Generación y Optimización de Controladores Difusos Utilizando el Modelo NEFCON

Generation and Optimization of Fuzzy Controllers Using the NEFCON Model

Erik V. Cuevas Jiménez[1,2], **Daniel Zaldívar Navarro**[1,2], **Marco Pérez Cisneros**[1] **y Ernesto Tapia Rodríguez**[2]

Departamento de Ciencias computacionales, Universidad de Guadalajara, CUCEI
Av. Revolución 1500, Guadalajara, Jal, México
{erik.cuevas, daniel.zaldivar, marco.perez}@cucei.udg.mx
[2]*Institut fur Informatik, Freie Universität Berlin*
Takustr. 9, Berlin, Alemania
tapia@inf.fu-berlin.de

Artículo recibido en Enero 07, 2008; aceptado en Marzo 26, 2009

Resumen. El diseño de algoritmos que operen sobre plantas con dinámicas no modeladas aún representa un reto en el área de control automático. Una solución podría ser el uso de algoritmos capaces de aprender en tiempo real mediante la interacción directa con la planta. El modelo NEFCON, permite construir la estructura de un controlador difuso del tipo Mamdani capaz de aprender las reglas y adaptar los conjuntos difusos. La principal ventaja del modelo NEFCON respecto a otros enfoques de aprendizaje, es que su diseño se reduce a expresar la calidad del error actual de la planta a controlar. Sin embargo, una desventaja del modelo NEFCON es la pobre exploración de los estados de la planta durante el aprendizaje, lo cual hace imposible su aplicación para sistemas dinámicos no lineales. En este trabajo se propone la adición de ruido Gaussiano a las variables de estado de la planta, con el objetivo de asegurar una exploración amplia de los estados, facilitando la convergencia del algoritmo de aprendizaje, cuando se aplica a sistemas no lineales. En particular, se muestra la efectividad de la propuesta en el control del sistema dinámico de la "pelota y el balancín" (Ball and Beam)
Palabras clave: Sistemas de control adaptativos, sistemas de control por aprendizaje, control inteligente, control no lineal.

Abstract. The design of algorithms that operate on un-modeled dynamics plants still represents a challenge in automatic control area. A solution could be the use of algorithms able to learn in real time by direct interaction with the plant. NEFCON, allows to build a Mamdani fuzzy controller able to learn rules and adapt the fuzzy sets. The main advantage of NEFCON compared with other learning approaches, is that its design express the current error state of the plant to be controlled. However, a disadvantage of NEFCON is its poor exploration of the states of the plant during the learning; disable its application on nonlinear dynamic systems. In this work the addition of Gaussian noise to the states of the plant is proposed with the objective to assure a wide exploration of the states, simplifying the convergence, when it is applied to nonlinear systems. In particular, the effectiveness of our proposal is shown in the control of the "ball and beam" dynamic system.
Keywords: Adaptive control systems, learning control systems, intelligent control, nonlinear control.

1 Introducción

La teoría de control tradicional ha sido extensamente aplicada, sin embargo, cuando se carece del modelo de la planta o bien es impreciso, es necesario establecer la estrategia de control con un enfoque diferente. El control inteligente, ha demostrado buenos resultados, incluso aun sin contar con un modelo del sistema a controlar [Brown y Harris, 1994; Jang *et al.*, 1997; Gupta y Sinha, 1999; Haykin, 1999)]. Diferentes esquemas de control inteligente han sido propuestos [Hangos *et al.*, 2004], entre los que se encuentran: la lógica difusa, las redes neuronales, los algoritmos genéticos, etc. La lógica difusa se puede utilizar para generar controladores basados en reglas y observaciones cualitativas de los procesos. Sin embargo, los controladores "difusos" carecen de la capacidad de optimización y aprendizaje. Existen en la literatura trabajos que podrían compensar alguna de estas carencias, como el de Moon y Jin [Moon y Jin, 2002], que propone un controlador organizado de manera jerárquica, compuesto de varios sistemas difusos del tipo Sugeno y donde dicho

controlador restringe el número de reglas que cada sistema difuso puede generar. Otro trabajo relacionado es el de Shi-Yuan *et al*. [Shi-Yuan *et al*., 2002], donde se propone un controlador basado de por lo menos dos sistemas difusos del tipo Sugeno, cada uno compuesto de 5 reglas. Aunque su propuesta garantiza convergencia, para el cálculo y calibración de los parámetros de los consecuentes de sus reglas, es necesario conocer diferentes características dinámicas del modelo de la planta, lo cual lo hace poco ventajoso en comparación a uno clásico. Lon-Chen y Huang-Yuan [Lon-Chen y Huang-Yuan, 2007], propusieron un controlador neuronal adaptativo de modos deslizantes. Dicho controlador se compone de dos sistemas desacoplados: una red neuronal y un controlador por compensación. La idea consiste en asegurar estabilidad mediante el uso del compensador, mientras que la salida deseada es aproximada por la actuación de la red neuronal. Mediante el uso de este enfoque, es posible obtener la actuación deseada en términos del factor de sobretiro y tiempo de estabilización. No obstante, los algoritmos usados en la calibración del compensador y en la modificación de los pesos de la red neuronal, complican la estructura y el tiempo de cómputo necesario; por lo que su aplicación se reduce a nivel de simulación, haciéndolo restrictivo para su uso en tiempo real.

La mayoría de estos trabajos proponen sistemas que adaptan y modifican sus parámetros, suponiendo que tanto los datos del proceso y la estrategia de control para la planta son conocidos. Lo anterior en la práctica, normalmente no se cumple. Una solución a este problema, es que el controlador aprenda durante su funcionamiento la estrategia de control mediante la interacción directa con la planta [Domínguez et al., 2004].

Los sistemas *neurodifusos* resultan de la combinación de redes neuronales con la lógica difusa [Harris et al., 2002]. Estos sistemas conservan la estructura difusa e incluyen una conveniente capacidad de aprendizaje y adaptación. Los diferentes tipos de sistemas *neurodifusos* resultan de la utilización de algoritmos de aprendizaje en estructuras difusas [Jang, 1993], de operadores difusos en redes neuronales y más recientemente, de redes neuronales con polinomios difusos [Sung-Kwun et al., 2007]. Las capacidades de los sistemas neurodifusos para la predicción, estimación e identificación son bien conocidas [Gonzalez y Tang, 2007].

El modelo NEFCON [Nauck *et al*., 1997], permite construir un controlador difuso del tipo Mamdani [Yen, et al., 1999]. El controlador se basa en el uso de una señal de refuerzo, la cual se usa para medir el desempeño, para configurar las reglas y también para modificar los conjuntos difusos del controlador. En este artículo se describe el modelo NEFCON y cómo puede utilizarse en el control de sistemas dinámicos no lineales. Una parte muy importante de este enfoque, consiste en la definición de la señal de refuerzo, que en este artículo se considera como la calidad del error actual de la planta a controlar. A manera de ejemplo, se utiliza el sistema dinámico no lineal de la "pelota y el balancín" (Ball and Beam).

El modelo NEFCON, presenta otras ventajas en comparación a los enfoques anteriormente citados. Al obtener, como resultado del aprendizaje, un controlador basado en un sistema difuso del tipo Mamdani, es posible obtener la estrategia de control a partir del análisis del conjunto de reglas **SI-ENTONCES** típicas del tipo de inferencia Mamdani [Yen y Langari, 1999]. En contraposición, para otros enfoques tales como [Moon y Jin, 2002] y [Shi-Yuan *et al*., 2002], no es posible obtener ningún tipo de información adicional, ya que la información obtenida del aprendizaje se encuentra codificada (caja negra) en los coeficientes de las funciones del consecuente del sistema de inferencia Sugeno [Yen y Langari, 1999]. La principal ventaja del modelo NEFCON, en comparación con otras propuestas de aprendizaje, es que su diseño se reduce a plantear cualitativamente el error actual de la planta a controlar (señal de refuerzo), por lo que no es necesario contar con un modelo de la planta tal y como se requiere en [Shi-Yuan *et al*., 2002]. Además, a diferencia de [Lon-Chen y Huang-Yuan, 2007], su estructura sencilla permite su empleo en tiempo real.

