Rastreador Mercurio ALCABOT´2001

Miguel Magán Corrochano (miguelmmc@terra.es Tlf: 91.433.68.39)

José Antonio Morales Rodríguez (ja021175@terra.es Tlf: 91.697.53.48)

0. Resumen

Fig 1. Miguel Magán, José Antonio Morales y Mercurio

Mercurio es un robot rastreador diseñado originalmente como práctica libre en la asignatura de Control Electrónico Digital, perteneciente a la titulación de Ingeniero Electrónico. Este robot debía ser capaz de rastrear una pista con curvas difíciles y ángulos rectos, implementando con controlador tipo P para realizar un control más sofisticado que un controlador todo-nada.

Las modificaciones que se le ha añadido a esta nueva versión de Mercurio son la detección de marcas para la elección del camino óptimo en las bifurcaciones y las mejoras en el código.

Miguel Magán es estudiante de 2º Ingeniería Técn. Telecomunicaciones Esp. Sistemas Telemáticos en la UAH.

José Antonio Morales Rodríguez es estudiante de 5º de Ingeniería Electrónica. Anteriormente cursó Ingeniería Técn. Telecomunicaciones Esp. Sistemas Electrónicos en la UAH.

1. Introducción

Mercurio es un rastreador basado en una estructura de tracción diferencial [1]. El material utilizado para su construcción ha sido una plancha de metacrilato donde se han colocado 2 servos trucados Futaba en su parte trasera y una rueda loca en la delantera.

Consta de tres tarjetas colocadas en una disposición de torre en el centro de la plancha. La placa inferior es la de sensores; utiliza ocho sensores CNY70. La placa intermedia es la placa de control; el MCU utilizado es un 68HC11E1 en modo boot-strap. La placa superior es la etapa de potencia; se implementa un puente en H basado en el L298N y un regulador LM7805.

El sistema se alimenta con una batería de 9.6V-650mAh de Nikko.

En la figura 2 se puede observar la disposición de los distintos elementos.

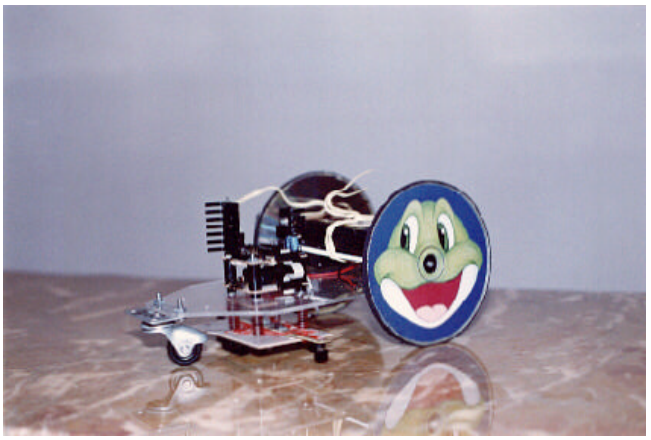

Fig 2. Mercurio

Los objetivos perseguidos con este rastreador es mejorar los algoritmos para obtener una buena relación control-velocidad.

2. Plataforma mecánica

La estructura de Mercurio está formada por una plancha de metacrilato transparente de 3mm de grosor. Se optó por esta solución, frente planchas de contrachapado o cartón endurecido, por ser un material económico, resistente y de fácil manipulación, que permite ser cortado y taladrado con gran comodidad. Es una buena oportunidad de comprobar el comportamiento de este material bajo los focos de la competición, aunque probablemente habrá que proteger los sensores.

Fig 3. Despiece de Mercurio

El detalle de los distintos elementos se puede observar en la figura 3.

Los motores están colocados en la parte trasera del robot. Se han ensamblado a la plancha mediante escuadras de

Meccano, y para ofrecer una mayor robustez, se han unido ambos motores por su parte superior mediante dos barras. Este lugar será el destinado a la ubicación de la batería.

La velocidad de los dos motores no es la misma, pudiendose comprobar este hecho dejando a Mercurio sobre una superficie blanca.

