Controlador borroso implementado en FPGA

Gerardo Aranguren, Mariano Barrón, Joseba López de Arroyabe y Alberto Chavarría
Departamento de Electrónica y Telecomunicaciones
E.T.S. Ingenieros Industriales y Telecomunicaciones
Universidad del País Vasco, Alameda. Urquijo s/n, 48013 Bilbao
jtpararg@bi.ehu.es

Resumen

Para el control de procesos en tiempo real se han propuesto varios sistemas hardware específicos en los que el número de variables de entrada y de salida se sitúa alrededor de cuatro, con una alta velocidad de procesado. En el presente artículo se presenta un diseño hardware que es capaz de gestionar un gran número de entradas, basándose en una estructura RISC segmentada. La configuración, similar a un microprocesador, dota de gran flexibilidad al sistema, y la aplicación de ciertas simplificaciones generadas mediante un preprocesado de las instrucciones consigue una reducción importante en el número de reglas a examinar, con lo que el tiempo de ejecución es apropiado para aplicaciones en tiempo real.

Palabras Clave: Fuzzy, Pipe-line, FPGA.

1 INTRODUCCIÓN

En la última década se han desarrollado diferentes diseños hardware orientados a ejecutar algoritmos borrosos en tiempo real. Por un lado están los que se basan en microprocesadores de propósito general adecuadamente programados. Por otro lado están los que utilizan un hardware totalmente específico. Ambos extremos tienen sus ventajas e inconvenientes.

El empleo del software proporciona una gran flexibilidad para definir los conjuntos borrosos, las reglas borrosas y escoger los algoritmos de inferencia. Pero debido a que todo el proceso de borrosificación, inferencia y desborrosificación se realiza de forma secuencial, se produce una drástica reducción de la velocidad de inferencia a medida que aumenta el número de variables o reglas a evaluar.

El empleo de un hardware específico consigue una alta velocidad a costa de una importante limitación en la flexibilidad. Entre las primeras realizaciones con esta solución hay que destacar las aportadas por Yamakawa [1], Katsumata [2] y Eichfeld [3].

Algunos de estos productos han adquirido carácter comercial como: NLX230 de American Neuralogix, FC110 de Togai Infralogic, Omron FP-3000, WARP de SGS-THOMSON [4]. En algunos casos poseen ciertas capacidades avanzadas como paralelismo o escalabilidad.

En el diseño que se presenta en este artículo se busca una solución intermedia, con flexibilidad y velocidad suficiente para el control de procesos en tiempo real. Evidentemente lo óptimo es tratar de conseguir las cualidades ventajosas de los dos tipos de diseño: un procesamiento secuencial, tipo microprocesador que permite una gran flexibilidad, pero con elementos paralelos, principalmente pipe-line que proporcionan una mejora del rendimiento en tiempos. Además, esta arquitectura puede alojarse en un circuito integrado de pocos pines y, por tanto, de un costo reducido.
Para optimizar el diseño y las prestaciones del circuito, se requiere aproximarse a la concepción de un procesador RISC (*Reduced Instruction Set Computer*), para ello es necesario adecuar el algoritmo borroso y realizar algunas simplificaciones, tal y como se expusieron en Estylf 96 [5], y que consiguen una reducción notable en el número de instrucciones a ejecutar. También se ha realizado un software de adquisición de las variables y reglas borrosas, presentado en Estylf 97 [6].

2 DIAGRAMA GENERAL

En la figura 1 se muestra el circuito completo para control borroso. Como se puede observar está compuesto de 3 bloques principales.

Microcontrolador. Se encarga del control general del sistema: obtiene las variables de entrada, las digitaliza y envía al sistema de control, ordena el comienzo del procesado del controlador borroso y recibe los resultados para enviarlos a los actuadores. Supone un interfaz entre el controlador borroso y la aplicación.

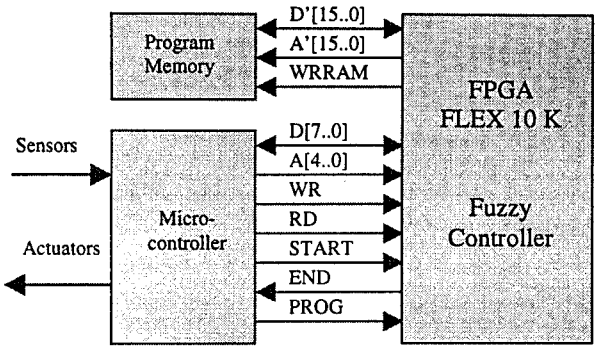


Figura 1 Diagrama de bloques del controlador

Memoria de programa. Contiene el programa objeto generado con el software presentado en Estylf 97 [6]. Las instrucciones son de 16 bits por tanto la anchura del bus de datos de esta memoria se ha tomado también de 16 bits. En sistemas expertos recibe el nombre de base de conocimientos.

FPGA. En una FPGA (*Field Programmable Gate Array*) se ha implementado el algoritmo necesario para efectuar todas las operaciones borrosas: cálculo del grado de pertenencia, comparaciones mínimo y máximo y la desborrosificación por el método de Media de los Máximos. Las entradas necesarias para su funcionamiento son las instrucciones de la Memoria de programa, y las variables de entrada suministradas por el microcontrolador.

El funcionamiento de todo el circuito está regulado por el programa generado por el software de definición de reglas [6], dispuestas en la memoria de programa. La definición de las instrucciones utilizadas en este programa, que rigen el funcionamiento de todo el circuito, se puede apreciar en la tabla 1.

Tabla 1. Conjunto de instrucciones del controlador

Código de Instrucción	Explicación
0000.XXXX.XXXX.XXXX	No operación
0001.DBBB.XXXV.VVVV Dir16	Salto a Dir16 si el bit BBB de la variable VVVVV vale D
0010.XXXX.XXXX.XXXX Dir16	Salto incondicional a la dirección absoluta Dir16
0011.AAAA.AXXX.XXXX	Final del actuador AAAAA. (Desborrosificar)
0100.CCCC.CUXX.XXXX	Then consecuente CCCCC. (Último consecuente si U=1)
0101.XXXX.XXXX.XXXX	Final del programa. Esperar un nuevo START
0110.TTTT.TTTT.XXXX	Máximo grado de activación de regla = TTTT.TTTT
1ZZZ.ZZYY.YYYV.VVVV	Subpremisa (If variable VVVVV is AS, o AZ)

Los siguientes apartados describen los circuitos más importantes del controlador borroso alojado en la FPGA.

3 CÁLCULO DEL GRADO DE PERTENENCIA

El software desarrollado para el controlador borroso descrito en este artículo, permite la definición de los términos lingüísticos de las variables de entrada, mediante funciones de pertenencia de perfil lineal, de cuatro tipos distintos: S-términos, Z-términos, Λ-términos, y Π-términos [5]. Sin embargo, el controlador borroso trabaja internamente con los dos literales saturados (S-términos y Z-términos), lo que posibilita una reducción del número de reglas a procesar, y la evaluación del grado de pertenencia de las variables de entrada a cada S-término o Z-término, en un solo pulso de reloj. Por otro lado, los literales saturados quedan completamente especificados mediante un par ordenado de valores (Zj,i e Yj,i) como puede observarse en la figura 2.

Para el cálculo del grado de pertenencia de las variables de entradas a sus literales saturados se han ideado varios circuitos. En primer lugar se describe el más evidente y a continuación se tratan las modificaciones del mismo que permiten la integración de la memoria necesaria dentro de una FPGA.

El cálculo del grado de activación de cada subpremisa $\mu(AS_{j,i})$ se puede realizar mediante una LUT (*Look Up Table*), implementada en una memoria ROM (*Read Only Memory*). El contenido de la ROM depende exclusivamente de la arquitectura utilizada, y es el mismo para cualquier sistema borroso. Esta LUT tiene por entradas el valor nítido de la variable de entrada a examinar V_i, definida por 8 bits, y los determinadores del literal saturado $Z_{j,i}$ e $Y_{j,i}$ definidos por 5 bits cada uno (figura 2).

Tras la orden de inicio de ejecución del algoritmo borroso, la memoria de programa comienza a proporcionar de forma ordenada el programa. Todas las instrucciones pasan por el calculador de grado de pertenencia, pero sólo las que tengan el código de instrucción adecuado servirán para el proceso. Estas instrucciones, cuyo bit de mayor peso es 1 (Tabla 1), contienen los términos $Z_{j,i}$, $Y_{j,i}$, según la simplificación primera presentada en [5], y los 5 bits de la dirección de la variable de entrada V_i. El valor V_i sirve para obtener el valor actual de la variable "i", que habrá sido almacenado previamente por el microcontrolador en la memoria de entrada (figura 3). Los otros dos datos son retrasados mediante registros en pipe-line para que el proceso se realice sin pérdida de pulsos de reloj. Con todos los datos

disponibles se direcciona la LUT, cuya salida proporciona el valor $\mu(AS_{j,i})$, según ilustra la figura 2.

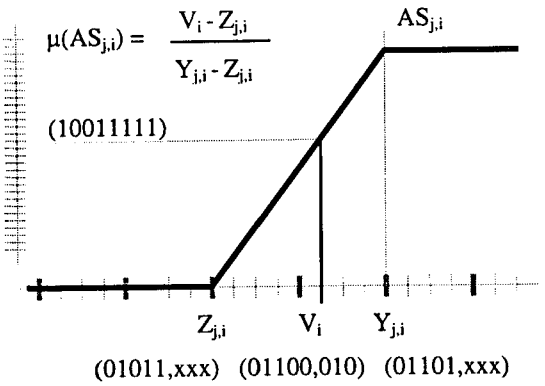


Figura 2. Cálculo del grado de pertenencia de una premisa mediante una LUT.

Todos los conjuntos borrosos se han reducido a dos tipos: S-términos y Z-términos. Expresado en forma de ecuación los S-términos:

$$\begin{aligned} &\text{Si } V_i < Z_{j,i} && \mu(AS_{j,i}) = 0 \\ &\text{Si } Z_{j,i} < V_i < Y_{j,i} && \mu(AS_{j,i}) = \frac{255 \cdot (V_i - Z_{j,i})}{Y_{j,i} - Z_{j,i}} \\ &\text{Si } Y_{j,i} < V_i && \mu(ASj,i) = 255 \end{aligned} \qquad (1)$$

Si la forma del literal corresponde a un Z-término las expresiones anteriores cambian a:

$$\begin{aligned} &\text{Si } V_i < Y_{j,i} && \mu(ASj,i) = 255 \\ &\text{Si } Y_{j,i} < V_i < Z_{j,i} && \mu(AS_{j,i}) = \frac{255 \cdot (Z_{j,i} - V_i)}{Z_{j,i} - Y_{j,i}} \\ &\text{Si } Z_{j,i} < V_i && \mu(ASj,i) = 0 \end{aligned} \qquad (2)$$

El término 255 sirve para escalar la salida y generar un dato de 8 bits.

El cálculo del grado de pertenencia, como se puede ver, es independiente de la aplicación, de las reglas y de los conjuntos borrosos, lo que representa una gran ventaja respecto de otros diseños.

El tamaño de la memoria está determinado por el número de bits de dirección que se utilizan: 8 para V_i, 5 para $Z_{j,i}$ y 5 para $Y_{j,i}$, hacen un total de 18 bits de direcciones que corresponden a una de memoria de 256 Kbytes.

Modificando el circuito mediante dos restadores que realicen las operaciones $B_{j,i}=Vi-Z_{j,i}$ y $Cj,i=Y_{j,i}-Z_{j,i}$, las ecuaciones (1) y (2) quedan reducidas a:

$$\begin{aligned} &\text{Si } V_i < Z_{j,i} && \mu(AS_{j,i}) = 0 \\ &\text{Si } Z_{j,i} < V_i < Y_{j,i} && \mu(AS_{j,i}) = \frac{255 \cdot B_{j,i}}{C_{j,i}} \\ &\text{Si } Y_{j,i} < V_i && \mu(AS_{j,i}) = 255 \end{aligned} \qquad (3)$$

El tamaño de la memoria LUT en este caso viene determinado por los 8 bits de $B_{j,i}$ y los 5 bits de $C_{j,i}$, quedando reducido a 8 Kbytes. Bien es cierto que hay que tener en cuenta los signos de las distintas operaciones y darles el tratamiento adecuado.

En este último circuito la mitad de la LUT está ocupada por el número 255. Haciendo un plegamiento de la memoria se puede reducir el tamaño hasta 4 Kbytes. Y perdiendo un bit de precisión en el dato de entrada V_i , el tamaño de la LUT queda reducido a 2 Kbytes, cantidad de memoria disponible en la FPGA empleada.

4 CÁLCULO DEL ACTUADOR

Todos los elementos de cálculo del actuador se pueden implementar en un dispositivo lógico programable. Este circuito trabaja secuencialmente en función de las instrucciones recibidas desde la base de conocimiento.

Las operaciones se realizan con varios niveles de segmentación, adicionales a los tres niveles de segmentación en el cálculo del grado de pertenencia, y que son: lectura de la instrucción, adquisición de la variable y cálculo del grado de pertenencia.

Los niveles de segmentación de este circuito son: cálculo del mínimo o del máximo, cálculo de los sumatorios y de la multiplicación, y cálculo de la división que exige el método de desborrosificación Media de los Máximos (figura 3).

El primer nivel de segmentación que agrupa al decodificador de instrucciones y a los comparadores de mínimo y máximo, no presenta dificultades de diseño.

Para el segundo nivel se han ensayado distintos multiplicadores, optando al final por una variante del multiplicador de Booth, que proporciona el resultado en un solo pulso de reloj. Los sumadores de este nivel de segmentación son sumadores rápidos.

En el tercer nivel de la segmentación aparece el divisor para el cálculo de la Media de los Máximos, es decir:

$$MoM = \frac{\sum B \cdot \mu_R}{\sum \mu_R} \qquad (4)$$

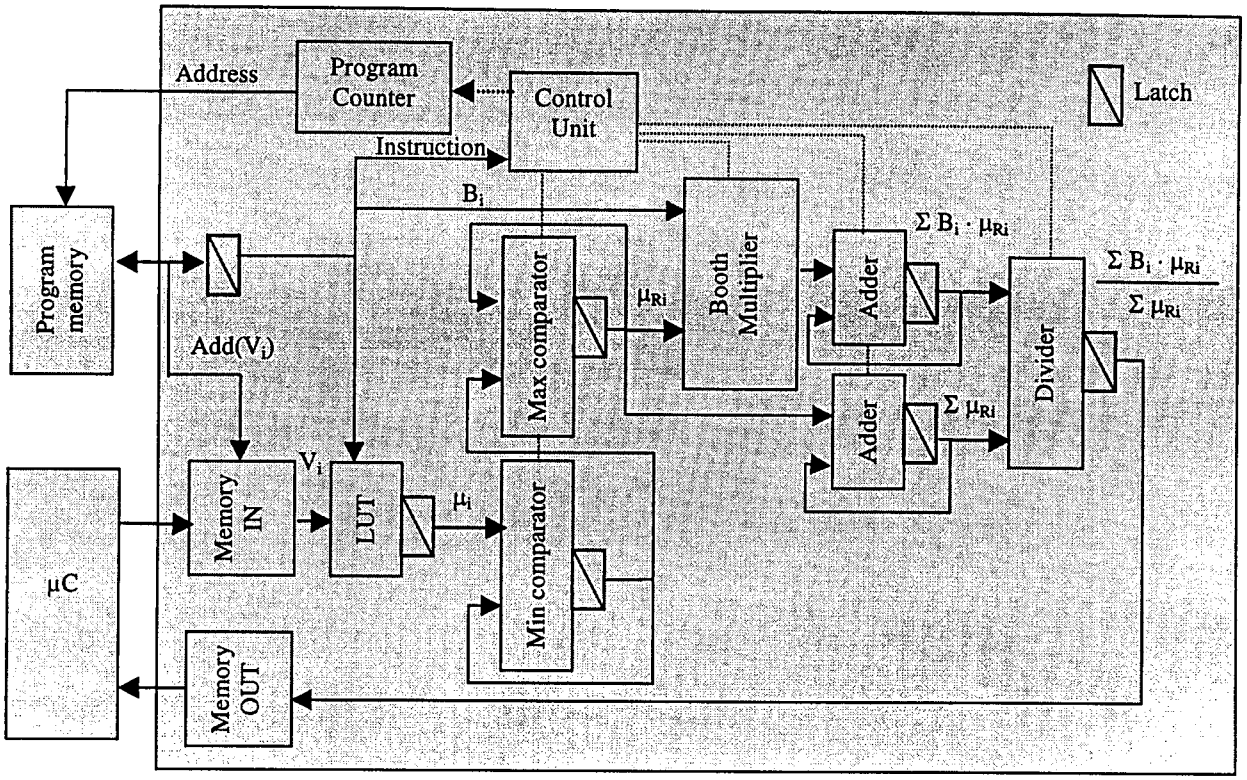


Figura 3. Diagrama de bloques de la FPGA que ejecuta el algoritmo mínimo-máximo-media de los máximos

El divisor se ha implementado mediante un circuito de restas y desplazamientos que consume 8 impulsos de reloj, simultáneos al análisis de las subpremisas correspondientes al siguiente actuador o variable de salida. Seguidamente se ofrece una breve descripción del método utilizado en la división.

