_{min}} = 0,6 \times VCC = 0,6 \times 5V = 3V$ es la mínima tensión para detectar un uno lógico en un pin configurado como entrada.

Cuando un sensor es situado sobre una celda blanca, la tensión en su emisor alcanza los 4.5 V, valor considerado por el microcontrolador como un "uno" lógico. En cambio, ubicado sobre una celda negra, la tensión medida es aproximadamente cero, ya que la luz no es reflejada en esta superficie y esto es interpretado como cero lógico.

El microcontrolador enciende los sensores durante un tiempo muy breve cada vez que el robot decide girar a su derecha, de esta forma se evita que las líneas blancas que dividen las celdas sean interpretadas como la salida del laberinto.

Comunicación serie

El presente proyecto se puede conectar a un PC, por medio de un puerto RS232. Esta función es utilizara para *debugging*.

Se configuró la *UART* para enviar y recibir datos, y se configuró una interrupción, que se activa cuando se recibe un dato por el puerto serie.

5. Desarrollo del proyecto

Decisiones de diseño

Para el diseño del robot, primero se analizó la forma en la cual éste se desplazaría dentro del laberinto. Como, entre los movimientos a realizar, hay que rotar 90° o 180° se pensó que la estructura del robot debía ser circular, para evitar complicaciones al momento de rotar.

Por otro lado, se diagramó el robot en diferentes módulos, con las partes importantes, como el microcontrolador, los puentes H y los motores, la alimentación y los sensores de pared, y los sensores de piso. Por lo tanto, se armaron distintas placas, para cada uno de los módulos. De esta forma, se pudo probar cada parte por separado y corregir los errores fácilmente. Ésto, tambien fue muy ventajoso, al momento de solucionar problemas o fallas.

Se dispusieron las placas pequeñas dentro de la placa circular de forma tal de colocar las baterías abajo, haciendo que el centro de gravedad este lo más cerca del piso.

Para colocar la placa con los sensores de piso, se utilizaron dos tornillos largos atornillados en el robot. De esta forma logramos que los sensores queden al ras del piso para medir mejor.

Armado del robot

El robot esta montado sobre los mismos PCB. La placa de alimentación es el disco central, donde se atornillaron los motores, y las placas del microcontrolador y los puente H. Las baterías se agarraron con presintos. Las placas estan conectadas entre si, por medio de cables planos.

Problemas hallados

Durante la realización del proyecto se hallaron varios problemas. Se lista a continuación algunos de ellos:

- Al momento de comprar los motores, no se analizó como encastrar las ruedas. Por lo que se tuvo que mandar a fabricar un adaptador especial.
- Las ruedas compradas eran muy pequeñas. Por lo que se adaptaron rodillos de una impresora, para cumplir la función de rueda.
- Se tuvo que crear un adaptador especial para sostener los motores.
- Durante toda la experiencia, hubo muchos problemas con las placas. Desde pistas cortadas, agujeros mal colocados o errores en el diametro de los mismos, errores en las dimensiones de los componentes. Por ejemplo, la placa de los puente H, se fabricó 3 veces, ya que se hallaron problemas, tanto de diseño como de fabricación.

6. Conclusiones

Los resultados obtenidos en el presente proyecto han sido satisfactorios. Se ha realizado un robot capaz de cumplir con el objetivo propuesto, salir de un laberinto por sus propios medios.

También, el proyecto a sido galardonado con el segundo premio en la categoría *Laberinto*, en la competencia de robótica *Megaman 2013* organizada por el *Club de Robótica de la Facultad de ingeniería de la Universidad de Buenos Aires*.

A futuro, se realizaran algunas mejoras, como por ejemplo:

- Mejorar el sistema de interrupciones. Como se utilizó una OR de diodos, para conectar varios sensores a la misma interrupción, a veces, el funcionamiento de un sensor, afecta el funcionamiento de otro sensor.
- Mejorar la estructura del robot.

7. Listado de componentes

Robot

Referencia	Descripción	Cantidad	Precio unitario	Total
C3, C9, C11	100 nF	3		
C25	100 uF	1		
C14, C15, C18, C19	10 n	4		
C4, C5, C6, C7, C8, C13, C17	1 uF	7		
C1, C12	12 p	2		
C12, C16	220 uF	2		
C20, C21	2,2 uF	2		
C23	330 uF	1		
R1	10 k	1		
R11	21 k	1		
R10	732	1		
R2, R3, R4, R5, R6, R7	R	6		
X1	8 MHz	1		
IC1	ATMEGA16-A	1		
U1	MAX232	1		
U2	PTN78000W	1		
U3	PTN78020W	1		
Puente1, Puente2	LMD18200	2		
L1	Inductor	1		
Conecotor 2x1	Conecotor 2x1	7		
Conecotor 3x2	Conecotor 3x2	1		
Conectores	Conectores	9		
SW1, SW2,	Pulsador NA	2		
SW3	Llave dos vías	1		
Q1	NPN	1		

8. Herramientas de software

- KiCAD (Build: 20100406 SVN-R2508)-final: Diseño de esquemáticos y ruteo de placas
- CuteCom (v0.22.0): Captura de datos por puerto serie (S.O. Linux - Ubuntu)
- L^AT_EXLenguaje para paginado de este informe
- Dia (v0.97.1) Creación de diagramas de flujo y bloques. (S.O. Linux - Debian)
- Kile (v2.0.85) Entorno gráfico para desarrollar código L^AT_EX. (S.O. Linux - Debian)
- Mercurial (v1.6.4) Sistema para el control de versiones.
- TortoiseHG (v1.1.1) Entorno gráfico para utilización de Mercurial. (S.O. Linux - Debian)

9. Bibliografía

- I.Scott Mackenzie, "The 8051 Microcontroller", Prentice Hall, 1995
- Datasheet del Microcontrolador ATMEGA16

10. Apéndice

Esquemáticos (*)

Código fuente (*)

*Ver páginas adjuntas a este informe.