Las ruedas son CD dobles recubiertos de una goma para ofrecer más adherencia al suelo. Es una rueda fácil de construir y lo suficientemente grande como para alcanzar grandes velocidades lineales con motores lentos. El material de contacto es la espuma de las alfombrillas de ratón. Este es un aspecto crítico en la construcción de todo robot, y un comportamiento positivo, permitirá la construcción sencilla de ruedas.

En la parte central del robot se ubican las placas en una disposición de torre. La placa inferior es la dedicada a los sensores y se encuentra situada por debajo a la plancha a ras del suelo. La placa de control está inmediatamente encima del metacrilato y la placa de potencia es la superior por llevar disipadores. Las placas están sujetas y alineadas mediante cuatro varillas enroscadas. Entre las placas de potencia y control se han puesto topes en las varillas, y entre el metacrilato y la placa de sensores se ha puesto muelles, para asegurar que esta placa no va a vibrar y que se va poder regular la altura de una forma más sencilla que si se pusieran también topes. Los extremos de las varillas tienen tuercas.

En la parte delantera se ha colocado una rueda loca para ofrecer un tercer punto de apoyo. Hay que prestar atención al hecho de que esta rueda gire libremente 360º y que no deba soportar mucho peso, de ahí que la pila esté en el otro extremo del robot.

Para mejorar la estabilidad ante los cambios de dirección bruscos, el centro de gravedad del robot debe caer en el eje de simetría longitudinal.

Mercurio posee una estructura de triciclo con dos ruedas tractoras independientes y una pasiva. Posiblemente sea la estructura óptima para esta categoría, ya que se puede realizar un control sencillo ante trayectorias suaves y es capaz de girar ±90º sobre el punto medio del eje de los motores.

3. Arquitectura hardware

- Tarjeta de sensores. Consta de 8 sensores CNY70 [2] dispuestos como se muestra en la figura 4.

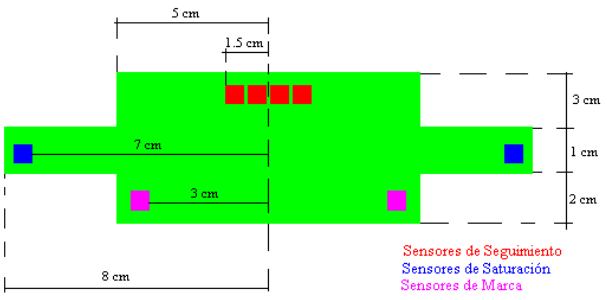

Fig 4. Disposición de los sensores

La característica más importante es que el CNY70 es un sensor de infrarrojos reflexivo con emisor y receptor en el mismo encapsulado. La distancia óptima de medida se encuentra dentro del rango 0-5mm, lo que obligará a que esta placa esté muy cerca del suelo.

El circuito eléctrico implementado se muestra en la figura 5 [3].

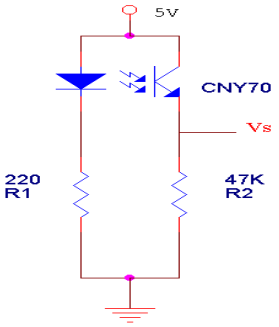

Fig 5. Circuito de polarización del CNY70

Por tanto, cuando el sensor se encuentre sobre la línea negra, la luz emitida por el diodo no se reflejará, el transistor estará cortado y $Vs \rightarrow 0$; del mismo modo, sobre el fondo blanco $Vs \rightarrow V_{cc}$. Realmente con un valor de R_E=47KΩ, la tensión de salida varía entre $1.35\ V \leq V_S \leq 4.75\ V$, debido a las tensiones de saturación en conducción y las corrientes de fugas en corte del optodispositivo.

Los sensores de seguimiento de línea están conectados al ADC del 68HC11. Sin embargo, la salida de los sensores de saturación o marca debe ser una señal digital, por lo que se debe introducir una puerta schmitt-trigger (74hc14) para regenerar la señal.

Mercurio dispone de 4 sensores de seguimiento de línea. Su función es seguir el centro de la línea siempre que sean rectas o curvas no muy pronunciadas.

Se hicieron numerosas pruebas variando el número de sensores (2, 3 y 4), la distancia entre ellos (desde completamente juntos hasta una distancia máxima entre sensores que fue la mitad del ancho de la línea. Utilizando el mismo algoritmo de control, se llegó a la conclusión de que la disposición más adecuada eran 4 sensores completamente juntos.