Con las asignaciones: **D** = dividendo, **d** = divisor, **C** = parte entera del cociente, y **R** = resto de la división entera, la igualdad:

$$\mathbf{D = d \cdot C + R} \qquad (5)$$

conduce a:

$$\mathbf{D \geq d \cdot C} \qquad (6)$$

y si el cociente **C** se expresa en base 2, utilizando n bits:

$$\mathbf{D \geq d \cdot (\Sigma\, c_i \cdot 2^i)},\ \text{con } \mathbf{i = (n\text{-}1)...0},\ \text{y con } \mathbf{c_i = 0 \text{ ó } 1} \qquad (7)$$

La expresión (7) permite ir calculando los coeficientes $\mathbf{c_i}$, de uno en uno y empezando por $\mathbf{c_{n-1}}$. En efecto, si al comparar D y $(d \cdot 2^{n-1})$ se tiene $D \geq (d \cdot 2^{n-1})$ entonces $\mathbf{c_{n-1}}$ valdrá 1, pero si $D < (d \cdot 2^{n-1})$, entonces $\mathbf{c_{n-1}}$ valdrá 0.

Si $\mathbf{c_{n-1}}$ vale 0, se tendrá:

$$\mathbf{D \geq d \cdot (\Sigma\, c_i \cdot 2^i)},\ \text{con } \mathbf{i = (n\text{-}2)...0} \qquad (8)$$

Si por el contrario $\mathbf{c_{n-1}}$ vale 1, haciendo la resta: $\mathbf{D' = D - (d \cdot 2^{n-1})}$, tendremos:

$$\mathbf{D' \geq d \cdot (\Sigma\, c_i \cdot 2^i)},\ \text{con } \mathbf{i = (n\text{-}2)...0} \qquad (9)$$

En ambos casos se llega a una expresión que permite calcular $\mathbf{c_{n-2}}$. Repitiendo el proceso de comparaciones y restas (condicionadas al resultado de la comparación), es posible calcular todos los coeficientes $\mathbf{c_i}$. El hardware necesario para realizar el proceso descrito, consiste simplemente en un restador del mismo número de bits que el denominador, y de un registro de desplazamiento que proporcione los valores $(\mathbf{d \cdot 2^i})$, que no son otra cosa que el denominador desplazado **i** lugares hacia la izquierda.

5 FUNCIONAMIENTO GENERAL

La forma de operar del controlador borroso se parece en cierto modo a la de un microcontrolador con arquitectura RISC y disposición de buses según modelo Harvard. Su funcionamiento se inicia con la búsqueda de instrucciones y prosigue con la ejecución de las instrucciones en modo segmentado.

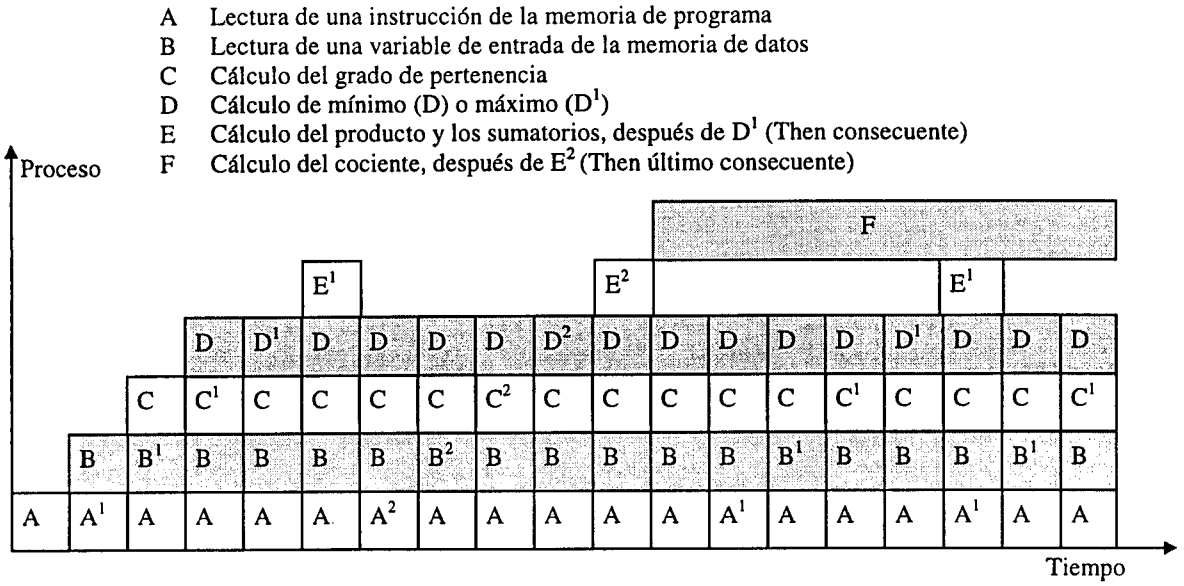


Figura 4. Diagrama proceso-tiempo mostrando la segmentación en el cálculo del algoritmo borroso

Tal vez lo más importante del circuito es haber conseguido los niveles de segmentación adecuados para que se pueda ejecutar una instrucción por ciclo de reloj.

En la figura 4 se muestran los niveles de segmentación utilizados por el controlador borroso. En ella se puede ver que en todo momento se están realizando las operaciones básicas (A, B, C y D), siendo el decodificador el encargado de seleccionar cual de las operaciones finales (E y F) se debe realizar. Todas las operaciones consumen un impulso de reloj, menos la división, que consume 8 impulsos. Esta distinta duración no supone ningún problema para la segmentación, ya que entre dos operaciones de división normalmente transcurren muchos más de 8 impulsos de reloj.

El circuito ha sido diseñado para calcular el valor de múltiples actuadores, sistema MIMO (*Multiple Input, Multiple Output*), por lo que después de una división y cálculo del valor de un actuador, puede empezar el cálculo de otro actuador. Cuando termina con el último actuador emite una señal de fin de proceso y el microcontrolador debe reinicializar un nuevo cálculo cuando los nuevos valores de las variables se encuentren escritos en la memoria de entrada de datos.

6 REALIZACIÓN FÍSICA

El circuito descrito ha sido realizado físicamente sobre un circuito impreso. El módulo microcontrolador está formado por un microcontrolador PIC16C74 con comunicación por línea serie RS-232. En este primer prototipo un ordenador se encarga de simular el proceso e indicar las variables procedentes del proceso a controlar.

El módulo de cálculo está implementado en una FPGA tipo EPF10K50RC240-3 de Altera más una EPROM de configuración y un conector para relacionarse con el programador Bit Blaster. Además se han dispuesto memorias NVRAM para el almacenamiento del programa de control borroso.

El controlador borroso dispone de dos modos de funcionamiento que pueden seleccionarse con la entrada PROG (figura 1). Cuando esta entrada vale 0 se comporta como el controlador borroso descrito en los apartados anteriores. Cuando la entrada PROG vale 1, la FPGA entra en un modo de programación que facilita al microcontrolador la actualización del programa de control borroso residente en las memorias NVRAM. El modo de programación resulta interesante en las fases de depuración o de ajuste del controlador, ya que permite cambiar rápidamente el programa con los datos recibidos desde un ordenador personal por la línea serie RS-232.

El circuito implementado en la FPGA se ha diseñado y simulado convenientemente con el software MAX+PLUS II, versión 8.1, de Altera.

Los niveles bajos de la jerarquía del diseño se han escrito en AHDL (*Altera Hardware Description Language*), mientras que el nivel superior se ha realizado gráficamente.

7 CONCLUSIONES

En comparación con otros controladores estudiados, nombrados en la introducción, podemos destacar que el circuito presentado [7]:

a) Admite hasta 32 variables de entrada con precisión de 8 bits y puede controlar hasta 32 actuadores definidos con 8 bits. Esta cantidad es mayor que cualquiera de los controladores hardware actuales, donde el número de entradas está entre 2 y 8, y el número de salidas suele ser una y raramente 2. En nuestro diseño no se pretende calcular 32 salidas dependientes de 32 entradas, sino generar una salida dependiente de 2, 3, 4 o las variables que necesite, a continuación otra salida dependiente de las mismas u otras variables, y así todas las salidas que se precisen. De esta forma se evita la necesidad de disponer de un controlador para cada función.

b) Trabaja a razón de una subpremisa por impulso de reloj. Con un reloj de 12 MHz puede analizar en un segundo alrededor de 10.000.000 de subpremisas, ya que requiere algunos impulsos de reloj para las instrucciones de control. Se trata de una velocidad notablemente superior a la de un microprocesador y algo inferior a la de los controladores borrosos de procesamiento paralelo SIMD (*Single Instruction Multiple Data*), pero con una arquitectura mucho más sencilla y sin las limitaciones de estos últimos.

c) Admite más de 60.000 subpremisas, lo que le sitúa muy por encima de las necesidades habituales. La mayoría de los controladores reducen el número de reglas a unos pocos cientos, debido a limitaciones: de anchura de los buses, de memoria, de tiempo o de capacidad de definición de las reglas. En nuestro procesador todas esas limitaciones se han eliminado, salvo la posible limitación de tiempo.

Agradecimientos

Este trabajo ha sido subvencionado por la Universidad del País Vasco / Euskal Herriko Unibertsitatea con el código: UPV 147.345 - TB058/96.

Referencias

[1] T. Yamakawa, "A Simple Fuzzy Computer Hardware System Employing Min and Max Operations - A Challenge to 6th Generation Computer", Proc. 2nd IFSA Cong., 1987.

[2] A. Katsumata, H. Tokunaga and S. Yasunobu, "Fuzzy Set Processor (FSP) for Fuzzy Information Processing", IFES'91, pp. 399-406, 1991.

[3] H. Eichfeld, M. Klimke, M. Menke, J. Nolles and T. Künemund, "A General-Purpose Fuzzy Inference Processor", IEEE Micro mag., vol. 15, n° 3, 1995.

[4] M. Lavorgna, "WARP: Weight Associative Rule Processor", Fuzzy Sets and Systems", vol. 61, n. 1, pp. 111-113, 1994.

[5] G. Aranguren, M. Rodríguez y M. Barrón, "Controlador Secuencial Segmentado para Lógica Borrosa Implementado en FPGA", ESTYLF'96, pp. 255-259, Oviedo, 1996.

[6] G. Aranguren, M. Rodríguez, J. Lz. de Arroyabe, N. Altarriba, "Software de compilación, simulación y emulación borrosos", ESTYLF'97, Tarragona, pp. 77-82, 1997.

[7] G. Aranguren, M. Barrón, J. Lz. de Arroyabe y G. García-Carreira, "A Pipe-line Fuzzy Controller in FPGA", FUZZ-IEEE'97, vol. II, pp. 635-640, Barcelona, 1997.

APROXIMACION A LA DETERMINACION DE PERFILES DE CLIENTES EN ENTIDADES FINANCIERAS UTILIZANDO TECNICAS FUZZY

Antonio Flores-Sintas[1]
Dpto. de Nuevas Tecnologías
Caja Ahorros de Murcia
Gran Vía,23
30005-Murcia
e-mail: aflores@cajamurcia.es

Joaquín Cánovas Páez
Subdirección General Comercial
Caja Ahorros de Murcia
Gran Vía,23
30005-Murcia
email: jcanovas@cajamurcia.es

Jorge García Pozzy
Dpto. de Marketing Estratégico
Caja Ahorros de Murcia
Gran Vía,23
30005-Murcia
email: jgpozzy@cajamurcia.es

RESUMEN

Presentamos en esta contribución una experiencia de aplicación de técnicas fuzzy, concretamente de clustering difuso, para la determinación de perfiles de clientes en entidades financieras, y enmarcamos dicha experiencia en un proyecto más ambicioso de determinación de perfiles de clientes utilizando inferencia difusa .

Palabras Clave: funciones de pertenencia, análisis cluster, fuzzy c-means.

1 INTRODUCCION.

"El cliente debe de presidir las actuaciones de las entidades financieras y éstas deben de ajustar la oferta de productos y servicios a la satisfacción de las necesidades del cliente. La prestación de servicios financieros debe basarse en líneas estratégicas de acción como son la segmentación de clientes, la paquetización de productos, y la calidad y personalización de los productos financieros. Sin embargo, mientras que los sectores institucional y empresarial no presentan problemas para su segmentación, no parece que sea tarea fácil en el sector de economías domésticas ya que se trata de un colectivo cuantitativamente amplio y al mismo tiempo heterogéneo. La segmentación debe estar orientada al mercado y por ello, entre los posibles criterios que se puedan utilizar, hay que dar una mayor relevancia a las ventajas buscadas por los clientes en sus relaciones con las entidades financieras y los productos y servicios que utilizan." [1]

El sector financiero es uno de los sectores más mecanizados. Además es un sector que realiza la venta directa de sus productos. Como consecuencia, la información sobre el uso que hacen sus clientes de los productos y servicios es muy alta. Sin embargo, el número de productos y servicios ofertados es, como en cualquier sector que realiza venta directa, bastante mayor que el número medio de productos y servicios consumidos por cliente. Esta hecho dificulta la aplicación de técnicas convencionales de clasificación en el espacio global de las características (productos y servicios). Por otra parte, el personal de las entidades financieras realiza diariamente una "clasificación" de sus clientes utilizando variables linguisticas del tipo: tiene saldos a la vista altos, préstamos bajos, fondos de inversión medios, etc. Parece pues razonable utilizar el mismo criterio y realizar clasificaciones unidimensionales para establecer automáticamente el perfil de los clientes.

2 TIPOS DE CARACTERISTICAS.

Para determinar el perfil de un cliente podemos utilizar cualquier característica cuantificable que proporcione información sobre el mismo. En este sentido, los intereses abonados a un cliente es una característica utilizable aunque no es un producto ni un servicio financiero, sino un coste para la entidad financiera. Como producto financiero podemos citar a los fondos de inversión, y como servicio el de nóminas.

Hemos detectado dos tipos de características cuyas distribuciones están representadas en las figuras 1 y 2. La correspondiente a la figura 1 es del tipo nómina: una buena parte de los clientes reciben una nómina distribuída alrededor del máximo de la figura, estando la media desplazada hacia la derecha del máximo. La característica correspondiente a la figura 2 es más frecuente y se presenta, por ejemplo, en los intereses abonados. Hay que señalar que la curva de la figura 2 es como la de la curva de la figura 1 desplazando el máximo hacia la izquierda.

1 Autor para correspondencia

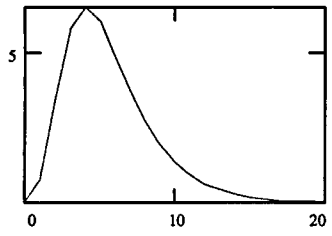


figura 1

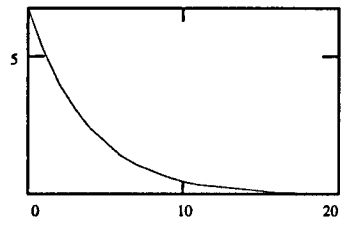


figura 2

De la forma de las distribuciones se puede deducir que no existen grupos naturales, y que la distancia Euclídea no es apropiada para realizar una partición. En cuanto al primer problema, nuestro objetivo no es detectar grupos naturales sino dividir una muestra en un número de grupos que se acomode a la forma de operar normal en las entidades financieras, teniendo en cuenta el error producido en la clasificación. En cuanto al segundo problema, lo hemos abordado partiendo del estudio de la matriz de covarianzas (en una dimensión es la varianza) presentado en [2]. Básicamente el trabajo citado deduce propiedades globales a partir de métricas locales definidas por la distancia de Mahalanobis, lo que ha permitido presentar en [3] una modificación del algoritmo Fuzzy C-Means (FCM) que se ha aplicado a la clasificación unidimensional. El error cometido al realizar la clasificación, inherente a cualquier proceso de clustering, se estudia en [4]. La formulación del algoritmo y del errror cometido se presenta a continuación.

