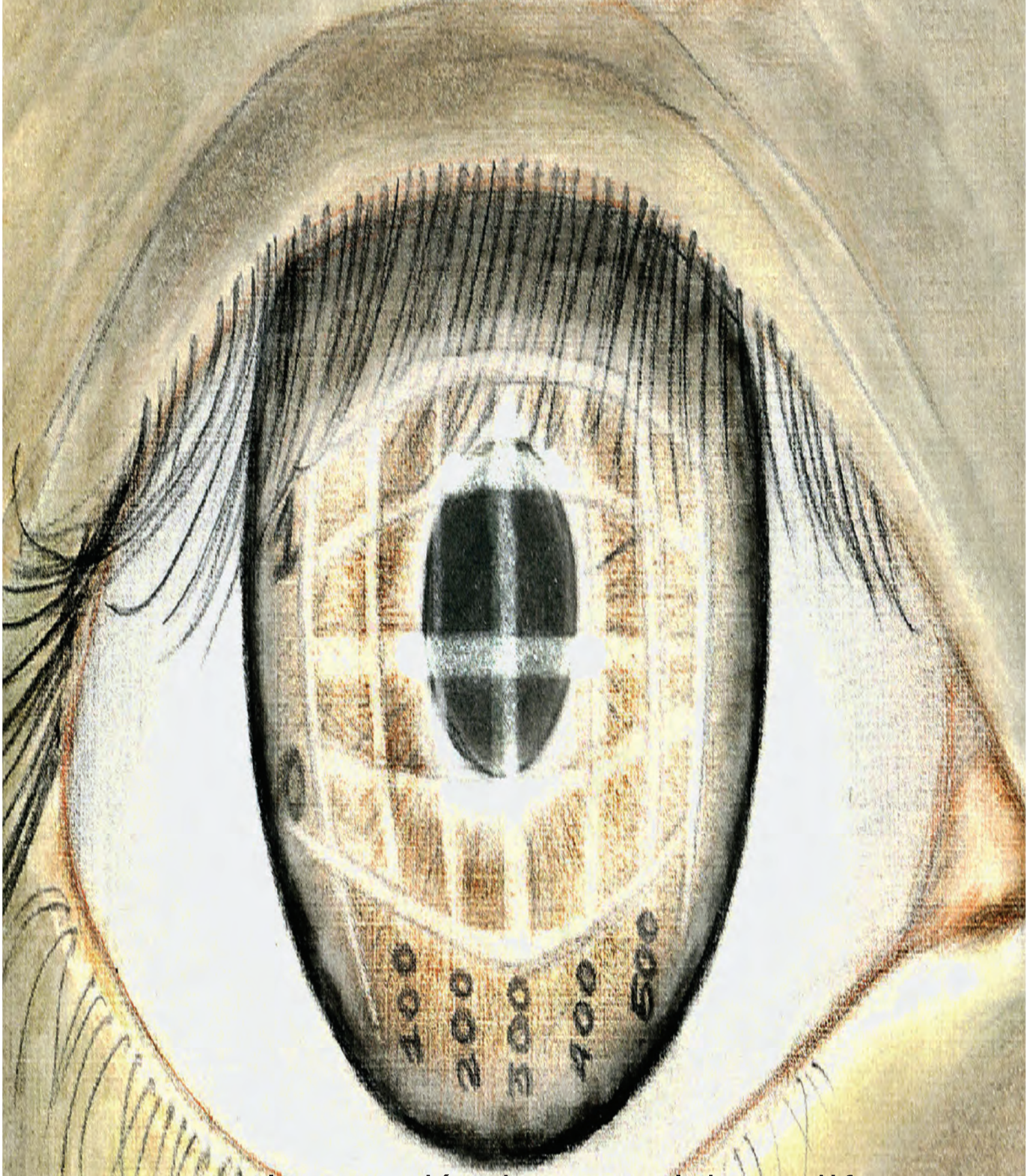

Implementación de controladores difusos con microcontroladores

Ingeniero Fernando Pinilla Ibs
Docente Facultad de Ingeniería
Programa Tecnología en Electrónica – Unimonitoto

IMPLEMENTACIÓN DE CONTROLADORES DIFUSOS CON MICROCONTROLADORES

1. INTRODUCCIÓN

La teoría de conjuntos borroso es una aproximación a la forma de pensamiento humano en la manera de describir los objetos y la forma de estos en agruparse. Esto permite modelar un suceso impreciso de manera más exacta que con la simple descripción que nos otorga la teoría de conjuntos clásica. La teoría de conjuntos borrosos toma en consideración la existencia de predicados vagos, es decir, predicados que al aplicarlos a cierto Universo del Discurso, los elementos del Universo no queda completamente clasificado como "pertenece" o como "no pertenece" a dicho predicado. Así por ejemplo, si consideramos la variable lingüística "Velocidad" sobre la cual hemos definimos un Universo del discurso dado por el conjunto de velocidades entre 0 y 1000 revoluciones por minuto (rpm) y el subconjunto dado por el predicado "rápido", conformado este por el conjunto clásico de todas aquellas velocidades entre 600 rpm y 1000 rpm, nuestro subconjunto nos indicaría que tanto una velocidad de 600 rpm como una de 1000 rpm pertenecen al conjunto "rápido" de igual manera, aunque nosotros tengamos una natural inclinación a considerar más rápido las velocidades más altas. La teoría de conjuntos borrosos permite dar cuanta de estos peculiaridades del lenguaje al definir una función de pertenencia al predicado vago o variable lingüística. Esta función indicará cual es el grado de pertenencia de un elemento cualquiera del Universo del Discurso al predicado vago.

