

μ Robot Diego

Andrés Vicente, Vidal Moreno, Belén Curto, Eduardo Espino, Ramiro Aguilar

Departamento de Informática y Automática

Universidad de Salamanca

Plaza de la Merced s/n. 37008 Salamanca

{bcurto, control}@abedul.usal.es, {andres, eduardo, ramiro}@tejo.usal.es

Resumen

En este artículo, se describe la visión, misión y detalles específicos de la concepción del microrobot rastreador “Diego”. Este robot está construido con elementos sencillos y baratos desde el punto de vista hardware, y se incorporan soluciones algorítmicas que van desde una estrategia de control que utiliza la lógica *fuzzy* hasta la incorporación de autómatas finitos deterministas.

Palabras Clave: Autómata Finito Determinista, Control Borroso, Lógica Borrosa, Máquina de Estados, Microrobot.

1 INTRODUCCIÓN

En este trabajo se presenta el diseño de un robot rastreador cuya misión es seguir un camino determinado por una línea negra sobre un fondo blanco. Decide qué ruta tomar cuando encuentra un cruce de dos caminos en función de una marca.

Se ha propuesto la utilización de componentes sencillos y baratos, que permitan con un coste reducido llevar a cabo la misión planteada. El esfuerzo se ha centrado en el diseño y desarrollo del software de control que, con la restricciones impuestas por el hardware utilizado, permita dirigir al robot a lo largo de la trayectoria de referencia (la línea negra) con un doble criterio de rapidez y precisión. Pero, cómo lograr mucha rapidez y alta precisión?. Las respuestas vienen respaldadas por la lógica borrosa que fundamentada por muchos trabajos desde sus inicios en 1965 [6] [5] hasta el momento se han observado muchos adelantos en aplicaciones de control así como en otras áreas [2] [1].

Así, el trabajo está estructurado de la siguiente manera, en la sección 2 se describen los aspectos mecánicos

y los materiales empleados para la construcción del microrobot (μ R), en la sección 3 se describen los sensores, actuadores y los aspectos de procesamiento; el software y las estrategias de control son descritos en la sección 4 y, finalmente, se emiten los resultados y principales conclusiones de este trabajo.

2 PLATAFORMA MECÁNICA USADA

El robot tiene un diseño propio compuesto de un chasis de material plástico con unas dimensiones de 20x30 cm., con 2 ruedas de 14 cm. de diámetro de fabricación propia que están acopladas a dos motores procedentes de campo del modelismo y dispone de una tercera rueda “loca”. Los motores están dispuestos en forma opuesta y hacen que las ruedas estén separadas 10 cm por lo que puede realizar giro diferencial con radio mínimo de 5 cm.

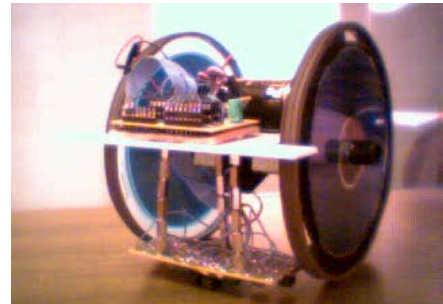


Figura 1: Vista parcial de Diego.

3 ARQUITECTURA HARDWARE

3.1 SENSORES Y ACTUADORES

Dispone de 8 sensores infrarrojos dispuestos en forma de ‘V’ invertida (Fig. 2) que detectan la línea negra. En principio la entrada es una palabra de 8 bits, pero

para la consecución de los objetivos de este trabajo esta información ha de ser procesada para que la dirección tome valores continuos.

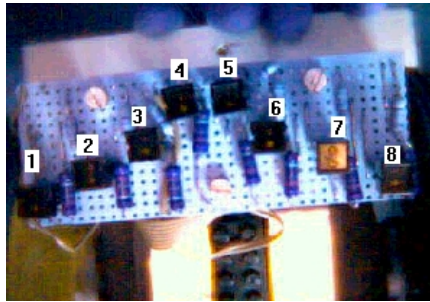


Figura 2: Disposición de los sensores de los 8 sensores de infrarojos de “Diego”.

