IMPLEMENTACIÓN DE NEUROCONTROLADORES EN LÍNEA. TRES CONFIGURACIONES, TRES PLANTAS

Resumen— En este artículo se implementa el algoritmo de aprendizaje backpropagation en línea para el entrenamiento de redes neuronales tipo feedforward. Se implementan tres neurocontroladores para tres sistemas, los cuales son: el circuito resistivo capacitivo, RC, un motor de corriente continua, emulado electrónicamente, y un sistema esfera-tubo. La primera estrategia que se prueba para todos los sistemas es un controlador Proporcional, Integral, Derivativo, (PID clásico), el cual es utilizado para comparar el desempeño de los otros controladores. El primer neurocontrolador comparte la responsabilidad de comandar al sistema con un PID; el siguiente es entrenado en línea y trabaja solo; y por último se encuentra un controlador PID neuronal, el cual cambia las ganancias del PID para hacerlo adaptable a la dinámica de la planta. El control se realiza en tiempo real, por medio de Simulink, junto con una tarjeta de adquisición de datos PCI 6024E. En el desarrollo del artículo se muestran los resultados obtenidos de cada sistema.

Palabras clave— Entrenamiento en línea, redes neuronales, neurocontrol, control híbrido, control en tiempo real.

Abstract— In this paper we develop a backpropagation learning algorithm for feedforward neural networks trained online. Three neurocontrollers are designed for three systems. Those systems are a RC circuit, a DC motor (electronically emulated) and a sphere-tube system. The first implemented strategy is a standard PID controller, which is used in order to compare the performance of the neurocontrollers. The first neurocontroller leads the system in parallel with a PID, the next one is trained online to work alone and the last one is a neural PID, which looks for making the controller adaptable to the dynamic of the plant trough changes on the PID gains. The control is done in real time using Simulink and a data acquisition card, it is a PCI 6024E. In addition, as the paper progresses, we present results for each system.

Index Terms—Online Training, Neural Networks, Neurocontrol, Hybrid Control, Real Time Control.

I. Introducción

En su forma más básica una red neuronal artificial está compuesta por neuronas y conexiones entre neuronas. El trabajo de cada neurona es comprimir el valor de todas las entradas en un único valor de salida, por medio de lo que se conoce como función de activación. A la conexión entre cada par de neuronas se le asocia un peso, el cual puede indicar el grado de importancia que tiene esa conexión en comparación con las demás.

Las redes neuronales fueron diseñadas para emular a las neuronas biológicas, y han servido para ayudar a contestar preguntas acerca de cómo funciona el cerebro. En este proceso resultó que estas redes pueden memorizar, abstraer, generalizar o predecir, entre otras funcionalidades que también se encuentra en el cerebro. Hay tres características que hace a las redes neuronales especiales: 1) las funciones de activación son no lineales, 2) por la forma en que se conectan, conocida como la estructura, realizan computación en paralelo, 3) existen algoritmos, llamados de entrenamiento, mediante los cuales se ajustan los pesos para que la red lleve a cabo una tarea determinada.

Con toda seguridad, la estructura de red más popular es la conocida como feedforward, del inglés relacionado con conexiones hacia delante. Esta estructura se caracteriza por la configuración en capas, una de entrada; cero, una o más capas intermedias; y una capa de salida. La información fluye de la entrada, a las capas intermedias, para finalmente, generar una o varias salidas en la capa de salida. En cada capa, excepto en la de entrada, existe un número de neuronas, cada una de las cuales recibe conexiones, ponderadas por pesos, de la capa anterior. Las conexiones son procesadas por las funciones de activación, y el resultado de cada neurona pasa a todas las neuronas de la capa siguiente.

Las características ya mencionadas de las redes neuronales han hecho que estas sean atractivas para proponer soluciones inteligentes en el control de sistemas dinámicos. Por ejemplo, en una de las

Eliminado: DC

primeras aplicaciones, en (Weerasooriya y El-Sharkawi, 1993) se utiliza una red tipo perceptron multicapa para el control de un motor DC, el cual vale la pena mencionar aquí dado que este tipo de motor es una de las plantas que se trabaja en este artículo. En (Rubaai *et al*., 2001) controlan un motor de inducción, y utilizan un algoritmo de aprendizaje en línea, pero no para hacer el control, sino para la construcción del modelo de la planta (mediante cinco redes neuronales trabajando en paralelo), el cual se requiere para el control de corriente con control adaptable. Un trabajo más, el cual se utilizó como motivación inicial para el desarrollo de este trabajo está en (Kadwane *et al*., 2006), en el cual controlan un conversor de voltaje con una red neuronal artificial, entrenada en línea, utilizando Simulink. Si bien los resultados son interesantes, no son concluyentes en cuanto al rendimiento de ese controlador en otras plantas, como por ejemplo las utilizadas en este artículo.

Un trabajo más, dedicado a exponer las ventajas de los neurocontroladores en línea está en (Noh, Lee y Jung, 2008). Sus autores proponen utilizar redes neuronales de base radial para controlar la trayectoria de un péndulo en movimiento, en el cual el objetivo es mantener el equilibrio, mientras se traza una trayectoria circular. También con respecto al control de temperatura, esta vez en (Hedjar, 2007), se prueban configuraciones con redes neuronales para identificar y controlar, con resultados satisfactorios. Este trabajo es interesante también porque además de las redes con conexiones hacia delante utiliza redes recurrentes, como se hará hacia el final de este artículo.

La organización de este artículo es la siguiente: en la segunda sección se trata el entrenamiento en línea de una red neuronal artificial; en la siguiente se explican tres estrategias de control neuronal, luego se presenta la implementación de los tres controladores sobre dos sistemas, un circuito RC y un motor de corriente continua, llamado en adelante motor DC; enseguida se trabaja con un sistema de complejidad alta, este es el sistema esfera-tubo, y finalmente están las conclusiones y recomendaciones.