Cuanto más adelantados estén estos sensores menos cabeceará el robot, pero hay que prestar atención al hecho de que la rueda loca debe girar con libertad. Retrasar los sensores de seguimiento hasta la posición del eje de los motores supondría que pequeños errores se magnifiquen en la cabecera.

Los sensores de saturación están conectados al recurso Input Capture del 68HC11. Su función es detectar la línea cuando la trayectoria es un ángulo recto o lo suficientemente brusca como para que los sensores centrales se pierdan. Se puede observar en la figura 4 que estos sensores están ligeramente retrasados respecto a los sensores de seguimiento (esto facilitará el algoritmo de recuperación de la trayectoria) y muy separados del eje longitudinal de la tarjeta (hay que evitar que se puedan detectar de forma accidental las marcas de bifurcación). Su ubicación, por tanto, es empírica, determinada por la capacidad de seguimiento de Mercurio.

Los sensores de marca permiten detectar las señalizaciones que se encuentran antes de las bifurcaciones. Están a unos 2 cm del eje de simetría. En principio, una posición más restrasada o adelantada no ha supuesto mejora significativa. Las salidas de estos sensores están conectadas al puerto C. Hubiese sido mejor conectarlos a otros Input Capture, pero no se disponía de los suficientes recursos.

- Tarjeta de control. El cerebro de Mercurio es un 68HC11E1 de Motorola [4][5][6]. Se ha pretendido hacer una tarjeta lo más pequeña posible, lo que ha implicado que sólo se implementara los recursos imprescindibles (esquema Circuito de control): oscilador basado en un cristal de 8.00MHz, circuito de reset basado en red RC, jumpers para modificar la velocidad y la posibilidad de detectar marcas.

Mercurio utiliza las siguientes características de la familia 68HC11E1:

Encapsulado	52 pines PLCC
Tamaño RAM	512 bytes
Tamaño E^2PROM	512 bytes
Entradas de ADC usadas	4
IC utilizados	2
OC utilizados	4
Entradas del puerto C	4
Salidas del puerto B	2

Tabla 1. Recursos usados del 68HC11E1

La reducidad memoria E^2PROM condiciona el uso del lenguaje ensamblador para la elaboración del programa. Usar un compilador facilita la generación del SW, pero el programa se haría muy grande y requeriría memoria externa. Otra alternativa hubiese sido usar la versión 68HC11E2 que dispone de 2KB de E^2PROM, pero durante el desarrollo de Mercurio, fue un componente que no estaba disponible.

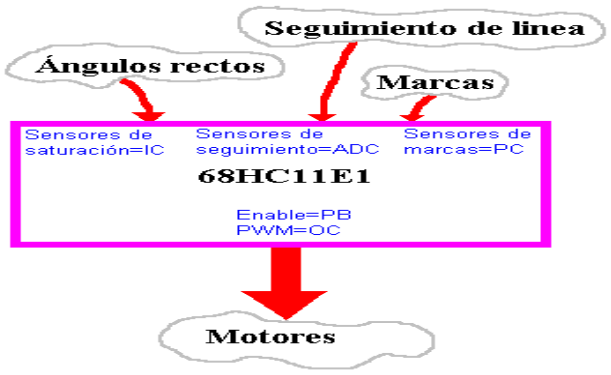

Fig 6. Recursos del 68HC11E1 utilizados

La figura 6 muestra la conexión de la tarjeta de control y sus recursos con el resto de elementos de Mercurio. Los sensores de seguimiento está conectados a AN1..3, los de saturación a IC2..3 y las marcas a PC0..1.