Los actuadores son dos motores con reductora incorporada y control PWM integrado. Giran hasta 45° en 0.12s y tienen una alimentación entre 4.5 y 6 V. por lo que alcanza una velocidad lineal máxima de 44 cm/s aproximadamente (ver Fig. 1). A éstos se le aplica una señal PWM de unos 4.5 ms por ciclo, teniendo 8 niveles de potencia hacia atrás y 8 hacia adelante (entre 85 y 101 unidades de potencia respectivamente). El control no es más que la escritura del ciclo PWM en un puerto de 8 bits (donde se realiza un filtrado de primer orden de la salida para mejorar el control, por ejemplo: $\alpha S_n + (1 - \alpha)S_{n-1}$).

3.2 CPU y PROGRAMACION

Se usa un μP 16F876A de Microchip con instrucciones RISC a 4 Mhz (1 millón de instrucciones por segundo) con 8 Kbytes de memoria y 3 Puertos E/S multi-propósito. Además de realizar control PWM, tiene conversores A/D, temporizadores, etc.

La programación se realiza con el compilador C2C++: se edita el código fuente (c y ensamblador) y después de compilarlo, se graba en el μP con la tarjeta SMT2 (tipo crack TV) mediante el puerto serie del PC.

4 SOFTWARE Y ESTRATEGIAS DE CONTROL

El modelo básico del software se basa en el típico ciclo entrada-proceso-salida. Los problemas a resolver son dos: 1) Que el robot se sitúe y siga la línea de la forma más eficiente posible tomando como solución la lógica borrosa y, 2) Detectar marcas de bifurcación y tomar bifurcaciones correctamente para lo que se utiliza una máquina de estados.

En las siguientes secciones se describe, en primer lu-

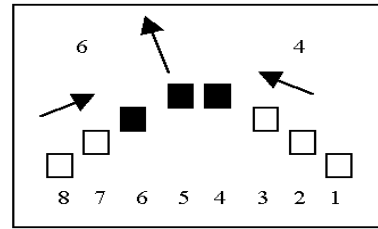


Figura 3: Ejemplo de dirección: $6+4=10$.

gar, el tratamiento de la información de entrada. A continuación se presenta el algoritmo borroso utilizado para, posteriormente presentar el autómata de estados que controla el comportamiento global. Finalmente se presenta el algoritmo de control.

4.1 ENTRADAS DE INFORMACIÓN

El aspecto crucial es la determinación de la información de entrada. Así, considerando la disposición de los sensores del μR se establece que:

- En cada instante de tiempo, se realiza la lectura de estímulos en los sensores.
- La entrada ha de ser procesada con un doble propósito: aislar el algoritmo del ancho de la línea y obtener la dirección a seguir. Para ello, se busca el primer bit puesto a "uno" por la derecha y el primero por la izquierda. Se suman sus posiciones y obtenemos la dirección de la línea a seguir. En este caso obtendremos 16 valores posibles entre 2 y 16 más el 0. Este valor que se utilizará para el algoritmo borroso utilizado (ver Fig. 3). Con ello, si la suma está por debajo de 9 se considera que se detecta línea negra a la derecha del μR . Si por otra parte, el valor de la suma esta por encima de 9, se detecta la línea negra a la izquierda del μR .
- Se puede considerar una entrada adicional que se corresponde con la dificultad del camino a seguir que, o bien se determina mediante la lectura de los sensores o se establece inicialmente.
- Un aspecto crucial es la detección de las marcas de bifurcación. Para detectar una marca de bifurcación, buscamos si entre el primer bit por la derecha y el primero por la izquierda hay un hueco (algún 0). Si además este hueco dura un tiempo determinado, consideramos que es una marca y no una lectura esporádica. Por diferencias, respecto a lecturas anteriores, podremos saber de que lado está la marca, y por consiguiente que lado debemos tomar en la bifurcación. Un ejemplo se puede observar en la Fig. 4

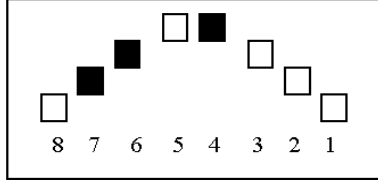


Figura 4: Ejemplo de detección de un hueco en la dirección del sensor 5.

A continuación se van a revisar los principales componentes software. En primer lugar, el sistema de control borroso que se encarga de que el robot haga un correcto seguimiento de la línea. A continuación se describe el autómata de estados finitos que determina el comportamiento global. Así, finalmente se presentará el algoritmo que soporta el funcionamiento del robot.