El modelo NEFCON fue concebido originalmente para el control de sistemas lineales. En sistemas no lineales, NEFCON no asegura la convergencia durante el diseño del controlador, dada la pobre exploración de los estados de la planta durante el aprendizaje, limitando en consecuencia la construcción de reglas.

La propuesta planteada en este artículo se inspira en el trabajo de Chapeau-Blondeau y

Rousseau [Chapeau-Blondeau y Rousseau, 2005], en donde se demuestra que la adición de ruido Gaussiano permite detectar de forma óptima señales de modelos no lineales. La idea es explorar rápidamente el espacio de estados para crear un mayor número de reglas durante el proceso de aprendizaje, lo cual permitirá regular una planta no lineal.

Este trabajo está organizado como sigue: la sección 2 describe el modelo NEFCON y presenta la nomenclatura utilizada en secciones posteriores. La sección 3 define el error difuso extendido. La sección 4 describe los algoritmos de aprendizaje, tanto para los conjuntos difusos como para las reglas. La sección 5 discute el problema de exploración de los estados de la planta y cómo el ruido Gaussiano contribuye a la convergencia del algoritmo de aprendizaje. La sección 6 muestra el modelo matemático y la definición del error para la planta no lineal de la "pelota y el balancín". Posteriormente, la sección 7 presenta el controlador difuso obtenido como resultado del aprendizaje. Por último, en la sección 8 se mencionan las conclusiones de este trabajo.

2 El modelo NEFCON

El NEFCON (del inglés NEuro-Fuzzy CONtroller), es un modelo que combina las técnicas de lógica difusa con la arquitectura neuronal del Perceptrón. De esta manera, NEFCON utiliza los principios de aprendizaje neuronal para la construcción y optimización de un sistema difuso. NEFCON utiliza una señal de error *E* (error difuso extendido) que informa al modelo acerca del desempeño del controlador sobre la planta (aprendizaje por refuerzo), proporcionando así la magnitud y el sentido de los cambios en la estructura. La señal de error depende del sistema a controlar y representa el punto crítico en la generación del controlador difuso.

La figura 1, muestra un sistema NEFCON con dos variables de entrada, una variable de salida y cinco reglas. Las variables de entrada ξ_1 y ξ_2, son las variables de estado del sistema a controlar *S*. La salida del sistema NEFCON η, es la acción de control aplicada a *S*. Las unidades de la capa oculta representan las reglas difusas. La unidad R_3 representa la regla:

Si ξ_1 es $A_2^{(1)}$ **y** ξ_2 es $A_2^{(2)}$ **entonces** η es B_2, (1)

donde $A_2^{(1)}$, $A_2^{(2)}$ y B_2 son términos lingüísticos representados por los conjuntos difusos $\mu_2^{(1)}$, $\mu_2^{(2)}$ y ν_2.

En lugar de valores reales, como los usados en redes neuronales, los pesos de las conexiones en NEFCON son conjuntos difusos. En donde algunas conexiones pueden compartir pesos, como se ilustra en la figura 1. Por ejemplo, las conexiones de la unidad de entrada a R_1 y R_2 comparten el mismo peso $\mu_1^{(1)}$ (conjunto difuso) y las conexiones de R_4 y R_5 a la unidad de salida η, comparten el mismo peso ν_3. El algoritmo de aprendizaje debe realizar cambios idénticos en los pesos compartidos, ya que de lo contrario se perdería la integridad de la estructura Mamdani. También, las normas-T, las conormas-T y el procedimiento de defusificación, se eligen de tal forma que correspondan al sistema de inferencia tipo Mamdani [Yen, et al., 1999].

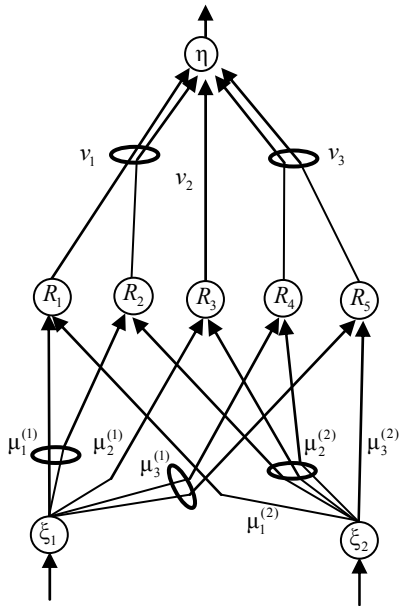


Fig. 1. Arquitectura del modelo NEFCON. Las elipses dibujadas alrededor de las conexiones indican los pesos compartidos

3 Conceptos Preliminares

Como se mencionó anteriormente, el modelo NEFCON utiliza para la construcción y optimización del controlador difuso, una señal de error que informa al modelo la calidad del desempeño del controlador sobre la planta, proporcionando de esta manera, la magnitud y sentido de los cambios en la estructura. Esta señal de error es llamada *error difuso extendido* (*E*), y es usada como señal de recompensa durante el proceso de aprendizaje por refuerzo.

Se considera al sistema como *controlado*, si las variables de estado de una planta *S* alcanzan y permanecen en sus valores ideales (punto de consigna), definidos por el vector $\mathbf{x}_{\text{opt}}$. De esta manera, se pueden distinguir dos casos en la evaluación del estado de la planta. En el primer caso, las variables de estado de la planta *S* se encuentran próximas al estado óptimo $\mathbf{x}_{\text{opt}} = (x_1^{(\text{opt})}, ..., x_n^{(\text{opt})})$, donde *n* representa el número de variables de estado que representan a la planta. Por lo tanto, cada variable de estado podría modelar su estado óptimo, mediante la definición de funciones de membresía que describan lingüísticamente su aproximación a él. En el segundo caso, las variables de estado pueden tener valores lejanos a su valor ideal. Sin embargo, el cambio en su magnitud indica si el sistema evolucionará o no, hacia un estado óptimo. A esto se le llama *estado compensatorio.*

Definición 1. Sea *S* un proceso con *n* variables de estado $\xi_1 \in X_1, ..., \xi_n \in X_n$, para el cual se conocen *s* estados compensatorios. De acuerdo al número de variables de estado existirán *n* conjuntos difusos que modelen el estado óptimo:

$$\mu_{\text{opt}^{(i)}}: X_i \to [0,1], \quad (i \in \{1,...,n\}) \tag{2}$$

y s relaciones difusas *n*-arias:

$$\mu_{\text{comp}}^{(j)} : X_1 \times ... \times X_n \to [0,1], \ (j \in \{1,...,s\}) \tag{3}$$

que representan el estado compensatorio. El estado actual está dado por $(x_1,...,x_n)$.