En nuestro ejemplo, podemos definir el conjunto borroso "rápido" a partir de el valor semántico de dicha etiqueta y la función de inclusión mostrada en la figura 1.

Si 1 representa pertenencia completa al conjunto borroso y 0 representa la no pertenencia a dicho conjunto, la función de inclusión indicaría que la pertenencia al conjunto borroso se da de manera gradual con el aumento de la temperatura. En este caso diremos que en 600 rpm no se pertenece al conjunto, que en 1000 rpm se pertenece por completo al conjunto, y que en 800 rpm se pertenece en un 50% al conjunto

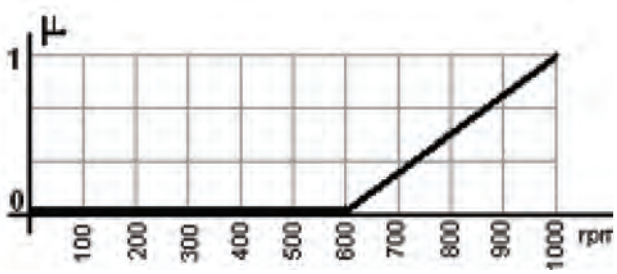

Figura 1 Función de inclusión del predicado vago "rápido"

Nuestro lenguaje es rico en el uso de predicados vagos y sin embargo vemos limitado el análisis que nuestra tradición da a nuestro modo de pensar claramente lógico – lingüístico, al tratamiento especulativo y analítico del lenguaje. La insuficiencia de la lógica clásica al reto que supone inferir con algún grado de certeza a partir de enunciados imprecisos es asumido por [Zadeh 1975] al establecer los fundamentos de la teoría de conjuntos borrosos y extrapolar los valores de la función de inclusión a niveles lógicos de lmismo modo que supone hacerlo en la lógica clásica. Estamos entonces ante el nacimiento de una lógica polivalente que plantea un nuevo enfoque al tratamiento del lenguaje y sus implicaciones en el pensamiento humano.

Los sistemas de control borroso vienen a ser una de las principales aplicaciones de la lógica borrosa y permiten al ingeniero de control usar reglas intuitivas acentuadas por la vaguedad de sus enunciados y formular estrategias que permitan encontrar esfuerzos de control, de un modo similar a los procedimientos seguidos por los sistemas expertos pero superando los inconvenientes de implementación que estos suponen.

Este artículo realiza una revisión de los fundamentos necesarios para el desarrollo de controladores difusos en problemas de control y propone esquemas de implementación de los controladores en sistemas de procesamiento basados en microcontroladores. Se tomará como referencia de los ejemplos de código el conjunto de instrucciones de l89C52

2. CONJUTOS BORROSOS

Definiremos un conjunto borroso P como la tripleta dada por el Universo de discurso U, la etiqueta del predicado vago P o variable lingüística asociada al conjunto U, y la función de inclusión de los elementos de U en el rango [0 1]

<U, P, μP>

La igualdad de dos conjuntos borrosos P y Q queda dada por la igualdad de sus funciones de inclusión, es decir,

$$(\forall <U, P, \mu P> \wedge <U, Q, \mu Q>) (\forall u \in U) (P=Q \Leftrightarrow \mu P(u)=\mu Q(u))$$

El complemento P´ de un conjunto borroso P queda definido por

$$(\forall <U, P, \mu P>) (\exists <U, P', \mu P'>) (\forall u \in U) (\mu P'(u)=1-\mu P(u))$$

El antónimo AntP de un conjunto borroso P se define como

$$(\forall <U, P, \mu P>) (\exists <U, AntP, \mu AntP>) (\forall u \in U) (\mu AntP(u)=\mu P(1-u))$$

El antónimo de P tiene las mismas consideraciones semánticas que supone tener el antónimo de un predicado vago. Por ejemplo el "Ant rápido", que podemos denominar lento, define la función de inclusión mostrada en la figura 2.