II. Entrenamiento en línea de una red neuronal

El algoritmo de entrenamiento para las redes con conexiones hacia delante es conocido como backpropagation, del inglés para propagación hacia atrás. El algoritmo tiene dos partes, en la primera, cada grupo o vector de entradas es presentado a la red, y esta, usualmente inicializada con pesos aleatorios, genera algunas salidas. Estas salidas de la red son comparadas con las salidas que se espera que la red asocie a esas entradas, y se calcula la diferencia o error. Enseguida se presenta el segundo vector de entrada, también conocido como ejemplo; nuevamente se propaga la red, y se evalúa el error entre la salida de la red y las salidas que se esperarían. El error debido a todos los ejemplos es acumulado, y se repite esto con todos los ejemplos que incluya el conjunto de datos que se tenga. En la segunda parte, con base en el error acumulado, se empiezan a cambiar los pesos de cada conexión en la capa de salida, de manera que el error disminuya (para mayores detalles con respecto al algoritmo de aprendizaje ir a (Kadwane *et al*., 2006). Luego se cambian también los pesos de todas las otras capas. Cuando se han realizado estas dos partes, se dice que se ha implementado una época, luego de lo cual se repite la parte uno, luego la dos, y se repite el procedimiento por tantas épocas como sea necesario, hasta que el error sea menor o igual a una cota que se considere adecuada.

El proceso ya descrito se conoce como entrenamiento en lote, o fuera de línea, porque se tienen todos los ejemplos y se dispone de tiempo para ajustar los pesos por medio del algoritmo de aprendizaje, y una vez se considera que la red tiene el conjunto de pesos adecuado, se detiene el entrenamiento y se utiliza la red. En este artículo se trabaja un problema relacionado, pero más difícil, debido a que el aprendizaje debe hacerse en línea. La red neuronal debe aprender, debe hacerlo rápido, y además debe hacerlo a partir de un ejemplo por época.

En su forma más simple, lo que la red aprende es la dinámica inversa de la planta, como en (Nouri *et al*., 2006). Los ejemplos que se le presentan a la red, compuestos por duplas de datos correspondientes a

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

Eliminado: Tanomaru

Con formato: Fuente: 11 pt, Fuente de escritura compleja: 11 pt

Eliminado: Omatu, 1992). Sus autores proponen dos estructuras básicas, una basada en el modelo inverso, y otra basada en la predicción del error. Estas dos estructuras son probadas con un ejemplo clásico como es el control de temperatura del agua en una bañera.

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

Eliminado:)

la entrada y la salida de la planta, se intercambian, de tal manera que la entrada de la red es la salida de la planta y la salida de la red es la entrada de la planta.

En general, en la capa de entrada de la red pueden tenerse *n* conexiones, pero para simplificar la explicación en la figura 1 se suponen sólo cuatro: p_1 a p_4. Entre la capa de entrada y la primera capa oculta existen conexiones entre todas las entradas y todas las neuronas, y cada conexión tiene asociado un peso, $iw^{1,1}$, denotado así por enlazar la capa de entrada con la primera capa oculta. De otra parte, cada peso tiene un subíndice, el cual indica la neurona destino y la entrada conectada por la conexión; para el ejemplo en la figura uno de los subíndices es 3,4. Cada neurona tiene un valor de ajuste, por ejemplo en la primera neurona es $b^1{}_1$, lo cual indica que es la primera neurona en la primera capa oculta.

La entrada de la función de activación de la capa oculta 1, es decir f^1, está en la ecuación 1.

$$n^1{}_1 = \sum_{R=1}^{4} iw^{1,1}{}_{1,R} * P_R + b^1{}_1 \quad (1)$$

La función de activación por lo general es sigmoidea, aunque en realidad puede ser cualquier función diferenciable.

$$a^1{}_1 = f^1\left(\sum_{R=1}^{4} iw^{1,1}{}_{1,R} * P_R + b^1{}_1 \right) \quad (2)$$

La salida de la capa oculta es llamada $a^1{}_1$, como se indica en la figura 1. Este valor es una de las entradas para la capa de salida, en la cual se suman las contribuciones de las neuronas de la capa oculta, como se indica en la ecuación 3, y finalmente se calcula la salida, utilizando f^2.

$$n^2{}_1 = \sum_{S=1}^{3} lw^{2,1}{}_{1,S} * a^1{}_S + b^2{}_1 \quad (3)$$

$$a^2{}_1 = f^2\left(n^2{}_1\right) \quad (4)$$

En la ecuación 4 $a^2{}_1$ es la salida de la red, la cual se compara con la salida deseada, y_d, para determinar la desviación entre la red y la salida esperada.

$$e = \frac{\left(y_d - a^2{}_1\right)^2}{2} \quad (5)$$

En la ecuación 5 *e* es el error cuadrático medio, según (Kadwane *et al.*, 2006), el cual se busca minimizar mediante el cambio de los pesos de la red. En ese proceso de minimización, las variables δ^1 y δ^2 indican la contribución de cada peso al error general, según se muestra en las ecuaciones 6 y 7.

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

$$\delta^2{}_1 = \left(y_d - a^2{}_1\right) * f^{2'}\left(n^2{}_1\right) \quad (6)$$

$$\delta^1{}_1 = f^{1'}\left(n^1{}_1\right) * \delta^2{}_1 * lw^{2,1}{}_{1,1} \quad (7)$$

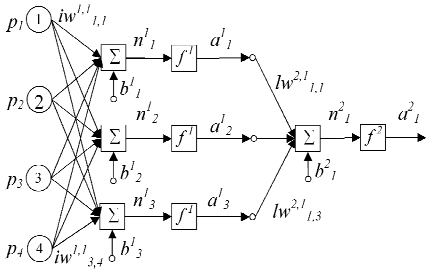


FIG 1. ESTRUCTURA DE UNA RED NEURONAL ARTIFICIAL FEEDFORWARD CON UNA CAPA DE ENTRADAS, UNA CAPA OCULTA Y UNA CAPA DE SALIDA. FUENTE: AUTORES DEL PROYECTO

$f^{1'}$ y $f^{2'}$ son las derivadas de las funciones de activación. Finalmente, el valor de los pesos corregidos es el que se indica en las ecuaciones 8 y 9.

$$lw^{2,1}{}_{1,1}(t+1) = lw^{2,1}{}_{1,1}(t) + \alpha * \delta^2{}_1 * a^1{}_1 \quad (8)$$

$$iw^{1,1}{}_{1,1}(t+1) = iw^{1,1}{}_{1,1}(t) + \alpha * \delta^1{}_1 * P_1 \quad (9)$$

La tasa de aprendizaje, α, es la encargada de definir la velocidad de aprendizaje. Para mayor detalle sobre la deducción del algoritmo *backpropagation* en línea ir a (Kadwane *et al.*, 2006).

