

ESCUELA POLITÉCNICA DEL EJÉRCITO

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

**CARRERA DE INGENIERÍA EN ELECTRÓNICA,
AUTOMATIZACIÓN Y CONTROL**

**PROYECTO DE GRADO PARA LA OBTENCIÓN DEL TÍTULO EN
INGENIERÍA**

**DISEÑO E IMPLEMENTACIÓN DE CONTROLADORES
INTELIGENTES PARA LA UNIDAD *AIR FLOW TEMPERATURE
CONTROL SYSTEM* MEDIANTE LA UTILIZACIÓN DE LA
HERRAMIENTA RTW (REAL-TIME WORKSHOP) DE MATLAB**

AUTORES: ADRIÁN CABEZAS Y.

DANIEL CAPELO

SANGOLQUÍ – ECUADOR

2012

CERTIFICACIÓN

Certificamos que el presente proyecto de grado titulado **“DISEÑO E IMPLEMENTACIÓN DE CONTROLADORES INTELIGENTES PARA LA UNIDAD AIR FLOW TEMPERATURE CONTROL SYSTEM MEDIANTE LA UTILIZACIÓN DE LA HERRAMIENTA RTW (REAL-TIME WORKSHOP) DE MATLAB”** Ha sido desarrollado en su totalidad por el señor PAUL ADRIAN CABEZAS YANEZ con CI: 1719189548 y el señor DANIEL ALEJANDRO CAPELO PARREÑO con CI: 1712083516 bajo nuestra dirección.

Atentamente:

Ing. Víctor Proaño

DIRECTOR

Ing. Paul Ayala

CODIRECTOR

AGRADECIMIENTO

“Hoy me siento afortunado de poder decirlo, y convertirlo en rap para que así pueda sentirlo.

Hoy quiero dar las gracias por existir, por sonreír y celebrar que aún queda mucho por vivir.

Gracias:

Por tener conciencia, tener el don de hacer ciencia que aboga por la no violencia.

A unos padres bondadosos que me arroparon con besos, voy hacerlos sentir orgullosos.

A mis hermanos Brito por su confianza, por enseñarme que la base es el trabajo y la constancia

A mis hermanas por su apoyo por su amor tan hardcore, que me ha ayudado cada día en esta labor.

Por no caer en la maldad y en la ignorancia, por nacer en tierra de luz y abundancia.

Por mis momentos de paz sin malas sorpresas, por una cama y un plato lleno en mi mesa.

A la música por refrescarme, única droga que logra calmarme.

Amigos por dejarme descansar, enemigos por hacerme reaccionar.

Por respirar, por poder mirar el mar y sentirme libre, por no tener que disparar balas de ningún calibre, por vivir y sentirme invencible.

Por ser fuerte en esta lucha, por no hacer música de masas porque las masas no escuchan.”

Adaptación, Nach Scratch – “Gracias”

ADRIAN CABEZAS Y.

AGRADECIMIENTO

Gracias a mis padres, por el gran apoyo que me dieron durante toda mi carrera y formación, que siempre confiaron en mí y gracias a ellos pude finalizar con esta última etapa de mi carrera; Gracias por su amor, cariño, confianza y paciencia. Con amor y agradecimientos infinitos.

DANIEL CAPELO

DEDICATORIA

A mi madre Ximena con mucho amor.

ADRIAN CABEZAS Y.

DEDICATORIA

Este proyecto va dedicado con mucho amor a mis padres, Jenny y Marcelo, a mis hermanas Paola y Diana y a mis abuelitos Juan y Laura.

DANIEL CAPELO

PROLOGO

El presente proyecto de tesis abarca el estudio, diseño e implementación de controladores: neuronales, difuso y predictivo para controlar la temperatura en la unidad AIR FLOW TEMPERATURE CONTROL SYSTEM mediante la utilización de la herramienta RTW (Real-Time Workshop) de Matlab que nos permite ejecutar en tiempo real los controladores obtenidos, la adquisición de datos de la planta se lo realiza con la tarjeta NI PCI 6221.

