risti
Revista Ibérica de Sistemas e Tecnologias de Informação
Revista Ibérica de Sistemas y Tecnologías de Información

Recebido/Submission: 09/08/2019
Aceitação/Acceptance: 01/09/2019

Configuración de un Control de Temperatura en un Sistema Embebido de Bajo Costo, usando Herramientas de Inteligencia Artificial y el Internet de las Cosas

Carlos Mario Correa Torres, Jhon Fredy Valencia Gómez

cmcorrea@sena.edu.co, jfvalenciag@sena.edu.co

Servicio Nacional de Aprendizaje, Diagonal 104 No. 69-120,50002, Medellín, Colombia.

DOI: 10.17013/risti.34.68–84

Resumen: El propósito de este trabajo es la implementación de un controlador difuso en línea, empleando una plataforma del internet de las cosas (IoT), para obtener un conjunto de datos que se procesan, empleando redes neuronales, con el objetivo de obtener un modelo que permita la identificación de la dinámica de un sistema térmico. Se calculan los parámetros de un controlador neuronal con base en el modelo estimado de la planta y posteriormente se transmiten a un sistema en chip (SoC), con el fin de configurar un sistema de control automático fuera de línea.

Palabras-clave: inteligencia artificial; internet de las cosas; sistemas embebidos; control automático.

Configuration of a temperature control in a low cost embedded system, using artificial intelligence and the internet of things tools

Abstract: The purpose of this paper is to implement an online fuzzy controller using an internet of things (IoT) platform to obtain a data set, that is processed using neural networks in order to build a model that permits to identify the dynamics of a thermal system. The parameters of a neural controller are calculated based on the estimated model of the plant. These parameters are transmitted to a System on a Chip (SoC) in order to configure an automatic offline control system.

Keywords*:* artificial intelligence; internet of things; embedded systems, automatic control.

1. Introducción

La cuarta revolución industrial ha llegado, como cada revolución, producto de la invención e introducción de herramientas tecnológicas que modifican drásticamente los métodos de producción, los empleos, las interacciones sociales y el entorno. En las primeras revoluciones, fueron la máquina de vapor, la energía eléctrica, la informática y automatización, las que impulsaron respectivamente cada una de ellas. Esta cuarta

revolución la promueve, no una tecnología específica, sino la sinergia de distintos desarrollos en áreas como biología sintética, fabricación aditiva, internet de las cosas, inteligencia artificial, *blockchain*, datos masivos, entre otras. La implementación de estas tecnologías y los productos de sus integraciones en entornos industriales, debe ser un tema de análisis prioritario para el desarrollo de la economía del país, tratar de resolver los retos que se presentan en estandarización, confiabilidad de sistemas computacionales, tráfico de datos, demanda energética, implementación de sistemas ciberseguros, costos de actualización tecnológica, entre otros, resulta imperativo si deseamos ser competitivos y tener una industria que satisfaga los requerimientos modernos. También deben considerarse las nuevas oportunidades en creación de productos y servicios soportados en estas, resulta obvio que con las revoluciones industriales desaparecen algunos tipos de trabajos y surgen otros que requieren nuevos conocimientos (Schwab, 2017), además de un alto componente de creatividad e innovación. El propósito de este trabajo es evidenciar que la aplicación de herramientas de la inteligencia artificial y el internet de las cosas en contextos industriales traerá beneficios en el desarrollo de procesos con sistemas más eficientes y con capacidad de autogestión.

2. Marco Teórico

Las técnicas de Inteligencia Artificial (IA) buscan básicamente que algoritmos informáticos emulen la observación, el entendimiento y el razonamiento humano. Por ejemplo, cuando intentamos describir muchos de los fenómenos de la naturaleza percibimos que la incertidumbre, la ambigüedad y la subjetividad hacen parte fundamental de nuestra representación y comprensión del mundo, podemos notar que algunos sistemas pueden ser definidos de manera más sencilla a partir de descripciones meramente cualitativas. Existe también, un gran conocimiento que a través de años de observación y experimentación han adquirido operadores humanos en sistemas industriales, que solo puede expresarse verbalmente, información que sería deseable ser interpretada y almacenada para su réplica. En el contexto anterior ha surgido como una solución de la representación y matematización de esta información, la lógica difusa.

2.1. Lógica Difusa

La lógica difusa tiene como base los denominados conjuntos difusos, que asumen valores de pertenencia en el intervalo [0 1]. A diferencia de la lógica clásica donde la pertenencia es exclusivamente 1 o 0, pertenece o no pertenece y en donde los enunciados solo pueden ser verdaderos o falsos. Un conjunto difuso A, se caracteriza por una función de pertenencia $A: \rightarrow X \in [0, 1]$, donde A denota tanto al conjunto como la función de pertenencia asociada. En 1965, Lofti A. Zadeh propuso el álgebra para la operación de estos conjuntos (Zadeh, 1965), y diferentes representaciones han hecho uso de esta, como, por ejemplo, los sistemas difusos Mamdani (Mamdani & Assilian, 1975), con variables lingüísticas de entrada y salida, que poseen sistemas de inferencia basados en reglas de la forma SI antecedente, ENTONCES consecuente, cuyo fin es obtener conclusiones de las variables lingüísticas de salida, a partir de los actuales valores de las variables de entrada. En los últimos años se han desarrollado un gran número de aplicaciones principalmente en sistemas de control automático, que se construyen a partir de reglas semánticas y el conocimiento experto (Olufunke, Charles, Charles, Abraham, & Snasel, 2013) (Bazmara & Donighi, 2014).