3 TECNICA DE CLASIFICACION.

Sea $X=\{x_1,x_2,...,x_n\}\subset \mathbf{R}^F$ una muestra F-dimensional de n patrones que queremos dividir en c grupos. Sea $v=\{v_1,v_2,...,v_c\}\subset \mathbf{R}^F$ el conjunto de los prototipos de los grupos que queremos obtener. El algoritmo FCM modificado en [3] obtiene los prototipos que minimizan la función

$$J(v)=\sum_{k=1}^{c}\sum_{x\in X}u_{xk}^2 d_{xk}^2 ,$$

siendo:

$$u_{xk}=\mu_{xk}\Big/\sum_{j=1}^{c}\mu_{xj}$$

$$\mu_{xk}=\sqrt{\left|C_k^{-1}\right|}\Big/\left(1+d_{xk}^2\right)^{3/2}$$

$$(C_k)_{ij}=\frac{\sum_{x\in X}u_{xk}^2\left(x_i-(v_k)_i\right)\left(x_j-(v_k)_j\right)}{\sum_{x\in X}u_{xk}^2}$$

u_{xk} es la probabilidad de que el patrón x pertenezca al grupo k; μ_{xk} es el grado de pertenencia del patrón x al grupo k; C_k es la matriz de covarianzas difusa del grupo k. $d_{xk}^2=(x-v_k)^T C_k^{-1}(x-v_k)$ es la distancia de Mahalanobis entre el patrón x y el prototipo v_k. Los prototipos que minimizan $J(v)$ son:

$$v_k=\sum_{x\in X}u_{xk}^2 x\Big/\sum_{x\in X}u_{xk}^2 .$$

Nótese que el intervalo de variación del grado de pertenencia es diferente de la unidad lo que no es contradictorio con la teoría de conjuntos difusos [6].

El algoritmo FCM es el siguiente:

FCM-1. Fijar $\varepsilon>0$ una constante positiva pequeña. Inicializar $u_{xk}^{(0)}=\{0,1\}$. Calcular $v_k^{(0)}$ y $C_k^{(0)}$. Para iteraciones $t=1,2,...,$ los siguientes pasos:

FCM-2. Calcular $\mu_{xk}^{(t)}$ usando $v_k^{(t-1)}$ y $C_k^{(t-1)}$. Calcular $u_{xk}^{(t)}$ usando $\mu_{xk}^{(t)}$. Calcular $v_k^{(t)}$ y $C_k^{(t)}$ usando $u_{xk}^{(t)}$.

FCM-3. Si $\left\|v_k^{(t)}-v_k^{(t-1)}\right\|<\varepsilon$, y $\left\|\left|C_k^{(t)}\right|-\left|C_k^{(t-1)}\right|\right\|<\varepsilon$, $1\le k\le c$, parar. En caso contrario ir a FCM-2.

El algoritmo FCM obtiene los prototipos por iteraciones sucesivas de Picard partiendo de una partición hard cualquiera. Para el caso unidimensional, basta sustituir la matriz de covarianzas difusa por la covarianza difusa σ, de tal manera que, por ejemplo, $d_{xk}^2=\sigma_k^{-1}(x-v_k)^2$. En general, el algoritmo FCM no garantiza la obtención de un mínimo absoluto, pudiendo obtener mímimos

locales diferentes dependiendo de la partición inicial. Sin embargo en el caso unidimensional, podemos partir de una partición conveniente simplemente ordenando la muestra por el valor de la característica y comenzando con una partición de grupos tal que el primero contenga los n/c valores menores, el segundo los n/c valores siguientes, etc.

Sean $u_{xk}^*, 1 \le k \le c$, las probabilidades de pertenencia del patrón x a los c grupos representados por los prototipos v_k^* calculados por el algoritmo FCM. El patrón x pertenecerá al grupo en el que tenga la probabilidad máxima de pertenencia. El error que cometemos al realizar esa afirmación es $1 - max\{u_{xk}^*\}$. El error de Bayes, es $B = E\{1 - max\{u_{xk}^*\}\}$, siendo E la esperanza matemática [5]. El error de Bayes es el error medio que cometemos al considerar que existen c grupos.

Para una característica del tipo de la figura 1, las funciones de pertenencia a los cinco grupos en que se ha dividido la muestra están representadas, cualitativamente, en la figura 3. Nótese que la distribución de la figura 1 es una envolvente de las distribuciones de la figura 3, es decir, las funciones de pertenencia representan distribución de densidades para cada grupo. Las probabilidades correspondientes están representadas en la figura 4. Estas funciones presentan un máximo en los prototipos, después decrecen al alejarnos de los prototipos, y vuelven a crecer debido a la dependencia de la función de pertenencia del cubo de la distancia. Debemos de entender que la validez de las funciones de probabilidad está comprendida entre los mínimos más cercanos a los prototipos. La figura 5 representa a las funciones de probabilidad teniendo en cuenta los intervalos de validez. Esta funciones se pueden utilizar como funciones de pertenencia para fuzzificar las entradas en un módulo de inferencia difusa.

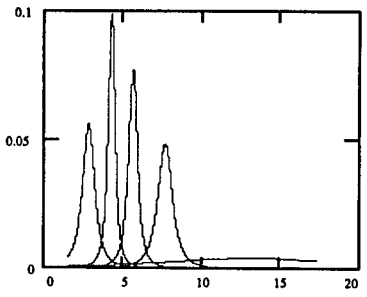


figura 3

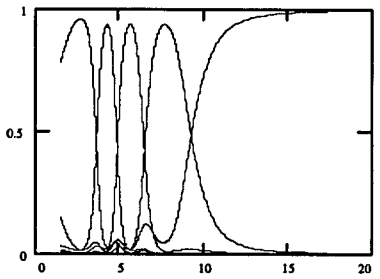


figura 4

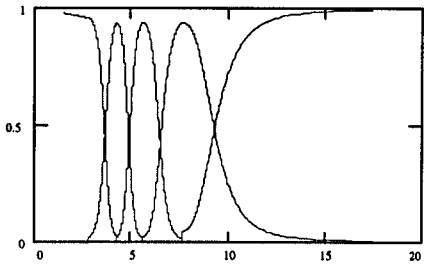


figura 5

4 APLICACION REALIZADA.

La utilización de un módulo de inferencia difusa implica investigar en las funciones de pertenencia de las salidas así como en la estructura de la base del conocimiento. Hemos preferido posponer dichas investigaciones y aplicar y contrastar la técnica de clasificación explicada en el párrafo anterior.

La aplicación realizada trataba de determinar aquéllos clientes con posibilidad medio-alta de necesitar un plan de pensiones. El equipo comercial, en base a su conocimiento del mercado, explicitó que, básicamente, el perfil de un potencial comprador de un plan de pensiones está marcado por una o más de las siguientes cualidades: capacidad de ahorro, búsqueda de beneficios fiscales, motivación de ahorro claramente finalista y cierto nivel de aversión al riesgo. Entre las muchas características que podíamos utilizar se seleccionaron seis: importe medio mensual de los ingresos por nóminas recibidos por los clientes, importe bruto total de intereses abonados a los clientes en todas sus cuentas de pasivo, valor de mercado (a una fecha determinada) de la cartera de valores de renta fija y variable, importe anual de las primas de seguros de riesgo sobre las personas (vida, responsabilidad civil,..) pagados por los clientes, valor patrimonial de los fondos de inversión en poder de los clientes, e importe de la liquidación positiva del IRPF pagado por los clientes. Las

dos primeras características nos miden un nivel de renta; la tercera y quinta nos miden directamente un nivel patrimonial y la segunda indirectamente; la cuarta característica el nivel de aversión al riesgo y/o predisposición al ahorro finalista; la quinta característica es un producto financiero con ciertas similaridades al que queríamos vender, y la sexta evidencia la necesidad de obtener beneficios fiscales.

Utilizando la técnica de clasificación explicada en la sección anterior, cada una de las características se clasificó en cinco variables linguísticas con fronteras difusas que determinaban el grado de "uso" por cada cliente en función del valor de la característica: bajo, medio-bajo, medio, medio-alto y alto. Aunque el grado de pertenencia a cada grupo está determinado por la técnica de clasificación, a efectos de fijar el número de clientes objetivo (necesario para calcular el coste de la campaña de venta), se consideró que un cliente x estaba en el grupo k de una característica si:

$$u^*_{xk} = max\left\{u^*_{xj} \middle| 1 \le j \le 5\right\}$$
$$ó\ , \left|u^*_{xk} - u^*_{x,k-1}\right| \le B, ó\ , \left|u^*_{xk} - u^*_{x,k+1}\right| \le B$$

siendo B el error de Bayes de la partición realizada para la característica correspondiente. El solapamiento existente no era importante porque la selección se realizaba a partir de un nivel determinado. Por ejemplo, clientes a partir de nómina medio-alta, es decir, clientes con nómina medio-alta y alta.

El nivel de selección de los clientes en cada característica fue diferente, determinado por criterio comercial y por el número de clientes objetivo a seleccionar. Un cliente pasó a formar parte del público objetivo de la acción de marketing, si satisfacía por lo menos uno de los criterios establecidos, es decir, por ejemplo: selecionar un cliente si su nivel de uso en la característica 1 es mayor de medio, ó si su nivel de uso en la característica 2 es mayor de medio alto, ó si su nivel de uso en la característica 3 es mayor de medio-bajo, etc. Es interesante señalar que si se cambiaba la conectiva "ó" por la conectiva "y" el número de clientes seleccionados era muy pequeño, hasta el punto de que no existía ningún cliente con nivel alto simultáneamente en las seis variables escogidas.

5 RESULTADOS.

Se seleccionaron dos grupos de clientes, uno de ellos, A, de clientes con posibilidad muy alta de tener necesidad de un plan de pensiones. El grupo A estaba contenido en un grupo global, B, tal que el conjunto B - A eran clientes con posibilidad medio-alta de necesitar un plan de pensiones. A todos los clientes se les realizó una acción de mailing, y a los clientes del grupo A, además, una acción de telemarketing con un nivel de contacto del 65%, y se les incluyó en la agenda comercial de la red de ventas. La venta del producto se realizó durante los dos últimos meses del año 97.

Durante los mismos meses del año 96 también se realizó una campaña de venta del mismo producto seleccionando los clientes por criterios socioeconómicos. Las respuestas positivas del mailing del año 97 quedaron multiplicados por el factor 3.11 respecto del año 96, en el caso del conjunto global B. Para el conjunto A el factor multiplicativo fue 6.39.

La selección realizada no era exclusiva para la consecución de los objetivos marcados a la red de ventas, pudiendo dedicarse los esfuerzos comerciales a otros públicos objetivos. Sin embargo, un plan de pensión vendido a un cliente perteneciente a la selección (grupo B) ha presentado un saldo cuatro veces superior que otro seleccionado directamente por la red de ventas. En el caso del grupo A la proporción es de seis a uno.

6 CONCLUSIÓN.

Los resultados se consideran muy positivos, aunque obviamente son debidos al conjunto de acciones realizadas (criterios de selección, clasificación, campaña de marketing y de telemarketing, y red de ventas), entre los que uno es el método de selección-clasificación realizado. Sin embargo, debemos destacar la sencillez del método y su adaptación a la manera usual de trabajar en las entidades financieras.

Creemos que el método de selección-clasificación debe ser incluído en un módulo global de inferencia difusa. Para ello hace falta desarrollar dos aspectos: uno es la selección de las características que nos proporcionan información sobre un producto, que además de incluir características seleccionadas por criterio comercial permita deducir otras. El otro es la definición de la variable de salida (grado de selección) que permita realizar una agregación de las características de entrada, en el sentido de que dos grados medios de dos características de entrada puede ser un valor alto en la variable de salida, tanto si las características son directamente agregables como si no. La agregación debe comtemplar también la ponderación de las características para conseguir un afinamiento del modelo.

AGRADECIMIENTOS.

Queremos agradecer al Departamento de Informática, Inteligencia Artificial y Electrónica de la Facultad de Informática de la Universidad de Murcia su inestimable ayuda para la realización de este trabajo, en especial a los Doctores F. Martin y J.M. Cadenas.

REFERENCIAS.

[1] Egea Krauel C,"Análisis Estratégico del Sector de Cajas de Ahorro en España",Tesis Doctoral, Universidad Autónoma de Madrid, 1988.
[2] Flores-Sintas A.,Cadenas J.M.,Martin F.,"A Local Geometrical Properties Application to Fuzzy Clustering", Fuzzy Sets and Systems, 100/1-3 (1998) 237-248.
[3] Flores-Sintas A.,Cadenas J.M.,Martin F.,"Membership Functions in the Fuzzy C-Means Algorithm", Fuzzy Sets and Systems, to be published.
[4] Flores-Sintas A.,Cadenas J.M.,Martin F.,"Partition Validity and Defuzzification", Fuzzy Sets and Systems, to be published.
[5] Fukunaga K.,"Introduction to Statistical Pattern Recognition",(Academic Press,1990).
[6] Zadeh L.,"Fuzzy Sets", Inform. and Control, 8-3 (1965) 338-353.

DESARROLLO DE UNA APLICACIÓN EXPERIMENTAL DE CONTROLADORES DIFUSOS

C.J. Jiménez, I. Baturone, A. Barriga y S. Sánchez Solano

Instituto de Microelectrónica de Sevilla - Centro Nacional de Microelectrónica.
Edificio CICA, Avda. Reina Mercedes s/n, 41012-Sevilla.
email: cjesus@imse.cnm.es

Resumen

En este artículo se describe la construcción de un sistema experimental para la realización de pruebas con prototipos ASIC de controladores difusos. Previamente se plantean descripciones y simulaciones del sistema a alto nivel. El problema de control abordado es el de mantener suspendida una pelota en el interior de un tubo de cristal mediante el flujo de aire suministrado por un ventilador situado en la parte inferior del tubo.

Palabras Clave: Lógica difusa, Aplicaciones, Hardware.

1 INTRODUCCIÓN

La lógica difusa ha demostrado suficientemente su validez en aplicaciones de control, sobre todo en sistemas cuya descripción matemática sea muy compleja, pero cuyas reglas de control puedan ser extraídas a partir de un conocimiento experto [1-4]. El control difuso puede llevarse a cabo utilizando entornos software o usando realizaciones hardware de controladores difusos [5][6].

En este artículo se describe la construcción de un sistema experimental para la realización de pruebas con los prototipos ASIC de controladores difusos diseñados y fabricados por los autores [7]. Como paso previo a su realización, se han realizado descripciones y simulaciones del sistema a alto nivel. De esta forma no sólo se ha obtenido la base de conocimientos más adecuada , sino que también se han estudiado los efectos de las variaciones en los parámetros y las no idealidades del sistema.

El problema de control que nos planteamos consiste en mantener suspendida una pelota en el interior de un tubo mediante el flujo de aire suministrado por un ventilador situado en la parte inferior del tubo. Controlando el ventilador, y por lo tanto de la intensidad del flujo de aire, se estabiliza la pelota a la altura deseada. El sistema permite la variación de la altura objetivo, situación ante la cual el controlador modifica la fuerza para poder alcanzar la nueva posición objetivo en el espacio de tiempo más breve posible.

Aunque la aplicación pueda parecer sencilla, se trata de un problema de control difícil como consecuencia de los fenómenos de fricción y turbulencias que se producen en el interior del tubo, lo que difuculta la obtención de una descripción matemática completa. Una complicación adicional es la existencia de un tiempo de retraso bastante significativo en la respuesta del sistema.

El sistema consta de los elementos que se esquematizan en la Fig. 1. Además del tubo de cristal y del ventilador, un conjunto de fotoemisores y un fotoreceptor, situados en la parte superior del tubo, se encargan de la detección de la posición de la pelota. Con esta posición y con el dato de la posición deseada se produce la acción de control que activa directamente la fuerza del ventilador.

2 MODELADO A ALTO NIVEL DEL SISTEMA

Antes de comenzar el montaje del sistema se han realizado una serie de simulaciones de alto nivel con objeto de poder analizar su comportamiento, así como para generar la base de conocimientos. En esta etapa se llevaron a cabo pruebas con controladores Fuzzy-PD y Fuzzy-PID para comprobar el comportamiento de ambos tipos de controladores. Las distintas simulaciones se realizaron con la ayuda de las herramientas de Matlab.