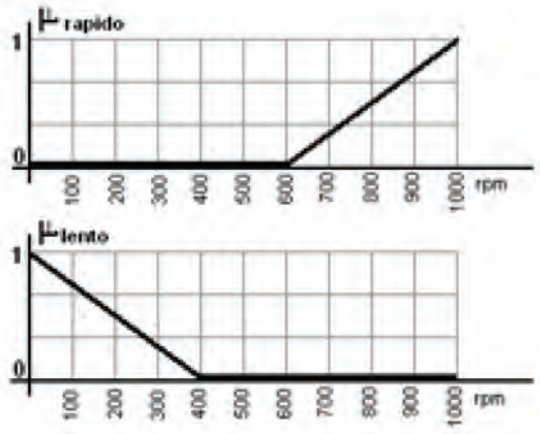

Figura 2 Función de inclusión de lpredicado vago "rápido" y su antónimo "lento"

Los operadores sobre predicados vagos también se pueden definir en términos de funciones de inclusión de conjuntos borrosos. Expresiones como "Muy rápido" o "Mas o menos rápido" definen nuevos conjuntos borrosos a partir del conjunto borroso rápido. Estos operadores se les reconoce como Concentración y Dilatación respectivamente y quedan definidos para un conjunto borroso P por

$$(\forall <U, P, \mu P>) (\exists <U, MP, \mu MP>) (\forall u \in U) (\mu MP(u)=\mu P^2(u))$$

y

$$(\forall <U, P, \mu P>) (\exists <U, \pm P, \mu \pm P>) (\forall u \in U) (\mu \pm P(u)=\mu P^{1/2}(u))$$

Las operaciones entre conjuntos borrosos de Unión e Intercepción quedan definidos por

$$(\forall <U, P, \mu P> \wedge <U, Q, \mu Q>) (\exists <U, P \cap Q, \mu P \cap Q>) (\forall u \in U) (\mu P \cap Q (u)=\min[\mu P(u), \mu Q(u)])$$

y

$$(\forall <U, P, \mu P> \wedge <U, Q, \mu Q>) (\exists <U, P \cup Q, \mu P \cup Q>) (\forall u \in U) (\mu P \cup Q (u)=\max[\mu P(u), \mu Q(u)])$$

Las funciones min( ) y max( ) son en realidad las más restrictivas de una serie de posibles funciones que cumplen las leyes de Morgan y a las que se les conoce como Normas Triangulares y Conormas Triangulares respectivamente.

Finalmente diremos que un conjunto borroso P puede definir un conjunto clásico conocido como el soporte de P y que denotaremos como sopP definido por

$$(\forall <U, P, \mu P>)\ (\exists\ sopP)\ (\forall u \in U)\ (sopP = \{u \in U / \mu P(u) \neq 0\})$$

$$(\forall \exists)(\forall \exists)(\forall \exists) \wedge \vee \Rightarrow \Leftrightarrow \exists \mu \in \notin \cup \cap$$

3. LOGICA BORROSA

La lógica borrosa permite modelar el cálculo con predicados vagos cualificados de verdad asignando factores de certeza entre 0 y 1. Asocia las funciones de pertenencia que definen los conjuntos borrosos a los valores de certeza y deriva nuevas funciones de inclusión a partir de las operaciones entre dichos conjuntos permitiendo encontrar la certeza de nuevos conjuntos. A esto se ha llamado Razonamiento Aproximado.

Los antecedentes de un razonamiento estarán cada uno asociados a conjuntos borrosos diferentes, con universos del discursos eventualmente diferentes, que conllevan a una nueva función de inclusión. Consideremos por ejemplo los predicados vagos P y Q sobre los cuales deseamos establecer un calculo proposicional. Tenemos entonces un nuevo conjunto borroso R que es una relación definida por

$$(\forall <U, P, \mu P> \wedge <V, Q, \mu Q>)\ (\exists <UxV, R, \mu R>)\ (\forall u \in U)\ (\forall v \in V)\ (\mu R: UxV \rightarrow [0, 1])$$

Relación entre el espacio UxV que denotaremos como $\mu P\text{-}Q$.

Las relaciones borrosas más interesantes son, por supuesto, las relaciones condicionales donde toman valor expresiones del lenguaje común como "Si x es P entonces y es Q" donde se ponen de manifiesto la relación condicional entre P y Q. En este orden de ideas es posible definir relaciones borrosas para establecer la Conjunción, Disyunción, y diferentes tipos de implicación entre las que se destacan el Modus Ponen y el Modus Tolen. Presentamos a continuación las diferentes expresiones para el calculo de las relaciones borrosas

Conjunción borrosa

$$(\forall <U, P, \mu P> \wedge <V, Q, \mu Q>)\ (\exists <UxV, P \rightarrow Q, \mu\ P \rightarrow Q>)\ (\forall u \in U)\ (\forall v \in V)\ (\mu P \rightarrow Q(u, v) = \mu P(u)\ \mu Q(v))$$

Disyunción borrosa

$$(\forall <U, P, \mu P> \wedge <V, Q, \mu Q>)\ (\exists <UxV, P \rightarrow Q, \mu\ P \rightarrow Q>)\ (\forall u \in U)\ (\forall v \in V)\ (\mu P \rightarrow Q(u, v) = \mu P(u) + \mu Q(v))$$