III. Tres configuraciones para el controlador

Relacionado con el objetivo final del artículo, el cual es presentar el desempeño de un neurocontrolador con aprendizaje en línea, en esta sección se explican las tres configuraciones que se probarán a lo largo de todo el trabajo. Estas no son escogidas al azar, sino que responden a un proceso heurístico de diseño de controladores, en el cual se parte de una configuración conocida, y se incrementa el nivel de complejidad de la arquitectura del controlador, hasta alcanzar el objetivo del diseño. Este incremento en la complejidad también se verá en el tipo de planta a controlar, con tres casos también, como se explica en la sección siguiente, a lo cual se adiciona un controlador PID clásico, utilizado como base de comparación. De esta manera se tienen doce experimentos, mediante los cuales se busca conocer el comportamiento de las plantas y lograr el patronamiento de los controladores.

La primera estrategia es un lazo de control, comandado por un PID clásico, como se observa en la figura 2.a. En la segunda estrategia, en la figura 2.b, se incluye el neurocontrolador, pero la responsabilidad de comandar la planta es compartida con un PID. En la figura 2.c está la tercera opción, esta es una red neuronal entrenada en línea para minimizar el error, entendido éste como la diferencia entre la salida deseada *yd(t)* y la que se tiene *y(t)*; aplicaciones similares se pueden ver en (Guo *et al.*, 2007; Kamalasadan y Ghandakly, 2007).

La cuarta configuración fue denominada PID neuronal, como en (Omatu, 2009) en donde se aplicó al control de un vehículo. En ésta las ganancias *Kp*, *Ki* y *Kd* del PID convencional son reemplazadas por variables que dependen del error, según se indica en la figura 2.d. El propósito de esta última prueba es utilizar los mejores resultados del control PID clásico (entre los que se está el tener en cuenta el valor actual, la historia y una predicción del error para controlar), y combinar esos resultados con las ventajas de control neuronal en línea.

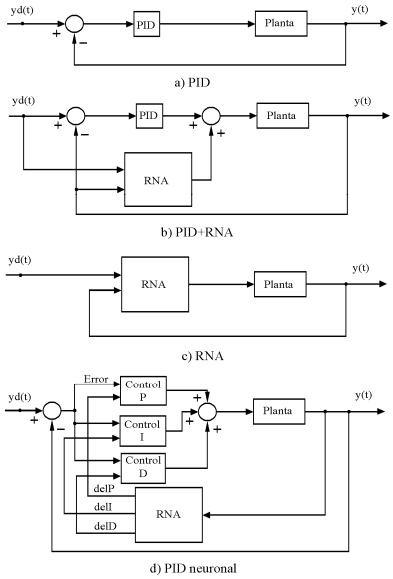


FIG 2. ESTRATEGIAS DE CONTROL, A) PID, B) PID+RNA, C) RNA Y D) PID NEURONAL. FUENTE: AUTORES DEL PROYECTO

Eliminado: ; Sztipanovits, 1992; Xiaosong et al., 1995

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

Eliminado: Boquete y Barea, 1998

Eliminado: vehiculo.

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

El cambio de las ganancias del PID neuronal consiste en definir incrementos (deltas) de ganancia, las cuales son la salida de la red neuronal, y esos valores se suman a las ganancias actuales en cada instante, como se bosqueja en la figura 2.d. Es importante anotar que la red neuronal ajusta sus pesos internos en función del error, por medio del método conocido como *backpropagation* en línea, explicado en la sección anterior. Este algoritmo permite que el proceso de ajuste de las ganancias se repita incluso hasta mil veces por segundo.

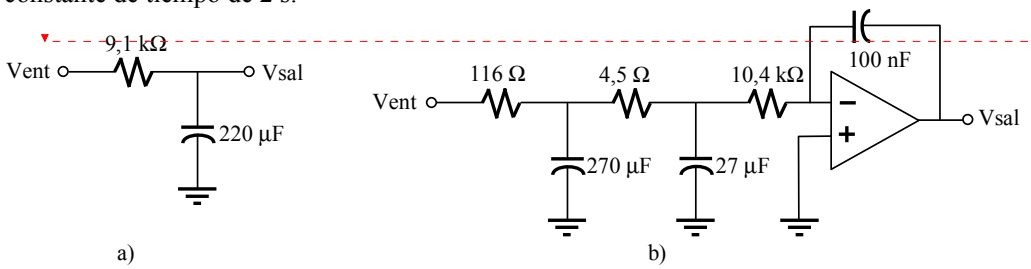
IV. Implementación de los controladores

En esta sección se implementan las estrategias de control descritas en la sección anterior. Primero en un circuito eléctrico RC y en seguida en un circuito doble RC en cascada, acoplado con un amplificador operacional, lo cual genera una función de transferencia equivalente a la de un motor DC.

A. *Circuito RC*

La primera opción cuando se prueba una estrategia de control es implementarla sobre un sistema lineal de orden uno. El hecho de que éste tenga solo un polo, estable, hace que sea fácil dirigir su dinámica hacia cualquier región del plano S. La opción más generalizada para controlar un sistema lineal consiste en interconectar al sistema con un sensor y un controlador, para que al formar un lazo cerrado, el sistema en su conjunto tenga los polos y los ceros que se requiera para alcanzar la dinámica deseada.

El sistema de orden uno pudo haber sido emulado de muchas formas, y de ellas se prefirió la implementación física, debido a que las ecuaciones son sólo abstracciones matemáticas del comportamiento físico, y por tanto siempre dejan algo de lado, lo cual puede ser importante para la red neuronal con aprendizaje en línea. El sistema de orden uno utilizado es un circuito eléctrico RC serie, como en la figura 3. La selección de R y C define la constante de tiempo. En este caso se seleccionó una constante de tiempo de 2 s.



Eliminado: <sp><sp><sp><sp><sp>¶
<sp><sp><sp><sp>¶
<sp><sp><sp><sp><sp><sp><sp>
<sp><sp><sp><sp><sp><sp><sp>
<sp><sp>¶
<sp><sp><sp><sp>¶
<sp><sp><sp><sp><sp><sp>¶

FIG 3. CIRCUITOS ELÉCTRICOS, A) CIRCUITO RC, B) CIRCUITO EQUIVALENTE A UN MOTOR DE CORRIENTE CONTINUA. FUENTE: AUTORES DEL PROYECTO

En la figura 4 se presenta el resultado del control del circuito RC. La referencia es la salida de un circuito de orden dos, con coeficiente de amortiguamiento de 0,5, y por tanto oscilatorio. La primera prueba se efectuó con un PID con ganancias $Kp = 10$, $Ki = 5$, $Kd = 0.1$. La red neuronal para la segunda prueba, como indica la parte b) de la figura 2, tiene cinco entradas; tres neuronas en cada una de las dos capas ocultas; y una neurona lineal de salida, la cual es combinada con el PID ya definido. En la tercera prueba se deja a la red neuronal sola para que trabaje como controlador, según se muestra en la parte c) de la figura 2.