Para la descripción a alto nivel del sistema se supone, en primera aproximación, un sistema totalmente ideal, que considera que la pelota está sometida únicamente a dos fuerzas, una vertical hacia arriba, debida al aire enviado por el ventilador, y la fuerza de gravedad sobre la pelota. La

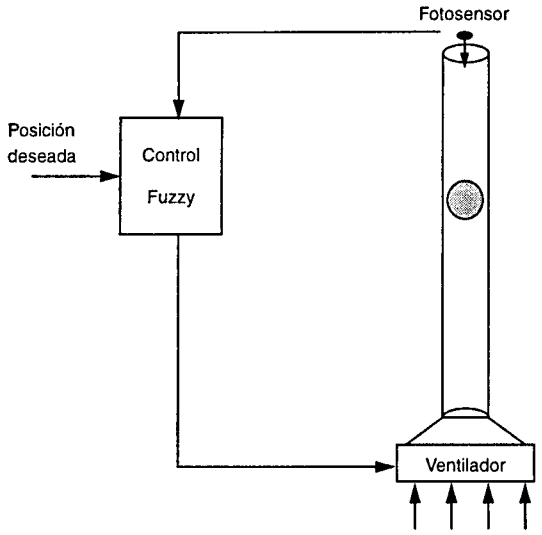


Fig. 1: Elementos del sistema experimental desarrollado.

fuerza vertical hacia arriba se corresponde con la salida del controlador difuso. Con estas dos fuerzas, la ecuación de movimiento de la pelota dentro del tubo es:

$$F = m \cdot a = F_{control} - mg \qquad (1)$$

La ecuación de la posición de la pelota tiene la forma:

$$S(t) = \iint \frac{(F_{control} - mg)}{m} dtdt \qquad (2)$$

Para controlar la posición de la pelota con este modelo del sistema consideramos, en primer lugar, un controlador Fuzzy-PD. Este controlador utiliza como entradas el error de la posición actual con respecto a la posición objetivo (error) y la variación de dicho error entre dos instantes de tiempo (Δerror). Para aproximarnos lo más posible a la realización experimental se ha discretizado el tiempo, de forma que las entradas "error" y "Δerror" no sean continuas. El diagrama de bloques del sistema con el controlador Fuzzy-PD se muestra en la Fig. 2.

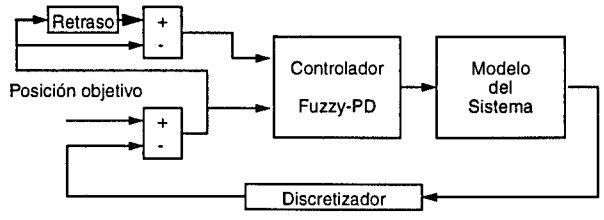
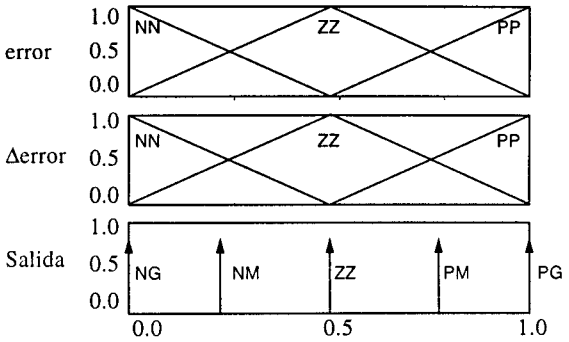


Fig. 2: Descripción del sistema con un control Fuzzy-PD.

La base de conocimientos empleada en este caso se muestra en la Fig. 3. Los antecedentes de cada una de las entradas son tres funciones triangulares equiespaciadas y los consecuentes son singularidades difusas que se sitúan también de forma equiespaciada. Se ha empleado el mínimo como conectivo de los antecedentes. La base de reglas utilizada se muestra en la parte (b) de la figura. Asimismo puede observarse en la parte (c) de la Fig. 3 la superficie de control resultante.

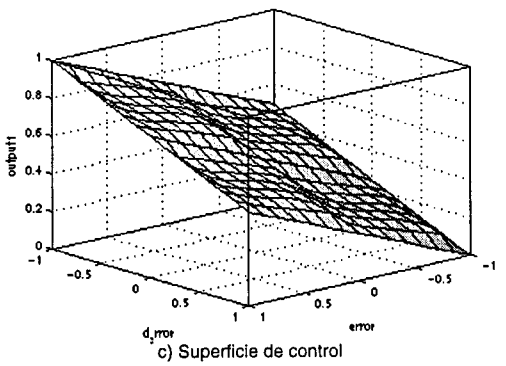
La calidad en la respuesta del control depende de otros parámetros además del ajuste de la base de conocimientos. Así, por ejemplo, al aumentar la frecuencia de operación del controlador se obtiene un mejor control. Sin embargo, ha de tenerse en cuenta que un incremento de la frecuencia provoca una disminución en la señal de variación de error,



a) Antecedentes y consecuentes

		error		
		NN	ZZ	PP
Δerror	NN	ZZ	PM	PG
	ZZ	NM	ZZ	PM
	PP	NG	NM	ZZ

b) Base de reglas



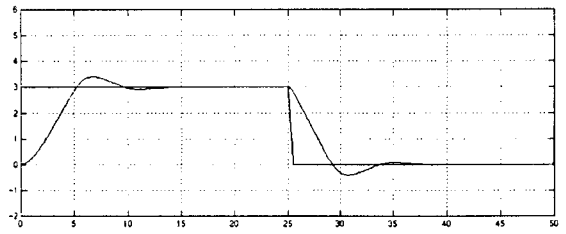
c) Superficie de control

Fig. 3: Base de conocimientos para el controlador Fuzzy-PD

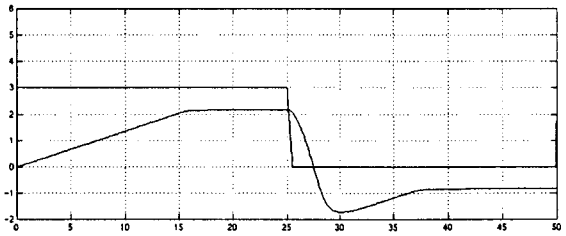
lo que puede provocar desajustes en la base de conocimientos. En la figura Fig. 4 (a) se muestran los resultados de la simulación para el modelo del sistema descrito anteriormente operando a una velocidad de 8 FLIPS. Se observa cómo el sistema alcanza la posición objetivo tanto en cambios ascendentes como en cambios descendentes.

En un segundo conjunto de simulaciones se han introducido "no idealidades" en la ecuación de modelado del sistema. En estas simulaciones se ha observado que cuando el valor del consecuente de la regla "si error = zz y Δerror = zz" no coincide con la fuerza necesaria para mantener la pelota en equilibrio, entonces, aunque sigue produciéndose una estabilización de la misma, la posición alcanzada no coincide con la deseada. Otra de las no idealidades introducidas es la respuesta del ventilador frente a la salida del controlador. Cuando esta relación no es lineal, tampoco se llegan a alcanzar las posiciones deseadas (Fig. 4 (b)).

Los problemas anteriores pueden solventarse utilizando un esquema de control Fuzzy-PID. Para ello se ha introducido una tercera entrada, la integral del error (∫error), que se encarga, mediante la acumulación de los errores, de la corrección de los comportamientos no ideales, de forma que en las circunstancias en las que el controlador Fuzzy-PD no alcanza en equilibrio la posición deseada, la acumulación del error provoca cambios en las reglas que se activan y así se modifica la salida del controlador para alcanzar la posición deseada. El diagrama de bloques del sistema con un control Fuzzy-PID se muestra en la Fig. 5.



(a) Resultados ideales



(b) Resultados con respuesta no ideal en el motor

Fig. 4: Resultados de simulación con un controlador Fuzzy-PD

La base de conocimientos del controlador, con 3 entradas y tres funciones de pertenencia en cada una de las entradas, tiene ahora 27 reglas. En el apartado (a) de la Fig. 6 se muestran las funciones de pertenencia de las entradas y las singularidades difusas de la salida. En la parte (b) se muestra la base de reglas (representada en tres tablas, cada una de ellas para un antecedente distinto de la entrada "∫error"). En la parte (c) se muestra la superficie de control, de la entrada "error" frente a la entrada "Δerror", para el valor de ∫error = 0. Distintos valores de la entrada "∫error" producirán el desplazamiento había arriba y hacia abajo de esta base de reglas.

Los resultados de simulación con un control Fuzzy-PID para el caso ideal (Fig. 7 (a)) son bastante parecidos a los obtenidos con el control Fuzzy-PD. Sin embargo, donde sí se aprecian bastantes diferencias es al introducir no idealidades en el sistema. Puede observarse en la parte (b) de la Fig. 7 el comportamiento del sistema cuando la respuesta del motor no es lineal con sus niveles de alimentación. Mientras que para el caso del control Fuzzy-PD no se conseguía alcanzar la posición objetivo, ahora, tanto en subida como en bajada, sí se alcanza.

Para este tipo de controladores se ha estudiado también la influencia de los retrasos en la respuesta del ventilador sobre el comportamiento del sistema. Como puede apreciarse en la parte (c) de la Fig. 7, al introducir un retraso de 0.25 segundos en la respuesta del ventilador, y manteniendo la no linealidad en la respuesta del motor, el sistema sigue alcanzando la posición objetivo, aunque aparece un ligero rizado en la respuesta.

3 CONSTRUCCIÓN Y CONTROL HARDWARE DEL SISTEMA EXPERIMENTAL

Una vez obtenidos en simulación unos resultados satisfactorios estamos en disposición de construir el sistema experimental.

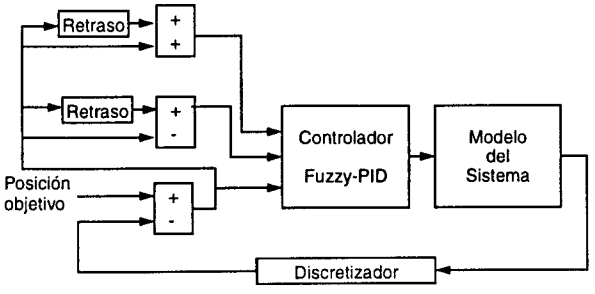
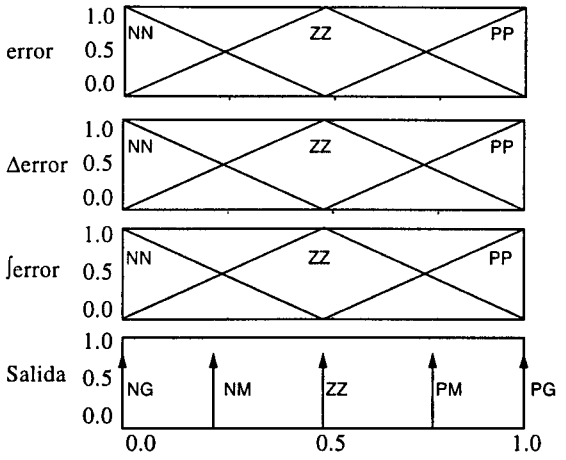


Fig. 5: Diagrama de bloques para un control Fuzzy-PID



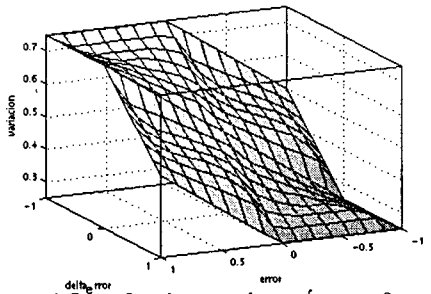
a) Antecedentes y consecuentes

∫error = NN	Δerror \ error	NN	ZZ	PP
	NN	NM	ZZ	ZZ
	ZZ	NG	NM	ZZ
	PP	NG	NG	NM

∫error = ZZ	Δerror \ error	NN	ZZ	PP
	NN	ZZ	PM	PM
	ZZ	NM	ZZ	PM
	PP	NM	NM	ZZ

∫error = PP	Δerror \ error	NN	ZZ	PP
	NN	PM	PG	PG
	ZZ	ZZ	PM	PG
	PP	ZZ	ZZ	PM

b) Base de reglas



c) Superfice de control para ∫error = 0

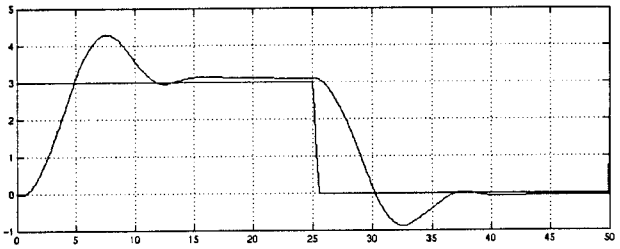
Fig. 6: Base de conocimientos del control Fuzzy-PID.

El sistema montado consta básicamente de los elementos reseñados en la introducción y mostrados gráficamente en la Fig. 1, donde el bloque denominado como "control fuzzy" tiene unos componentes encargados de la adaptación de señales y de la carga de la base de conocimientos del controlador difuso. Estos componentes se muestran en la Fig. 8.

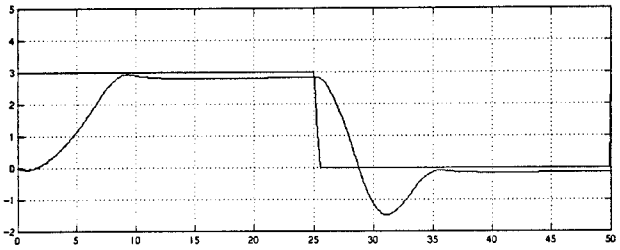
El fotoreceptor produce una corriente que depende de la distancia a la que se encuentre la pelota. Esta señal analógica es pasada a digital con un convertidor A/D. Debido a que la respuesta del fotoreceptor no es lineal con la posición de la pelota, introducimos, a la salida del convertidor tal en otra que representa posiciones equiespaciadas de la pelota a lo largo del tubo.

El cálculo de las entradas al controlador (error, variación de error e integral del error) a partir de la posición objetivo y de la posición actual se hace con la ayuda de una FPGA.

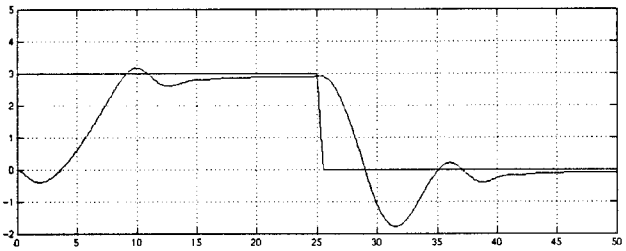
Un aspecto importante es el ajuste de los parámetros para el sistema experimental. El principal problema es el ajuste de los rangos de variación de cada una de las tres entradas al controlador difuso. Para ello, la propia FPGA que genera las señales de entrada al controlador difuso, introduce una multiplicación programable, con objeto de poder ajustar los rangos en los que se mueven cada una de las entradas.



(a): Resultados ideales



(b) Resultados con respuesta no ideal en el motor



(c) Resultados con respuesta no ideal en el motor y con retraso en dicha respuesta

Fig. 7: Resultados de simulación con un controlador Fuzzy-PID

Como controlador difuso se ha utilizado, entre los prototipos disponibles [8], uno con 3 entradas y una salida (de 6 bits cada una de ellas), 8 funciones de pertenencia en cada una de las entradas, 512 reglas y método de desborrosificación simplificado. Los antecedentes están almacenados en memoria, pudiendo tener cualquier forma, con la única restricción de que su grado de solapamiento sea como máximo 2. Este prototipo tiene la capacidad de ser programado, leyendo, la base del conocimiento a partir de un bus de 8 bits y almacenándola en memorias RAM internas. Los datos necesarios para la programación se han almacenado en una memoria EEPROM.

En la construcción del sistema experimental, el control de la velocidad del ventilador se ha hecho introduciéndole una señal de ancho de pulso modulado como señal de alimentación (el motor funciona con una tensión en continua de 12 voltios). La salida del controlador difuso, que es una señal digital de 6 bits, es convertida con la ayuda de un circuito programado también en una FPGA.

Los resultados obtenidos con el sistema experimental aparecen en la Fig. 9. En esta gráfica se muestra la posición de la pelota junto con la posición objetivo. En un primer instante, cuando la pelota está en la parte inferior del tubo, se fija la posición deseada en una zona baja. Tras la estabilización de la pelota, se cambia la posición objetivo a una zona alta del tubo y finalmente se vuelve a cambiar la posición objetivo a una zona baja del tubo. Con estos cambios, no sólo puede observarse un buen comportamiento del sistema una vez alcanzada la condición de equilibrio, sino también un buen comportamiento en las transiciones.