2.2. Redes Neuronales

De la inspiración de los procesos biológicos ocurridos en el sistema nervioso central, donde una neurona, recibiendo a través de sus dendritas impulsos nerviosos, procesados luego por el cuerpo principal y transmitidos por medio de la sinapsis a otras neuronas en una estructura inmensa de conexiones e información, las redes neuronales artificiales buscan procesar información en estructuras similares, para conseguir así, un tipo de asociación y aprendizaje artificial. En la red neuronal artificial, cada neurona se alimenta de entradas que a través de una función, obtienen una salida, producto de una suma ponderada de sus entradas, pueden obtenerse también relaciones no lineales, con el uso de la llamada función de activación, a cada entrada corresponde un factor de ponderación que influye en la participación de esta en el cálculo del valor de salida, a este valor se le conoce como peso, el proceso de entrenamiento consiste en ajustar estos pesos para que la salida calculada se aproxime al valor de una salida esperada. En sus inicios las redes neuronales básicamente se usaban para clasificaciones simples, más sin embargo el desarrollo de algoritmos y estrategias de optimización (Rumelhart, Hinton, & Williams, 1986) (Rumelhart, Hinton, McClelland, 1986) han conseguido hacerlas aplicables a múltiples procesos tales como: reconocimiento de patrones, identificación de sistemas, entre otros (Bevilacqua, Cerrone-Szakal & Siegfried, 2007).

2.3. Internet de las Cosas

La miniaturización de los circuitos electrónicos y el desarrollo de sistemas embebidos, ha permitido que tecnologías de procesamiento digital y comunicación en Sistemas en Chips (SoCs) pueden integrarse prácticamente a cualquier objeto, sus grandes volúmenes de producción y la competencia entre grandes empresas que quieren participar del rentable y prometedor negocio, ha generado una reducción considerable en el costo de estos dispositivos, logrando así, que se puedan desarrollar prototipos con inversiones mínimas de dinero. Herramientas de *software* de código abierto, una comunidad enorme de desarrolladores trabajando colaborativamente también han promovido la facilidad, simplicidad en el desarrollo de aplicaciones y servicios soportados en estas tecnologías. Lo anterior ha brindado las bases para lo que conocemos como el internet de las cosas (IoT). El IoT desarrolla la idea de sensores, actuadores, máquinas, electrodomésticos etc. inmersos en redes, que permiten además de su interconexión, la presentación y el análisis de datos. Estos dispositivos encuentran aplicación en diversas áreas cuando intercambian información en servicios públicos, transporte, equipos biomédicos, inmótica, domótica, industria automovilística, entre otras (Lee & Lee, 2015) (Ning & Hu, 2012) (Gubbi, Buyya, Marusic, & Palaniswami, 2013), el verdadero potencial del IoT se evidenciará en un escenario donde máquinas y objetos a través de conexiones generen productos o servicios de forma autónoma.

3. Desarrollo de la Implementación del Sistema

La teoría del control clásico a través de ecuaciones diferenciales y el análisis en la frecuencia, ha permitido la identificación de sistemas dinámicos y el desarrollo de estrategias de control, principalmente en sistemas lineales e invariantes en el tiempo (LTI), de una entrada y una salida (SISO). La teoría de control moderna por su parte

ha permitido el análisis de sistemas más complejos; sistemas de múltiples entradas y múltiples salidas (MIMO), inclusive variantes en el tiempo. Una de las técnicas de este análisis es la representación de sistemas dinámicos por el espacio de estados, en este, un sistema dinámico pude definirse de forma matricial a través de sus entradas, las variables de estado y sus salidas (Narendra & Parthasarathy, 1990) donde:

$$\frac{dx}{dt} \triangleq \dot{x}(t) = \boldsymbol{A}(t)x(t) + \boldsymbol{B}(t)u(t) \tag{1}$$

$$y(t) = \boldsymbol{C}(t)x(t) + \boldsymbol{D}(t)u(t)$$

Se conoce a $\boldsymbol{A}$ como la matriz de estados, $\boldsymbol{B}$ la matriz de entrada, $\boldsymbol{C}$ la matriz de salida y $\boldsymbol{D}$ la Matriz de transmisión directa, en general, esta última se hace cero por simplicidad, asumiendo que el sistema no tiene transmisión. Una representación por ecuaciones diferenciales sería:

$$\begin{aligned} &\dot{x} = \varphi[\left(x(t), u(\ )\right] \quad t \in R^{+} \\ &y(t) = \varnothing\left[x(t)\right] \\ &\boldsymbol{x}(t) \triangleq \left[x_1(t), x_2(t), \ldots, x_n(t)\right]^T \\ &\boldsymbol{u}(t) \triangleq \left[u_1(t), u_2(t), \ldots, u_p(t)\right]^T \\ &\boldsymbol{y}(t) \triangleq \left[y_1(t), y_2(t), \ldots, y_m(t)\right]^T \end{aligned} \tag{2}$$

La ecuación 2 representa un sistema de orden n con p entradas y m salidas, $u_1(t)$ representa las entradas del sistema, $y_1(t)$ representa las salidas y $x_1(t)$ representa las variables de estado. El vector $x(t)$ denota el estado del sistema en el tiempo t y es determinado por el estado en un tiempo $t_o < t$ ty la entrada $u(t)$, definida en el intervalo $[t_o$ t). t_o La salida $y(t)$es determinada completamente por el estado del sistema en el tiempo t. La ecuación 2 es conocida como la representación del sistema entrada-estado-salida. Para el caso de un sistema discreto lineal e invariante en el tiempo la ecuación 2 puede representarse como una ecuación en diferencias así:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) \end{aligned} \tag{3}$$

Donde $y(k)$, $x(k)$, $u(k)$, representan secuencias. En el caso particular de que el sistema sea un sistema SISO la ecuación 3 (Narendra & Parthasarathy, 1990) puede escribirse como:

$$y(k+1) = \sum_{i=0}^{n} \propto_i y(k-i) \ \sum_{j=0}^{m} \beta_j u(k-j) \tag{4}$$

Donde α_i, β_j son parámetros constantes desconocidos. La ecuación 4 muestra que con el cálculo de estos parámetros podríamos obtener la identificación del sistema o un modelo de la salida de la planta a partir de la información de datos de entrada-salida en instantes previos. Por la anterior, nos proponemos identificar el sistema térmico usando el método serie-paralelo (Figura 1), usando una red neuronal para obtener los valores $\hat{\alpha}_i$, $\hat{\beta}_j$, y así esta estimación:

$$\hat{y}_p(k+1) = \sum_{i=0}^{n-1} \hat{\alpha}_i(k) y_p(k-i) \;\; \sum_{j=0}^{m-1} \hat{\beta}_j(k) u(k-j) \tag{5}$$

Como el propósito es implementar el sistema en tiempo real, se propone usar inicialmente un controlador difuso en línea, que en tanto se encuentra conectada la planta satisfaga las necesidades de control y a su vez pueda generar variabilidad en los datos de entrada, para que estos, en conjunto con sus salidas correspondientes, proporcionen la información necesaria para el entrenamiento de la red neuronal y con esto, la identificación del sistema. Una vez realizada la identificación del sistema térmico, se define a partir del modelo estimado de la planta, un controlador, que se implementa en el sistema embebido, para evitar la necesidad de conexión a la red. A continuación, se presenta el controlador difuso y la red neuronal para la identificación de la planta que se usaron, para luego mostrar le estructura del sistema de control.