Este proyecto pretende brindar más información sobre el diseño e implementación de los controladores inteligentes mencionados en un sistema real de temperatura, los modelos de simulación y los modelos experimentales se desarrollan en la herramienta Simulink de Matlab.

CONTENIDO

CAPITULO I	10
INTRODUCCION.....	10
1.1 Definición del Proyecto.....	10
1.1.1 Antecedentes.....	10
1.1.2 Justificación e Importancia.....	11
1.1.3 Alcance del Proyecto	12
1.1.4 Objetivos	13
CAPITULO II.....	14
2 TEORIA DE LOS CONTROLADORES A IMPLEMENTAR.....	14
2.1 Control Neuronal	14
2.1.1 Controlador neuronal indirecto por modelo inverso.....	17
2.1.2 Controlador neuronal directo por modelo de referencia	18
2.2 Control Difuso	19
2.3 Control Predictivo	21
CAPITULO III.....	24
3 MODELAMIENTO DE LA PLANTA.....	24
3.1 Características de la unidad Air Flow Temperature Control System	24
3.2 Modelo Matemático del Sistema	25
3.3 Identificación	28
CAPITULO IV.....	33
4 DISEÑO DE LOS CONTROLADORES INTELIGENTES.....	33
4.1 Controlador Difuso.....	33
4.1.1 Especificaciones del Problema	33
4.1.2 Solución del Problema	34
4.1.3 Simulación	38
4.1.4 Implementación.....	40
4.2 Controlador Predictivo.....	42
4.2.1 Especificaciones del problema	42
4.2.2 Solución del Problema	42
4.2.3 Simulación	45
4.2.4 Implementación.....	47
4.3 Controlador Neuronal Indirecto por Modelo Inverso.....	49
4.3.1 Especificaciones del problema	49
4.3.2 Solución del Problema	49

4.3.3 Simulación	52
4.3.4 Implementación.....	55
4.4 Controlador Neuronal Directo Basado en Modelo de Referencia con Acción integral	59
4.4.1 Especificaciones del problema	59
4.4.2 Solución del Problema	59
4.4.3 Simulación	64
4.4.4 Implementación.....	68
CAPITULO V	71
5 PRUEBAS Y RESULTADOS.....	71
5.1 Controlador Difuso.....	73
5.2 Controlador Predictivo.....	76
5.3 Controlador Neuronal Indirecto por Modelo Inverso.....	78
5.4 Controlador Neuronal Directo Basado en Modelo de Referencia ..	80
CAPITULO VI	83
6 CONCLUSIONES Y RECOMENDACIONES	83
6.1 Conclusiones	83
6.2 Recomendaciones.....	85
REFERENCIAS BIBLIOGRÁFICAS.....	87

CAPITULO I

INTRODUCCION

1.1 Definición del Proyecto

1.1.1 Antecedentes

Diseñar controladores que no requieran un excesivo trabajo matemático es uno de los problemas que posee la teoría de control. Los controladores más usados a nivel industrial son los controladores PID y si bien representan un método eficiente y de bajo costo, requiere un amplio trabajo de ingeniería. Los métodos de control inteligente representan un método alternativo de solución de problemas de control.

Actualmente el software MATLAB es comúnmente utilizado como simulador para el diseño de sistemas de control por su gran capacidad de cálculo, versatilidad de librerías y porque dispone de una amplia colección de funciones aplicables al campo de control inteligente, que mediante la herramienta Real-Time Workshop (RTW), permite probar los algoritmos modelados en tiempo real.

En la asignatura de Control Inteligente dictada en el área de Automatización y Control, se realiza la implementación en tiempo real de controladores diseñados en MATLAB y plantas físicas de algunos sistemas del laboratorio de servo - mecanismos, en el Departamento de Ciencias de la Energía y Mecánica de la ESPE.