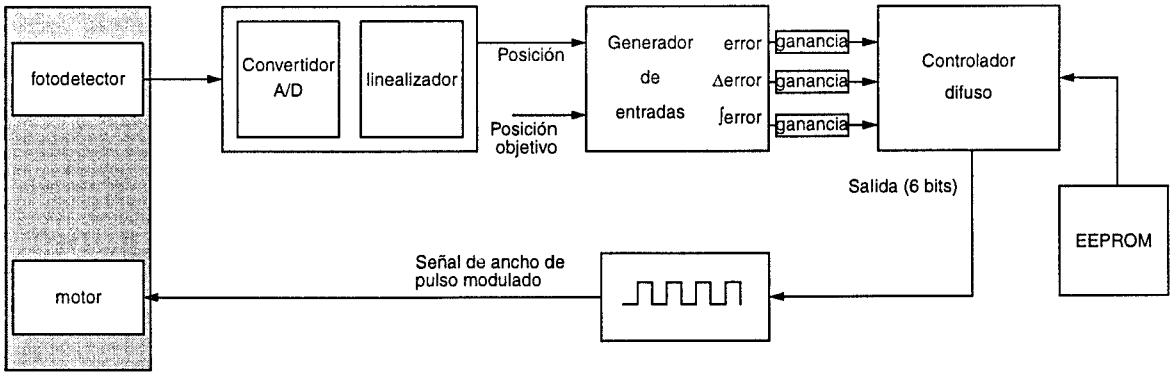


Fig. 8: Esquema de bloques del sistema de control

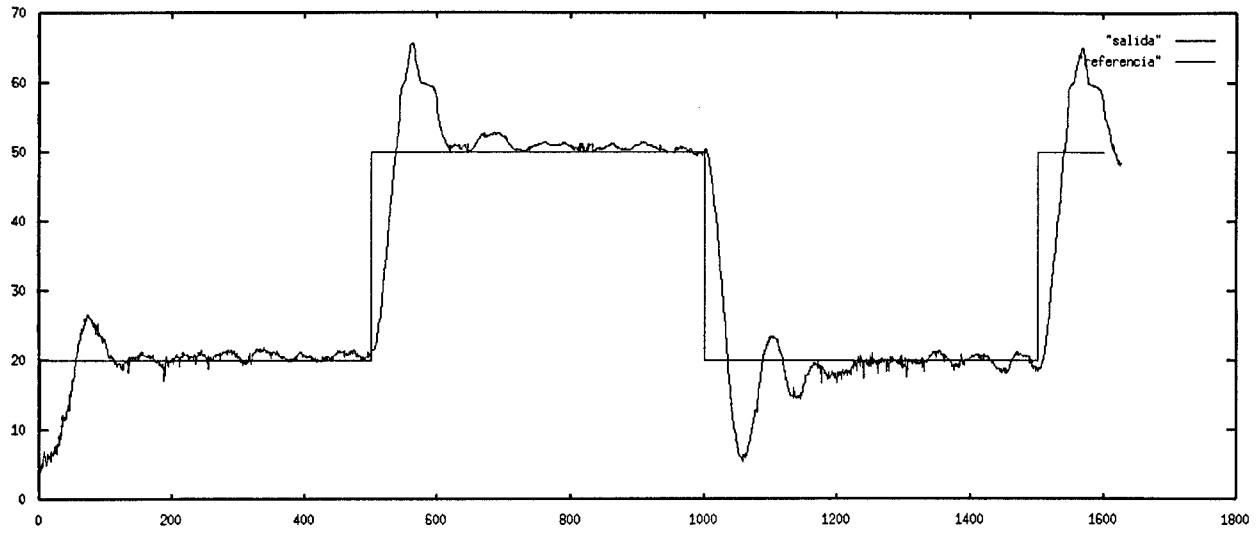


Fig. 9: Resultados experimentales

4 CONCLUSIONES

Se ha construido un sistema experimental para realizar pruebas con los prototipos ASIC de controladores difusos desarrollados por los autores. Empleando un modelo simple de alto nivel del sistema y tras realizar simulaciones se seleccionó un control del tipo Fuzzy-PID y se generó la base de conocimientos. Las características de los controladores ASIC utilizados permiten resultados de control adecuados, semejantes a las simulaciones de alto nivel.

5 REFERENCIAS

[1] R.M. Hilloowala and A.M. Sharaf, "A Rule-Based Fuzzy Logic Controller for a PWM Inverter in a Stand Alone Wind Energy Conversion Scheme", *IEEE Trans. on Industry Applications*, Vol. 332, No. 1, Enero/Febrero 1996, pp. 57-65.

[2] P. Guillemin, "Fuzzy Logic Applied to Motor Control", *IEEE Trans. on Industry Applications*, Vol. 332, No. 1, Enero/Febrero 1996, pp. 51-56.

[3] H. H. Li, N. Godfrey y Y. Ji, "A Fuzzy Logic Beam-and Ball Controller Prototype", *IEEE Micro*, Diciembre 1995, pp. 1-7.

[4] L. Wang y R. Langari, "Complex Systems Modeling via Fuzzy Logic", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 26, No. 1, Febrero1996, pp. 100-106.

[5] A. Costa, A. de Gloria, P. Farabvoschi, A. Pagni y G. Rizzotto, "Hardware Solutions for Fuzzy Control", *Proceedings of the IEEE*, Vol. 83, No. 3, Marzo 1995, pp. 422-434.

[6] K. Nakamura, N. Sakashita, Y. Nitta, K. Shimomura y T. Tokuda, "Fuzzy Inference and Fuzzy Inference Processor", *IEEE Micro*, vol. 13, n. 5, Octubre 1993, pp. 37-48.

[7] C.J. Jiménez, S. Sánchez-Solano y A. Barriga, "Hardware Implementation of a General Purpose Fuzzy Controller", *Proc. of the VI IFSA Word Congress* 1995, Sao Paulo, Vol. 2, pp. 185-188.

[8] :S. Sánchez-Solano, A. Barriga, C.J. Jiménez y J.L. Huertas, "Design and Application of Digital Fuzzy Controllers", *Proc. of the Sixth IEEE International Conference on Fuzzy Systems*, 1997, Barcelona, Vol. 2, pp. 869-874.

UNA ARQUITECTURA DE AGENTES DIFUSOS PARA ROBOTS AUTÓNOMOS MÓVILES*

Antonio Gómez Skarmeta
Dep. de Informática, Inteligencia Artificial, y Electrónica
Universidad de Murcia
skarmeta@dif.um.es

Humberto Martínez Barberá
Dep. de Informática, Inteligencia Artificial, y Electrónica
Universidad de Murcia
humberto@fcu.um.es

Pedro García López
Dep. de Informática, Inteligencia Artificial, y Electrónica
Universidad de Murcia
pedro@aries.dif.um.es

Resumen

En este artículo se describe el trabajo que se está realizando con robots autónomos móviles. Se muestra la plataforma hardaware desarrollada, y se plantea una arquitectura de agentes con los siguientes objetivos: flexibilidad, sencillez, y tolerancia a fallos. Para conseguir esto se hace un uso extenso de la lógica difusa en todos los niveles: definición de los agentes y fusión de los resultados. Además, esta arquitectura está bastante indicada para el uso de técnicas de aprendizaje.

Palabras Clave: Navegación Reactiva, Control Difuso, Agentes Inteligentes, Robots Autónomos

1 INTRODUCCIÓN

La operación de un robot autónomo móvil en un entorno no estructurado, tal como se presenta en el mundo real, requiere tener en cuenta múltiples detalles. Principalmente, el controlador debe ser capaz de operar bajo condiciones de imprecisión e incertidumbre. Por ejemplo, el conocimiento a priori del entorno, en general, es incompleto, incierto, y aproximado. La información perceptual adquirida típicamente es también incompleta y afectada por el ruido. Además, la ejecución de un comando de control no es completamente fiable puesto que la dinámica del mundo real es compleja e impredecible. Para tener en cuenta todas estas dificultades, el controlador debe responder reactivamente a eventos no previstos tan pronto como sean percibidos.

El uso de agentes inteligentes cooperativos, que implementan el control de navegación por medio de lógica difusa, es altamente recomendado debido a la naturaleza del comportamiento que debe observar un robot autónomo. Por un lado, la literatura en robots autónomos contiene numerosos ejemplos de aproximaciones que buscan descomponer la función de control global en otras más pequeñas llamadas comportamientos [6]. Cada comportamiento es responsable de producir ciertas acciones (típicamente acciones de percepción y acción) de forma que se cumpla o mantenga un determinado objetivo como apartarse de los obstáculos. Esta interpretación encaja con la noción de agentes. Los agentes son entidades autónomas capaces de desarrollar tareas específicas por ellas mismas, o bien a través de la cooperación con otros agentes. Si además queremos que sean capaces de operar en ambientes de imprecisión e incertidumbre, la lógica difusa ofrece un marco bastante solvente para ello [3].

Por otro lado, la inteligencia es el grado de razonamiento y comportamiento aprendido: la habilidad del agente para aceptar los objetivos dados por el usuario y llevar a cabo la tarea encomendada. Como mínimo, debe haber algún tipo de órdenes, tal vez en forma de reglas, con un motor de inferencia u otro tipo de mecanismo de razonamiento para actuar sobre ellas. También aquí es posible aplicar lógica difusa para tratar la imprecisión e incertidumbre [14]. Niveles de inteligencia superiores incluyen un modelo de usuario o alguna forma de conocimiento y razonamiento acerca de lo que el usuario quiere que se haga, y planificación de los medios para alcanzar dicho objetivo.

En los siguientes apartados se describirá la arquitectura hardware sobre la que se trabajará, la arquitectura de agentes que se ha definido, así como una descripción de la forma de trabajo de los mismos. Por último se analizarán el grado de desarrollo del sistema y los resultados obtenidos.

2 DESCRIPCIÓN DE LA PLATAFORMA

La plataforma sobre la que se está desarrollando y probando la arquitectura propuesta es un robot autónomo móvil holonómico, de construcción propia (parte del actual desarrollo fue probado en otro robot más

* Trabajo parcialmente financiado por el proyecto CICYT TIC97-1343-C002-02

simple y no holonómico [10]). La planta del robot es cuadrada y mide 40x40 cm, con una altura de 55 cm. Presenta un sistema de dirección diferencial, con dos ruedas motoras grandes y dos pequeñas de giro libre para proporcionar estabilidad en todos los rumbos. El peso total del conjunto es de unos 10 Kg.

Para el procesamiento se cuenta con dos CPUs: una placa basada en un 486DX2 con 10 Mb de memoria y 500 Mb de disco corriendo el sistema operativo Linux; y una placa HandyBoard basada en el microcontrolador MC68HC11 con 32 Kb de memoria estática y diversos puertos de entrada/salida con conversores A/D programada en C y en ensamblador. La primera es la encargada de correr todo el software de planificación y control, mientras que la segunda actúa como esclava para procesamiento de los sensores y ejecución de comandos sobre los actuadores.

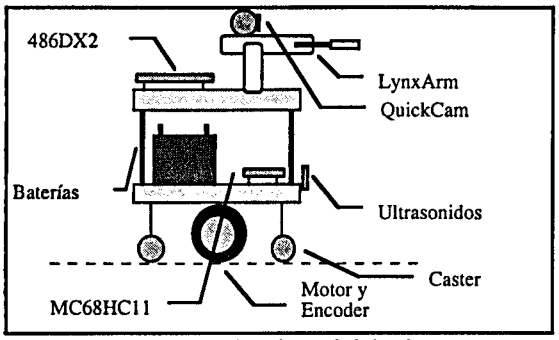


Figura 1: Vista lateral del robot.

El robot (Figura 1) cuenta con encoders ópticos montados sobre las dos ruedas tractoras que son actuadas por sendos motores de contínua, cuatro sensores ultrasónicos Polaroid 6500 trabajando a la misma frecuencia, un anillo de pulsadores para detectar colisiones, tres sensores propioceptivos para detectar el nivel de carga de las baterías, un brazo mecánico LynxArm con 4 grados de libertad y una cámara QuickCam de 256 tonos de grises montada sobre él. Para alimentar el sistema se utilizan tres baterías: una NiCd de 1.2 Ah para los motores, una NiCd de 2 Ah para los ultrasonidos, una de Acido-Pb de 15 Ah para la alimentación del resto del sistema.

Para comunicar los distintos subsistemas (Figura 2) se utilizan dos puertos serie RS232C (486 a 68HC11, y 486 a LynxArm), un puerto paralelo Centronics (486 a QuickCam), y un enlace Ethernet fijo para el desarrollo.

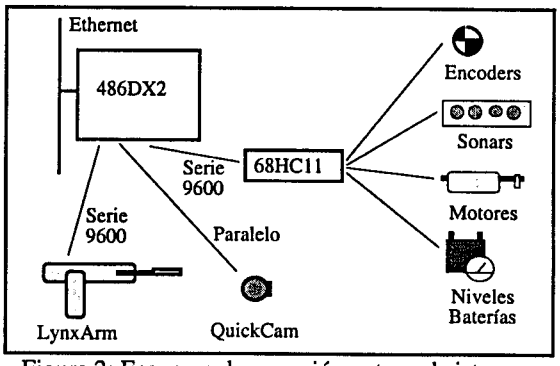


Figura 2: Esquema de conexión entre subsistemas.

3 DESCRIPCIÓN DE LA ARQUITECTURA DE AGENTES

Se ha tomado como paradigma principal, de la arquitectura que se presenta, una arquitectura de pizarra [11][17]. Con este modelo una serie de agentes (o bases de conocimiento) comparten la información disponible sobre el estado del sistema y del entorno. Cada agente leerá de la pizarra la información que sea relevante para él, y tras realizar el correspondiente procesamiento escribirá en la misma un resultado. La arquitectura propuesta se ha diseñado de acuerdo a los siguientes principios:

- uso de mecanismos de encapsulación, aislamiento y control local: cada agente es una entidad semiautónoma e independiente.
- no se hacen asunciones acerca del conocimiento de cada agente o del método de resolución que implemente.
- organización flexible y dinámica.
- adecuación de la arquitectura al uso de técnicas de aprendizaje.

Con estos principios se ha realizado una descomposición de los elementos principales del sistema [1] y se han definido una serie de agentes relacionados mediante una pizarra jerárquica (Figura 3). Se tiene un primer nivel con una serie de agentes que se ejecutan concurrentemente y de forma asíncrona. El número de los mismos variará en función de la dificultad de la tarea que se quiera realizar, y el tipo de sensores y actuadores disponibles en el sistema. En nuestro caso, puesto que el objetivo último es el de realizar tareas del tipo búsqueda, recogida y reparto de objetos en la planta del laboratorio (con diferentes salas). Los agentes que se han identificado son:

- Control Reactivo. Agente encargado de la navegación reactiva.
- Planificación. Agente encargado de la planificación de alto nivel y supervisión del grado de consecución del objetivo global.

- Localización. Agente encargado de crear un modelo del entorno, así como la situación del robot dentro del mismo.
- Control del Vehículo. Agente que se encarga del control y filtrado de los elementos de bajo nivel del sistema: sensores y actuadores.

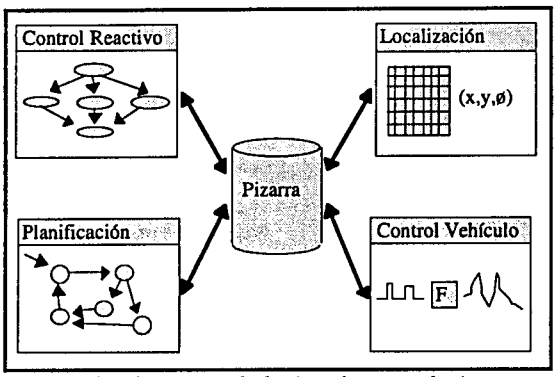


Figura 3: Elementos de la Arquitectura de Agentes

A su vez, cada uno de estos agentes de primer nivel está formado por una serie de agentes de segundo nivel. Para facilitar el desarrollo de los mismos se ha definido un lenguaje de programación de alto nivel (BG), apoyado en la semántica del lenguaje C y basado en COLBERT [13]. En este lenguaje, además de las estructuras de control y elementos tradicionales, se tienen: autómatas finitos deterministas, conjuntos difusos, bases de reglas difusas, un mecanismo de fusión difuso del resultado de los agentes, y librerías de bajo nivel para interactuar con los elementos de la plataforma hardware. De esta forma, basados en la arquitectura descrita, se implementan los agentes de los distintos niveles. El compilador de BG genera directamente un ejecutable que se utiliza como software de control.

3.1 CONTROL REACTIVO

El agente de control reactivo es, a su vez, una pizarra sobre la cual trabajan una serie de agentes de segundo nivel. Estos son los encargados de implementar los distintos comportamientos reactivos. La idea principal que se ha tenido en cuenta es reducir al maximo la complejidad a base de descomponer en comportamientos muy simples y después fusionarlos [8]. La ventaja de este método está relacionada con la escalabilidad y el aprendizaje: mientras que en una arquitectura de subsumisión [6] el control de los agentes es dependiente del comportamiento que implementen [10], con la arquitectura propuesta el control recae en el método de fusión. Esta fusión se realiza a partir de una base de metareglas difusas que operan habilitando/deshabilitando la salida de los distintos agentes, ya sea total o parcialmente [15][2][10]. De esta forma se puede entrenar el sistema con unos pocos comportamientos básicos, e ir añadiendo nuevos sin necesidad de modificar los anteriores: sólo hay que volver a reentrenar.