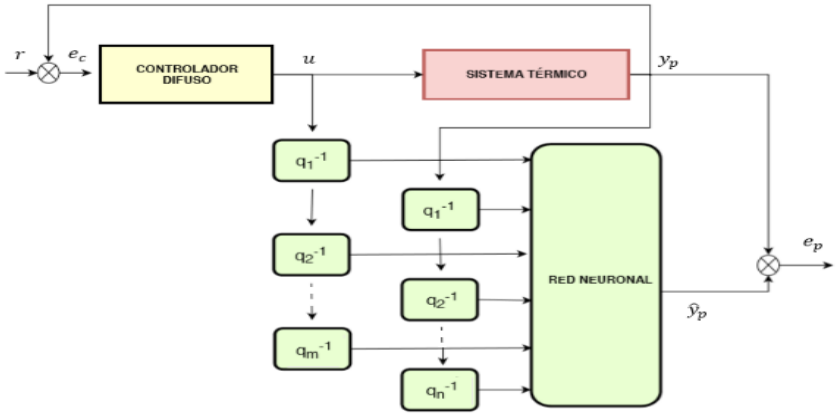

Figura 1 – Fase de identificación de la planta usando el método serie paralelo con controlador difuso en línea. q^{-1} representa las unidades de retardo de las entradas y salidas de la planta.

3.1. Controlador Difuso

Para el controlador difuso se utiliza un sistema difuso tipo Mamdani, se define el *Error* como variable lingüística de entrada, el error resulta de la diferencia entre el valor deseado o *setpoint* y el valor actual de temperatura, este se compone de cuatro conjuntos o valores difusos: *Negativo Grande, Negativo Mediano, Negativo pequeño y Positivo*. Se definieron funciones de pertenencia triangulares para el error, en el intervalo [-70 10], el rango del error se define partiendo de una temperatura ambiente

y un *setpoint* para entrenamiento de 60 grados, los rangos para cada función de pertenencia triangular se definieron subjetivamente. Como variable lingüística de salida se define la *Potencia*, con cuatro conjuntos o valores difusos; *Potencia grande, Potencia media, Potencia pequeña y Potencia cero*, en un intervalo [0 100], se definieron 3 funciones triangulares en intervalos subjetivos sin traslapamiento y una función *singleton* que se definió para el caso de salida de una potencia cero, se buscó que el sistema tuviese una respuesta aproximadamente proporcional o tipo rampa como puede apreciarse en la Figura 2. La simplicidad y generalidad de este sistema se plantea con el propósito de que pueda ser usado en cualquier otro sistema dinámico lineal, las variables, sus correspondientes conjuntos difusos y rangos se muestran en la Figura 2. Para la implementación de este sistema de inferencia difusa se utilizó el software *LabVIEW* y su herramienta para el desarrollo de sistemas difusos *Fuzzy System Designer* perteneciente a la versión 2018. La Figura 2 muestra la ventana de evaluación del controlador difuso propuesto. Para la operación del sistema de inferencia se definieron 4 reglas:

1. Si el Error es negativo grande entonces la potencia es potencia grande.
2. Si el Error es negativo mediano entonces la potencia es potencia media.
3. Si el Error es negativo pequeño entonces la potencia es potencia pequeña.
4. Si el Error es positivo entonces la potencia es potencia cero.

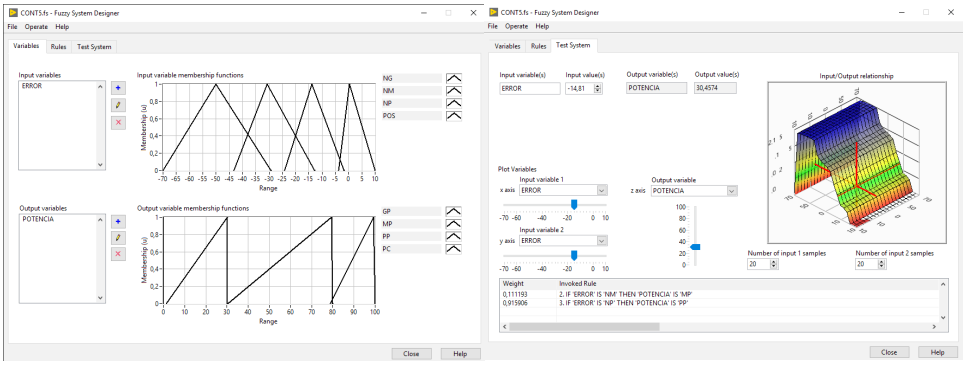

Figura 2 – Variables de entrada, salida y conjuntos difusos del controlador implementado en LABVIEW(Izq.), Respuesta del Controlador (Der.).

Cabe notar que sistemas más robustos pueden ser implementados, con mayor número de variables, conjuntos y reglas de inferencia, pueden también definirse otros apoyados en el conocimiento experto.

3.2. Red Neuronal propuesta para la identificación

Para la identificación del sistema, como se mencionó anteriormente, se propone el uso de redes neuronales, en específico el uso de un perceptrón simple (Torrubia, 2010), que pueda predecir la salida de la planta a partir de datos de entradas-salidas previos, la estructura del perceptrón propuesto se muestra en la Figura 3.