Implicación material

$$(\forall <U, P, \mu P> \wedge <V, Q, \mu Q>)\ (\exists <UxV, P \rightarrow Q, \mu\ P \rightarrow Q>)\ (\forall u \in U)\ (\forall v \in V)\ (\mu P \rightarrow Q(u, v) = \mu P'(u) + \mu Q(v))$$

Calculo proposicional

$$(\forall <U, P, \mu P> \wedge <V, Q, \mu Q>)\ (\exists <UxV, P \rightarrow Q, \mu\ P \rightarrow Q>)\ (\forall u \in U)\ (\forall v \in V)\ (\mu P \rightarrow Q(u, v) = \mu P'(u) + \mu P(u)\ \mu Q(v))$$

Modus Ponen Borroso

$$(\forall <U, P, \mu P> \wedge <V, Q, \mu Q>)\ (\exists <UxV, P \rightarrow Q, \mu\ P \rightarrow Q>)\ (\forall u \in U)\ (\forall v \in V)\ (\mu P \rightarrow Q(u, v) = \sup \{x \in [0, 1] / \mu P(u)^* c \leq \mu Q(v)\})$$

Modus Tolen Borroso

$$(\forall <U, P, \mu P> \wedge <V, Q, \mu Q>)\ (\exists <UxV, P \rightarrow Q, \mu\ P \rightarrow Q>)\ (\forall u \in U)\ (\forall v \in V)\ (\mu P \rightarrow Q(u, v) = \inf \{x \in [0, 1] / \mu Q(v) + c \leq \mu P(u)\})$$

4. CONTROLADORES BORROSOS

Un sistema de control automático está representado por los elementos mostrado en la figura 3

$$(\forall <U, P, \mu P >)\ (\exists <U, AntP, \mu AntP>)$$
$$(\forall u \in U)\ (\mu AntP(u)=\mu P(1-u))$$

Figura 3 Diagrama de bloques de un sistema de control automático

En los procesos industriales la señal controlada puede ser temperatura, presión, nivel, posición, velocidad, etc. Nuestro interés es garantizar que esta variable presente estabilidad absoluta, es decir, que los valores de esta estén confinados dentro de ciertos límites transcurrido el tiempo donde tienen incidencia los fenómenos transitorios. El otro problema que interesa es la estabilidad relativa que supone problemas como garantizar que la variable controlada responda en los tiempos adecuados dada una perturbación, que no presente sobrepicos más allá de ciertos umbrales o que su respuesta al disturbio no presente oscilaciones.

El objetivo fundamental del controlador es mantener iguales las señales de referencia y la señal sensada, o dicho de otro modo, mantener en cero la señal de error. A esto se agrega el responder de manera rápida a los cambios del error (derivada del error), o al error acumulado (integral del error); lo anterior supone la forma más clásica de control análogo: PID (proporcional, integral y derivativo).

Un PID en un lazo de control como el mostrado en la figura 3 puede ser fácilmente implementado con Amplificadores Operacionales, con procedimientos de diseño claramente definidos, pero solo tiene validez si nuestro proceso se modela con ecuaciones diferenciales lineales invariantes en el tiempo. Para el 90% de las aplicaciones de control esto resulta suficiente, y en caso de contar con plantas no lineales, estas se podrán linealizar alrededor de un punto de operación. Sin embargo no todos los procesos son susceptibles de dichos procedimientos.

El otro inconveniente que presenta un control PID clásico o en general, cualquier sistema de compensación implementado con amplificadores operacionales, resulta en la dificultad o imposibilidad de gozar de las ventajas de un sistema digital: mandos secuenciales, comunicaciones, visualización, registro estadístico, etc. Esto se puede resolver implementando sistemas de control digital. Para ello requerimos establecer un periodo de muestreo, lo más grande posible pero garantizando estabilidad en la planta. Durante cada muestreo la planta va a estar en lazo abierto por lo cual resulta imprescindible dimensionar correctamente el tiempo de muestreo. Es típico fijar el umbral máximo de este tiempo como 20 veces 1/f donde f representa la frecuencia máxima del sistema. Un tiempo menor a este si bien resulta conveniente para el propósito de control, dejará menos tiempo de procesador para las actividades complementarias de un sistemas digital como las mencionadas anteriormente. Las técnicas de diseño en control digital buscan encontrar estos tiempos máximos, pero revisten otro inconveniente: las ecuaciones de diferencia para implementar el controlador cuentan con constantes con más de 6 cifras decimales. El truncar el número de decimales implica cambiar las dinámicas del controlador considerablemente, pero el cargar con ellas demandará en el uso de un sistema de procesamiento que soporte punto flotante como DSP, o el diseño del controlador en un lenguaje de alto nivel

(como lenguaje C) cuando se requiera usar micro contradores de uso comercial para nuestras aplicaciones de control.
Los procedimientos de control difuso está en un camino intermedio al permitir usar microcontroladores comerciales de gama media, proporcionando controladores con rendimientos comparables a los usados con técnicas de control digital, pero usando menor capacidad computacional en el procesador elegido, reglas de control intuitivas, y aplicaciones a sistemas multivariables no lineales.