Cada agente implementa un comportamiento básico reactivo a base de reglas difusas de tipo MISO. Estas reglas se obtienen bien a partir de la experiencia, o bien a través de algún método de aprendizaje, como el aprendizaje evolutivo [12] o el clustering difuso [16]. Una vez que se tienen definidos los agentes, se aprenden las metareglas difusas, que tienen en cuenta los contextos de activación de los distintos agentes [4].

3.2 CONTROL DEL VEHÍCULO

El agente de control del vehículo es, a su vez, una pizarra sobre la cual trabajan una serie de agentes de segundo nivel. Por la función particular de este agente, no se necesita ningún mecanismo de control. Se pueden distinguir aquí dos tipos de agentes: agentes sensores y actuadores. Los primeros se encargan de procesar y filtrar la información que proviene de los sensores, y los segundos de enviar las señales oportunas a los actuadores para que realizen determinados comandos de control. En nuestra plataforma tendremos los siguientes agentes:

- adquisición y filtrado de ultrasonidos.
- chequeo de colisión.
- adquisición y filtrado de la cámara.
- ejecución de vectores de control (v, θ) para la plataforma.
- ejecución de vectores de control (x, y, z) para el brazo.

La navegación del robot depende por completo de la información de los ultrasonidos. Estos se caracterizan por producir una percepción imprecisa del entorno, debido a problemas bien conocidos: crosstalking, ruido, y baja fiabilidad para ángulos de incidencia mayores de 15 grados sobre supercifies pulidas. Debido a esto, un fallo en el subsistema de ultrasonidos puede condicionar el funcionamiento del sistema completo. Para evitar que se traten especialmente estas situaciones por todos los agentes que dependan de esta información, se destinan agentes para recoger la información de los ultrasonidos, procesarla y dar un resultado fiable. Este método reduce la complejidad del sistema y, además, incrementa la fiabilidad del mismo. Para ello se aplica un filtrado difuso [7] a cada uno de los sensores de ultrasonidos. Básicamente, estos filtros constan de tres elementos:

- una memoria de las n-últimas lecturas.
- un bloque de predicción de la siguiente lectura a partir de la memoria (generalmente la media).
- un bloque de inferencia (Figura 4). La entrada es la diferencia entre el valor leído y la predicción. La salida es un factor de ponderación entre ambas. Esto se realiza con un conjunto de tres reglas SISO difusas.

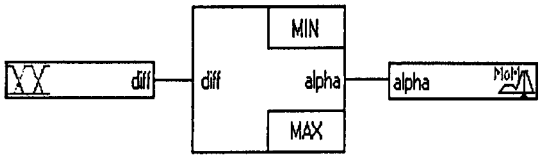


Figura 4: Bloque de inferencia del filtrado difuso.

Con este método se consiguen mejores resultados que con filtros de Kalman simples. Además, cuando se desactivan los filtros se observan muchos movimientos erráticos generalmente producidos por errores de reflexión.

Para ejecutar los vectores de control (v, θ) de la plataforma, hay que tener en cuenta que nuestro robot posee dirección diferencial. Para ello hay que convertir este vector en velocidades de los motores. Para cada motor se tiene un agente encargado de mantener una velocidad constante usando como realimentación la información de los encoders. Esto se realiza con un conjunto típico de cinco reglas SISO difusas.

Las marcas que se desean detectar son discos de un determinado color. Para ello, el agente encargado del proceso de la imagen, aplica filtros comunes a la imagen: filtrado de frecuencias, segmentación, y extracción de contornos. Cuando detecta una marca estima la distancia a partir del tamaño de la misma, y la posición relativa a la plataforma.

3.3 PLANIFICACIÓN

Debido a la simplicidad de las tareas que se van a encomendar al robot, el agente de planificación es bastante sencillo. Actualmente es un autómata finito determinista, cuyas salidas son órdenes del tipo: busca una marca, ve a la posición (x, y), coge un objeto, deja un objeto. Estas órdenes serán recogidas por un agente de control reactivo, que generará, a su vez, los correspondientes objetivos.

3.4 LOCALIZACIÓN

El agente de localización es una pizarra simple, sin mecanismo de control. Estará formado por tres agentes que se encargarán de: elaborar un mapa del entorno, calcular la posición del robot dentro del mismo, y posicionar las marcas.

Para la elaboración del mapa [9] se definen una matriz NxM de celdas, de una determinada anchura en centímetros. Una de estas se toma como origen de coordenadas. Cada celda tendrá un valor que corresponderá al grado de certeza de que sea un obstáculo. Según se vaya moviendo el robot, se irán actualizando las celdas. Esta información puede ser utilizada en el bloque de predicción del filtrado difuso. Sobre el mapa se posicionará el robot por medio de odometría. Para corregir los errores odométricos se usará la posición de las marcas, que será conocida de antemano.

4 CONCLUSIONES

En este momento el sistema está todavía en fase de desarrollo, y, por tanto, no está completo. Sin embargo, se han realizado una serie de pruebas para probar el funcionamiento parcial del sistema. El objetivo es conseguir un *safe-wander* (deambular por la planta del laboratorio, sin colisionar ni quedarse inmóvil). Para ello sólo han sido usados los agentes de Control del Vehículo y de Control Reactivo. En el primer caso se han utilizado los conjuntos de reglas definidos en [7] y [10]. El agente de Control Reactivo, descrito en el lenguaje BG, implementa los siguientes comportamientos:

- *contact-obstacle-detector*: detecta la colisión del robot con un obstáculo y lo detiene.
- *obstacle-avoider*: dirige el robot en dirección opuesta al obstáculo más próximo (Figura 5).
- *align-right*: mantiene el robot a una distancia prudencial de la pared derecha (Figura 6).
- *align-left*: mantiene el robot a una distancia prudencial de la pared derecha (Figura 7).

```
behaviour avoid priority 2.0
{
    fusion turn;
    rules
    {
        if (sonar1 is CLOSE)turn is TR;          // Right
        if (sonar2 is CLOSE)turn is TL;          // Left

        if (left is CLOSE)   turn is TR;         // Right
        if (right is CLOSE)  turn is TL;         // Letf

        if (left is NEAR)    turn is TSR;        // Small right
        if (right is NEAR)   turn is TSL;        // Small left
    }
    if ((sonar1 is DANGER) || (sonar2 is DANGER))
    {
        if (left > right)    turn = -10.0;       // Left
        if (right > left)    turn = 10.0;        // Right
    }
}
```

Figura 5: Comportamiento básico *obstacle-avoider*.

```
behaviour alignR priority 0.5
{
    fusion turn;
    rules
    {
        if (right is FAR)    turn is TSR;        // Small right
        if (right is MED)    turn is TSL;        // Small left
        if (right is NEAR)   turn is TSL;        // Small left
        if (right is CLOSE)  turn is TL;         // Left
    }
}
```

Figura 6: Comportamiento básico *align-right*.

```
behaviour alignL priority 0.5
{
    fusion turn;

    rules
    {
        if (left is FAR)      turn is TSL;      // Small left
        if (left is MED)      turn is TSR;      // Small right
        if (left is NEAR)     turn is TSR;      // Small right
        if (left is CLOSE)    turn is TR;       // Right
    }
}
```

Figura 7: Comportamiento básico *align-left*.

Estos comportamientos básicos reactivos se fusionan teniendo en cuenta una serie de metareglas (Figura 8), cuyos consecuentes son los grados de activación de los comportamientos reactivos básicos. La obtención de las reglas difusas (tanto comportamientos básicos como metareglas) ha sido realizada a base de experiencia previa (ya sea propia como la presente en la literatura). Sin embargo, como actividad prioritaria se está desarrollando un módulo que sea capaz de aprender las metareglas, por medio de refuerzo. Hay que tener en cuenta que, en la actualidad, estamos más interesados en desarrollar y probar el sistema que en conseguir unos buenos conjuntos de reglas.

```
blender
{
    rules
    {
        background (0.1)                avoid is LOW;
        background (0.1)                alignR is LOW;
        background (0.1)                alignL is LOW;
        background (0.1)                stop is LOW;

        if (collision)                  stop is HIGH;

        if ((left is DANGER) || (front is DANGER) || (right is
DANGER))        avoid is HIGH;

        if (left is MED)                alignL is HALFH;
        if (right is MED)               alignR is HALFH;
        if (front is MED)               avoid is HALFL;

        if ((left is FAR) || (front is FAR))        alignL is HIGH;
        if ((right is FAR) || (front is FAR))       alignR is
HIGH;
    }
}
```

Figura 8: Metareglas de fusión.

Para comprobar el funcionamiento del sistema se han realizado dos tipos de pruebas: con el robot real y con un simulador. El resultado ha sido satisfactorio. En las pruebas reales (una de ellas de 20 minutos de duración) el robot fué capaz de estar circulando por el pasillo (de forma circular con ramales, y con tráfico moderado de personas), entrando y saliendo de salas, sin entrar en un bucle infinito y sin colisionar. En el simulador también actuó de igual forma (Figura 9), si bien la planta era un poco más complicada. Aún así, hay que tener en cuenta que, debido a su simplicidad, se pueden preparar situaciones para las que el sistema no esté preparado, y ante las cuales el robot se queda oscilando.

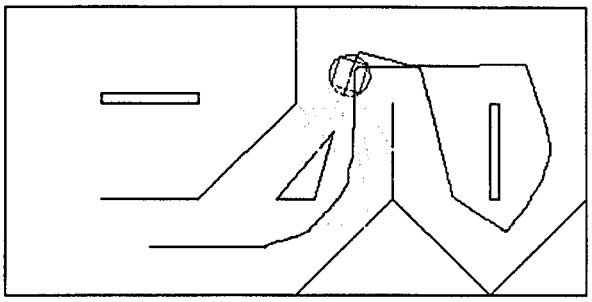


Figura 9: Trayectoria del robot en el simulador.

Teniendo en cuenta el estado actual del desarrollo, se han podido comprobar una serie de puntos:

- los interfaces son simples, lo que permite acelerar la fase de desarrollo, ya que cada agente se desarrolla y valida por separado.
- el sistema es bastante fiable en cuanto a tolerancia a fallos y protección contra ruidos.
- la arquitectura permite integrar distintos métodos de resolución de problemas, así como el uso de técnicas de aprendizaje en en el desarrollo de agentes y fusión de los mismos.

5 REFERENCIAS

[1] D.G. Armstrong, C.D. Crane, "*A Modular, Scalable, Architecture for Intelligent, Autonomous Vehicles*", 3rd IFAC Symposium on Intelligent Autonomous Vehicles, Universidad Carlos III, Madrid, Spain, pp 62-66, 1998

[2] B.C. Arrúe, F. Cuesta, B. Braunstingl, A. Ollero, "*Fuzzy Behaviours Combination to Control a Non Holonomic Mobile Robot using Virtual Perception Memory*", Sixth IEEE Intl. Conf. on Fuzzy Systems (FuzzIEEE'97), Barcelona, Spain, pp 1239-1244, 1997

[3] H.R. Beom, H.S. Cho, "*A Sensor-Based Navigation for a Mobile Robot using Fuzzy Logic and Reinforcement Learning*", IEEE Trans. Syst. Man Cyber., vol 25, no. 3, pp 464-477, 1995

[4] A. Bonarini, F. Basso, "*Learning to Compose Fuzzy Behaviors for Autonomous Agents*", Intl. Journal of Approximate Reasoning, no. 17, pp 409-432, 1997

[5] J. Borenstein, H.R. Everett, L. Feng, "*Navigating Mobile Robots: Systems and Techniques*", A.K. Peters, Wellesley, MA, 1996

[6] R.A. Brooks, "*A robust Layered Control System for a Mobile Robot*", IEEE J. Robotics and Automat., vol. RA-2, no. 1, pp 14-23, 1986

[7] M. Delgado, A.F. Gómez Skarmeta, H. Martínez Barberá, J. Gómez, "*Fuzzy Range Sensor Filtering for Reactive Autonomous Robots*", Fith Intl. Conference on Soft Computing and Information / Intelligent Systems (IIZUKA'98), Fukuoka, Japan, 1998

[8] M. Dorigo, M. Colombetti, "*Robot Shaping: an Experiment in Behavior Engineering*", The MIT Press, Cambridge, MA, 1998

[9] J. Gásos, A. Martín, "*Mobile Robot Localization using Fuzzy Maps*", in Lectures Notes in Artificial Intelligence (A. Ralescu and T. Martín Eds.), Springer-Verlag, 1996

[10] A.F. Gómez Skarmeta, H. Martínez Barberá, "*Fuzzy Logic Based Intelligent Agents for Reactive Navigation in Autonomous Systems*", Fith International Conference on Fuzzy Theory and Technology, Raleigh, USA, pp 125-131, 1997

[11] B.A. Hayes-Roth, "*Blackboard for Control*", Artificial Intelligence, vol. 26, pp 251-324, 1985

[12] F. Hoffmann, G. Pfister, "*Evolutionary Learning of Fuzzy Control Rule Base for an Autonomous Vehicle*", Proc. Sixt Int. Conf. Information Procesing and Management of Uncertainty in Knowledge-Based Systems (IPMU'96), Granada, Spain, pp 1235-1240, 1996

[13] K. Konolige, "*COLBERT: a Language for Reactive Control in Saphira*", German Conference on Artificial Intelligence, Freiburg, 1997

[14] A. Saffiotti, et al., "*Robust Execution of Robot Plans using Fuzzy Logic*", in Fuzzy Logic in Artificial Intelligence: IJCAI'93 Workshop. Lectures Notes in Artificial Intelligence 847 (Ralescu A. Ed.). Springer-Velag, pp 24-37, 1994

[15] A. Saffiotti, "*Fuzzy Logic in Autonomous Robotics: behavior coordination*", Sixth IEEE Intl. Conference on Fuzzy Systems (FuzzIEEE'97), Barcelona, Spain, pp 573-578, 1997

[16] J.R. Velasco, F.E. Ventero, "*Some Applications of Fuzzy Clustering to Fuzzy Control Systems*", Third International Conference of Fuzzy Theory and Technology, Durham, USA, 1994

[17] S. Vranes, M. Stanojevic, "*Integrating Multiple Paradigms within the Balckboard Architecture*", IEEE Transactions on Software Engineering, vol. 21, no. 3, pp 244-262, 1995

CONTROLADOR FUZZY PARA LA GESTIÓN DE TRÁFICO EN UN CRUCE AISLADO

Jon Zabala
Dpto. de Inteligencia Artificial
IKERLAN
Centro de Investigaciones Tecnológicas
E-Mail: jzabala@ikerlan.es

Gotzon Arregi
Dpto. de Inteligencia Artificial
IKERLAN
Centro de Investigaciones Tecnológicas
E-Mail: garregi@ikerlan.es

Juan Pedro Uribe
Dpto. de Inteligencia Artificial
IKERLAN
Centro de Investigaciones Tecnológicas
E-Mail: jpuribe@ikerlan.es

Resumen

En la presente comunicación se describe el diseño y desarrollo de un sistema Fuzzy para la gestión de los semáforos en un cruce aislado, utilizando información de los sensores de presencia instalados en cada carril del cruce. A partir del preprocesamiento de la información, se construyen las entradas del sistema en las que subyace información de carácter global (proactivo) e instantáneo (reactivo), que se combinan en una base de reglas simple en la que se decide el cambio de estado de la fase en verde. El sistema tiene una estructura modular donde las unidades son la fase y el carril, lo cual hace posible su generalización a diferentes cruces de forma normalizada.

Por último, se presentan los resultados comparativos del sistema propuesto con dos sistemas convencionales, uno con tiempos fijos y otro de carácter reactivo para un cruce genérico, y se observan las mejoras obtenidas tanto en los tiempos medios de espera como de colas.

Palabras clave: Sistema Fuzzy, control reactivo, control proactivo, Vehicle Actuated, Fixed Time.

1 INTRODUCCIÓN

La gestión eficiente de las distintas infraestructuras viarias es una prioridad desde hace ya bastantes años. Se ha observado que año tras año el parque automovilístico crece y que el consiguiente incremento de la capacidad de canalización eficiente de dicho tráfico topa con un importante obstáculo, la imposibilidad física de aumentar la capacidad de canalización a través de la construcción de nuevas vías en algunos casos, y el alto coste que conlleva en otros.