En el caso del controlador neuronal, RNA en la figura 4, se utilizó una tasa de aprendizaje variable "$t.a(t)$", de la forma $t.a(t) = 9{,}9\exp(-t/2{,}8) + 0{,}1$. Esto permite que el algoritmo aprenda la dinámica del sistema cuando la tasa es alta; y deja que la neurona siga aprendiendo, aunque más lento, cuando la tasa es baja. Una ventaja adicional de la tasa pequeña es que se reduce el riesgo de oscilaciones, las cuales

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

pueden presentarse por búsquedas inestables alrededor de mínimos locales por parte del algoritmo de aprendizaje.

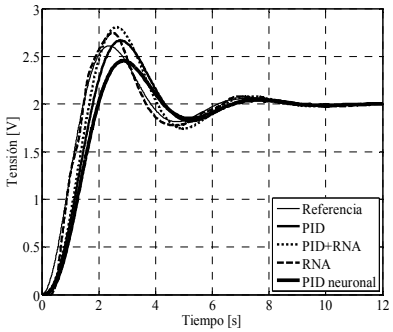


FIG 4. RESPUESTA DEL CIRCUITO RC ANTE LAS CUATRO ESTRATEGIAS DE CONTROL. FUENTE: AUTORES DEL PROYECTO

Por motivo de espacio no se presentan los resultados de todas las pruebas, pero cabe mencionar que, entre otras, se analizó la respuesta ante referencias tipo escalón y senoisal; también se experimentó con señales de referencia aleatorias, y se estudió la respuesta ante disturbios, entendidos como señales aleatorias de voltaje en algunos puntos del circuito. En todos los casos se llegó a resultados satisfactorios.

B. ***Control de posición de un motor DC***

El que las estrategias de control logren su objetivo con un circuito de orden uno no es una victoria muy grande, pero sí indica que se va por buen camino. Para incrementar la complejidad del problema, en esta sección se implementa una planta con un polo inestable, en el origen, más dos polos estables, como se presenta en la ecuación 10. El modelo equivale al cambio de posición de un motor de corriente directa, como se explica en (Rairán y Fonseca, 2011).

$$H(S)=\frac{Vp(S)}{Vf(S)}=\frac{253.782}{S\cdot(S^2+41\cdot S+263)} \qquad (10)$$

El denominador de la función de transferencia fue obtenido mediante la herramienta de Matlab para identificación de parámetros aplicada a un motor de imán permanente de marca Faulhaber con velocidad sin carga de 8.100 rpm, el cual indica constantes de tiempo de 30 ms y 125 ms. El valor del numerador fue seleccionado para que una entrada tipo escalón de amplitud 10 produzca una salida de 360 en un tiempo aproximado de 210 ms, como sería en el motor real para la entrada en voltios y la salida en grados, respectivamente, cuando la velocidad inicial es cero. El circuito eléctrico que emula la función de transferencia en la ecuación 10 está en la figura 3.

En la realización de los experimentos se siguió el orden indicado en la figura 2. Primero se sintonizó un controlador PID, por medio de Sisotool en Matlab, utilizando como modelo de la planta el resultado de un proceso de identificación, el cual fue alimentado con datos de salida del circuito ante entradas aleatorias en amplitud y duración. Con el controlador PID se logró el comportamiento indicado en la figura 5, el cual presenta un sobrepico menor al 30% y un tiempo de estabilización de alrededor de 3 s. Las ganancias son $Kp = 5$, $Ki = 6$, $Kd = 0$.

En la siguiente prueba, es decir PID+RNA en la figura 5, se diseñó una red neuronal con cinco entradas, dos capas ocultas y una capa de salida. Las funciones de transferencias son tangentes sigmoidales, excepto en la capa de salida donde es una línea con pendiente unitaria. Las entradas de la red son: el instante actual y el anterior de la señal de referencia; el valor anterior, y diez veces anterior de la salida de la planta; y la variable de control, filtrada con una función de transferencia de orden uno con constante de tiempo de 0,2 s, lo cual se hizo para reducir oscilaciones.

Eliminado: et al.

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

Como tercer experimento, la red neuronal anterior se deja sola para que controle el sistema, el resultado está en la figura 5 bajo el título RNA. El sistema presenta oscilaciones alrededor del punto de equilibrio, lo cual es una indicación de que puede hacerse más trabajo en la definición de esa red. La experimentación demostró que la tasa de aprendizaje es determinante en el algoritmo. Los mejores resultados se logran con la aplicación de reglas empíricas consignadas en reglas difusas, en las cuales el valor de la tasa depende de la diferencia entre la referencia y la salida del sistema, o error. Esta tasa va de 0,5 a 0,001. A mayor error mayor tasa. Esta lógica permite que cuando se presenta un incremento de error los pesos de la neurona cambien con rapidez para que el sistema se reajuste, y cuando se fija un valor bajo en la tasa es porque el error está dentro de un rango que se supone aceptable, para este caso dentro del 1% con respecto al valor de referencia.

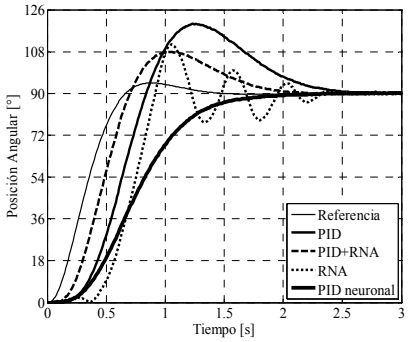


FIG 5. RESPUESTA DE LOS CONTROLADORES PARA EL CONTROL DE POSICIÓN DE UN MOTOR DC. FUENTE: AUTORES DEL PROYECTO

El último controlador en esta sección es denominado PID neuronal. La red tiene tres entradas: la referencia actual, la anterior, y el valor anterior de salida del sistema controlado. Con base en los datos de entrada se incrementa o reduce el valor de las ganancias, las cuales multiplican al error, a la acumulación del error, y a la tendencia de cambio del error. Esta estrategia presenta la curva más rápida y sin sobrepico, como se observa en la figura 5. En la figura 6 se presenta la evolución de las ganancias, y se muestra la forma particular como cambian algunos de los pesos dentro de la red neuronal. En la parte a) se observan las ganancias del controlador PID neuronal escalizadas por objeto de comparación; en la parte b) está la variación individual de algunos de los pesos en la red neuronal que forma parte del controlador PID neuronal. La notación utilizada para la identificación de los pesos corresponde a la explicada en la figura 1.