En el proyecto de tesis realizado por los alumnos Oswaldo Ibarra y Byron Acuña en el 2010, se logró el control de la unidad *air flow temperature control system* utilizando el software MATLAB, mediante un controlador PID implementado en Simulink y que se comunica con la planta mediante la herramienta RTW. En este proyecto se pretende continuar la investigación con métodos de control avanzado ya que presentan estabilidad ante dinámicas indeseadas inclusive sin conocer un modelo exacto de la planta. Mediante la aplicación Simulink de MATLAB, se conoce que se puede construir y simular modelos de sistemas de control, proporcionando flexibilidad en el diseño de los controladores inteligentes.

1.1.2 Justificación e Importancia

Con el fin de contribuir al desarrollo de investigación del Departamento de Eléctrica y Electrónica, el proyecto pretende consolidar la teoría impartida en la cátedra de Control Inteligente, mediante la creación de un entorno para realizar prácticas en los laboratorios propios del Departamento. Así se podrá analizar pruebas en tiempo real de los controladores diseñados.

Los controladores inteligentes hoy en día son ampliamente utilizados por su capacidad de adaptación, aprendizaje y adecuado funcionamiento en ambientes variables, con este trabajo se espera entregar una mayor información acerca del procedimiento de diseño, las ventajas, desventajas, y el posible uso de cada esquema de control. Además se analizará las dificultades al realizar controladores neuronales, difusos y predictivos.

1.1.3 Alcance del Proyecto

El presente proyecto de tesis abarca el estudio, diseño e implementación del código en MATLAB para la unidad *AIR FLOW TEMPERATURE CONTROL SYSTEM* (control de temperatura) en tiempo real mediante la herramienta RTW, de los siguientes métodos de control:

- Controlador Neuronal
 - Controlador Neuronal Indirecto por Modelo Inverso
 - Controlador Neuronal Directo Basado en Modelo de Referencia.
- Control Difuso
- Control Predictivo

Se utilizará una tarjeta de adquisición y generación para interconectar la planta real con el sistema SIMULINK de MATLAB.

Se realizará el acondicionamiento de señales para la conexión de la tarjeta con el proceso. Se configurará la tarjeta de adquisición y se realizará los archivos .m y .mdl de MATLAB para facilitar el uso de la herramienta

1.1.4 Objetivos

1.1.4.1 General

Realizar el diseño e implementación de controladores inteligentes de temperatura para la unidad *AIR FLOW TEMPERATURE CONTROL SYSTEM* mediante la utilización de la herramienta RTW (*real time workshop*) de MATLAB.

1.1.4.2 Específicos

1. Estudiar y analizar la herramienta RTW (*Real Time Workshop*) de MATLAB
2. Verificar el adecuado funcionamiento de la tarjeta de adquisición de datos *Real-Time Windows Target* con MATLAB.
3. Modelar la planta *AIR FLOW TEMPERATURE CONTROL SYSTEM* y los diferentes controladores como bloques en SIMULINK.
4. Verificar el funcionamiento de la interfaz de adquisición de datos usando el software controlador creado en MATLAB.
5. Describir y verificar mediante la implementación, el adecuado funcionamiento del los sistemas de control inteligente para la unidad *AIR FLOW TEMPERATURE CONTROL SYSTEM* mediante el software creado en MATLAB específicamente en la herramienta SIMULINK.

CAPITULO II

2 TEORIA DE LOS CONTROLADORES A IMPLEMENTAR

2.1 Control Neuronal

Las redes neuronales en el ambiente industrial se los utilizan como identificadores de procesos o como sistemas que optimizan el funcionamiento de un controlador.

Existen varios controladores basados en redes neuronales, llamados neurocontroladores.