4.2 LÓGICA BORROSA

En primer lugar se van a describir los conjuntos borrosos definidos para las variables de entrada y salida del control utilizado en el robot. A continuación se presentarán las reglas utilizadas.

4.2.1 VARIABLES DE ENTRADA

Se considerarán, las siguientes variables de entrada al sistema.

A = Situación de la línea negra respecto al frente del robot.

B = Tipo del camino del entorno

Para cada variable, se calculan los grados de pertenencia respecto de los siguientes conjuntos borrosos. Entonces:

A_- = {Dirección negativa} (camino a la derecha); A_0 = {Dirección cero} (camino al frente) y; A_+ = {Dirección positiva} (camino a la izquierda). Es decir, $A_- = \{x | \mu_{A_-}(x)\}$; y:

$$\mu_{A_-}(x) = \begin{cases} 1 & \text{si } x < \frac{a+b}{2} \\ \frac{1}{2} - \frac{x-b}{l} & \text{si } \frac{a+b}{2} \leq x \leq \frac{b+c}{2} \\ 0 & \text{si } x > \frac{b+c}{2} \end{cases} \quad (1)$$

siendo a , b , c y d los valores que dividen en tres intervalos el universo de valores $[min, max]$ para cada variable, donde $a = min = 2$ y $d = max = 16$ (l es el ancho de intervalo para cada conjunto borroso).

Análogamente,

$$\mu_{A_0}(x) = \begin{cases} 0 & \text{si } x < \frac{a+b}{2} \\ \frac{x - \frac{a+b}{2}}{l} & \text{si } \frac{a+b}{2} \leq x \leq \frac{b+c}{2} \\ 1 - \frac{x - \frac{b+c}{2}}{l} & \text{si } \frac{b+c}{2} \leq x \leq \frac{c+d}{2} \\ 0 & \text{si } x > \frac{c+d}{2} \end{cases} \quad (2)$$

y

$$\mu_{A_+}(x) = \begin{cases} 0 & \text{si } x < \frac{b+c}{2} \\ \frac{1}{2} + \frac{x-c}{l} & \text{si } \frac{b+c}{2} \leq x \leq \frac{c+d}{2} \\ 1 & \text{si } x > \frac{c+d}{2} \end{cases} \quad (3)$$

El gráfico de las funciones μ_{A_-} , μ_{A_0} y μ_{A_+} se muestra en la Fig. 5.

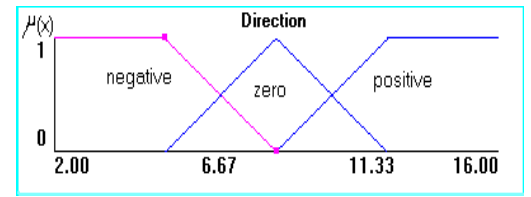


Figura 5: Gráfica de las funciones de pertenencia μ_{A_-} , μ_{A_0} y μ_{A_+} para la variable lingüística "dirección".

Por otra parte, para la variable "entorno" se definen los siguientes conjuntos borrosos:

B_e = {Entorno de camino fácil}, B_r = {Entorno de camino regular} y B_d = {Entorno de camino difícil} cuyas funciones de pertenencia tienen la misma forma que las funciones de pertenencia descritas en las ecuaciones 1, 2 y 3 resaltando $a \approx 0$ y $b \approx 1$. El gráfico de estas últimas puede apreciarse en la Fig. 6.

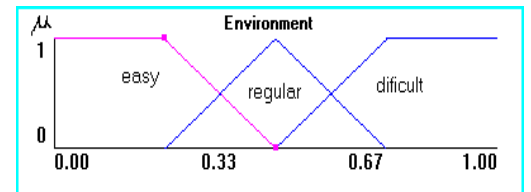


Figura 6: Gráfica de las funciones de pertenencia μ_{B_e} , μ_{B_r} y μ_{B_d} para la variable lingüística "dificultad del entorno".

Cabe resaltar que los conjuntos borrosos de la variable "entorno" se generan a través de la variable "dirección".