5. ESTRUCTURA INTERNA DE UN CONTROLADOR DIFUSO

La figura 4 ilustra las diferentes partes que conforman un controlador borroso. Describimos a continuación los elementos del controlador:

Figura 4 Estructura interna de un controlador borroso

5.1 BORROSIFICADOR

Su función es establecer una relación entre los diferentes puntos de entrada no borrosos (que suponemos de ahora en adelante entradas digitales del sistema), y los correspondientes conjuntos borrosos. El borrosificador de más fácil implementación será el tipo Singleton donde cada vector en el espacio de entrada define en si un conjunto borroso con valor de μ igual a 1 en el punto de entrada y valor de μ igual a 0 en cualquier otro punto. En la figura 5 se muestran las 256 funciones de inclusión que supone tener las 256 posibles valores de una variable lingüística de entrada que se ha digitalizado con un conversor ADC de 8 bits.

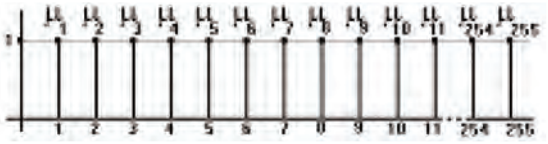

Figura 5 Borrosificador tipo Singleton para las 256 entradas posibles provenientes de un ADC de 8 bits

Otros tipos de borrosificador presumen un tratamiento estadístico de cada entrada. En general se asocia una función de inclusión exponencial tipo campana centrada en el valor digital que interesa con una amplitud determinada por el espaciamiento entre otras entradas. Un tratamiento de este tipo es factible en sistemas donde la cuantificación de la señal muestreada sea con pocos bits (conversores ADC de 4 bits).

5.2 BASE DE REGLAS

Las reglas borrosas asocian a uno o varios conjuntos borrosos de entrada, uno o varios conjuntos borrosos de salida. Los conjuntos borrosos de entrada están asociados mediante conjunciones y su estructura toma la forma

IF $x1 \in P1 \wedge x2 \in P2 \wedge \ldots xn \in Pn$ THEN $y \in Q$
Donde X=(x1, x2, ... xn) representa el vector de entrada y "y" representa la salida.

El determinar las reglas borrosas es el asunto principal del diseñador de un controlador borroso. El diseñador debe hacer uso de su intuición en el cálculo del esfuerzo de control. Los conjuntos borrosos de entrada y el conjunto borroso de salida serán conjuntos borrosos (particiones borrosas) de la variable lingüística medida o

controlada. Si por ejemplo tenemos el conjunto borroso "velocidad Muy Alta" (VMA) entonces el esfuerzo de control podría indicar "Mínima Corriente" (MC) y expresar la anterior como

IF velocidad es VMA THEN corriente es MC

En el caso de contar con dos variables lingüísticas de entrada y una variable de salida ayuda el considerar arreglos bidimensionales, denominados Memoria Asociativa Borrosa FAM, que den cuenta de todas las posibles combinaciones que pueden conformar reglas. La figura 6 muestra las particiones de dos variables lingüísticas de entrada: velocidad y derivada de la velocidad; y una variable lingüística de salida: voltaje de armadura. Cada una de las variables lingüísticas está digitalizada con un ADC y borrosificada por un borrosificador tipo Singleton. El bloque de preprocesado escalado los valores a 255 (FF hexadecimal) y cada variable se ha particionado en 5 particiones según se muestra en la figura. Los nombres de las particiones son genéricos y representan "Pequeño Pequeño" PP, "Pequeño" P, "Regulada" R, "Grande" G, y "Grande Grande" GG. En la figura 7 se ha establecido una FAM de las variables lingüísticas de la figura 6 con 25 reglas.

La regla señalada en la figura 7 se puede leer entonces como

IF "velocidad" es G es "Derivada de la velocidad" es P THEN "Voltaje de armadura" es R

Es importante tener en cuenta las siguientes normas a la hora de construir una base de reglas:

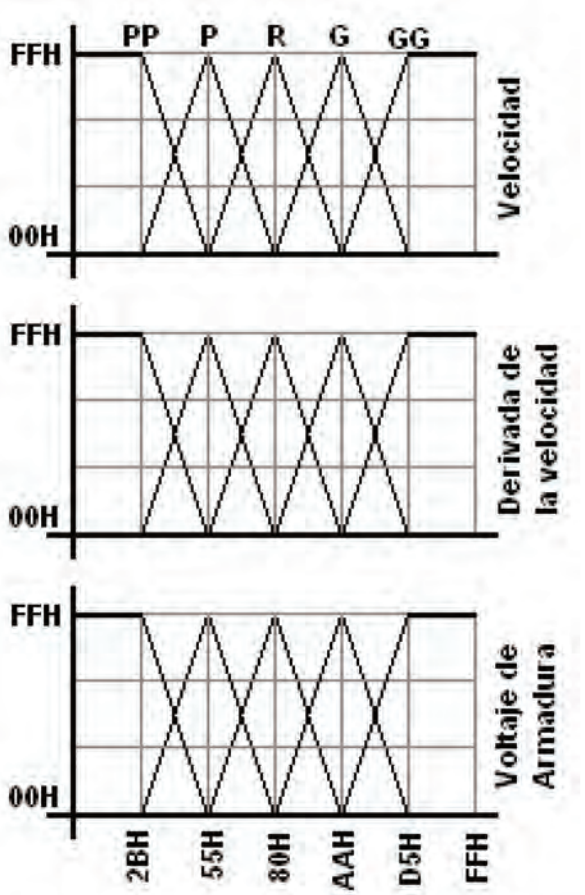

Figura 6 Particiones de las variables lingüísticas

Derivada de la velocidad \ Velocidad	PP	P	R	G	GG
PP	GG	G	G	R	PP
P	GG	R	R	R	PP
R	G	R	R	P	P
G	G	G	P	PP	R
GG	R	G	PP	PP	G

Figura 7 FAM de las variables lingüísticas de la figura 6

1. Cada variable lingüística debe tener un número impar de particiones: 3, 5 y 7 son los números más comunes de particiones
2. Las particiones soportan entre si un solapamiento de hasta un 50% típicamente
3. Se recominendan particiones completas, es decir, que todos los elementos que conforman la variable lingüística pertenezcan al soporte de alguna partición de la variable.
4. Son preferibles las particiones tipo triangular y tipo trapezoidal donde los parametros de las particiones son puntos singulares de la variable como la referencia de lcontrol, los valores extremos, etc.
5. El uso de más de dos entradas linguisticas se puede representar mediante varias Memorias Asosiativas Borrosas, sin embargo no es recomendable por el gran tamaño que puede tomar la base de reglas.

5.3 DISPOSITIVOS DE INFERENCIA BORROSA

Se denominan dispositivos de inferencia borrosa a los dispositivos que interpretan las reglas borrosas para obtener un conjunto borroso de la relación borrosa entre los antecesdentes y el consecuente de la regla. La implicación borrosa entre antecedente y consecuente se formula generalmente en una de dos reglas

Regla de lmínimo
P->Q(x1,x2, … , xn, y) = min [μP(x1, x2, … xn), μQ(y)]

Regla de lproducto
μP->Q(x1,x2, … , xn, y) = μP(x1, x2, … xn) μQ(y)

donde a su vez la relación entre los antecedentes se puede definir por regla de l mínimo o regla de lproducto

Es importante hacer notar la gran sencillez que implica implementar un dispositivo de inferencia borroso de este tipo al considerar que la mayoría de micro controladores de gama media poseen instrucciones para comparar y multiplicar.

Cada regla proporcionará entonces un conjunto borroso de salida. El conjunto borroso de salida de las M reglas (o las que apliquen) se define como la unión de los conjuntos borrosos de salida de cada regla. Lo anterior está determinado por el uso de lmáximo para cada función de inclusión en cada uno de los conjuntos de salida.

5.4 DESBORROSIFICADOR

El desborrosificador tiene la función de encontrar la salida no borrosa de un conjunto borroso. Las estrategias a seguir toman en consideración la manera como se realizaron las particiones en el conjunto de salida.

Desborrosificador por máximo
En este interesa el valor máximo de la partición que resulta del dispositivo de inferencia

Borrosificador por media de centros
Usa la media ponderada de los valores centrales de cada partición de salida.

Borrosificador por centro de área
Usa el área de cada partición de salida. De esta manera se tiene en cuenta los aportes que particiones no simétricas pueden tener sobre la respuesta final

6 UNIDADES FUNCIONALES DE CONTROLADORES BORROSOS

Consideremos controladores difusos con borrosificadores tipo singleton y dispositivos de inferencia borrosos por regla de lmínimo. La implementación de este controlador se logra mediante la división del proceso en unidades funcionales pequeñas implementadas mediante subrutinas de código de programa para microcontroladores. Las siguientes son las unidades funcionales:

Bloque de Preprocesado
Circuito de Función de Pertenencia CFP
Función Extractor de Minimo
Función TRUNC
Unidad Módulo de Regla MR
Función Union
Función Desborrosificador

Describiremos a continuación la implementación de estas unidades en código de programa tomando como punto de partida un diseño hipotético de un controlador borroso para la velocidad en un motor DC.

6.1 DESCRIPCIÓN DEL PROBLEMA

Deseamos realizar un controlador para la velocidad de un motor de DC. La variable medida será la velocidad y la variable controlada será el voltaje de armadura de lmotor.

6.2 DESCRIPCIÓN DEL CONTROLADOR

El controlador que usaremos para resolver el problema de control y describir las unidades funcionales se puede especificar así:

Estrategia de control PD (Proporcional-Derivativo), donde consideramos dos entradas al sistema: velocidad y diferencia de la velocidad actual y la anterior. La salida estará determinada por la variable lingüística Voltaje de Armadura.