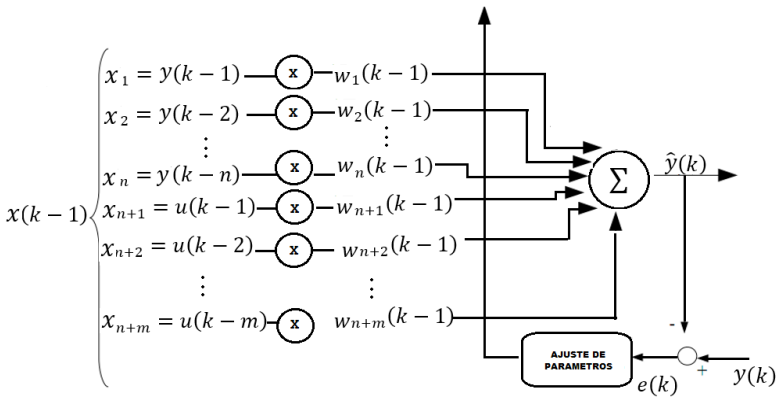

Figura 3 – Perceptrón para identificación del sistema térmico.

El vector de pesos correspondiente al perceptrón se define como:

$$\boldsymbol{w}(k)=\left[w_1,w_2,\ldots w_n,w_{n+1},\ldots w_{n+m}\right]^T \tag{6}$$

El vector $x(k)$ de entrada de la red se define como:

$$\boldsymbol{x}(k)=\left[x_1(k),\ldots x_n(k),x_{n+1}(k),\ldots,x_{n+m}(k)\right]^T \tag{7}$$

cuyas componentes para la fase de entrenamiento serán las entradas y salidas correspondientes en instantes anteriores así:

$$\boldsymbol{x}(k-1)=\left[y(k-1),\ldots y(k-n),u(k-1),..u(k-m)\right]^T \tag{8}$$

El valor estimado de salida de la planta en el instante k se definirá entonces como:

$$\hat{y}(k)=\sum_{i=1}^{n+m} w_i(k-1)x_i(k-1) \tag{9}$$

Así un valor futuro de la planta se calculará con:

$$\hat{y}(k+1)=\sum_{i=1}^{n+m} w_i(k)x_i(k) \tag{10}$$

donde

$$\boldsymbol{x}(k)=\left[y(k),\ldots y(k-n+1),u(k),..u(k-m+1)\right]^T \tag{11}$$

Para estimar el valor de salida de la planta en el instante k, $\hat{}y(k)$, el vector $x(k-1)$ se obtiene con el sistema *online* funcionando en lazo cerrado (Figura 1). El sistema parte midiendo el valor de la temperaturaen el horno, inicialmente, temperatura ambiente, que se transmite al controlador difuso. Este calcula, basado en las reglas de inferencia, la acción de control u_i necesaria para alcanzar el *setpoint* definido en 60 grados Celsius, luego de dos segundos se toma de nuevo una muestra de temperatura y_{ip} y se repite el proceso, este termina cuando se han tomado 300 datos: 150 correspondientes a la entrada u y 150 de su respectiva respuesta y_{ip}. Se organiza el data set en una matriz donde cada elemento del vector $v_p = [y_{11} \quad \dots \quad y_{21} \quad \dots \quad y_{141}]$, se predice con los valores previos del vector columna que los precede en la Matriz *Data Set,* que se muestra en 12.

$$Data\,set: \begin{vmatrix} y_1 & y_{11} & y_{21} & \cdots & y_{141} \\ y_2 & y_{12} & y_{22} & \cdots & y_{142} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ y_{10} & y_{20} & y_{30} & \cdots & y_{150} \\ u_1 & u_{11} & u_{21} & \cdots & u_{141} \\ u_2 & u_{12} & u_{13} & \cdots & u_{142} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ u_{10} & u_{20} & u_{30} & \cdots & u_{150} \end{vmatrix} \tag{12}$$

La evaluación de la estimación del modelo del sistema térmico se realiza usando el error cuadrático medio:

$$mse = \frac{1}{l}\sum_{i=1}(\hat{y}_{ip} - y_{ip})^2 \tag{13}$$

Donde $l = 14$, que es el número de elementos del vector en el vector v_p extraídos del *data set*. Una vez alcanzado un valor predeterminado de este estimador el proceso de ajuste de los pesos del perceptrón termina y se considera que la estimación de la planta es aceptable y a partir de este modelo puede ya calcularse el controlador.

3.3. Controlador Neuronal

El propósito de un controlador es suministrar, a través de una variable manipulada, una entrada en la planta, que produzca que la variable controlada alcance un valor de referencia o *setpoint*. Una vez identificada la planta con el modelo que se describe en la ecuación 10, podemos calcular a partir de esta, una acción de control, despejando el valor de $u(k)$, luego dividiendo por su correspondiente coeficiente o peso w_{n+1}, y reemplazando el valor $y(k+1)$ por el valor de referencia (Torrubia, 2010):

$$u(k) = \sum_{i=1}^{n+m} w_i'(k) * z_i(k) \tag{14}$$

El vector de coeficientes del controlador w_i' puede escribirse a partir de los pesos obtenidos del modelo de identificación de la planta así:

$$w(k)=\frac{1}{w_{n+1}}\left[1,w_1(k),w_2(k),\cdots,w_n(k),w_{n+2}(k),w_{n+m}(k)\right]^T \tag{15}$$

El vector ***z*** es definido a partir de los estados del modelo

$$\mathbf{z}(k)=\left[y(k+1),-x(k),\cdots-x_n(k),-x_{n+2}(k),\cdots-x_{n+m}(k)\right]^T \tag{16}$$