4.2.2 VARIABLES DE SALIDA

Se definen las siguientes variables de salida:

D = Potencia del motor derecho y,

$I = \text{Potencia del motor izquierdo.}$

Con lo que se definen sus conjuntos borrosos; para la variable “potencia del motor derecho”:

$D = \{\text{Potencia negativa alta}\}$ (la rueda gira rápidamente hacia atras), $\mu_{D_{nh}}$

$D = \{\text{Potencia negativa media}\}$ (la rueda gira regularmente hacia atras), $\mu_{D_{nm}}$

$D = \{\text{Potencia cero}\}$ (la rueda gira un poco hacia atras o hacia adelante), μ_{D_z}

$D = \{\text{Potencia positiva media}\}$ (la rueda gira regularmente hacia adelante), $\mu_{D_{pm}}$

$D = \{\text{Potencia positiva alta}\}$ (la rueda gira rápidamente hacia adelante), $\mu_{D_{ph}}$

El gráfico de las funciones de pertenencia de los conjuntos borrosos de variable “potencia del motor derecho” se puede apreciar en la Fig. 7, en el universo de discurso [85, 101]. De forma análoga se realiza la definición de los conjuntos borrosos que determinan la “potencia del motor izquierdo”, donde se ha de tener en cuenta que están configurados de forma opuesta.

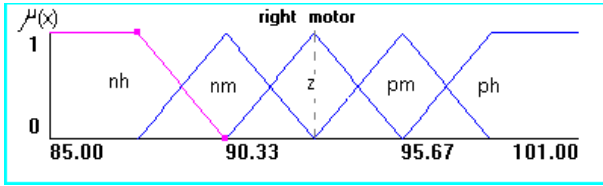


Figura 7: Gráfica de las funciones de pertenencia $\mu_{D_{nh}}$, $\mu_{D_{nm}}$, μ_{D_z} , $\mu_{D_{pm}}$ y $\mu_{D_{ph}}$ para la variable lingüística “potencia del motor derecho”.

4.2.3 REGLAS BORROSAS

En las tablas 1 y 2 se agrupan las reglas borrosas que generan una matriz con acciones de control sobre los motores derecho e izquierdo respectivamente.

Tabla 1: Acciones de control para el motor derecho.

Entorno	Dirección		
	Negativa	Cero	Positiva
Fácil	z	ph	ph
Regular	nm	ph	ph
Difícil	nh	ph	ph

Así, por ejemplo, las siguiente regla permitirá al móvil realizar giro diferencial hacia la derecha en un camino dificultoso.

Regla 9: IF *entorno* es **difícil** AND *dirección* es **negativa** THEN *potencia del motor derecho* es **negativo**

alto (nh) AND *potencia del motor izquierdo* es **positivo alto (ph)**.

Tabla 2: Acciones de control para el motor izquierdo.

Entorno	Dirección		
	Negativa	Cero	Positiva
Fácil	ph	ph	z
Regular	ph	ph	nm
Difícil	ph	ph	nh

La matriz de acciones de control se genera *off-line* debido a las limitaciones que ofrece el microprocesador utilizado. Mediante simulación, se generarán todos los posibles valores de las entradas, se realiza la evaluación de las reglas borrosas y se procede a la obtención de la variable de salida.

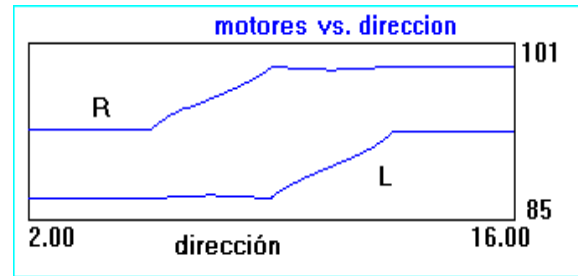


Figura 8: Función de transferencia “ambos motores” vs. “dirección” en un entorno fácil.

Así, por ejemplo, haciendo variar la entrada *dirección* (de 2 a 16) y estableciendo *entorno fácil*, el comportamiento conjunto de las relaciones “motor derecho” vs. “dirección” y “motor izquierdo” vs. “dirección” se pueden observar en la Fig. 8. Como el accionar del μ robot se refiere al control de potencias sobre ambos motores, esto significa que: cuando se detecta la línea muy a la izquierda o muy a la derecha el giro del robot no es diferencial (ya que el entorno es fácil, se supone que no hay curvas pronunciadas).