- Tarjeta de potencia. Funcionalmente se puede dividir en una parte de alimentación, donde con un regulador 7805 se transforman los 9.6 V de la batería en 5V, y otra parte de driver de los motores, formada por un puente en H bipolar L298 y un buffer 4050 para atacar de forma correcta al puente desde el MCU y poder así controlar bidireccionalmente 2 motores con PWM. En la figura 7 se muestra esta idea

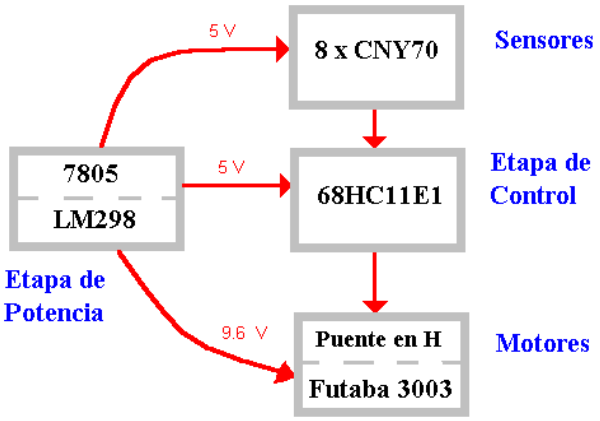

Fig 7. Etapa de potencia

Esta tarjeta, al igual que el resto, está grapinada, pero con cable de mayor grosor, pues debe ser capaz de soportar corrientes mayores. Tanto el puente como el regulador disponen de disipador.

4. Software y estrategias de control

Mercurio está regido por un sistema de control en tiempo real. Para ello utiliza la interrupción en tiempo real programada a 4.1ms.

El programa debe contemplar las 3 posibles situaciones que se pueden dar:

- Rutina del PID. Para el seguimiento de la línea se ha implementado un controlador tipo P. Recordando un poco la teoría de control clásica [7][4], este controlador tiene la característica de que la señal de control que se aplica es $u(k)=Kp*\varepsilon(k)$. La ventaja de este controlador es que es sencillo de implementar; para Kp grandes, el robot es capaz de seguir curvas muy pronunciadas, pero oscilará en las líneas rectas (es equivalente a decir que en régimen permanente siempre mantiene un error); para Kp pequeñas, las oscilaciones en las líneas rectas son menores, pero se pierde más fácilmente en las curvas (es equivalente a decir que ante cambios en la consigna, la respuesta del sistema es muy lenta). El valor de Kp=28h se ha ajustado de forma empírica, llegando a un compromiso entre estas dos características. Hubiese sido mejor desarrollar un PID completo, pero al programar en ensamblador, el código se complicaba algo más, y crecía muchísimo en tamaño.

La señal de error es $\varepsilon(k)$=Sensores de la Izquierda-Sensores de la derecha. Como en cada lado hay dos sensores de infrarrojos, es necesario combinar y filtrar esta información: (Externo&F0h + Interno>>4)>>1. Los desplazamientos a derecha equivalen a un filtro paso bajo, y hacen al sistema más robusto ante pequeños cambios en la lectura de los sensores.

La velocidad de los motores se controla mediante PWM, cuyo período es 4.1ms, aprovechando la rutina de tiempo real. Cada motor tiene asociado una variable con la que se controla el tiempo a nivel alto de la señal que ataca a los motores, denominadas PWMDER y PWMIZQ. Hablar de velocidad es lo mismo que poner código entre 0h y 2004h (durante 4.1ms la señal estaría a nivel alto). Es conveniente

recordar que si PWMDER=PWMIZQ, teóricamente el robot debe ir en línea recta, pero como los dos motores no son exactamente iguales, el robot se tuerce a la izquierda.

Si el robot va por el centro de la línea, el error sería cero. Esto implica que las dos ruedas deben ir a la misma velocidad, PWMDER=PWMIZQ=mitad de la máxima velocidad posible. Se ha comprobado que un valor mayor de velocidad media hace que el sistema se vuelva oscilatorio

La figura 8 muestra un ejemplo con ε(k)>0 (Izq=1, Der=0). Esto implica que la rueda de la izquierda debe aumentar su velocidad, del mismo modo que la rueda de la derecha debe disminuir para ser capaz de seguir la línea. Entonces PWMxxx varía entre 0 y la máxima velocidad: PWMxxx=Vmedia±U(k). Si Vmedia tiene un valor alto, |U(k)| debe tener un valor más pequeño, se tiene menos capacidad de regulación, es decir, Kp debe ser menor, con lo que eso implica.