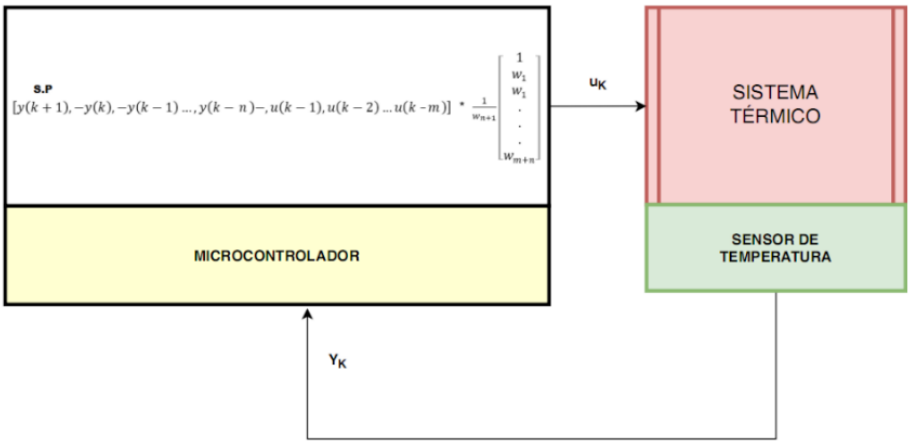

Figura 4 – Diagrama de Bloques Controlador Neuronal.

Se debe satisfacer que w_{n+1} > 0.1. para la convergencia del modelo. Para el entrenamiento de la red neuronal se utilizó el software Matlab versión 2018a y su herramienta para el trabajo con redes neuronales, utilizando el 80% de los datos para entrenamiento y el 20% para validación, con datos tomados en tiempo real, tanto de la salida de la planta, como del controlador, como se explicó en la sección 2.2. Durante el proceso de evaluación del controlador se entrenó la red varias veces usando sesgo o bias diferente de cero, y se observó que su valor fue siempre cercano a cero, como se evidencia en la ecuación 5, que muestra la dependencia del valor futuro principalmente de la suma ponderada de los valores previos de entrada-salida, sin termino adicional, por lo anterior en las simulaciones posteriores se consideró eliminar el sesgo.

3.4. Funcionamiento del Sistema

El funcionamiento del sistema se describe en los siguientes pasos:

1. Una vez establecida la conexión, el microcontrolador en el sistema térmico transmite el valor actual de temperatura.

2. El controlador difuso en *LabVIEW,* basado en la temperatura recibida del microcontrolador y en sus reglas de inferencia, calcula la potencia y transmite esta al microcontrolador.
3. Paralelamente a los pasos 1 y 2 los datos de temperatura y potencia son recibidos por Matlab para usarse en el entrenamiento de la red neuronal.
4. Una vez son alcanzados la cantidad de datos determinados (data set) se inicia el proceso de entrenamiento y se evalúa el desempeño usando 13, se debe satisfacer que (mse<1.0) y $w_{n+1} > 0.1$, si estos criterios no se satisfacen, la acción de control sigue siendo calculada por el controlador difuso.
5. Si los criterios se cumplen, el vector $\boldsymbol{w'}$ que contiene los coeficientes calculados de la red neuronal para el controlador, es transmitido al SoC, este ignora ahora la potencia transmitida desde el controlador difuso y comienza a trabajar sin conexión.
6. El SoC calcula la potencia necesaria usando el vector $\boldsymbol{w'}$, el vector $\mathbf{z}$ y la ecuación 13, el vector $\mathbf{z}$ es actualizado en cada instante de muestreo para el próximo cálculo de la potencia (Figura 4).
7. Los coeficientes son guardados en la memoria EEPROM del microcontrolador para que permanezcan, inclusive, el sistema se desconecte de la red eléctrica.

La transmisión y recepción de estos datos esta soportada en herramientas del IoT, específicamente, en el protocolo *Message Queuing Telemetry Transport (MQTT)* y el formato *Java Script Object Notation* (JSON). MQTT es un protocolo *Machine To Machine*, usado ampliamente en el IoT, usa un modelo, publicar - suscribirse, donde básicamente los datos que se transmiten o publican pueden ser recibidos por cualquier

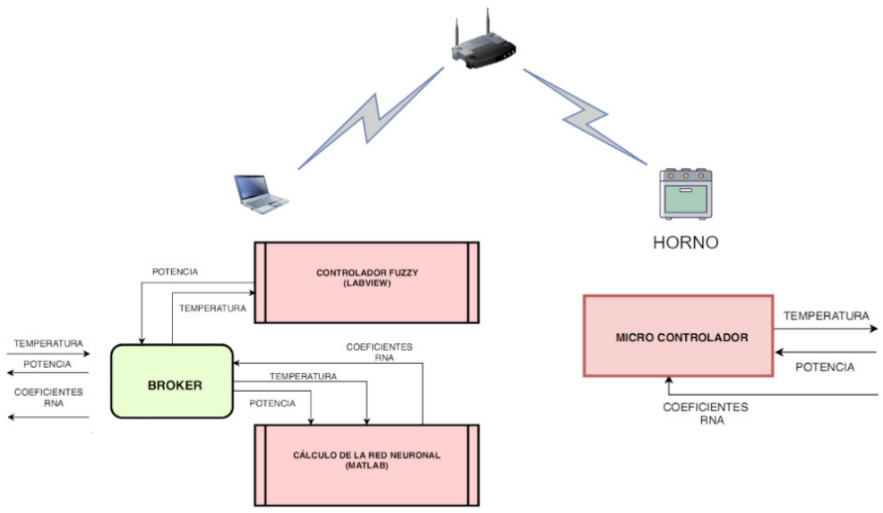

Figura 5 – Topología para prueba del sistema propuesto para configuración del controlador mientras se encuentra en línea.

dispositivo en la red que se suscriba a estos, esta característica permite compartir información paralelamente de forma eficiente. La intermediación de estos datos se consigue a través de un *Broker* que se encarga de la gestión de transporte, la topología implementada se muestra en la Figura 5.