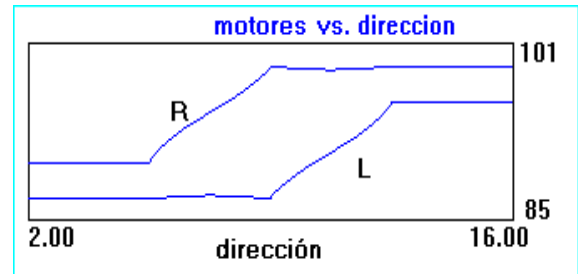


Figura 9: Función de transferencia “ambos motores” vs. “dirección” en un entorno regular.

Similarmente, variando *dirección* y estableciendo *entorno regular* se tiene el control sobre las potencias de los motores cuyo comportamiento puede observarse en la Fig. 9: un entorno regular con curvas no muy cerradas que necesitan giros regularmente pronunciados realizan giro diferencial mínimo.

Análogamente, variando *dirección* y estableciendo *entorno difícil* se tiene el control sobre las potencias de los motores cuyo comportamiento detallado se observa en la Fig. 10: un entorno difícil con curvas muy cerradas que necesitan giros muy pronunciados requieren de giros diferenciales (en el concurso de microrobots, las pistas de prueba eran dificultosas).

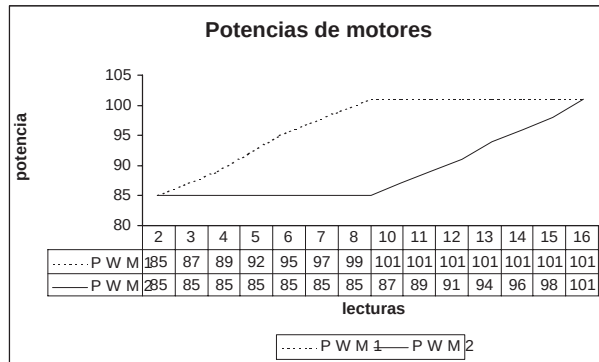


Figura 10: Acciones de control “óptimas” sobre los actuadores de Diego en un entorno difícil.

4.3 MÁQUINA DE ESTADOS

Para que el robot pueda identificar una marca de dirección en la pista y para que pueda decidir qué hacer cuando encuentre una bifurcación se define un Autómata Finito Determinista (AFD) compuesto de estados, eventos y acciones (ver Fig. 11). A continuación se describen estos elementos:

- Estados
 - 0: control normal
 - 1: detección de marca de bifurcación.
 - 2: control lateral según la marca leída.
- Eventos
 - A: marca falsa (hueco esporádico).
 - A': marca verdadera de bifurcación (hueco durante un tiempo determinado).
 - B: línea detectada sin marcas (sin huecos).
 - C: temporizador si se alcanzó un tiempo determinado.

• Acciones

En 0: control normal (lectura de la suma de los bit izquierdo y derecho).

En 1: control lateral (opuesto al lado de la marca: 2*BitNoMarca).

En 2: control lateral (del lado de la marca: 2*BitLadoMarca).

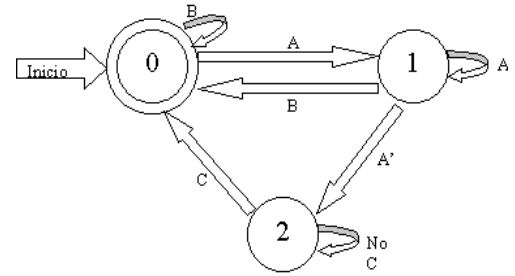


Figura 11: Máquina de estados para el control de bifurcaciones.

Diego sigue la línea siempre que sus sensores no detecten huecos o marcas (estado 0 o normal), si detecta alguna marca, se cerciora de que sea una marca adecuada (considerando el largo de las marcas establecidas en el concurso), si lo es (estado 1), en función de la posición de la marca, realiza un control lateral del camino en un tiempo (estado 2), finalmente, retoma su estado normal (ver Fig. 12).

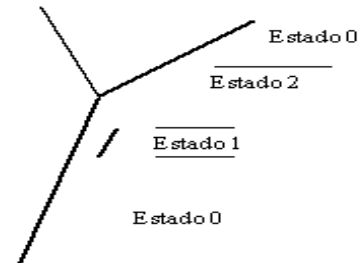


Figura 12: Ejemplo de funcionamiento con el AFD.