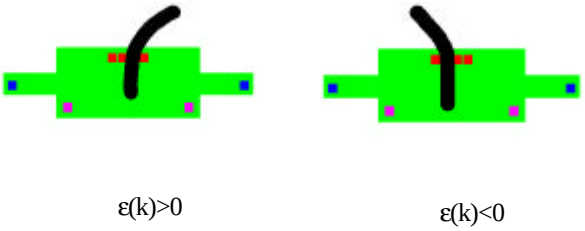

***Fig 8**. Obtención del error*

De forma equivalente, si ε(k)<0, la rueda de la derecha debe aumentar su velocidad.

- Rutinas de Saturación Izq/Der. Intervienen cuando el robot no es capaz de seguir la trayectoria o ésta es un ángulo recto. Cuando los sensores de saturación detectan la trayectoria, se activa una interrupción que habilita un nuevo estando, en el que el robot debe volver otra vez a la línea lo más rápido posible, y así, seguir ejecutando el PID.

***Fig 9**. Obtención del error*

La figura 9 muestra esta situación. La actuación de mercurio es girar bruscamente hacia el lado donde se ha detectado la saturación hasta que se detecta que los sensores de seguimiento están otra vez sobre la línea.

Sólo puede activarse un sensor de saturación en un determinado momento. Si el robot está girando bruscamente hacia un lado, aunque el otro sensor de saturación se active, no se tendrá encuenta está nueva situación hasta que no se vuelva al estado de seguimiento de línea.

Las rutinas de saturación tienen más prioridad que la de seguimiento.

- Rutina de Marca. Cuando se activa uno de los sensores de marca, se pasa a una nueva situación; la espera de una bifurcación. En función de la velocidad y de la distancia entre las marcas y la bifurcación, se inicializan unos contadores que indicarán cuando se deben desactivar las marcas. Mientras tanto, se ejecutará el PID y luego, en función de las marcas y los sensores de saturación, se elegirá una rama u otra de ls bifurcación.

Básicamente se pueden presentar los cuatro casos de la fígura 10. En 10.a y 10.c, el robot debe tomar el mismo camino que indica la marca, debe girar en el sentido del sensor de saturación que se active. Sin embargo, en 10.b y 10.d, el robot debe en el sentido contrario al indicado por el sensor de saturación. Este hecho lleva a la conclusión de que cuando se active la marca de un lado, se debe desactivar el sensor de saturación del lado opuesto durante el tiempo necesario para dejar atrás la bifurcación.

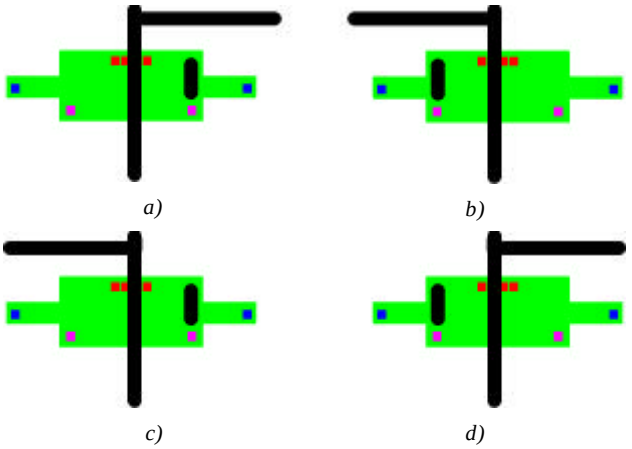

***Fig 10**. Obtención de marcas*

En la figura 11 se muestra una máquina de estado-resumen del funcionamiento del algoritmo

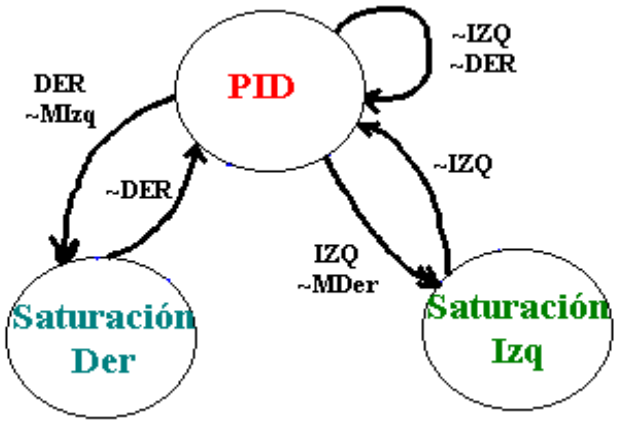

***Fig 11**. Máquina de estado*

A continuación se muestra el pseudocódigo del programa con sus distintas rutinas. Todo el código relacionado con una determinada función se ha representado en un color.