Por otro lado, diferentes estudios han demostrado que la utilización de las infraestructuras actualmente existentes está lejos de ser óptima en la mayoría de los casos, y que por lo tanto existe una oportunidad de mejora a través de una gestión más eficiente, lo que conlleva una inversión mucho más reducida.

El presente trabajo persigue como objetivo principal la exploración de las oportunidades que las nuevas tecnologías como Redes Neuronales, Lógica Fuzzy o Algoritmos Genéticos ofrecen para la mejora de la gestión del tráfico urbano de vehículos. Para ello, el proyecto centra su atención preferente en la regulación o control de tráfico mediante técnicas avanzadas en un elemento básico: "Cruce aislado regulado por semáforos", contemplando su extensión a áreas de regulación más amplias. Este control puede ser de tipo reactivo o proactivo (predictivo).

Para evaluar el comportamiento del control fuzzy desarrollado se ha construido una serie de controles clásicos. Estos controles clásicos son el Vehicle Actuated (VA) y el Fixed-Time (FT).

El control VA que se ha construido dispone de los mismos medios físicos que el control fuzzy implementado, es decir, la misma sensorización. El control VA desarrollado varía respecto del control fuzzy reactivo únicamente en la función de evaluación de cambio. El plan aplicado es el mismo que en el control fuzzy reactivo. Los tiempos máximos y mínimos utilizados para las fases variables tanto en el control fuzzy reactivo como en el VA han sido los mismos.

La función de cambio de fase del VA consiste en lo siguiente. Si durante el transcurso de una fase variable (sensorizada) se tiene que llega un vehículo, se alarga la fase en un tiempo de prolongación de 2 segundos, y si no se da paso a la siguiente fase. Utilizando el citado tiempo de prolongación se ha obtenido un buen control reactivo.

Por otra parte, se ha considerado un control FT (las fases son de duración fija) optimizado para la hora crítica, ya que de otra manera hubiera resultado ser un control insuficiente para dicho intervalo.

2 CONTROL FUZZY REACTIVO

Los controladores reactivos son aquellos que, a partir de la información procesada de la medida de los detectores, toman la decisión de dar por terminada o prolongar la fase que actualmente está en verde.

El controlador Vehicle Actuated (VA) es un ejemplo de controlador reactivo clásico. Pero el controlador VA únicamente tiene en cuenta información instantánea del sistema, dejando a un lado información global de gran utilidad que se puede obtener a partir de las señales de los detectores, como son las estimaciones de flujos de entrada y colas de cada uno de los carriles de aproximación.

El control fuzzy reactivo construido dispone de los mismos medios que el VA (un detector por carril), pero hace uso tanto de la información global como de la instantánea recogida para la evaluación del cambio de fase.

2.1. ESTRATEGIA DE CONTROL

Hay una amplia bibliografía que trata los controles fuzzy reactivos, pero en la mayoría de los artículos se realizan controles sobre cruces específicos [1,2,3,4]. En estos casos un mismo controlador no es aplicable a diferentes cruces.

En un principio, en el sistema Fuzzy a realizar se planteó que el control debía ser independiente del cruce. Es decir, el control debía ser el mismo para cualquier cruce y, luego, para cada cruce particular, se tendría una serie de parámetros en los que se describiría el plan a seguir, de qué carriles se dispondría, la longitud de las colas permisibles en cada carril etc. relacionados con la topología del cruce y medibles de forma estática.

En un sistema Fuzzy Reactivo típico, se tiene por entrada la información del estado de todos los carriles del cruce y por salida la decisión de cambio de fase. Si se quiere hacer que el control sea independiente del cruce, las reglas de control deben ser comunes para cualquier fase. Sin embargo, de una fase a otra se tiene diferente número de carriles en la fase de verde, por lo que a la hora de construir las reglas Fuzzy se debe tener en cuenta que el número de carriles de la fase no debe verse reflejado en el número de antecedentes de las reglas, pero sin embargo se debe reflejar la información de todos ellos. Se podría solventar esta dificultad considerando únicamente el carril más problemático de la fase en verde, pero para decidir cuál es el carril más problemático se requeriría de otro sistema Fuzzy.

Para solventar esta dificultad, se decidió variar el esquema del sistema Fuzzy reactivo típico. En vez de hacerse una llamada en cada segundo de simulación para decidir si se termina la fase o no, se realizan llamadas para cada uno de los carriles en la fase en verde, teniéndose como salida si la fase se puede dar por terminada o no en lo que respecta a un carril dado. Cuando todos los carriles la dan por terminada, se termina la fase. De esta manera se elimina la necesidad de construir un sistema añadido para la elección del carril más problemático, que será el último que dé la fase por terminada. De esta forma el sistema resulta totalmente modular y generalizable en cuanto al número de fases y carriles asociados.

2.2. PARÁMETROS DE CONTROL

Por un lado, se requiere información referente al plan de control, que consiste en la secuencia de fases e interfases a seguir, tiempos máximos y mínimos de las fases y la existencia o no de detectores asociados a cada fase. Por otro lado, se tiene información relacionada con los diferentes carriles sensorizados, como son las longitudes de carril, los flujos de saturación del mismo y las velocidades a las que circulan los vehículos.

2.2.1 Estructura del plan

El plan consiste en la secuencia de fases e interfases de los semáforos. Las fases pueden ser de duración variable (con un tiempo máximo y uno mínimo), o de duración fija (como ocurre en el control clásico fixed-time). Las interfases son cortos espacios de tiempo de duración fija en los que los semáforos permanecen invariables, cuya finalidad es enlazar las diferentes fases cumpliéndose ciertas normas de seguridad.

El control está preparado para funcionar en cruces mediante un plan de secuencia fija de fases.

2.2.2 Parámetros del carril sensorizado

Los parámetros que requiere el control fuzzy reactivo para la construcción de las entradas hacen referencia a las características de los carriles sensorizados. Estas características son la longitud del carril, el flujo de saturación, las velocidades medias de los vehículos que circulan por el carril, así como los flujos de entrada para la inicialización.

2.3. ENTRADAS AL SISTEMA FUZZY

A continuación se explica un total de cinco entradas al sistema fuzzy reactivo que se han implementado durante el presente proyecto.

2.3.1 Tiempo de Verde Estimado: fzTDif

A partir de los flujos de los carriles, se puede hacer una buena estimación del tiempo de verde necesario para dar salida a los coches que llegan a cada uno de los carriles.

Partiendo del supuesto de que en el ciclo anterior no quedaron coches acumulados tras la fase de verde y suponiendo que la densidad de salida durante la fase en verde será igual al flujo de saturación del carril, se tiene la siguiente ecuación:

$$(t_{RojoAnterior} + t_{VerdeEstimado}).F_{Entrada} = t_{VerdeEstimado}.F_{Saturacion} \quad [1]$$

Es decir, los coches que entran deben salir. De donde se tiene lo siguiente:

$$t_{VerdeEstimado} = \frac{t_{RojoAnterior}}{\frac{F_{Saturacion}}{F_{Entrada}} - 1} \quad [2]$$

Este tiempo se calcula al principio de la fase, para cada uno de los carriles.

Un carril dará por terminada la fase cuando el tiempo de la fase esté en un valor en torno a esta referencia. Así, se evitarán variaciones fuertes en el comportamiento del control, limitándose la reactividad debida a medidas instantáneas, que no suelen ser muy correctas.

Para llevar a cabo esta idea se define la variable Fuzzy de entrada fzTDif de la forma siguiente.

$$fzTDif = t_{VerdeEstimado} - t_{VerdeTranscurrido} \quad [3]$$

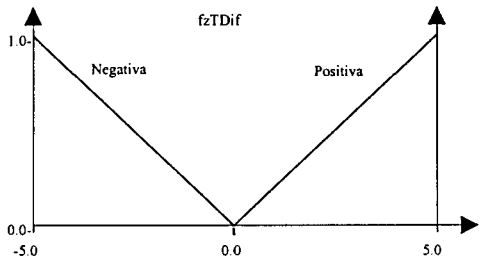


Figura 1

2.3.2 Longitudes de cola estimadas: fzLRelMax, fzElMasLleno

Sea un carril de longitud L. Se pretende que el tamaño de las colas en el carril no exceda de esa longitud L. Para ello, al principio de la fase que le corresponde al carril se estimará el tamaño de la cola inicial, y a partir de ella se calculará hasta dónde llegará esa cola, antes de que todos los coches que la forman se pongan en movimiento.

Si suponemos que en el final de la fase anterior no quedaban coches acumulados en la cola, a partir del flujo de entrada estimado calculamos la longitud inicial de la cola de la forma siguiente.

$$L_0 = F_{Entrada} . T_{Rojo} . l_{Media} \quad [4]$$

Sea T_{Rojo} el tiempo que lleva la fase en rojo, l_{Media} la longitud media de separación entre coches parados, y $F_{Entrada}$ el flujo de entrada.

Cuando se pone la fase en verde, la cola comenzará a vaciarse. Pero, mientras el último coche no arranque, los coches que lleguen seguirán acumulándose, con lo que la cola seguirá creciendo hasta un momento en el que todos los vehículos estén en movimiento. Queremos calcular por lo tanto la distancia a la que se sitúa el último coche parado.

Para ello se tiene que calcular la velocidad con la que crece la cola, $V_{LLenado}$, y la velocidad con la que se propaga el arranque de los coches, $V_{Arranque}$, que dependen de la velocidad, v, con la que circulan los coches por el carril, además del flujo de saturación y el flujo de entrada.

Se tiene que la velocidad de llenado es la siguiente:

$$V_{LLenado} = \frac{F_{Entrada} . l_{Media}}{1 - \frac{F_{Entrada} . l_{Media}}{v}} \quad [5]$$

Por otra parte, se tiene que la velocidad de propagación del arranque es la siguiente.

$$V_{Arranque} = \left(\frac{1}{F_{Saturacion} . l_{Media}} - \frac{1}{v} \right)^{-1} \quad [6]$$

Cuando se empieza a mover el último vehículo se tiene un tiempo t en el que se cumple la ecuación:

$$L_0 + V_{LLenado} . t = V_{Arranque} . t \quad [7]$$

De donde se obtiene el instante en el que se produce la longitud máxima de cola, y la propia longitud.

$$t = \frac{L_0}{V_{Arranque} - V_{LLenado}} \quad [8]$$

$$L_{Maximo} = L_0 + V_{Llenado} . t = \frac{L_0}{1 - \frac{V_{LLenado}}{V_{Arranque}}} \quad [9]$$

Cuanto mayor es la longitud del carril, mayor se puede permitir que sea esta cantidad. Es decir, conviene una medida relativa a la longitud del carril. Esto da una medida de lo lleno que está el carril.

$$L_{Rel} = \frac{L_{Maximo}}{L_{Carril}} \quad [10]$$

Se pretende que su valor no sobrepase la unidad, ya que esto significaría que hay algún carril en el que se generan colas más largas que las deseadas.

Al llamar al sistema Fuzzy, se construyen dos entradas basadas en las medidas de las longitudes relativas de los distintos carriles. Por un lado, se tiene el grado de llenado máximo fzLRelMax, que se obtiene teniendo en cuenta el máximo entre los valores de LRel de cada uno de los carriles. Por otra parte, una medida de lo lleno que está el carril en comparación con el más lleno. Esta medida la denominamos fzElMasLleno.

$$fzLRelMax = Maximo(L_{Rel}) \quad [11]$$

$$fzElMasLleno = \frac{L_{Rel}}{fzLRelMax} \quad [12]$$

La entrada fzLRelMax indica en qué medida se puede estar teniendo problemas debidos a las colas.

fzElMasLleno nos dice cómo de lleno está respecto del más lleno. Toma valores en el rango (0,1). Cuando el carril que está más lleno es el que llama al sistema Fuzzy, se tiene que es uno. Cuanto más vacío esté, más cercano estará a cero.

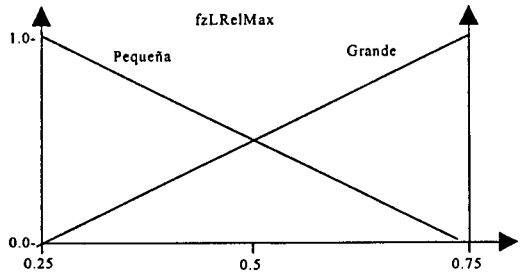


Figura 2

A continuación se muestran las funciones de pertenencia de los dos adjetivos de entrada, Pequeña y Grande, que se consideran para la fuzzyficación de la variable de entrada fzElMasLleno.

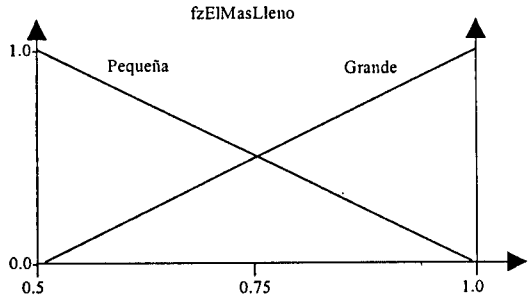


Figura 3

2.3.3 Densidades: fzDensRel, fz_dDensidadActual_dt

Las entradas al sistema Fuzzy vistas hasta ahora han sido creadas a partir de información global del sistema, ya que utilizaban únicamente la información disponible al principio de la fase. La reactividad vendrá dada por las entradas Fuzzy generadas a partir de la información obtenida por los detectores en la fase actual de verde.

A partir del filtrado de la señal del carril se obtienen dos medidas. Por una parte, la densidad actual o instantánea en el carril (número de coches por unidad de tiempo), y, por otra, la densidad media del carril durante la fase.

d_{Actual} = número de coches por unidad de tiempo que sobrepasan el detector. [13]

d_{Media} = número de coches que han atravesado el detector dividido entre el tiempo de fase transcurrido [14]

Cuando comienza la fase de verde de un carril dado, la densidad actual (instantánea) y la densidad media comienzan a crecer, yendo la densidad actual por encima de la densidad media, y según transcurre el tiempo se va aproximando lentamente a la densidad actual. Cuando dejan de pasar vehículos, o dejan de hacerlo en la medida que lo hacían al principio, la densidad actual decrece. Si pasa a ser inferior a la densidad media, ésta también empezará a decrecer. En esta situación no tiene sentido prolongar la fase, ya que la oleada de descarga ha pasado, por lo que tendrá que producirse el cambio.

Se construye una entrada Fuzzy, fzDensRel, a partir de las dos densidades del carril que llaman al sistema. Esta entrada dirá lo grande que es la densidad actual en relación con la densidad media.

$$fzDens\,\mathrm{Re}l = \frac{d_{Actual}}{d_{Media}} \qquad [15]$$

Por otra parte, se construye una variable que nos da la tendencia de la densidad instantánea o actual:

$$fz\_dDensidadActual\_dt = d_{Actual}(t) - d_{Actual}(t-1) \qquad [16]$$

Es decir, se mide la tendencia de la densidad actual restando los valores de la densidad filtrada en el instante actual, y el anterior

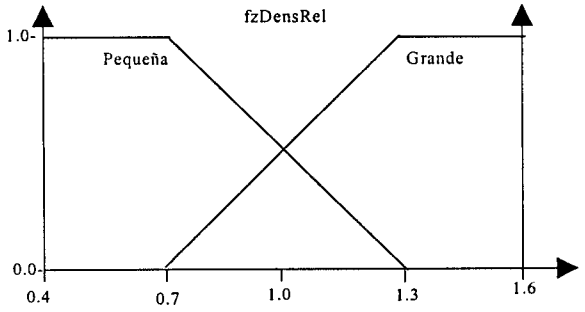


Figura 4

Por otra parte, en la siguiente gráfica se muestran las funciones de pertenencia de los dos adjetivos de entrada Positiva y Negativa que se consideran para la fuzzyficación de la variable de entrada fz_dDensidadActual_dt.

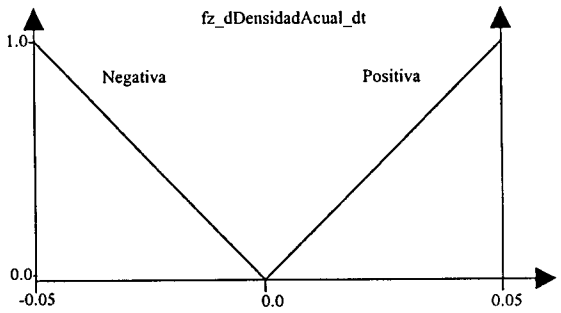


Figura 5

2.4. REGLAS FUZZY

Combinando las cinco entradas al sistema fuzzy reactivo (fzTDif, fzLRelMax, fzElMasLleno, fzDensRel, fz_dDensActual_dt) en las diferentes reglas de control se provoca la activación de los adjetivos de salida en distinto grado. Luego se combinarán estos adjetivos de salida para formar la salida del control, que es la decisión de dar por terminada o no la fase en lo que respecta al carril que llama al sistema fuzzy.