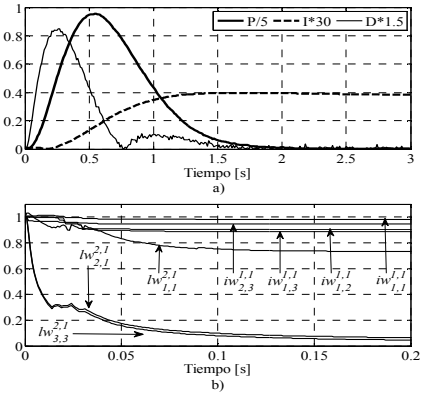


FIG 6. GANANCIAS Y VARIACIÓN DE LOS PESOS DEL CONTROLADOR PID NEURONAL. FUENTE: AUTORES DEL PROYECTO

V. Sistema esfera - tubo

Este sistema, a diferencia de los expuestos en la sección anterior, no es la emulación eléctrica de una planta, sino que es una planta real, la cual fue construida para verificar la efectividad de las estrategias de control explicadas en la sección III. También se conoce con el nombre de levitador neumático. Este ajusta la velocidad de un ventilador, y como resultado se obtiene un flujo de aire, el cual eleva y mantiene una esfera liviana en un nivel determinado. Por lo general la esfera está confinada en un tubo, lo que le da el nombre a la configuración (Ouyang *et al.*, 2007).

Eliminado: Pereira, J.B. Bowles, 1996

En esta sección se expone el montaje realizado, y se detalla el proceso de linealización de los sensores, debido a que durante el desarrollo del trabajo se observó que el control depende de la calidad de la medición, en gran medida. Se finaliza con la explicación de los resultados más sobresalientes.

A. *Descripción del montaje*

En la planta hay tres componentes principales: un ventilador, un tubo y una esfera. El ventilador genera una corriente de aire para elevar una esfera de icopor. Con el fin de simplificar el control, ese aire es confinado en un tubo; así la tarea de control se reduce a ajustar la velocidad del ventilador para que la esfera no caiga ni se pegue arriba, sino que se mantenga a cierta altura, llamada referencia.

Además de los tres componentes básicos, el montaje requiere sensores para medir la posición de la esfera. Si bien uno sería suficiente, en este trabajo se utiliza uno en la parte superior (con referencia comercial GP2D12), y otro en la inferior (GP2D120). El resultado es que la ventana de observación es más amplia, pero el precio es que la combinación de las dos lecturas exige un trabajo especial, como se explica en la sección siguiente. En la figura 7 puede verse la ubicación de los sensores y del resto del montaje.

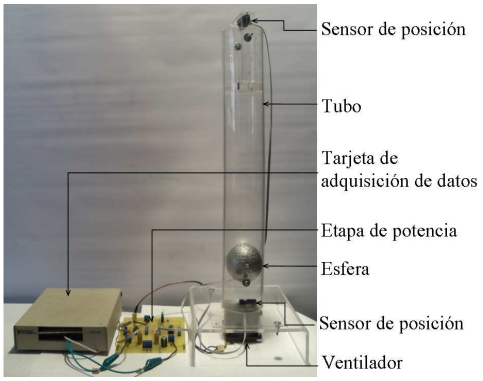


FIG 7. MONTAJE DE LABORATORIO DEL SISTEMA ESFERA – TUBO. FUENTE: AUTORES DEL PROYECTO

Un componente más de la planta es la etapa de potencia. Esta recibe una señal proveniente de la tarjeta de adquisición de datos, y la amplifica en potencia, para hacer girar el motor del ventilador. La salida de la tarjeta es un tren de pulsos, el cual alimenta al puente H con referencia L298N. Este puente puede manejar hasta 4 A.

La parte de programación del montaje se hizo en Simulink, como se muestra en el anexo, por dos razones: permite la ejecución del algoritmo de control en tiempo real; y es versatil en el manejo de bloques. Esto último facilita la implementación de la red neuronal con aprendizaje en línea. De acuerdo con el esquema en la figura 8, el primer bloque es el algoritmo de control, el cual puede ser PID, PID+RNA, RNA, PID neuronal, según se ha trabajado en todo el artículo. La conexión entre la parte física y la de programación es una tarjeta de adquisición de datos de National Instrumentes, la PCI-6024E, la cual cuenta con entradas y salidas análogas y digitales. El computador en que se trabajó tiene

las características siguientes: Intel Pentium 4, de 3Ghz, con 512 Mb de RAM, lo cual permite un tiempo de discretización de un (1) ms. Este tiempo resulta adecuado para la aplicación.

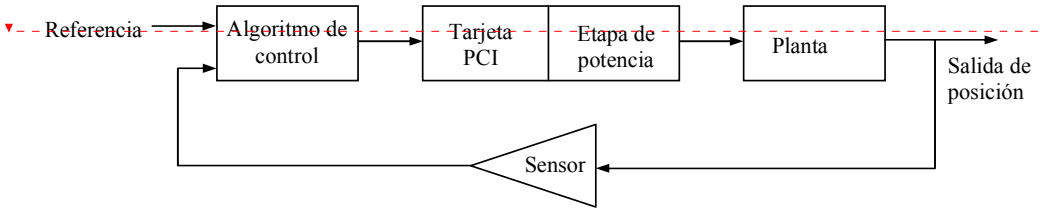


FIG 8. ESQUEMA GENERAL DE CONTROL. FUENTE: AUTORES DEL PROYECTO

B. ***Linealización de los sensores***

En esta sección se explica el proceso de adecuación y linealización de la lectura de los sensores que se utilizaron para medir la posición de la esfera. El problema inicial fue el rango de medida de los sensores comerciales, dado que en el mejor de los casos este llegó a 14 cm, mientras se requieren 24 cm. La solución fue unir la lectura de un sensor en la parte superior con la lectura de otro en la parte inferior.

La adecuación de la lectura de los sensores se realiza en tres etapas: 1) se comienza con un escalamiento, realizado con amplificadores operacionales; 2) enseguida se utiliza una red neuronal, para linealizar los datos, y finalmente 3) se integran las dos lecturas. La primera etapa, el escalamiento, consiste en ajustar la lectura de manera que el mínimo corresponda a cero voltios, y el máximo a algún valor cercano a diez voltios, como se observa en la figura 9.