PROGRAMA PRINCIPAL. Se encarga de inicializar las variables y habilitar las correspondientes interrupciones. El núcleo del programa es un ejecutivo cíclico que ejecuta despues de cada conversión del ADC un controlador P, una rutina de detección de marca y opcionalmente una rutina de saturación a izquirda o derecha.

```
;**************************************
 INICIALIZAR VARIABLES Y RUTINAS
 WHILE (FLAG.ADC==TRUE)
        EJECUTAR CONTROLADOR PID
        EJECUTAR RUTINA DE MARCA
        IF (OPCION.IZQ==TRUE)
```

```
                RUT_IZQ
        ENDIF
        IF (OPCION.DER==TRUE)
                RUT_DER
        ENDIF
        FLAG.ADC=FALSE
 ENDWHILE
;*************************************
RUT_PID. Ejecuta el controlador PID
 OBTENER E(k)
 Up(k)=Kp*E(k)
 TRUNCAR Up(k)

 PWMDER=VMID+Up(k)
 PWMIZQ=VMID-Up(k)
;*************************************
RUT_IZQ. Comprueba si los sensores centrales estan sobre la linea o gira
de forma brusca a la izquierda
 IF (CONTADOR SATURACION!=0)
        PWMDER =VMAX
        PWMIZQ =0
 ELSE IF (OPCION.SENSOR IZQ SOBRE LINEA==TRUE)
        OPCION.IZQ = FALSE
 ENDIF
;*************************************
RUT_DER. Comprueba si los sensores centrales estan sobre la linea o gira
de forma brusca hacia la derecha
 IF (CONTADOR SATURACION!=0)
        PWMDER =0
        PWMIZQ =VMAX
 ELSE IF (OPCION.SENSOR DER SOBRE LINEA==TRUE)
        OPCION.DER = FALSE
 ENDIF
;*************************************
RUT_RTI. Rutina que interrumpe de forma periódica y actualiza el giro
y la velocidad de los motores generando el PWM. También se utiliza para decrementar contadores
 BORRAR FLAGS RTII y OC2-5
 PWMINIT=TCNT

 TOC2=PWMDER+PWMINIT      Motor de la derecha
                          {VMIN<Inc(TOC2)<VMAX}
 TOC3=PWMDER+PWMINIT

 TOC4=PWMIZQ+PWMINIT   Motor de la izquierda
 TOC5=PWMIZQ+PWMINIT

 CONFIGURAR LOS MOTORES CON LA VARIABLE GIRO
 PROGRAMAR LOS MOTORES PARA QUE SE FRENEN CUANDO
 SE CUMPLA TOCx=TCNT

 DECREMENTAR CONTADOR DE MARCA
 DECREMENTAR CONTADOR DE SATURACION
;*************************************
RUT_IC2. Interrupción provocada por que el sensor de la derecha ha cortado con la línea. El Controlador no se debe ejecutar hasta que los sensores centrales no se encuentren sobre la línea
 BORRAR FLAG IC2F
 IF (OPCION.IZQ == TRUE)
        RETORNO
 ELSE
        OPCION.DER=TRUE
         INICIALIZAR CONTADOR DE SATURACION
 ENDIF
;*************************************
RUT_IC3. Interrupción provocada por que el sensor de la izquierda ha cortado con la línea. El Controlador no se debe ejecutar hasta que los sensores centrales no se encuentren sobre la línea
 BORRAR FLAGS IC3F
 IF (OPCION.DER == TRUE)
        RETORNO
 ELSE
        OPCION.IZQ=TRUE
        INICIALIZAR CONTADOR DE SATURACION
 ENDIF
;*************************************
INIT_TIMER. Inicializacion de la Interrupción de Tiempo Real
 RTI COMO INTERRUPCION MAS PRIORITARIA
 BORRAR RTIF
 HABILITAR RTII
 INICIALIZAR PSEUDOVECTOR RTI
;*************************************
INIT_IC. Inicializacion de las IC para detectar la linea con los sensores laterales IC2,IC3 para detectar flancos
 HABILITAR IC2I,IC3I
 INICIALIZAR PSEUDOVECTOR IC2,IC3
 BORRAR IC2F,IC3F
;*************************************
INIT_ADC. Inicializacion del conversor
 HABILITAR CONVERSION CONTINUA DE AN0-3
 BORRAR FLAG
;*************************************
RUT_E. Rutina de calculo del error de los 4 sensores conectados al ADC. Los errores se dan en modulo y signo
 FILTRAR SENSORES DERECHOS
 FILTRAR SENSORES IZQUIERDOS
 ERROR=ADCIZQ-ADCDER (Módulo-Signo)
;*************************************
```

```
;TRUNCAR. Evita que el incremento de velocidad de una rueda sea superior al máximo permitido
 IF (|Up(k)|>VSPAN)
        |Up(k)|=VSPAN
 ENDIF
;*************************************
FILTRAR. Leer, combina y filtra los datos de los sensores de un lado.
   LECTURA DE SENSOR = (Externo&F0h + Interno>>4)>>1
;*************************************
MARCA. Detecta marcas de bifurcaciones. Debe ser capaz de discriminar entre entre una marca o una trayectoria
 IF (FLAG SATURACION==TRUE)
        RETURN
 ENDIF

 LEER SENSORES DE MARCAS
 IF (MARCA DERECHA==TRUE && OPCION.MIZQ==FALSE)
        OPCION.MDER=TRUE
        INICIALIZAR CONTADOR DE MARCA
        DESHABILITAR INTERRUPCION LADO IZQUIERDO
 ENDIF

 IF (MARCA IZQUIERDA==TRUE && OPCION.MDER==FALSE)
        OPCION.MIZQ=TRUE
        INICIALIZAR CONTADOR DE MARCA
        DESHABILITAR INTERRUPCION LADO DERECHO
 ENDIF
;*************************************
```