Bloque de Preprocesado: la función de este bloque es entregar las señales de la variable lingüística al borrosificador. En este caso el bloque preprocesado está conformado por la lectura periódica de la entrada sensada y el cálculo inmediato de la diferencia de la entrada actual con la entrada anterior. Es importante indicar que implícitamente los valores están escalados a 255 que es el valor máximo que nos entrega un ADC de 8 bits. En este ejercicio no vamos a suponer cambios en la referencia de velocidad, y simplemente consideraremos que la velocidad sobre la cual regularemos corresponde a 128 (80H) en la lectura de la velocidad del ADC, de la escala de medida 0-255 (00H-FFH).

Borrosificador tipo singleton: usaremos las entradas del ADC como conjuntos borrosos en si. También el cálculo que nos arroje el bloque preprocesado de la diferencia de velocidades.

Desborrosificador por media de centros: buscamos la expresión para encontrar los valores de la salida a partir de los centros de cada partición de la variable de salida Voltaje de Armadura. Estos valores estarán escalados en un rango de 0-255 (00H-FFH).

Dispositivo de inferencia borrosa por regla de lmínimo: el procesamiento de las reglas de la base de reglas se realizará mediante comparaciones de acuerdo a la relación borrosa que se establece entre antecedentes y consecuentes de cada regla de la FAM. A su vez la relación entre antecedentes tambien obedecerán a la regla del mínimo.

La base de reglas: estará descrita por la Memoria Asosiativa Borrosa de la figura 6.

Bloque de Postprocesado: su función es retornar los valores escalados obtenidos del desborrosificador y convertirlos en los niveles de tensión y potencia para el manejo de lmotor. Asumiremos que 255

(FFH) representará el voltaje de armadura nominal de la máquina. El postprocesado no estará representado por código de programa sino por el DAC y el amplificador de potencia para el manejo del motor como bien puede ser un puente rectificador totalmente controlado donde el voltaje de control maneja el ángulo de disparo del puente.

6.3 BLOQUES DE UNIDADES FUNCIONALES

6.3.1 Circuito de Función de Pertenencia CFP

Su función es extraer la función de pertenencia de cada una de las entradas de las variables lingüísticas con una partición de dicha variable. Si suponemos particiones tipo triangular y tipo trapezoidal es posible elaborar un programa que permita encontrar el valor de la función de inclusión usando simplemente la definición por parámetros que implica cada una de estas particiones. Otra alternativa es elaborar en una tabla los valores para cada una de las posibles entradas. La primera solución requiere mayor tiempo de procesamiento para encontrar el valor solicitado pero conlleva un uso más racional de la memoria de programa. La segunda alternativa es una solución rápida en especial cuando las funciones de inclusión son definidas bajo condiciones muy particulares.

6.3.1.1 Funciones de inclusión mediante calculo computacional

La función de inclusión trapezoidal mostrada en la figura 8 presenta la ventaja de poder implementar cualquiera de las funciones de inclusión de la figura 6 cambiando los valores de los parámetros. En el siguiente código fuente el registro acumulador A tiene el valor de la variable lingüística. Las posiciones de memoria PAR1, PAR2, PAR3 y PAR4 representan cada uno de los cuatro parámetros que definen la función trapezoidal. Son posiciones de la memoria de datos que deben ser cargados previamente con datos de la memoria de programa.

La subrutina RECTA toma como parámetros los valores de "u" extremos de la recta, en los registros R1 y R2, y retorna en el acumulador A el valor correspondiente a la función.

```
TRAPEZ:  MOV     B, A
         MOV     A, PAR1
         SUBB    A,B
         JNC     TRAMO2
         MOV     A, #000H
         RET
TRAMO2:  MOV     A,PAR2
         SUBB    A,B
         JNC     TRAMO3
         MOV     A, PAR1
         MOV     R1,A
         MOV     A, PAR2
         MOV     R2,A
         CALL    RECTA
         RET
TRAMO3:  MOV     A, PAR3
         SUBB    A,B
         JNC     TRAMO4
         MOV     A, #0FFH
         RET
TRAMO4:  MOVA,PAR4
         SUBB    A,B
         JNC     TRAMO5
         MOV     A, PAR3
         MOV     R1,A
         MOV     A,PAR4
         MOV     R2,A
         CALL RECTA
         SUBB    A,#0FFH
         RET
TRAMO5:  MOV A,#0FFH
         RET
```

Código de la función trapezoidal

6.3.1.2 Funciones de Inclusión mediante tablas

Una implementación más práctica para el cálculo de la función de inclusión se logra mediante tablas. En estas se tiene menor

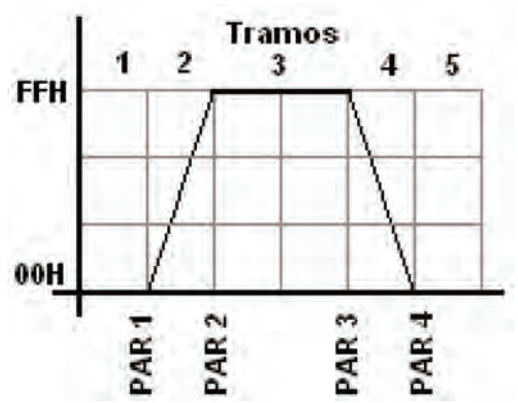

Figura 8 Función de Inclusión Trapezoidal definida mediante 4 parámetros en 5 tramos

complejidad computacional. La dirección de la tabla representará el valor de la variable lingüística y el valor leído representará el valor de la función de inclusión.