La *bondad difusa G* de la planta *S* está dada por:

$$G : X_1 \times \ldots \times X_n \to [0,1] \tag{4}$$

$$G(x_1,...,x_n) = g(G_{\text{opt}}(x_1,...,x_n), G_{\text{comp}}(x_1,...,x_n)) \tag{5}$$

donde g es una función apropiada que permite combinar los valores de bondad G_{opt} y G_{comp}, y que depende de la planta S. La bondad difusa *G* depende de la bondad difusa acumulativa G_{opt}:

$$G_{opt} : X_1 \times \ldots \times X_n \to [0,1] \tag{6}$$

$$G_{opt}(x_1,\ldots,x_n) = \text{norma-T}\left\{\mu^{(1)}{}_{opt}(x_1),\ldots,\mu^{(n)}{}_{opt}(xn)\right\} \tag{7}$$

y de la bondad difusa compensatoria G_{comp}

$$G_{comp} : X_1 \times \ldots \times X_n \to [0,1] \tag{8}$$

$$G_{comp}(x_{1,\ldots,}x_n) = norma - T\left\{\mu^{(1)}{}_{comp}(x_{1,\ldots,}x_n), \ldots, \mu^{(s)}{}_{comp}(x_{1,\ldots,}x_n)\right\} \tag{9}$$

La bondad difusa acumulativa G_{opt}, está basada en las descripciones del estado óptimo deseado y es calculada como la *norma-T* de las funciones de membresía, que modelan la proximidad de cada variable de estado. Las funciones de membresía $\mu^{(j)}{}_{\text{comp}}$, que describen el estado compensatorio de la bondad difusa G_{comp}, no están necesariamente definidas sobre todas las variables de estado, así que es posible que ésta dependa de sólo dos o más de ellas. La función *g*, que determina la bondad difusa (acumulativa) a partir de los valores de bondad G_{opt} y G_{comp}, debe ser seleccionada de tal forma que combine de manera adecuada ambos estados. Una función generalmente utilizada para combinar *g* es la operación mínimo.

Definición 2. Sea *S* un proceso con *n* variables de estado $\xi_1 \in X_1, ..., \xi_n \in X_n$ con una bondad difusa *G*. Si *S* es controlada por un sistema

NEFCON y $(x_1,...,x_n)$ es el estado actual de *S*, entonces el error difuso *f* del sistema NEFCON se define como:

$$f : X_1 \times ... \times X_n \rightarrow [0,1] \quad (10)$$

$$f\,(x_1,...,x_n) = 1 - G(x_1,...,x_n) \quad (11)$$

Para implementar el aprendizaje se debe calcular cómo cada regla es modificada, dependiendo de su desempeño en función del estado actual de la planta *S*. Una regla se "premia" aumentando su influencia en la salida del control, siempre y cuando la señal por refuerzo muestre un resultado positivo. Por el contrario, una regla se "castiga" disminuyendo su influencia, de manera que contribuya menos a la salida de control.

Para decidir si la contribución de una regla tiene una influencia positiva o negativa en el control de la planta, es necesario conocer el signo de la salida del controlador. Si se asume que para un estado óptimo (todas las variables de estado tienen sus valores ideales), la salida del controlador es cero, como a menudo sucede, entonces se puede saber si es necesario que el controlador aplique una señal positiva o negativa al proceso, dependiendo de su desviación con respecto al estado óptimo.

Definición 3. El error difuso extendido *E,* de un sistema NEFCON se define por:

$$E(x_1,...,x_n) = \mathrm{sgn}\,(\eta_{opt}) \cdot f(x_1,...,x_n) \quad (12)$$

donde $(x_1,...,x_n)$ es el estado actual, *f* es el error difuso, y sgn(η_{opt}) es el signo de la señal de control óptima.

En lugar de calcular el error difuso como en la definición 3, es posible usar reglas difusas. Para hacer esto, el espacio de estados y el intervalo del error [-1,1] deben ser divididos en funciones de membresía [Nauck y Kruse, 1992].

4 Algoritmos de aprendizaje

Los algoritmos de aprendizaje se basan en el valor de *E*, que se describe en la definición 3. También son conocidos como algoritmos de *retropropagación del error difuso* ("fuzzy error backpropagation"). Estos algoritmos pueden ser considerados como una clase de aprendizaje reforzado sin crítico adaptativo [Sutton y Barto, 1998]. En donde el error es usado como una señal negativa de refuerzo. El proceso de aprendizaje, es similar al utilizado en el *algoritmo de retroprogacion del error* (backpropagation) para perceptrones multicapa en donde, el valor del error se propaga hacia atrás en toda la red, tomando como inicio la capa de salida. De esta manera, cada unidad toma una función de la señal de error y la utiliza para modificar localmente sus pesos (funciones de membresía). Los algoritmos de aprendizaje de NEFCON se dividen en dos tipos: *aprendizaje de conjuntos difusos* (aprendizaje de parámetros) y *aprendizaje de reglas* (aprendizaje de estructura).

4.1 Algoritmo de aprendizaje de conjuntos difusos

El algoritmo de aprendizaje de conjuntos difusos modifica las funciones de membresía de NEFCON, a fin de mejorar el desempeño del controlador. Se asume la existencia de una base de reglas adecuada y que el inadecuado desempeño del controlador, se deba a una representación no óptima de los conjuntos difusos usados en las reglas. De este modo, sólo se cambian aquellas funciones de membresía que pertenezcan a reglas cuyo grado de activación haya sido mayor que cero (reglas disparadas).

Para premiar (incrementar) la contribución de una regla en la respuesta del controlador, se debe aumentar el grado de membresía en el estado actual $\xi = (x_1,...,x_n)$ y el valor numérico t_r aportado por la regla a la salida η. El grado de satisfacción de la regla aumenta, si se incrementa el soporte (intervalo de valores de la variable) de las funciones de membresía $\mu_{jr}^{(i)}$ del antecedente de dicha regla. Por otro lado, también el valor numérico de t_r se incrementa, si el soporte del conjunto difuso ν_{jr} del consecuente se reduce. Para castigar (disminuir) la contribución de una regla, debe hacerse exactamente lo contrario al caso anterior; esto es, disminuir el soporte de los conjuntos difusos del antecedente y aumentar los del consecuente. La figura 2 muestra las configuraciones de las funciones de membresía antes y después del proceso de premiar o castigar a la regla R_r. En esta figura, es posible observar que el aumento y reducción de los soportes en el antecedente y en el consecuente respectivamente, corresponden a un

incremento en el valor numérico t_r de la salida η. En otras palabras, se fortalece la influencia de la regla R_r en la señal de control.

Considerando que se pretende controlar un sistema *S* de *n* variables de estado, con una variable de control η, a partir de *k* unidades de regla $R_1,...,R_k$, entonces es posible utilizar iterativamente el algoritmo de aprendizaje que adaptará las funciones de membresía. El ciclo de adaptación se repite hasta que *E* alcance un valor mínimo deseado o bien, se haya aplicado un número de ciclos fijo. A continuación, se muestran los pasos que se siguen durante una iteración:

1. Se calcula la salida o_η del sistema NEFCON usando el estado actual, se aplica o_η a *S* y se determina el nuevo estado.

2. Se determina *E* del nuevo estado producido por *S*.

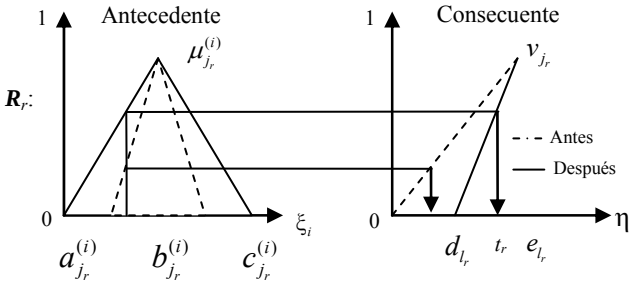


Fig. 2. Modificación del soporte de las funciones de membresía

3. Se determina para cada unidad de regla R_r su contribución t_r al valor de salida o_η y se calcula el error difuso de cada regla E_{Rr}, definido por:

$$E_{Rr} = o_{Rr} \cdot E \cdot \operatorname{sgn}(t_r) \tag{13}$$

para cada unidad de regla R_r con $r \in \{1,...,k\}$.