Otra herramienta en la que se apoya este trabajo es el *Node Red,* entorno de trabajo que permite la programación basada en flujogramas, además de integrar el *Broker* para la gestión de transporte de datos, permite realizar la depuración de estos, y también su presentación usando el *dashboard,* esta herramienta se usó para visualizar las respuestas de los controladores. *Node Red* fue instalado en un computador portátil haciendo la función de servidor, con el software LabVIEW y Matlab, configurando en ellos la comunicación con el protocolo MQTT, para la implementación del sistema. El SoC que se utiliza es el Node MCU de la empresa *Espressif*, dispositivo de muy bajo costo, que integra un Procesador dual Tensilica de 32 bits, de doble núcleo a 240MHz y una interfaz WI-FI con Protocolo 802.11 b/g/n. Este microcontrolador a través del modulador de ancho de pulso (PWM) controla la potencia a través de un relevador de estado sólido que está conectado a la resistencia eléctrica del sistema térmico, la potencia calculada se encuentra limitada en el intervalo [0 100], en el sistema se integró una pantalla táctil para la visualización local de la temperatura y reconfiguración del punto de referencia. La Figura 6 muestra su implementación.

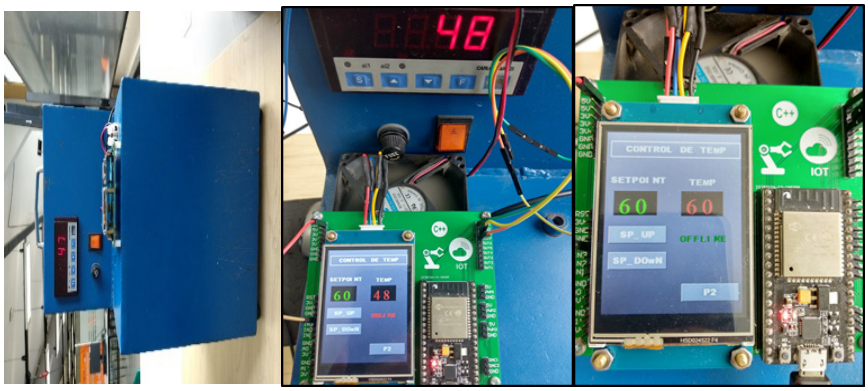

Figura 6 – (Izq.) Sistema térmico de prueba, (Centro) trabajando en línea con controlador difuso durante la identificación, (Der.) trabajando fuera de línea con el controlador neuronal.

4. Resultados

Se implementó en el sistema térmico de prueba, para tener una referencia, un controlador *Maxthermo* MC5438, usando el algoritmo Proporcional Integral Derivativo (PID), con la función de autoajuste y se estableció como referencia 58 °C, la respuesta del controlador se muestra en la Figura 7.

A continuación, en la Figura 8, se presentan los resultados obtenidos de la implementación del sistema de control difuso en línea con un *setpoint* de 60 °C, sin la aplicación de la red neuronal para la identificación.

Posteriormente se implementó el sistema completo. La Figura 9 presenta la respuesta del controlador, primero usando el controlador difuso en línea para la identificación del sistema y luego el controlador neuronal propuesto funcionando en el sistema embebido desconectado de la red. El tiempo de entrenamiento fue de 72.64 segundos, luego de haber tomado los datos para su entrenamiento.

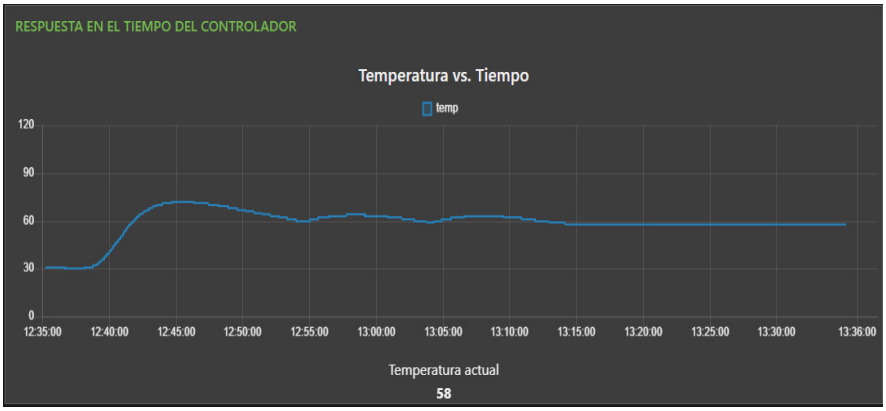

Figura 7 – Desempeño del controlador Maxthermo con algoritmo PID con autoajuste.

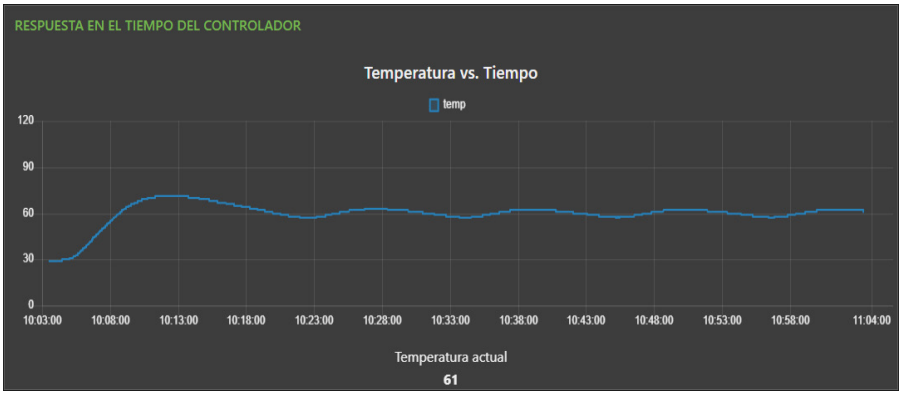

Figura 8 – Respuesta del controlador Difuso en línea implementado en la planta sin el uso del controlador neuronal.