En la segunda etapa, la linealización, dos redes neuronales con aprendizaje supervisado, son alimentadas con el voltaje escalizado. Las redes son entrenadas para que sus salidas correspondan a la posición real de la esfera, desde arriba y desde abajo, según corresponda. Las redes son de tipo feedforward, con una única capa oculta, con 30 neuronas con función de transferencia tangente sigmoidal, y una salida con función lineal pura. El algoritmo de entrenamiento es Levenberg-Marquardt, conocido en Matlab como trainlm, el cual es fuera de línea, a la manera que se expone en (Rairán *et al.*, 2009).

Es importante mencionar que en la generación de datos de entrada y salida para el entrenamiento de las redes se tomaron lecturas cada centímetro, las cuales fueron interpoladas por medio de una ecuación polinomial de orden seis, de la cual se obtuvieron 1.800 duplas entrada/salida para el sensor GP2D120 y 1.700 para el GP2D12. Así, un número típico de épocas durante el entrenamiento fue 1.000.

La tercera etapa es la integración de las lecturas, la cual se rige por la expresión siguiente:

$$b = c_1 \cdot S_1 + c_2 \cdot S_2$$

$$c_1 = \begin{cases} 1, & S_1 \le 12{,}75 \\ -S_1 + 13{,}75, & 12{,}75 < S_1 < 13{,}75 \\ 0, & S_1 > 13{,}75 \end{cases} \qquad (11)$$

$$c_2 = \begin{cases} 0, & S_2 \le 12{,}75 \\ S_2 - 12{,}75, & 12{,}75 < S_2 < 13{,}75 \\ 1, & S_2 > 13{,}75 \end{cases}$$

La salida integrada es la señal b; la salida linealizada del sensor en la parte inferior es S_1, y en la parte superior es S_2; c_1 y c_2 son los parámetros que sirven para efectúar la integración.

Siempre y cuando la lectura del sensor inferior sea inferior a 12,75, entonces la salida integrada, b, es igual a S_1. Cuando la lectura del sensor superior es mayor a 13,75, entonces b es igual a S_2. Para valores intermedios se realiza una ponderación lineal. El resultado se observa en la figura 9.

Eliminado: <sp>
Con formato: Izquierda
Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

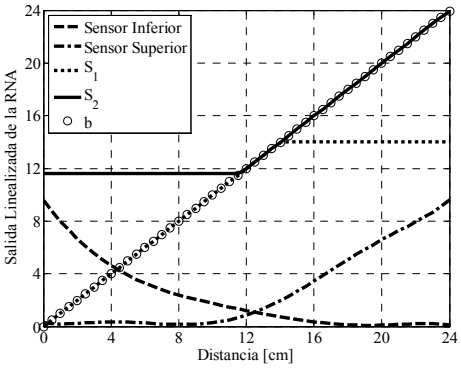


FIG 9. LINEALIZACIÓN DE LOS SENSORES. SE LINEALIZA LA REGIÓN CON MAYOR VARIACIÓN EN CADA SENSOR, Y SE COMBINAN PARA GENERAR UNA SOLA LECTURA. FUENTE: AUTORES DEL PROYECTO

C. ***Implementación de los controladores***

El levitador neumático es un problema difícil de solucionar, debido a los efectos no lineales que rigen su dinámica. Por ejemplo, experimentalmente se verificó que si las ganancias del controlador son fijas, entonces la respuesta del sistema es distinta en función de la altura de la esfera. Esto significa que un controlador con ganancias que se adapten puede resultar en un control más adecuado, lo cual es una de las motivaciones más fuertes para utilizar la red neuronal con aprendizaje en línea en el control de esta planta.

Una dificultad adicional es el tiempo de respuesta del sistema, el cual se expresa como el retardo entre el momento cuando se indica una referencia hasta cuando la planta responde en realidad. Por ejemplo, el retardo desde cuando se da la orden al ventilador para que gire, al momento cuando ya existe un flujo de aire proporcional al comando. Además existen dificultades durante la construcción, como el flujo turbulento que genera el ventilador y la tendencia de la esfera a girar, según describe la ley de Bernoulli y el efecto Coandă (Popescu *et al*., 2009).

Eliminado: Ziwei

Con formato: Fuente: Cursiva, Fuente de escritura compleja: Cursiva

Eliminado: 2007

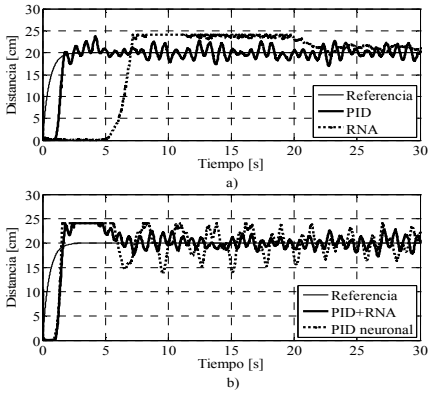


FIG 10. COMPORTAMIENTO DE LA PLANTA ANTE UNA SEÑAL ESCALÓN. A) PID Y RNA, B) PID+RNA Y PID NEURONAL. FUENTE: AUTORES DEL PROYECTO

En la figura 10 se observa el comportamiento de las cuatro estrategias utilizadas a lo largo del artículo, pero esta vez controlando el levitador a una referencia de 20 cm. El controlador que presenta la oscilación menor es el PID+RNA, aunque el tiempo para llegar por primera vez a la referencia es mayor que con el controlador PID, debido al tiempo que la RNA demora en adaptar sus pesos. Puede decirse que el control PID tiene el segundo mejor comportamiento, en cuanto a la amplitud de las oscilaciones,

las cuales deberían ser cero. El mayor inconveniente con el PID, como ya se mencionó, es que la utilidad de este algoritmo depende del punto de operación. La estrategia PID neuronal presenta el tercer mejor comportamiento, aunque se esperaba que tuviera el mejor, según se logró con los dos sistemas en la sección IV. Las ganancias del PID se definen por ensayo-error, de la experiencia en el manejo de la planta. Las ganancias iniciales para el PID neuronal son las mismas que para el PID. En el caso del controlador neuronal todos los pesos iniciales son uno y luego se les deja variar libremente según diga el algoritmo de aprendizaje. En el PID+RNA el PID tiene las mismas ganancias de los otros casos y la red es iniciada con pesos unitarios.

Las estrategias PID+RNA y RNA tienen cinco entradas:
– la referencia actual y la referencia del instante anterior,
– la salida de un instante anterior y de dos instantes antes, y
– la salida del controlador del instante anterior.
Esta última entrada convierte a esta configuración en recurrente. La red cuenta además con dos capas ocultas, una con función de activación tansig, y la otra logsig; en cada capa oculta hay tres neuronas. La capa de salida tiene una neurona con función de activación lineal.