4.4 DESCRIPCIÓN DEL ALGORITMO

Como se ha presentado previamente el robot utiliza una entrada evaluada a partir de la lectura de los sensores que determina la dirección a seguir con 16 valores posibles entre 2 y 16 mas el 0. Este valor es el que utilizamos para obtener la potencia a aplicar a cada motor (ver Fig. 3).

Asimismo, es capaz de detectar marcas que definen la bifurcación, de forma que una vez detectado el lado de la marca y pasada ésta (Estado 2), se hace el control

solamente por el lado donde ésta estaba, durante un tiempo suficiente, de forma que se ignoran las lecturas de la parte contraria a la marca.

Así, para conseguir estas funcionalidades, en la práctica se han implementado las funciones:

- bitDerecho: Retorna la posición del primer bit por la derecha a 1
- bitIzquierdo: Retorna la posición del primer bit por la izda a 1
- hayHueco: Retorna si hay o no hueco en la lectura
- ladoMarca: Retorna de que lado está la marca, tomando como referencia la última lectura sin hueco.
- calculaEstado: Retorna el nuevo estado de funcionamiento, según el estado actual y la entrada. (Ver Fig. 11).
- actuaMotores: Según el sentido de la entrada, aplicamos la potencia que nos indique la tabla de constantes, precalculada por el sistema borroso explicado (Ver Fig. 10).

El pseudocódigo del algoritmo de funcionamiento se puede observar en la Fig. 13.

```
Estado=0 //Control normal
//T es una cte con el numero de ciclos para
//considerar que es una marca y no una lectura espúrea
direccion=0; //Está en función del estado y la entrada.
//Valdrá entre 0 y 16 (ver. Fig 10);
Mientras (1==1)
{
    estado=calculaEstado(estado,entrada);
    si estado=0 o estado =1 //Control normal
    direccion =bitDerecha+bitIzquierda;
    si no si estado = 2
    {
        si ladoMarca=derecha
            direccion =2*bitDerecha; //ignora izda
        si no si ladoMarca=izda
            direccion =2*bitIzquierda; //ignora derecha
    }
    actuaMotores(direccion);
}
```

Figura 13: Algoritmo global de funcionamiento.

5 RESULTADOS Y CONCLUSIONES

El robot construido tiene una importante capacidad de rastreo. Así alcanza una velocidad lineal de unos 40

cm/s, incluso en curvas, ya que siempre hay un motor que está funcionando a la máxima potencia. Puede realizar giros diferenciales y conseguir tomar curvas de hasta 5 cm de radio. También gira correctamente en ángulos rectos. Ha sido probado en el concurso de microrrobots “ALCABOT 2002” celebrado en Alcalá de Henares donde consiguió la primera posición en la modalidad de rastreadores ante más de 50 robots competidores de toda España. No penalizó ni una sola vez por tomar un camino incorrecto o salirse de la pista. (<http://www.alcabot.uah.es>).

Ha sido necesario realizar un esfuerzo de optimización en su implementación por las restricciones impuestas por el hardware utilizado. Se trata de una primera versión por lo que en futuras implementaciones, se espera introducir las mejoras suficientes que redunden en un comportamiento optimizado del robot.

Agradecimientos

Agradecimientos a todas las personas que de algun modo u otro han influido en la realización de este trabajo así como al Departamento de Informática y Automática de la Universidad de Salamanca.

Referencias

- [1] D. Dumitrescu, et.al. *Fuzzy Sets and Their Application to Clustering and Training*, CRC Press. 2000.
- [2] A. Kandel *Fuzzy Mathematical Techniques with Applications*, Addison-Wesley, Menlo Park.
- [3] Microchip Technology Inc. PICmicro Mid-Range MCU Family, Reference Manual. 1997.
- [4] Microchip Technology Inc. PIC16F87X Data Sheet, 28/40-Pin 8-Bit CMOS FLASH, Microcontrollers. 2001.
- [5] E. Trillas. Sobre Funciones de Negación en la Teoría de Conjuntos Borrosos, *Stochastica*. 1979.
- [6] L. Zahde. Fuzzy Sets. *Information and Control*. Vol 8. 1975.