El control que se ha llevado a cabo sólo consta de seis reglas de fácil comprensión. Cada regla activa un adjetivo de salida a favor o en contra del cambio, dependiendo del caso.

Estas seis reglas se agrupan en tres pares. El primer par de reglas tiene en cuenta la información instantánea del detector correspondiente al carril. Es ésta la parte más reactiva del sistema. El segundo par de reglas tiene en cuenta la información global del sistema. Este par de reglas tiene en cuenta el tiempo de verde que se estima va a requerir el carril. El tercer par de reglas, en el que se tienen tanto información global como instantánea, tiende a mejorar el comportamiento del control en cuanto a colas se refiere.

Las primeras reglas, asociadas a la densidad, son:

SI fz_dDensidadActual_dt ES Negativa AND fzDensidadRel ES Pequeña

ENTONCES fzCambioFase SiDens;

SI fz_dDensidadActual_dt ES Positiva OR fzDensidadRel ES Grande

ENTONCES fzCambioFase NoDens;

El segundo par de reglas dice lo siguiente:

SI fzTDif ES Negativa

ENTONCES fzCambioFase NoT;

SI fzTDif ES Positiva

ENTONCES fzCambioFase SiT;

Por último, el tercer grupo de reglas (relacionadas con las colas) dice lo siguiente:

SI fzLRelMax ES Grande AND fzElMasLleno ES NO Grande

ENTONCES fzCambioFase SiCola;

SI fzLRelMax ES Grande AND fzElMasLleno ES Grande AND fzDensidadRel ES Grande

ENTONCES fzCambioFase NoCola;

2.5. SALIDA FUZZY

Como ya se ha visto en las reglas, tenemos por un lado adjetivos de salida en favor del cambio (SiDens, SiT, SiCola), y, por el otro, adjetivos que se oponen al cambio (NoDens, NoT, NoCola). Los adjetivos de salida se combinarán, teniendo en cuenta el grado de activación de las reglas correspondientes, para dar lugar a la salida del sistema en la defuzzyficación.

Se ha optado por utilizar la defuzzyficación vía centroide, utilizando adjetivos de salida tipo Singleton, que son escalados vía producto, y se combinan vía suma. En estas condiciones, los adjetivos de salida se pueden considerar como un conjunto de masas con peso igual al grado de activación de la regla correspondiente, ya que a cada adjetivo le corresponde una sola regla. La posición de los adjetivos definirá la importancia relativa y carácter de éstos. La defuzzyficación en estas circunstancias resulta sencilla, ya que consiste en obtener el centro de masas de un conjunto de masas puntuales. Esta simplicidad en la inferencia facilita el análisis y ajuste de los adjetivos de salida. En resumen, un TSK de orden 0.

Dado el buen comportamiento del control Vehicle Actuated (VA) en cruces aislados para situaciones no próximas a la saturación, se ha considerado oportuno dar mucha relevancia a la reactividad, por lo que se le ha dado a las reglas correspondientes a la parte reactiva un mayor peso, colocando los adjetivos correspondientes en los valores máximos. Es decir, SiDens se sitúa en el valor 1.0, y NoDens en el valor -1.0.

Para un buen comportamiento en situaciones de mucho tráfico, en las que se forman colas excesivas, se ha considerado oportuno dar un valor a los adjetivos correspondientes a las reglas de las colas comparables a los correspondientes a las reglas de densidades. Se ha situado el adjetivo SiCola en el valor 0.75 y NoCola a -0.75.

Finalmente, a los adjetivos correspondientes a las reglas de tiempos se les ha dado un valor de 0.5 para SiT, y -0.5 para NoT, inferiores a los de las otras reglas ya que se consideran menos prioritarias.

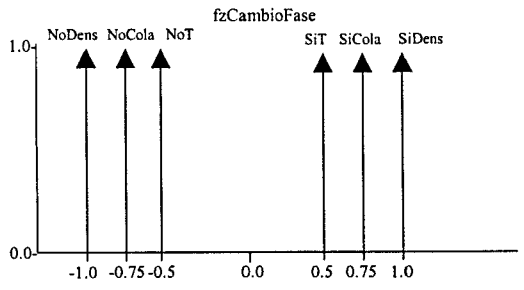


Figura 6

3 RESULTADOS

A continuación se comparan los resultados obtenidos mediante el controlador fuzzy reactivo con los obtenidos utilizando el control Vehicle Actuated (VA) y el control

Fixed-Time (FT), teniendo en cuenta, por un lado, las longitudes de las colas y considerando, por otro, los retardos de los vehículos.

3.1. CRUCE TEÓRICO

El cruce teórico consta de cuatro aproximaciones: Norte, Sur, Oeste y Este. Cada una de ellas tiene tres carriles. El izquierdo para girar a la izquierda, el central para seguir de frente y el derecho para seguir de frente o girar a la derecha.

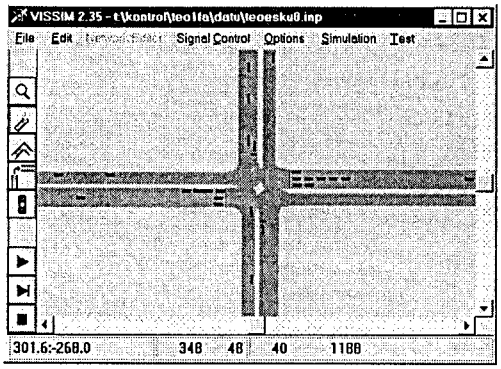


Figura 7

El plan que es aplicado en el cruce consta de cuatro fases: la primera permite el tránsito de los vehículos correspondientes a los carriles centrales y derechos de Norte y Sur; la segunda permite el giro a izquierda de los vehículos que provienen de Este y Oeste; la tercera permite el tránsito de los vehículos correspondientes a los carriles centrales y derechos de Este y Oeste; finalmente, la cuarta fase permite el giro a izquierda de los vehículos que provienen de Este y Oeste.

3.1.1 Controles implementados

Para evaluar el comportamiento del control fuzzy desarrollado se ha construido una serie de controles clásicos. Estos controles clásicos son el Vehicle Actuated (VA) y el Fixed-Time (FT).

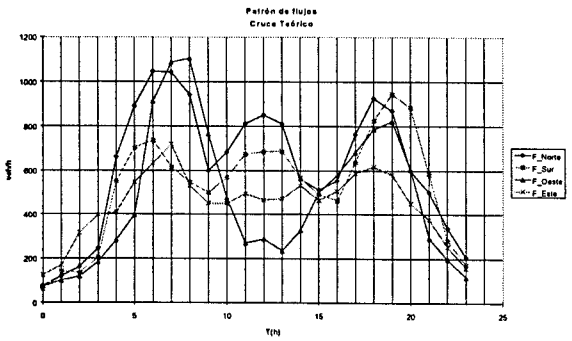


Figura 8

3.1.2 Tiempos de espera

A continuación se muestra la evolución de los retardos durante el día propuesto para el control fuzzy reactivo, el control VA y el FT propuestos.

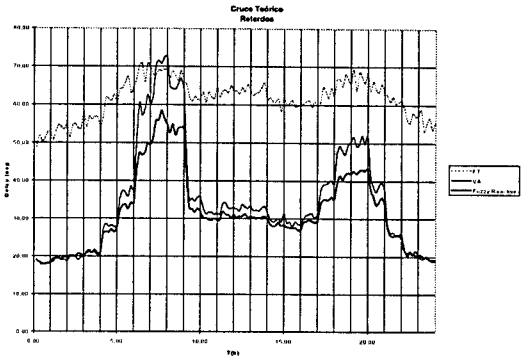


Figura 9

Se observa que el sistema fuzzy reactivo resulta ser mejor que el VA en cuanto a retardos se refiere, sobre todo en las horas de más tráfico. El VA tiene un buen comportamiento, que es difícilmente superable en las zonas de poco flujo, pero tiende a alargar en exceso las fases en las zonas de mucho tráfico debido a su filosofía, que consiste en prolongar las fases mientras se detecten vehículos.

Por su parte, el fixed-time construido para funcionar correctamente en la hora punta tiene un tiempo de ciclo largo, lo que supone que el tiempo de espera medio de los vehículos sea grande durante todo el día, independientemente del tráfico que haya.

El siguiente cuadro muestra los tiempos de espera medios promediados para todo el día, así como los tiempos de espera (medios) máximos que se han producido en la hora punta, para cada uno de los controles en el caso de test del cruce teórico.

Tabla 1

	FT	VA	Fuzzy Reactivo
Tiempo de Espera Medio del día	63.49	39.88	35.24
Tiempo de Espera Máximo	70.72	72.68	58.47

El FT funciona mucho peor a lo largo de todo el día que los controles sensorizados. El control fuzzy reactivo, por su parte, mejora considerablemente los tiempos del VA, tanto en lo que se refiere a la hora crítica como al promedio diario.

Las mejoras que trae el control fuzzy reactivo respecto al VA dependen en gran medida del patrón de flujos de entrada, además de depender también de la topología y

las limitaciones físicas del cruce. Con otros patrones de flujos se obtendrían mejoras diferentes. Para un patrón de flujos en el que se tiene mucho tráfico durante todo el día, se pronunciaría aún más la mejora del fuzzy reactivo respecto del VA. Por el contrario, si se utilizara un patrón de flujos en el que se tiene menos tráfico a lo largo del día, la mejora respecto al fixed-time aumentaría, mientras que la mejora respecto al VA se reduciría considerablemente, ya que el VA es un controlador ideal para estas condiciones.

3.1.3 Colas

Para evaluar las colas, se ha considerado oportuno hacer un análisis de las colas máximas que se producen. Se considera que, si algún carril tiene una cola que excede su longitud permitida, el control no se comporta adecuadamente. Por ello, se han tomado las colas máximas producidas en cada aproximación a intervalos de diez minutos y se han promediado entre las diez pruebas realizadas. Lo preocupante no es qué longitud tienen las colas, sino cómo de largas son respecto a la longitud del carril correspondiente. Por ello se dividen las colas por la longitud del carril, obteniéndose la longitud de colas normalizada en cada aproximación. De todas las longitudes normalizadas, la más grande es la más preocupante, ya que está más cerca de sobrepasar la longitud establecida del carril, trayendo probablemente un perjuicio de gran magnitud en el punto que, por alguna razón, se ha considerado principio de carril.

A continuación se muestra la evolución de las colas máximas producidas durante el día propuesto para el control fuzzy reactivo, el control VA y el FT propuestos.

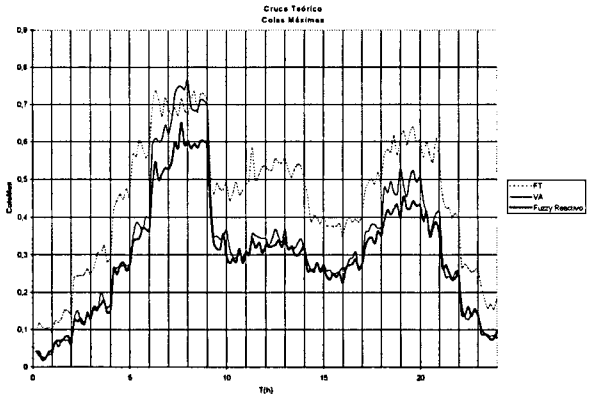


Figura 10

Se observa que, tanto en la zona problemática de la mañana como en la zona problemática de la tarde, las longitudes de las colas que se producen son inferiores en el control fuzzy reactivo que en el control VA, no habiendo apenas diferencia en el resto del patrón diario.

Las colas producidas por el FT, por su parte, son más largas que las que se producen en los controles sensorizados.

En este caso de test se puede apreciar que, mientras que con el control VA se producen colas que suponen el 75% de la longitud del carril, en el control fuzzy reactivo se producen colas en torno al 60%, lo que supone una mejora importante.

4 CONCLUSIONES

En este trabajo se ha desarrollado un controlador fuzzy modular que puede ser aplicado a cualquier cruce aislado. Este control fuzzy requiere de un solo detector por carril de aproximación, abaratándose así considerablemente el coste en una hipotética implantación en cruces reales.

Este controlador se ha comparado con controladores clásicos como son el Fixed-Time (FT), controlador de tiempos fijo, y el Vehicle Actuated (VA), que requiere de los mismos medios físicos (detectores) que el control fuzzy implementado.

Este control hace uso de información global (recogida en ciclos anteriores) y de la información instantánea (que se tiene en las fases de verde), a fin de determinar cuándo se debe dar una fase por terminada. Estima los flujos que se tienen en cada uno de los carriles de aproximación y, a partir de estos flujos, estima las colas máximas que se están produciendo. De esta manera, el control fuzzy reactivo tiene consciencia de la problemática de las colas, pese a utilizar sólo un detector, y actúa en consecuencia, intentando evitar que éstas lleguen al máximo permitido en cada uno de los carriles.

Haciendo uso de la información global e instantánea, este control consigue mejoras importantes respecto a los controladores clásicos, tanto en lo que se refiere a tiempos de espera como a colas. Las mejoras dependen de los flujos de entrada y la topología del cruce.

Así, en el caso de test del cruce teórico se tienen mejoras medias diarias en los tiempos de espera del 12% sobre el VA, llegando al 20% en la hora crítica. Respecto a los FT implementados, se tiene una mejora de más del 30% de media diaria y de más del 50% en zonas de poco tráfico. Con el control VA se forman colas de hasta el 75%, mientras que con el fuzzy reactivo éstas llegan al 60%.

El FT que se ha implementado ha sido optimizado para la hora crítica, ya que de otra manera no hubiera sido capaz de dar salida a todos los vehículos que llegan a la intersección.

En resumen, en caso de disponer de sensorización en un cruce aislado, es preferible un control como el fuzzy reactivo implementado, que utiliza información global e instantánea teniendo consciencia de las colas, cosa que no ocurre con el VA.

5 BIBLIOGRAFÍA

[1] "Fuzzy Traffic Light Controller". Devinder Kaur, Elisa Konga, Esa Konga. Proceedings of the 37th Midwest Symposium on Circuits and Systems. 1994. USA.

[2] "Fuzzy Logic Based Control of Traffic Intersection". Vratislav Jerabek, Gerard Lachiver. 1995 Canadian Conference on Electrical and Computer Engineering.

[3] "An Advanced Fuzzy Controller for Traffic Lights". R. Hoyer, U. Jumar. Artificial Intelligence in Real-Time Control. 1994. Valencia.

[4] "Traffic Fuzzy Control: Software and Hardware Implementations". M. Jamshidi, R. Kelsey, K. Bisset. Fifth IFSA world congress. 1993.

[5] "Traffic Signals". Webster F.V. & Cobbe B.M. Road Research Laboratory. Technical Paper Nº 56. London 1966.

[6] "Fuzzy Controler for Intersection Group". Jee-Hyong Lee, Keon-Myung, Hyung Leekwang. 1995 International IEEE/IAS Conference on Industrial Automation and Control: Emerging technologies. Taipei. Taiwan.

[7] "Traffic responsive signal control using Fuzzy Logic. A practical modular approach". Tessa Sayers, Michael G.H. Bell. EUFIT 1996.

[8] "Induction of Fuzzy Rules for a distributed traffic signal timing system". J.R. Clymer, D.R. Wirkkala. Proceedings of the 1994 First International Joint Conference of NAFIPS, IFIS and NASA. Texas. 1994.

[9] "Sistemas de Inteligencia Artificial para ayuda a la decisión en Tiempo Real". J. Cuena, J. Hernández, M. Molina. Revista Carreteras. Vol 74. 1994.

[10] "The role of knowledge based models in traffic management and their design". M. Boero, J. Cuena, H. Kirschfink, H. Traetteberg, D. Wild. First World Congress on ATT and IVHS. Paris. 1994.

[11] Reports anuales 1995. "University of London", "University of Leeds", "University of Westminster", "University of Newcastle", "University of Southampton", "University of Oxford". Revista Traffic Engineering + Control.

[12] "On some aspects of decentralized intelligent urban traffic control system with fuzzy logic". H. Guoguang, Liang Jung, Gerhard Noth. IFAC. Transportation systems: theory and application of advanced technology. 1994.

[13] "Simuladores de Tráfico: Herramientas para ayudar al diseño y evaluación de estrategias". J. Barceló, J.L. Ferrer, R. Grau. Revista Carreteras. Vol 74. 1994.

[14] "A Neural Network application in signal timing control". D.C. Chin, R.H. Smith. Proceedings of the International Conference on Neural Networks. 1996.