Para la estrategia PID neuronal, la red tiene tres entradas: la señal de referencia, la retroalimentación de la planta y, una vez más, la salida de la red. Esta vez se trabaja con una capa oculta con tres neuronas, las cuales tienen función de activación tansig. La capa de salida tiene tres neuronas, una por cada ganancia del PID, con función de activación lineal.

La tasa de aprendizaje en esta sección fue evaluada a partir de la aplicación de un controlador difuso, lo que hace a esta tasa variable. La entrada del controlador difuso es generada de la siguiente forma: primero se calcula el error medio cuadrático, el cual comprime en un único número las últimas diez diferencias entre la salida deseada y la real; en seguida, para facilitar la definición de las funciones de membresía del controlador, se toma el valor absoluto del error medio cuadrático y se le aplica logaritmo con base 10. Se utilizan cuatro funciones de membresía para la variable de entrada, y otras cuatro para la variable de salida. Como resultado se obtiene una relación no lineal, la cual genera una salida que está entre -2 y -6. La tasa de aprendizaje corresponde a diez elevado al valor de la salida del difuso; de esta manera, la tasa de aprendizaje está entre 0,01 y 1e-6. En general, la lógica del controlador difuso consiste en que entre más grande es el error, más grande es la tasa de aprendizaje.

VI. Conclusiones

En este artículo se verificó que la red *feedforward* con aprendizaje en línea puede aproximar la dinámica de las tres plantas utilizadas, de tal manera que al conectar cada red con el sistema correspondiente es posible hacer que cada sistema se estabilice en un valor de referencia dado. Sin embargo, el control del levitador neumático resultó más efectivo al utilizar una conexión recurrente; esto es, con realimentación de la salida de la red a la capa de entrada de la red. Aún así, el número de entradas fue sólo cinco. Dos de ellas son realimentaciones de la salida de la planta, otras dos corresponden a la señal de referencia, y la última es la conexión recurrente.

El desarrollo de los experimentos demostró que el hecho de estar aprendiendo con base en un único ejemplo por cada vez que se corre el algoritmo *backpropagation* en línea, en contraste con el aprendizaje por lote o fuera de línea, es compensado de dos maneras: 1) al correrlo cada milisegundo, mientras que las constantes de tiempo de los sistemas están en el orden de los segundos, y 2) por la tasa de aprendizaje variable. La tasa variable hace que el algoritmo sea rápido en un inicio; que no olvide cuando ya ha logrado aprender lo que se necesita; y finalmente posibilita el reaprendizaje, si las condiciones de operación han cambiado lo suficiente, lo cual hace al algoritmo adaptable.

A partir de la experimentación con las tres plantas no es posible indicar cuál de las tres configuraciones con redes neuronales es la mejor (un PID en paralelo con una red neuronal, una red sola

o una red ajustando los pesos de un PID). Sin embargo, en los tres casos sí resultó que el peor comportamiento es el de la red sola, por lo cual puede pensarse que la combinación entre estrategias puede presentar ventajas que no vale la pena desaprovechar en aras de exaltar una estrategia en particular. Una de las razones por las cuales el rendimiento más bajo es el de la red sola es porque ésta aprende la dinámica inversa, y entonces el ruido y los disturbios con frecuencias altas pueden resultan más influyentes en la red que la dinámica de la planta. Por el contrario, el control PID puede verse como un filtro pasabajos, lo cual hace que el algoritmo de control se centre en las frecuencias bajas, donde está la información importante de la planta.

Referencias

GUO, C.; SONG, Q. y CAI, W. A Neural Network Assisted Cascade Control System for Air Handling Unit. *IEEE transactions on industry applications*. Febrero 2007, vol. 54, núm 1, pp. 620 – 628.

HEDJAR, R. Online Adaptive Control of Non-linear Plants Using Neural Networks with Application to Temperature Control System. *Journal of King Saud University, Computer & Information Sciences*. 2007, vol. 19, núm 1, pp. 75 – 94.

KADWANE, S.; KUMAR, A. y KARAN B.; Ghose T. Online Trained Simulation and DSP Implementation of Dynamic Back Propagation Neural Network for Buck Converter. *ACSE Journal*, Enero 2006, vol. 6, núm 1, pp. 27 – 34.

KAMALASADAN, S. y GHANDAKLY, A. A Neural Network Parallel Adaptive Controller for Dynamic System Control. *IEEE Transactions on Instrumentation and Measurement*, Octubre 2007, vol. 56, núm 5, pp. 1786 - 1796.

NOH, J.; LEE, G. y JUNG, Motion Control of a Mobile Pendulum System Using Neural Network. *Advanced Motion Control, 2008. AMC '08. 10th IEEE International Workshop.* 2008, pp. 450 – 454.

NOURI, K.; DHAOUADI, R. y BRAIEK, N. Nonlinear speed control of a dc motor drive system with online trained recurrent neural network. *9th IEEE international workshop on advanced motion control*. 2006, pp. 704 – 708.

OMATU, S. Neuro-control and Its Applications to Electric Vehicle Control. *IWANN'09 Proceedings of the 10th Interntaional Work-Conference on Aritificial Neural Networks: Part II: Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living*. 2009, Berlin, pp. 1 – 12.

POPESCU, M.; BALAS, V.; MUSCA. S. y COSMA, D. Modeling the Phenomenon of Refreshing Air Inside the Spaces of the Fungi's Culture. *ISCIII 2009, 4th International Symposium on Computational Intelligence and Intelligent Informatics*, Octubre 21–25, 2009 Egipto, pp. 85-92.

RAIRÁN, J.; FONSECA, J. Doble lazo de control para regular la posición y la velocidad en un motor de corriente directa. *Revista ingeniería y universidad*. Julio – diciembre 2011, vol. 15, núm 2, pp. 337 – 357.

RAIRÁN, J.; PÉREZ, J. y OSORIO, J. Implementación de una red neuronal para la medición indirecta de posición. *Revista avances en sistemas e informática*. Diciembre 2009, vol. 6, núm 3, pp. 79 – 85.

RUBAAI, A.; KOTARU, R. y KANKAM, M.. Online Training of Parallel Neural Network Estimators for Control of Induction Motors. *IEEE transactions on industry applications*. Septiembre/Octubre 2001, vol. 37, núm 5, pp.1512 – 1521.