Para la elaboración del SW fue necesario recurrir a la tarjeta EVA2 [8], ya que mercurio no dispone de módulo de comunicaciones max233 propio.

Los programas utilizados para compilar y cargar los programas en E^2PROM fueron IASM11 y PCBUG11, respectivamente [4].

5. Características físicas y eléctricas más relevantes

Dimensiones (L x An. x Al.) (cm)	26 x 15 x 12
Peso del robot (batería incluida) (g)	650
Peso batería (g)	200
Corriente nominal de la batería (mAh)	650
Tensión nominal de la batería (V)	9.6

Tabla 2.Características de Mercurio

Una vez finalizado la exposición del trabajo, es conveniente evaluar las prestaciones del robot.

Se pueden dividir en físicas (velocidad máxima alcanzable, peso y dimensiones, etc) y eléctricas (tensión de alimentación, consumo, etc).

Se aconseja estructurar esta información mediante tablas.

6. Agradecimientos

Mercurio ha sido una realidad gracias al estímulo que fue el ver funcionar por primera vez al Neuronbot de Javier García Castaño, en la primavera del 99.

Se quiere agradecer a la Asociación de Electrónica E.U.I.T. de Telecomunicación de U.P.M. su apoyo logístico y sus ideas para la construcción de Mercurio. En especial a Manuel Sánchez y a Jesús Donate.

Finalmente, la ayuda de Laila Oudda Santos fue imprescindible para la elaboración de las fotografías y del poster.

Referencias

[1] J. Borenstein, H.R. Everett, and L. Feng. *Where am I? Sensors and Methods for Mobile Robots Positioning. University of Michigan. Disponible en internet.*

[2] CNY70. *Hoja de características de Temic Semiconductors. Disponible en internet.*

[3] José Mª Angulo. *Microbótica. Ed Paraninfo.*

[4] Francisco Javier Rodríguez y otros. *El microcontrolador MC68HC11 y herramientas de desarrollo. Publicaciones UAH.*

[5] P. Spasov. *Microcontroller technology. The 68HC11.Prentice Hall international. 2nd Edition.*

[6]MC68HC11. *Reference manual. Motorola. Disponible en internet.*

[7] K. Ogata. *Discrete-time control systems. Prentice Hall international. 2nd Edition.*

[8] S. Romero. *Monitor para familia de microcontroladores MC68HC11. Versión II.*