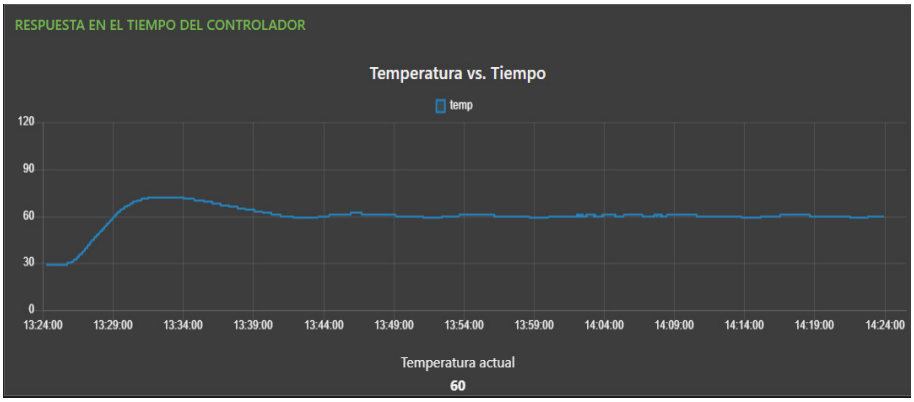

Figura 9 – Respuesta del controlador Neuronal trabajando inicialmente *online* con el controlador difuso y luego sin conexión con una referencia de 60 grados, mse= 0.6699.

Se modificó luego del establecimiento del punto de referencia de 60°C a un valor de 70°C, para verificar su funcionamiento con un valor distinto al de su entrenamiento. La Figura 10 muestra el desempeño del controlador.

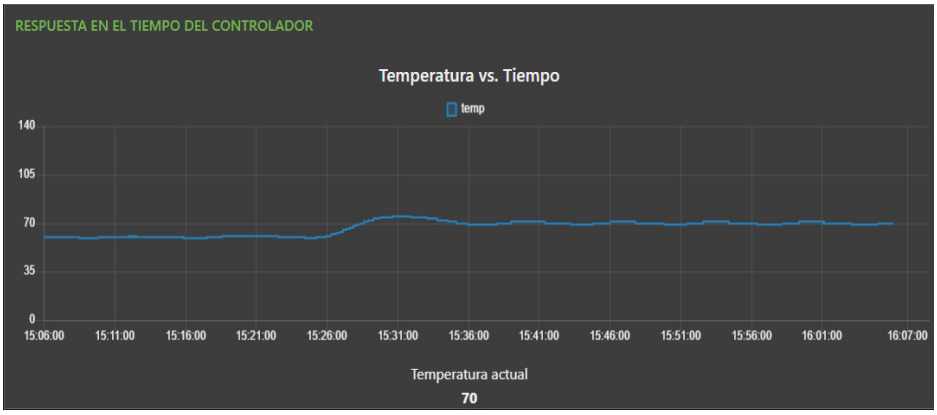

Figura 10 – Respuesta del controlador Figura 13, redefiniendo después de una referencia de 60 una setpoint de 70 grados

Las figuras 11 y 12 muestran otros resultados obtenidos de la implementación del sistema completo con sus respectivos valores de *mse* obtenidos.

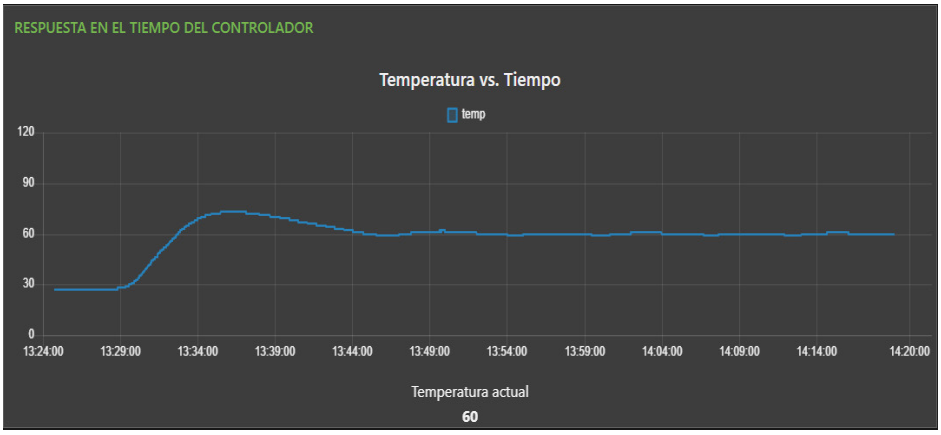

Figura 11 – Respuesta del controlador Neuronal trabajando inicialmente en línea con el controlador difuso y luego *offline* con un *Setpoint* de 60 grados, *mse*= 0.8011, 198.65 seg.

Con el objetivo de evaluar la robustez del controlador, este fue sometido a dos perturbaciones. Luego de haber llegado al punto de referencia 60° C y estabilizarse, fue introducido un flujo de aire frio constante a las 6:29 P.M, una vez el controlador consigue volver al punto de referencia 6:41 P.M, desde otra zona del horno se introduce otra corriente de aire a las 6:43 P.M como puede observarse en la Figura 13.

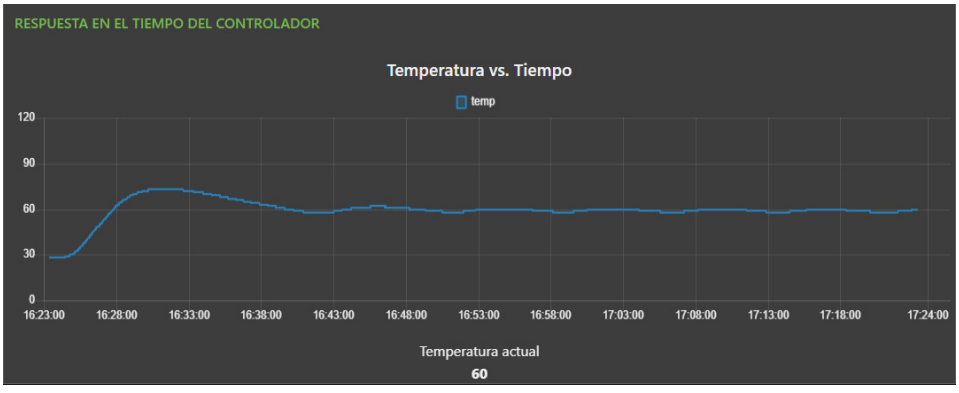

Figura 12 – Respuesta del controlador Neuronal trabajando inicialmente *online* con el controlador difuso y luego *offline* con un *Setpoint* de 60 grados, mse= 0.5015, 157,45 segundos.