WEERASOORIYA, S. y EL-SHARKAWI, M. Laboratory implementation of a neural network trajectory controller for a DC motor. *IEEE transactions on energy conversion*. Marzo 1993, vol. 8, núm 1, pp. 107 – 113.

OUYANG, Z.; SCHNELL, M. y WEI, K. The experiment Ball-in-tube with Fuzzy-PID controller based on dspace. *IEEE International Conference on Systems, Man and Cybernetics*. Octubre 2007, pp. 877 – 881.

Eliminado: BOQUETE, L. y BAREA, R. Control neuronal. *Servicio de publicaciones, Imprenta de la Universidad de Alcalá*, Universidad de Alcalá, Departamento de Electrónica, Monografía, 1998.¶

Eliminado: LIMA

Eliminado: .; CAVALCANTI, J. y DEEP, G. On-line Training of Adaptive Neural Network Controllers. *IECON '94., 20th International Conference on Industrial Electronics, Control and Instrumentation*. Septiembre 1994, vol. 2

Eliminado: 1392 – 1395

Eliminado: PEREIRA, J. y BOWLES, J. Comparing Controllers With the Ball In a Tube Experiment. *Proceedings of the Fifth IEEE International Conference on Fuzzy Systems*. Septiembre 1996, vol. 1

Con formato: Fuente: 12 pt, Fuente de escritura compleja: 12 pt

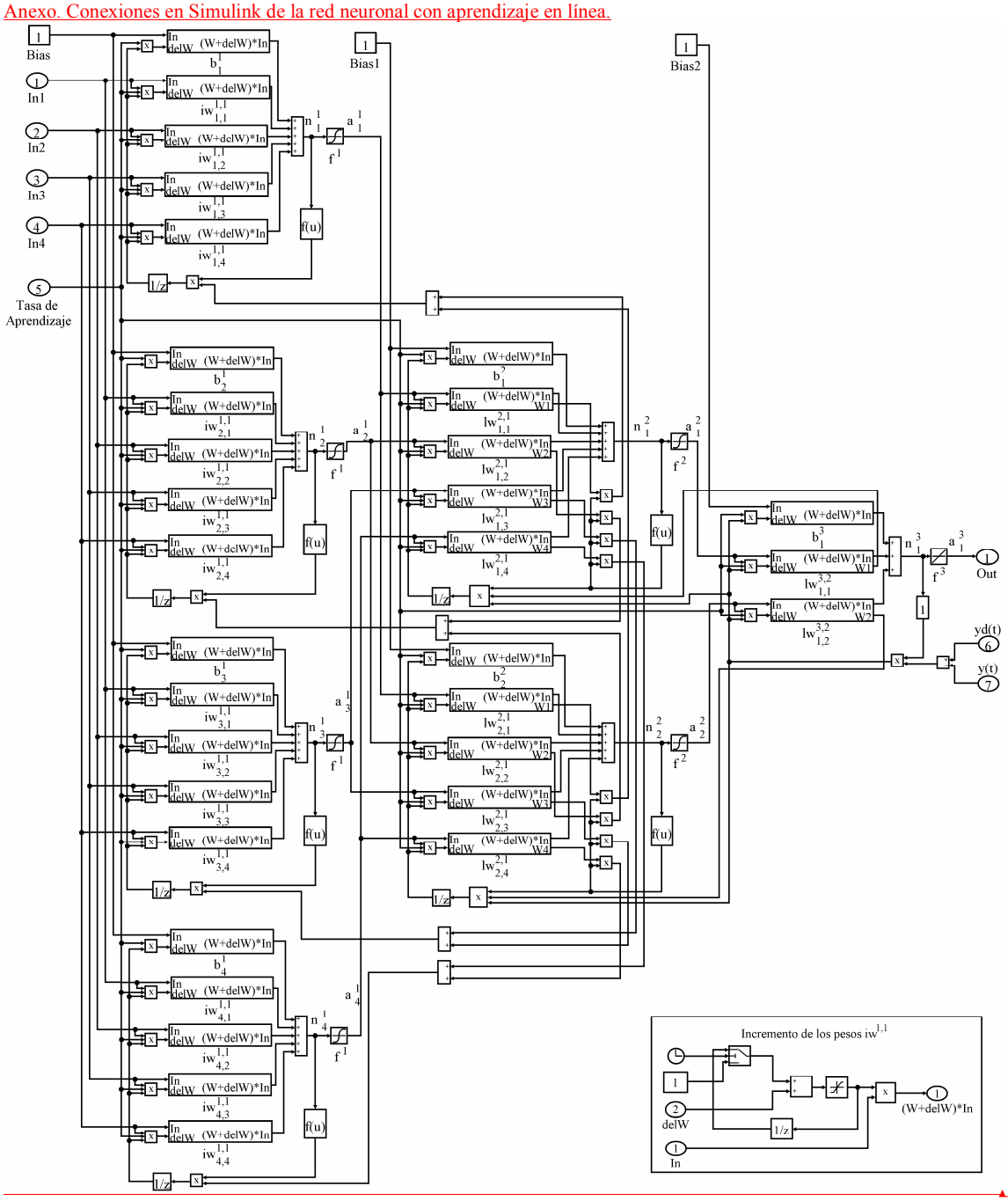
Eliminado: 504 – 510

Eliminado: . En proceso de publicación

Eliminado: SZTIPANOVITS, J. Dynamic Backpropagation Algorithm for Neural Network Controlled Resonator-Bank Architecture. *IEEE Transactions on circuits and systems-II: Analog and Digital Signal Processing*. Febrero 1992, vol. 39, núm 5, pp. 99 – 108.¶
TANOMARU, J. y OMATU, S. Process Control by On-Line Trained Neural Controllers. *IEEE Transactions on Industrial Electronics*. Diciembre 1992, vol. 39, núm 6, pp. 511 – 521.¶

Eliminado: W. y JIANG, J.P. A new learning controller based on neural networks for robot trajectories tracking. *IEEE International Symposium on Intelligent Control*. Aug, 1994, Columbus OH , USA, pp. 57 – 62.¶
XIAOSONG, D.; POPOVIC, D. y SCHULZ-EKLOFF G. The backpropagation neural network being capable of real-time identification. *Proceedings of the 4th IEEE Conference on Control Applications*. Septiembre 1995, pp. 572 – 577.¶
ZIWEI, O.; MICHAEL, S. y KEXIN, W.

Anexo. Conexiones en Simulink de la red neuronal con aprendizaje en línea.



Con formato: Fuente: 10 pt, Fuente de escritura compleja: 10 pt