4. Se modifican los parámetros de las funciones de membresía en los consecuentes v_{jr}, $j \in \{1,...,q\}$ y $r \in \{1,...,k\}$, utilizando:

$$\Delta d_{jr} = \sigma \cdot E_{Rr} \cdot (e_{jr} - d_{jr}) \tag{14}$$

donde *q* representa el número de funciones de membresía en los que se dividió el universo de discurso de la salida o_η, mientras que *d* y *e*, son los puntos que definen la función de membresía para el consecuente, como se muestra en la figura 2. El intervalo $0 > \sigma \leq 1$, representa los valores posibles para σ (índice de aprendizaje) y determina la rapidez con la que el algoritmo realiza las modificaciones.

5. Se determinan los cambios en los parámetros de las funciones de membresía $\mu_{jr}^{(i)}$ $i \in \{1,...,n\}$, $j \in \{1,...,p\}$ y $r \in \{1,...,k\}$ de los antecedentes de las reglas:

$$\Delta a_r^{(i)} = \sigma \cdot E_{Rr} \cdot (b_{jr}^{(i)} - a_r^{(i)}) \tag{15}$$

$$\Delta c_{jr}^{(i)} = -\sigma \cdot E_{Rr} \cdot (c_{jr}^{(i)} - b_{jr}^{(i)}) \tag{16}$$

donde *p* representa el número de funciones de membresía en los que se dividió el universo de discurso para cada una de las *n* variables de estado, mientras que a b y c son los puntos que definen la función de membresía para el antecedente, como se muestra en la figura 2.

4.2 Aprendizaje de reglas

El modelo NEFCON permite aprender en dos partes la estrategia de control. La *primera parte*, tiene por objetivo aprender la base de reglas. La *segunda parte*, optimiza las reglas ya aprendidas (véase la sección 4.1). Ambas partes, se basan en *E* para lograr sus objetivos.

Los algoritmos utilizados para aprender una base de reglas se pueden dividir en tres clases: *aleatorios*, *vacíos* o *completos*. NEFCON posee algoritmos sólo para las dos últimas clases. En este trabajo, se considera un algoritmo vacío llamado incremental, el cual, se inicia sin ninguna regla y estas se van agregando o incrementando. Este algoritmo resulta conveniente por ser menos costoso en términos de carga computacional.

4.3 Algoritmo incremental

El problema en el aprendizaje de reglas se origina dado que se desconoce la salida correcta del controlador para cada estado. En consecuencia, no es posible generar reglas a partir de la observación de los estados. Sin embargo, se puede tratar de inferir una salida conveniente mediante *E.*

El algoritmo se divide en tres fases. En la *primera fase,* se construye el antecedente de las reglas y se infiere el valor de salida a partir de *E.* En la *segunda fase,* se optimiza la base de reglas según *E*. Por último, en la *tercera fase* se contempla la adaptación de las funciones de membresía aprendidas.

Si se considera ahora una planta *S,* con *n* variables de estado $\xi = (x_1,...,x_n)$, las cuales están divididas por *p* conjuntos difusos cada una y una variable de control $\eta \in [y_{min}, y_{max}]$, dividida en *q* conjuntos difusos. Supóngase además, que se tiene un sistema NEFCON con *k'* unidades de reglas predefinidas, donde *k'* puede ser cero. El algoritmo de aprendizaje de reglas incremental estaría dado por los siguientes pasos:

1. Para un número fijo de repeticiones m_1, se llevan a cabo los siguientes pasos:

   a) Para el vector de entrada actual $(x_1,...,x_n)$, encontrar aquellos conjuntos difusos $\mu^{(1)},...,\mu^{(n)}$ para los cuales se cumpla que:

   $$(\forall ij)(\ \mu^{(i)}(x_i) \geq \mu_j^{(i)}(x_i)) \tag{17}$$

   donde $i \in \{1,...,n\}$ y $j \in \{1,...,p\}$.

   b) Si no se tiene ninguna regla con el antecedente de la forma:

   Si ξ_1 es $\mu^{(1)}$ **y** ... **y** ξ_n es $\mu^{(n)}$ (18)

   Entonces, encontrar el conjunto difuso, ν tal que se cumpla que:

   $$(\forall k)\ (\nu(\eta) \geq \nu_k(\eta)) \tag{19}$$

   tal que $k \in \{1,...,q\}$, donde el valor heurístico de salida η, está determinado por

$$\eta = \begin{cases} m + |E| \cdot (y_{\max} - m) \text{ si } E \geq 0 \\ m - |E| \cdot (m - y_{\min}) \text{ si } E < 0 \end{cases} \qquad m = \frac{y_{\max} - y_{\min}}{2} \tag{20}$$

   c) Se incorpora la regla:

Si ξ_1 es $\mu^{(1)}$ **y** ...**y** ξ_n es $\mu^{(n)}$ **entonces** η es ν (21)

al controlador.

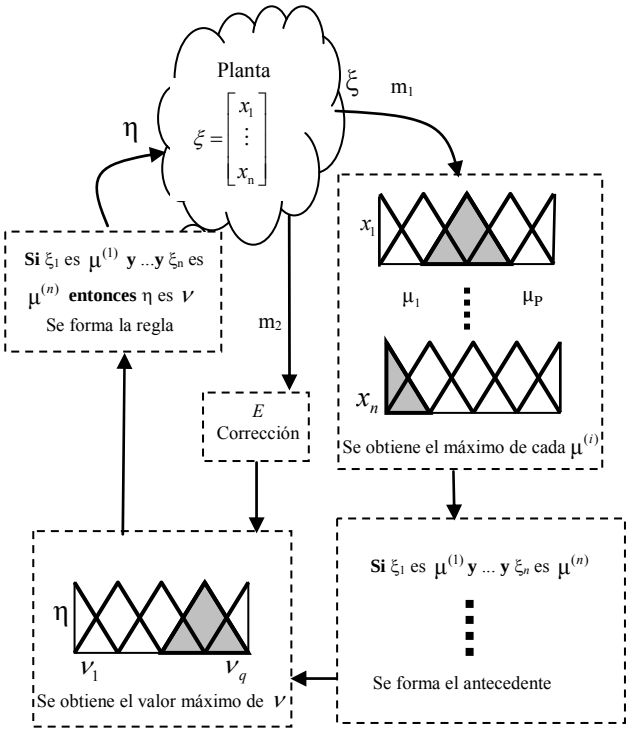


Fig. 3. Ciclo de aprendizaje de la estructura del controlador realizada por NEFCON

2. Para un número fijo de repeticiones m_2, se llevan a cabo los siguientes pasos:

   a) Se propaga el vector de entrada actual a través del controlador. Se calcula la contribución t_r de cada regla R_r, al valor de salida η. A partir de t_r, se estima la contribución ideal de cada regla mediante:

$$t_r^* = t_r + \sigma \cdot \eta \cdot E \quad (22)$$

donde σ, representa el índice de aprendizaje. La contribución ideal t_r^*, representa un ajuste de la contribución t_r de la regla. El ajuste se realiza mediante una estimación del error, producido durante la asignación de funciones de membresía.

b) Para cada unidad de regla R_r, se determina la nueva función de membresía ν_r del consecuente, de manera que se cumpla:

$$(\forall k)\ (\nu_r(t_r^*) \geq \nu_k(t_r^*)), \quad (23)$$

donde $k \in \{1,...,q\}$. Lo anterior, puede ser considerado como un reajuste de las funciones de membresía.

3. Se eliminan todas las reglas que no han sido utilizadas, en más de un porcentaje predefinido durante el aprendizaje. En otras palabras, serán eliminadas aquellas reglas que sólo fueron usadas ocasionalmente. Este fenómeno, sucede normalmente al inicio del aprendizaje.