El código mostrado a continuación permitirá el cálculo de una función de inclusión en una tabla

```
TRAPEZ:  MOV  DPTR,#TABLA
         MOV      A,@A+DPTR
         RET
```

En principio la tabla contiene con 256 bytes de memoria de programa, pero atendiendo las simetrías de la función es posible reducir considerablemente este número. Una alternativa intermedia sugiere implementar tablas a través de bloques de decisión como si se tratara de cálculo computacional, pero implementando en tabla los tramos que requieren cambios.

Resulta claro que tanto en las funciones calculadas mediante cálculo computacional como las calculadas mediante tabla pueden servir de base el cálculo de otras funciones de inclusión.

Por ejemplo en el cálculo de la función de inclusión para PP se puede hacer a través de la función GG teniendo en cuenta que PP = Ant GG I

6.3.2 Función Extractor de Minimo

El bloque Función Extractor de Mínimo tiene la función de determinar cuales el valor por regla del mínimo de todos los valores de las funciones de inclusión de las premisas de una regla. Se trata de una comparación entre los diferentes valores de la función de inclusión de las premisas. El código mostrado a continuación permite encontrar el menor valor entre las dos funciones de inclusión implementadas mediante tablas de las premisas de una regla. Las subrutinas LEERP1 y LEERP2 determinan el valor de cada una de las variables lingüísticas implementadas mediante las tablas TABLA1 y TABLA2

```
MININ:   CALL     LEERP1
         MOV      DPTR,#TABLA1
         MOV      A,@A+DPTR
         MOV      AUX,A
         CALL     LEERP2
         MOV      DPTR,#TABLA2
         MOV      A,@A+DPTR
         MOV      B,A
         MOV      A,AUX
         SUBB     A,B
         JC       MIN1
         MOV      A,AUX
         RET
MIN1:    MOV      A,B
         RET
```

6.3.3 Función TRUNC

La función TRUNC establece un truncamiento entre la partición de salida correspondiente a la regla y el valor

derivado de la función MININ. Su implementación es importante en sistemas donde se requiera encontrar el conjunto borroso de salida.

6.3.4 Unidad Módulo de Regla MR

La Unidad Módulo de Regla es una estructura computacional capaz de procesar una regla. La figura 9 ilustra la arquitectura de un MR para la regla señalada en la figura 7

IF x1 ∈ P1 ∧ x2 ∈ P2 ∧ ... xn∈Pn THEN y ∈ Q
Donde X=(x1, x2, ... xn) representa el vector de entrada y "y" representa la salida.

Figura 9 Módulo de regla para la regla señalada en la figura 7

6.3.5 Función Unión

Los diferentes conjuntos borrosos de salida, obtenidos con cada MR, deben unirse para producir un único conjunto de salida. Como se definió anteriormente, la unión de varios conjuntos borrosos se determina por el máximo de los valores de las diferentes funciones de inclusión.

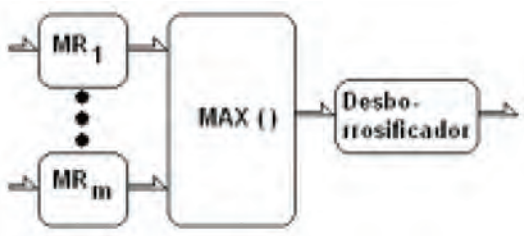

Figura 10 Función Unión en paralelo

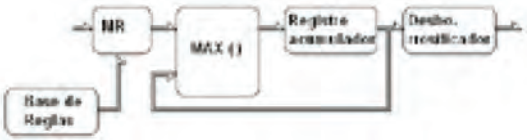

Figura 11 Función Unión secuencial

La figura 10 y 11 muestran dos versiones de implementación; la figura 10 es una implementación en paralelo que requiere por cada regla un MR. La figura 11 es una versión de implementación secuencial donde se hace uso de un registro acumulador y un MR reprogramable con cada regla, donde en cada iteración se actualizan los parámetros del módulo de regla y el valor del registro acumulador con el valor del MR y el valor del mismo registro que se inicializa en cero luego de las m reglas.

6.3.6 Función Desborrosificador

En el caso de usar un desborrosificador por media de centros, se debe implementar la función

$$y = \frac{y_1 \min 1 + y_2 \min 1 + y_3 \min 1 + y_4 \min 1 + y_5 \min 1}{y_1 + y_2 + y_3 + y_4 + y_5}$$

donde yi son los centros de la partición de salida y mini representan los valores obtenidos de la función unión. De esta manera los valores de min() ponderan los valores de yi.

(∀∃)(∀∃)(∀∃)∧∨⇒⇔∃μ∈∉∪∩