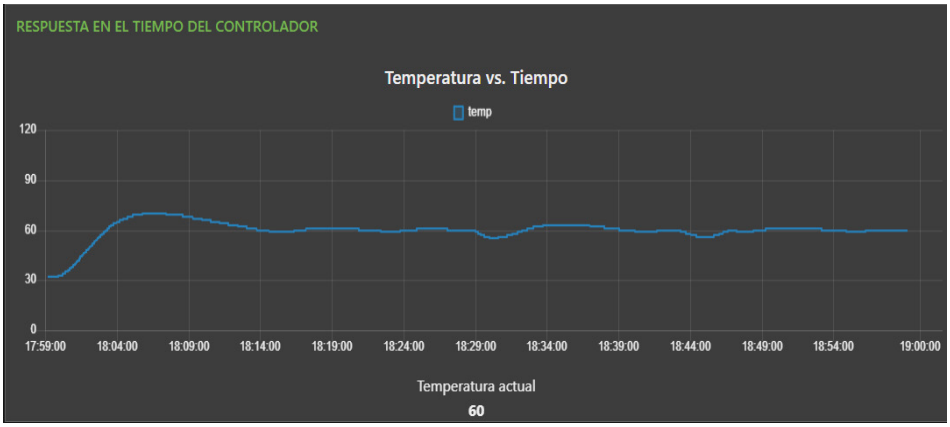

Figura 13 – Respuesta del controlador neuronal frente a dos perturbaciones de entrada

5. Conclusiones

Puede observarse que el desempeño del controlador difuso (Figura 8) es aceptable, aun siendo diseñado a partir de información verbal muy general, tiene un menor tiempo de establecimiento que el controlador *Maxthermo* (Figura 7), aunque presenta una oscilación leve y permanente alrededor del *setpoint*, también demanda una constante conexión para su operación. Cabe notar que el objetivo de este es suministrar variabilidad en los datos para alimentar el perceptrón manteniendo la planta controlada en tanto este es entrenado. Se espera que este controlador funcione solo en la etapa inicial. Por otra parte, el desempeño del controlador neuronal trabajando en desconexión con el modelo de la planta presenta un buen desempeño (Figuras 9,11,12), su tiempo de establecimiento es menor que el de controlador *Maxthermo*, esto inherente al proceso de autoajuste ejecutado por el *Maxthermo*. El controlador presenta una oscilación aún más leve que el controlador difuso, el tiempo de conexión es bajo y además no demanda de ninguna intervención en aspectos como programación o configuración, el controlador también muestra que funciona, luego de configurado para otros valores de referencia (Figura 10). La Figura 13 evidencia que se comporta bien frente a perturbaciones ya que es capaz aun siendo estas permanentes, regresar al punto de referencia.

La implementación de este sistema estaría en capacidad de configurar un controlador para cualquier sistema térmico ya que se logra a través de la información obtenida de sus entradas y salidas, este puede ser monitoreado en línea bajo demanda eliminando así el uso de interfaces hombre máquina, logrando una más efectiva gestión de la información del proceso y una menor necesidad de tiempo de conexión, todo esto soportado en un hardware de bajo costo. Implementar este sistema en la nube es

posible usando herramientas ya disponibles, bibliotecas para *machine learning*, lógica difusa y manejo de protocolos para el IoT como el MQTT que son gratuitas y de uso libre.

Referencias

Bazmara, A., & Donighi, S. S. (2014). Bank customer credit scoring by using fuzzy expert system. *International Journal of Intelligent Systems and Applications*, 6(11), 29–35. doi: 10.5815/ijisa.2014.11.04.

Bevilacqua, P. C., Cerrone-Szakal, A., & Siegfried, N. A. (2007). Insight into the functional versatility of RNA through model-making with applications to data fitting. *Quarterly Reviews of Biophysics*, 40(1), 55–85. doi: https://doi.org/10.1017/S0033583507004593

Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7), 1645–1660. doi: https://doi.org/10.1016/j.future.2013.01.010

Lee, I., & Lee, K. (2015). The Internet of Things (IoT): Applications, Investments, and Challenges for Enterprises. *Business horizon,* 58(4), 431–440. doi: https://doi.org/10.1016/j.bushor.2015.03.008

Mamdani, E.H., & Assilian, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, 7(1), 1–13. doi:https://doi.org/10.1016/S0020-7373(75)80002-2

Narendra, K. S., & Parthasarathy, K. (1990*)*. Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*, 1(1), 4–27. doi :10.1109/72.80202

Ning, H., & Hu, S. (2012). Technology classification, industry, and education for Future Internet of Things. *International Journal Communication Systems*, 25(9), 1230–1241. doi: 10.1002/dac.2373

Olufunke, O.O., Charles, U.O., Charles, A.K., Abraham, A., & Snasel, V. (2013). A fuzzy-mining approach for solving rule based expert system unwieldiness in medical domain. *Neural Network World*, 23(5), 435–450. doi: http://www.nnw.cz/doi/2013/NNW.2013.23.027.pdf

Rumelhart, D.E., Hinton, G.E., & McClelland, J.L. (1986). A general framework for parallel distributed processing. In D.E. Rumelhart, J.L. McClelland, & the PDP Research Group (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition:* Vol. 1. Foundations (pp.45- 76). Cambridge, MA: MIT Press.

Rumelhart, D.E., Hinton, G., & Williams, R. (1986). Learning Representations by Back-Propagation Errors. *Nature*, 323, 533–536. doi: https://doi.org/10.1038/323533a0

Schwab, K. (2017). *The Fourth Industrial Revolution*. New York: Crown Business.

Torrubia, S. (2010). *Redes Neuronales Multimodelo Aplicadas al Control de Sistemas*. (Tesis de maestría). Universidad Autónoma de Barcelona, España. Recuperado de https://ddd.uab.cat/pub/trerecpro/2013/hdl_2072_207064/TorrubiaCaravacaSergioR-ETISa2009-10.pdf

Zadeh, L.A., (1965). Fuzzy sets. *Information and Control*, *8*(3), 338–353. doi: https://doi.org/10.1016/S0019-9958(65)90241-X