4. Se utiliza el algoritmo de aprendizaje de conjuntos difusos que se discutió en la sección 4.1.

La figura 3 muestra el proceso en que NEFCON construye la base de reglas del controlador. De esta figura pueden distinguirse 2 tipos de flujos de información: uno exterior que denota el proceso de creación de reglas desempeñado durante las primeras repeticiones m_1 y otro interior ejecutado en las siguientes repeticiones m_2, que representan la corrección de los consecuentes a partir de la evaluación de *E*.

5 Exploración de los estados de la planta

NEFCON permite construir un controlador difuso a partir de los estados explorados durante el aprendizaje. Es muy importante considerar que para obtener una adecuada base de reglas, debe asegurarse que el espacio de estados sea explorado ampliamente durante el proceso de aprendizaje. En el caso de plantas lineales, es posible lograrlo aumentando deliberadamente el número de ciclos de aprendizaje y/o alargando la duración de los mismos.

Como se mencionó en la sección 4, durante la construcción del controlador, se añaden reglas en función de la relación entre el estado actual de la planta y la salida inferida mediante *E.* El proceso de añadir reglas en el instante *k* y usarlas en el ciclo siguiente *k+1,* depende de la cantidad de estados que puedan ser explorados.

El modelo NEFCON fue concebido originalmente para el control de sistemas lineales. En sistemas no lineales, NEFCON podría no converger durante el aprendizaje. Lo anterior, se debe a la pobre exploración de los estados de la planta y que en consecuencia limitaría la construcción de reglas. Por esta razón, para un sistema no lineal, el aumentar los ciclos de aprendizaje no seria suficiente.

La solución propuesta en este artículo, se inspira en el trabajo de Chapeau-Blondeau y Rousseau [Chapeau-Blondeau y Rousseau, 2005], en donde se demuestra que al añadir ruido Gaussiano, es posible detectar de forma óptima señales en modelos no lineales. La idea consiste en explorar más ampliamente el espacio de estados, con el objetivo de crear un mayor número de reglas durante el proceso de aprendizaje, permitiendo regular una planta no lineal. Un análisis riguroso de las pruebas de convergencia obtenidas al añadir ruido Gaussiano puede ser consultado en [Chapeau-Blondeau y Rousseau, 2005].

6 Modelo matemático y definición del error para el sistema de la "pelota y el balancín"

El sistema dinámico no lineal de la "pelota y el balancín", es frecuentemente utilizado como planta de referencia para evaluar el rendimiento de un controlador. Ejemplos de ello, se encuentran en [Perez-Cisneros, *et al.*, 2004]; en donde se presenta la aplicación de diferentes controladores clásicos a esta planta y en [Shi-Yuan *et al.*, 2002], donde se utiliza la misma planta para probar enfoques de control inteligente. A continuación, se describe el modelo matemático del sistema de la "pelota y el balancín" y el uso de *E*, implementado como señal

de refuerzo para el aprendizaje de reglas y adaptación de funciones de membresía.

6.1 El modelo matemático

La figura 4 muestra al sistema dinámico de la "pelota y el balancín". La posición *r* de la pelota, la cual se desplaza libremente sobre el balancín B**,** es controlada mediante el giro θ del servomotor M, el cual al encontrarse acoplado al balancín B mediante la barra C, permite modificar la inclinación α del balancín. El comportamiento dinámico del sistema es modelado mediante la ecuación Lagrangiana de movimiento definida como:

$$(\frac{J}{Rb}+m)\ddot{r}+m\cdot gr\cdot\sin(\frac{d}{L}\theta)-m\cdot r\frac{d^2}{L^2}(\dot{\theta})^2 \qquad (24)$$

donde *J* el momento de inercia de la pelota, *m* es la masa de la pelota, *Rb* el radio de la pelota, $\ddot{r}$ es la aceleración de la pelota, *gr* es la gravedad, *d* la posición de la barra y θ es el ángulo de compensación (salida del controlador). Los parámetros del sistema utilizados en este trabajo se resumen en la Tabla 1.

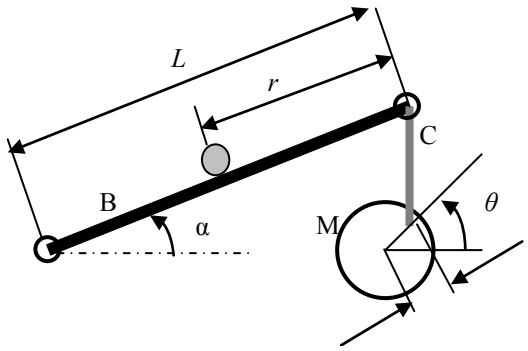


Fig. 4. Descripción del sistema de la "pelota y el balancín"

Tabla 1. Parámetros del sistema de la "pelota y el balancín"

Parámetro	Valor
m	0.11Kg
Rb	0.015m
d	0.03m
gr	9.8m/s^2
L	1m
J	9.99e-6 Kgm^2

6.2 Definición del error extendido *E*

El problema de control para este sistema, consiste en mantener la posición de la pelota en un punto de referencia del balancín B. Para esto, es necesario calcular el error *e*, que se define como:

$$e = referencia - r \qquad (25)$$

En donde *referencia,* es el punto a seguir por el controlador y *r* la posición real de la pelota. Considerando lo anterior, el sistema tendrá dos variables de estado: el error e y el cambio del error $\dot{e}$, este último definido como:

$$\dot{e} = e - e\_ant \qquad (26)$$

donde *e_ant,* representa el error anterior.

Para el caso en que las variables e y $\dot{e}$, alcanzan aproximadamente su valor ideal (estado óptimo), se definen las funciones de membresía que modelan su proximidad:

$$\mu_{opt}^{(1)}(e)=\begin{cases}1-\dfrac{|e|}{0.2}, & \text{si } |e|\le 0.2\\ 0, & \text{en otro caso}\end{cases} \qquad (27)$$

$$\mu_{opt}^{(2)}(\dot{e})=\begin{cases}1-\dfrac{|\dot{e}|}{0.5}, & \text{si } |\dot{e}|\le 0.5\\ 0, & \text{en otro caso}\end{cases} \qquad (28)$$

Si e no es cero, pero el valor de $\dot{e}$ indica que en el estado siguiente se reducirá el valor de e, entonces se tendrá un estado compensatorio, el cual puede ser definido por:

$$\mu_{comp}(e,\dot{e}) = \begin{cases} 1 - |10 \cdot e + \dot{e}|, & \text{si } |10 \cdot e + \dot{e}| \le 1 \\ 0, & \text{en otro caso} \end{cases} \quad (29)$$

A partir de las ecuaciones 7 y 8, se calcula la bondad difusa acumulativa G_{opt}

$$G_{opt}(e,\dot{e}) = \min\left\{\mu_{opt}^{(1)}(e), \mu_{opt}^{(2)}(\dot{e})\right\}, \quad (30)$$

y la bondad difusa compensatoria G_{comp}

$$G_{comp}(e,\dot{e}) = \mu_{comp}(e,\dot{e}) \quad (31)$$

Considerando la ecuación 5, se obtiene la expresión para la bondad difusa, tal que:

$$G(e,\dot{e}) = \begin{cases} G_{opt}(e,\dot{e}), & \text{si sgn}(e)=\text{sgn}(\dot{e}) \\ G_{comp}(e,\dot{e}), & \text{en otro caso} \end{cases} \quad (32)$$

De esta manera puede calcularse a partir de las ecuaciones 11 y 12, el error difuso y *E*, tal que:

$$f(e,\dot{e}) = 1 - G(e,\dot{e}) \quad (33)$$

$$E(e,\dot{e}) = \text{sgn}(e) \cdot f(e,\dot{e}) \quad (34)$$

7 Resultados del aprendizaje

El aprendizaje se realizó en seis ciclos de 20 segundos. Cuatro de ellos correspondieron al aprendizaje de reglas, mientras que los dos restantes a la optimización de los conjuntos difusos. Un ciclo consiste de 3000 iteraciones, considerando m_1=1500 y m_2=1500. La optimización de parámetros también usó 1500 iteraciones. Durante el aprendizaje de reglas, se agregó ruido aditivo Gaussiano de media cero y desviación estándar 0.1 a los estados. Se eliminaron aquellas reglas que no fueron utilizadas en más de un 2%. Como condición inicial fue seleccionada *r*=0.3. El punto de referencia fue 0.5 y la simulación de todo el sistema utiliza un intervalo de muestreo de 0.5 segundos. Para realizar la simulación de este trabajo, se utilizó la implementación en MatLAB del sistema NEFCON sugerida en [Nürnberger, *et al.*, 1999].

Para el aprendizaje de reglas, se dividió a cada variable de entrada e y $\dot{e}$ en tres conjuntos difusos, etiquetados como: **ne**, **ce** y **po**, mientras que para la variable de salida θ, se dividió en 5 conjuntos difusos, etiquetados como: **ne**, **nm**, **ce**, **pm** y **po**.

Como resultado del aprendizaje, se obtuvieron las siguientes seis reglas:

1. **Si** (e es ce) y ($\dot{e}$ es ce) **entonces** (θ es ce)
2. **Si** (e es ce) y ($\dot{e}$ es ne) **entonces** (θ es ce)
3. **Si** (e es ce) y ($\dot{e}$ es po) **entonces** (θ es pm)
4. **Si** (e es ne) y ($\dot{e}$ es ne) **entonces** (θ es ne)
5. **Si** (e es ne) y ($\dot{e}$ es ce) **entonces** (θ es ne)
6. **Si** (e es ne) y ($\dot{e}$ es po) **entonces** (θ es nm)

La figura 5 muestra la evolución del proceso de aprendizaje obtenido al aplicar el algoritmo NEFCON. Como puede verse en esta figura, en los dos primeros ciclos el efecto de añadir ruido se traduce en una mejor exploración del espacio de estados, así se puede formar una regla para cada combinación de variables de entrada. En los dos siguientes ciclos de aprendizaje (3 y 4), resulta evidente como el proceso de depuración de reglas trae consigo una mejora en la respuesta dinámica del sistema, disminuyendo la magnitud de los sobretiros. En la figura 6, se muestra la superficie de control obtenida con la base de reglas aprendidas por el sistema NEFCON.

Por último, en los ciclos 5 y 6, la base de reglas aprendidas es optimizada mediante la modificación de las funciones de membresía (véase 4.1), de tal manera que su efecto se percibe al atenuar el sobretiro. Las figuras 7a, 8a y 9a muestran las funciones de membresía iniciales utilizadas para e, $\dot{e}$ y θ respectivamente. Estas fueron modificadas durante el proceso de aprendizaje, quedando finalmente como muestran las figuras 7b, 8b y 9b.

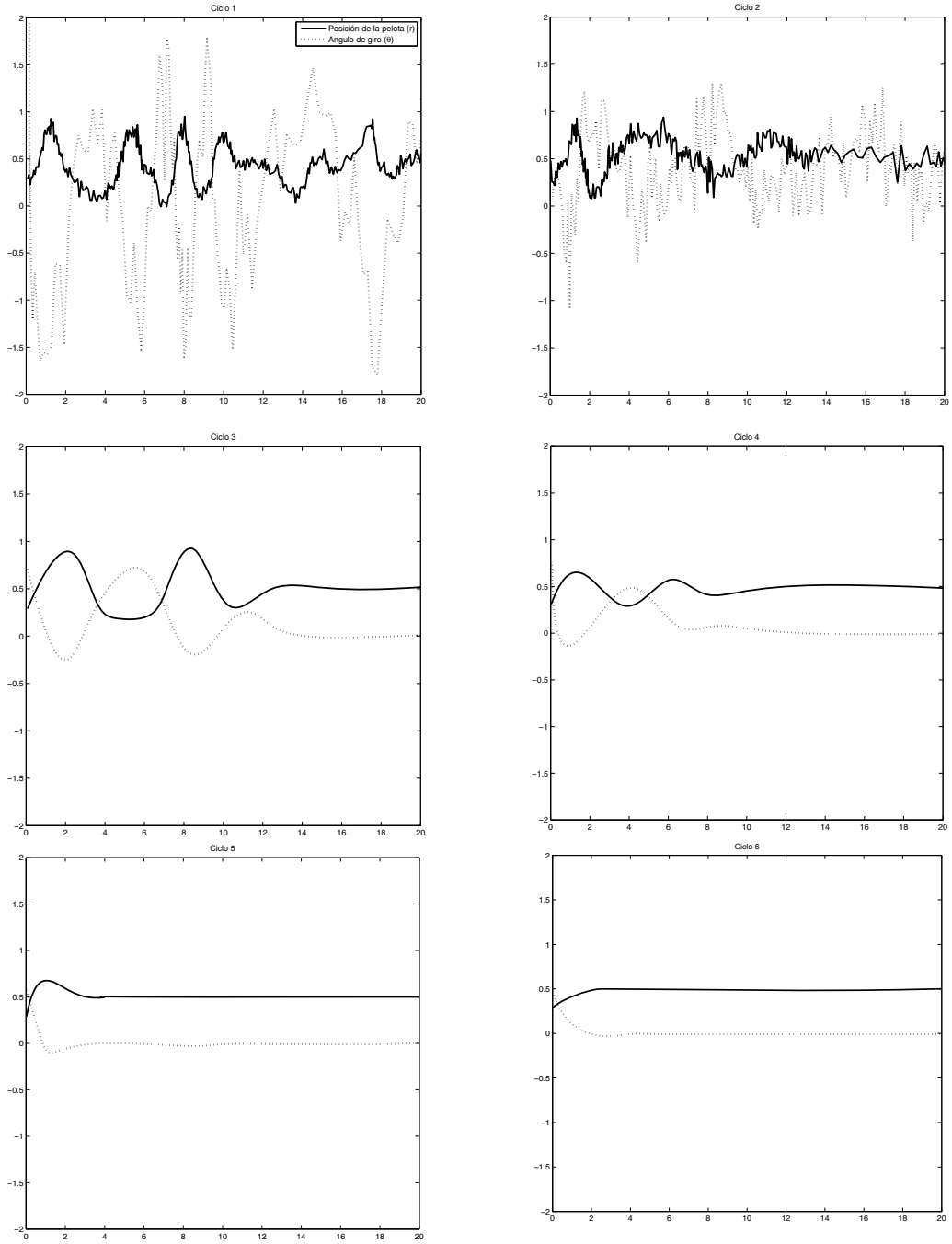


Fig. 5. Proceso de 6 ciclos de aprendizaje. La señal continua corresponde a la posición de la pelota r , mientras que la señal punteada corresponde al ángulo de compensación θ del servomotor (salida del controlador)

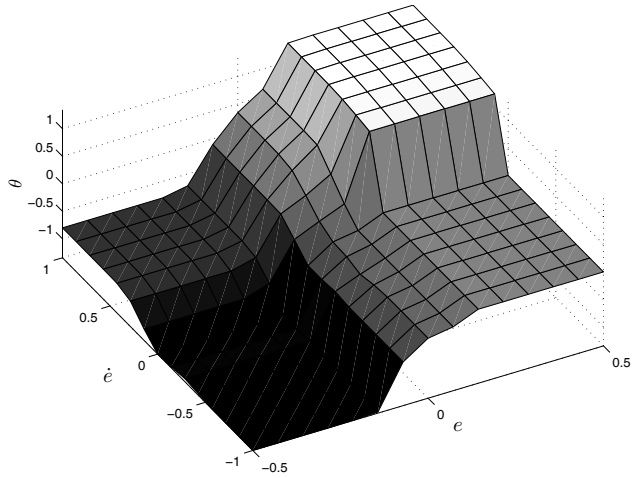
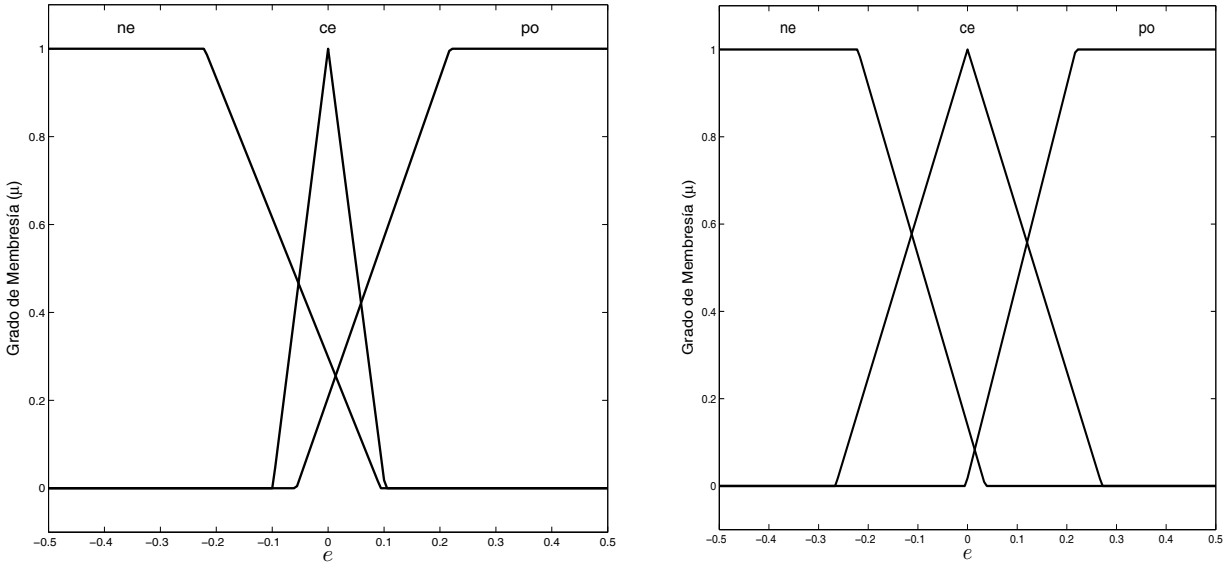


Fig. 6. Superficie de control obtenida con la base de reglas aprendidas



(a) (b)

Fig. 7. (a) Funciones de membresía iniciales y (b) funciones de membresía modificadas por el aprendizaje para e

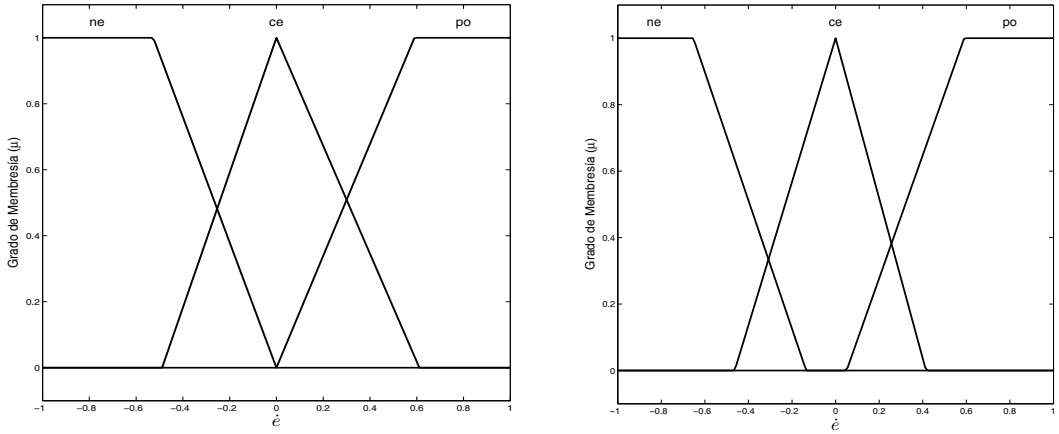


Fig. 8. (a) Funciones de membresía iniciales y (b) funciones de membresía modificadas por el aprendizaje para $\dot{e}$

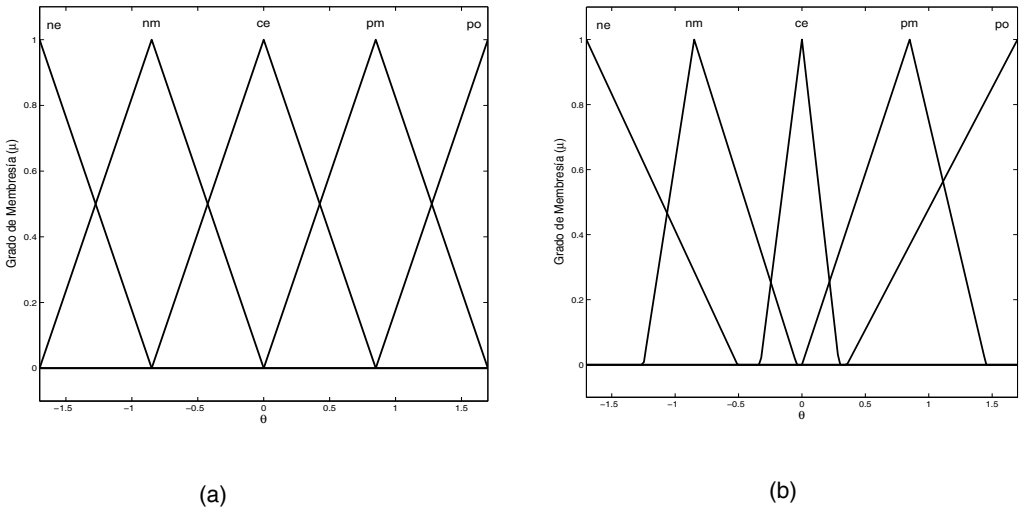


Fig. 9. (a) Funciones de membresía iniciales, (b) funciones de membresía modificadas por el aprendizaje, para θ

8 Conclusiones

En este trabajo se propone el uso del modelo NEFCON para generar y optimizar un controlador difuso capaz de controlar una planta no lineal.

NEFCON permite a partir de un algoritmo de aprendizaje, construir las reglas y optimizar los parámetros del controlador, al utilizar una señal de refuerzo obtenida por interacción directa con el sistema. En este artículo se presenta la aplicación del modelo NEFCON en el control de sistemas dinámicos no lineales. Lo anterior se hizo posible

mediante la adición de ruido Gaussiano a las variables de estado de la planta. El ruido añadido, al permitir una rápida exploración de los estados, asegura una mejor convergencia del aprendizaje durante el diseño del controlador. Con el objetivo de probar los alcances de esta propuesta, se utilizó como planta el sistema de la "pelota y el balancín". Se comprobó que de no añadirse ruido Gaussiano durante el aprendizaje, el modelo NEFCON es incapaz de construir un controlador, que regule la planta.

Para el control del sistema de la "pelota y el balancín", se definieron particiones iniciales de los conjuntos difusos para describir tanto las variables de entrada como las de salida. Después, se aplicó el algoritmo de aprendizaje incremental para el aprendizaje de reglas. Por último, se mejoró el desempeño del controlador mediante la optimización de los conjuntos difusos. El resultado comprende un controlador difuso compuesto por seis reglas y por conjuntos difusos optimizados, los cuales fueron modificados respecto a sus particiones iniciales.

El controlador generado automáticamente a partir del aprendizaje, fue capaz de controlar al sistema de la "pelota y el balancín" sin experimentar sobretiro y con un tiempo de establecimiento dentro del intervalo de 2 a 3 segundos. Los experimentos sugieren que es posible generar controladores para el sistema de la "pelota y el balancín" con la utilización de tan solo 3 ciclos de aprendizaje. Sin embargo, aunque estos controladores permiten someter a la variable de control después de 16 segundos, muestran una respuesta dinámica inadecuada (sobretiros de más del 40%). Por lo que, para este sistema es recomendable la utilización de más de 5 ciclos de aprendizaje, si se desea atenuar el sobretiro y obtener tiempos de establecimiento menores.

La principal ventaja del modelo NEFCON, en comparación con otras propuestas, es que su diseño se reduce a plantear cualitativamente el error actual de la planta a controlar (señal de refuerzo), por lo que no es necesario contar con un modelo de la planta tal y como se requiere en otros enfoques difusos [Shi-Yuan *et al*., 2002]. Además, a diferencia de gran parte de los enfoques inteligentes [Lon-Chen y Huang-Yuan, 2007], su estructura sencilla permite su empleo en tiempo real. Como desventaja, NEFCON no permite explícitamente definir índices de desempeño del controlador, debido a sus características heurísticas, al contrario de otros modelos de control no lineal [Perez-Cisneros, *et al*., 2004] u otras técnicas inteligentes [Lon-Chen y Huang-Yuan, 2007]. Por lo que, NEFCON no se recomienda para aquellos casos, en donde se deban satisfacer parámetros de respuesta del controlador predefinidos, como lo son, por ejemplo, tiempos de establecimiento y valores de sobretiro máximo.

Referencias

1. **Brown, M. & Harris, C.J. (1994)**. Neurofuzzy Adaptive Modelling and Control. Hemel Hempstead: Prentice Hall.
2. **Cuevas, E. & Zaldívar, D. (2006).** *Sistemas de Control Neurodifuso*. Göttingen: Cuvillier Verlag.
3. **Dominguez, J. A., Damper, R. I. & Harris, C. J. (2004).** Adaptive neurofuzzy control of a robotic gripper with on-line machine learning. *Robotics and Autonomous Systems*, 48 (2), 93–110.
4. **Gonzalez, M. A. & Tang, Y. (2007).** A new recurrent neurofuzzy network for identification of dynamic systems. *Fuzzy Sets and Systems*, 158 (10), 1023-1035.
5. **Gupta, M.M. & Sinha, N.K. (1999).** *Soft Computing and Intelligent Systems: Theory and Applications*. Amsterdam: Elsevier.
6. **Hangos, K.M., Lakner, R. & Gerzson, M. (2001).** *Intelligent Control Systems: An Introduction with Examples*. Netherlands: Kluwer Academic Publishers.
7. **Harris, C., Hong, X. & Gan, Q. (2002).** *Adaptive Modelling, Estimation and Fusion from Data: A Neurofuzzy Approach*. Berlin: Springer-Verlag.
8. **Haykin, S. (1999).** *Neural Networks: A Comprehensive Foundation*. Upper Saddle River, NJ: Prentice Hall.
9. **Jang, J.-S.R. (1993).** ANFIS: Adaptive-Network-based Fuzzy Inference System, *IEEE Transactions on Systems, Man, and Cybernetics* 23 (3), 665-685
10. **Jang, J-S.R., Sun, C-T. & Mizutani, E. (1997).** *Neuro-Fuzzy Modeling and Soft Computing: A Computational Approach to Learning and Machine Intelligence*, Englewood Cliffs. NJ: Prentice-Hall.
11. **Lon-Chen, H. & Hung-Yuan, C. (2007).** Decoupled control using neural network-based sliding-mode controller for nolinear systems, *Expert Systems with Application*, 32 (4), 1168-1182.
12. **Moon, G. & Jin, S. (2002).** Universal approximation by hierarchical fuzzy system with constraints on the fuzzy rule, *Fuzzy Sets and Systems*, 130 (2), 175-188.
13. **Nauck, D. & Kruse, R. (1992).** A Neural-Fuzzy Controller Learning by Fuzzy Error Propagation, *North American Fuzzy Logic Processing Society*, Puerto Vallarta, Mexico, 388-397.
14. **Nauck, D. (1994).** Fuzzy Perceptron as a Generic Model for Neuro-Fuzzy Approaches, *Fuzzy Systeme'94, 2nd*

GI-Workshop Siemens Corporation, Munich, Germany. 32-37.
15. **Nauck, D., Kruse, R. & Klawonn, F. (1997).** *Foundations on Neuro-Fuzzy Systems,* Chichester: Wiley.
16. **Nauck, D., Klawonn, F. & Kruse, R. (1996).** *Neuronale Netze und Fuzzy-Systeme (2nd extented ed.)*, Wiesbaden: Vieweg Verlag.
17. **Nürnberger, A., Nauck, D. & Kruse, R. (1999).** Neuro-Fuzzy Control Based on the NEFCON-Model (2007), retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.28.6827
18. **Perez-Cisneros, M.A. & Wellstead, P. (2004).** Sistema de Balancín y Pelota: Principios Básicos*, (notas de aplicación). México: División de electrónica y computación, Universidad de Guadalajara*.
19. **Rousseau, D. (2005).** Constructive action of additive noise in optimal detection, *International Journal Bifurcation Chaos,*15 (9), 2985–2994.
20. **Shi-Yuan, C., Fang-Ming, Y. & Huang-Yuan, C. (2002).** Decoupled fuzzy controller design with single-input fuzzy logic. *Fuzzy Sets and Systems*, 129 (3), 335-342.
21. **Sung-Kwun, O., Seok-Beom, R. & Pedrycz, W. (2007).** IG-based genetically optimized fuzzy polynomial neural networks with fuzzy set-based polynomial neurons, *Neurocomputing*, 70 (16-18), 2783-2798.
22. **Sutton, R., & Barto, A. G.(1998).** *Reinforcement Learning: An Introduction*. Cambridge: MIT Press.
23. **Yen, J., & Langari, R. (1999).** *Fuzzy Logic, Intelligence, Control, and information*, Upper Saddle River: Prentice-Hall.

Erik Cuevas Jiménez

Estudió la carrera de Ingeniería Electrónica en la Universidad de Guadalajara, posteriormente la maestría en Electrónica Industrial en el ITESO y el doctorado en la Universidad Libre de Berlín, Alemania. Sus áreas de interés son: la visión artificial, robótica y control automático.

Daniel Zaldívar Navarro

Es Ingeniero en Comunicaciones y Electrónica por la Universidad de Guadalajara, posteriormente obtuvo la maestría en Electrónica Industrial en el ITESO, México y el doctorado en la Universidad Libre de Berlín, Alemania. Sus áreas de interés son: la robótica de humanoides, visión artificial y control automático.

Marco Antonio Pérez Cisneros

Es Ingeniero en Comunicaciones y Electrónica por la Universidad de Guadalajara en 1995, obtuvo e grado me maestría en 2001 en Electrónica industrial en el ITESO. En el 2005 recibió el grado de doctor en Robótica por UMIST en el Reino Unido. Sus áreas de interés son: el control visual y el control de robots manipuladores.

Ernesto Tapia

Obtuvo el grado de maestría en Ciencias en Matemáticas Computacionales por el CIMAT, México y el grado de doctor por la universidad Libre de Berlín (FU), Alemania. Actualmente es profesor asociado en el departamento de ciencias computacionales de la FU. Sus áreas de interés son: reconocimiento de patrones, sistemas de aprendizaje e inteligencia artificial.