El concepto básico de una red neuronal artificial se lo observa en la Figura 2.1

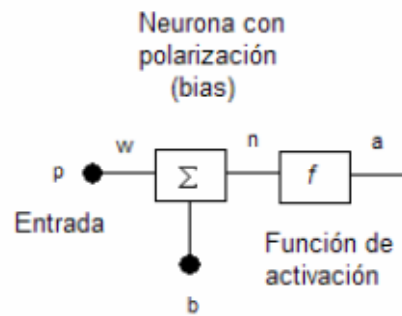


Figura 2.1 Neurona Artificial

Donde p es el valor de la señal de entrada, b es un valor conocido como bias, w es el valor que se ajusta al proceso de entrenamiento llamado peso y f es la función de activación, donde normalmente se lo conoce por dejar o no pasar la información.¹

Una red neuronal es la unión de varias neuronas artificiales como las mencionadas anteriormente, conectadas entre sí para manejar la información según su entrenamiento.

En los controladores neuronales existen varias maneras de realizar el control, así como también el entrenamiento y se clasifican de la siguiente manera:

- Control neuronal indirecto
- Control neuronal directo

¹ Ochoa José, *Control Clásico y Control Inteligente*

Para los sistemas de control neuronal indirecto una red neuronal no envía una señal directamente al proceso, si no que una red neuronal se usa como un indicador de las características del proceso, al hablar de indicador se quiere decir que se crea un modelo que imita el comportamiento del proceso real o un controlador, así el controlador neuronal tiene las referencias para comportarse según sea apropiado.²

Existen varios tipos de control neuronal indirecto, clasificados de la siguiente manera:

- Control neuronal basado en un modelo de proceso
- Control neuronal basado en un modelo inverso
- Desarrollo de una red neuronal autosintonizada

En los sistemas de control neuronal directo, una red neuronal se lo utiliza como un control retroalimentado, y envía señales de control directamente al proceso. Existen varios tipos de diseño y se los clasifican de la siguiente manera:

- Modelo de un controlador
- Control neuronal libre de modelo
- Control neuronal basado en modelo
- Control neuronal basado en modelo robusto

² Ochoa José, *Control Clásico y Control Inteligente*

Los modelos utilizados en esta tesis son controladores neuronales indirectos por modelo inverso y controladores neuronales directos por modelo de referencia, son modelos aplicables al sistema que se desea implementar, a continuación se procederá a dar una breve explicación de estos dos controladores.

2.1.1 Controlador neuronal indirecto por modelo inverso

Una red neuronal puede ser entrenada para imitar el comportamiento de la planta pero de forma inversa. La entrada a la red es la salida del proceso y la salida de la red es la entrada del proceso.³

Cabe recalcar que el modelo inverso es una estructura de estado estable, la cual puede ser usada para el control; el esquema de un controlador neuronal por modelo inverso se lo puede observar en la Figura 2.2.

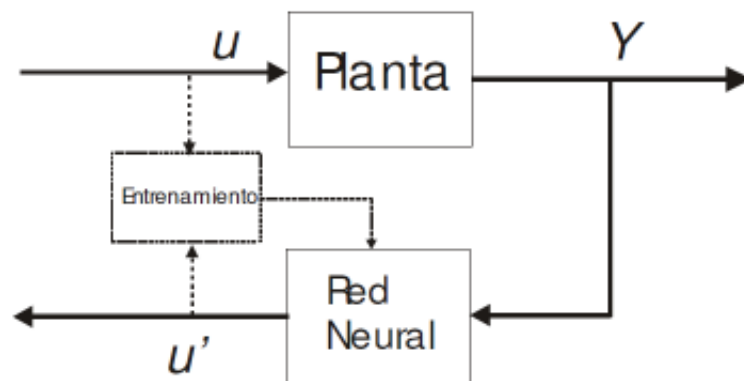


Figura 2.2 Red neuronal con modelo inverso

³ Ochoa José, *Control Clásico y Control Inteligente*

2.1.2 Controlador neuronal directo por modelo de referencia

El control neuronal por modelo de referencia consiste en diseñar un sistema que modifique el comportamiento natural de la planta con el objetivo que se aproxime a la respuesta que tiene un modelo de referencia establecido.

Este sistema que modifica el comportamiento de la planta, se lo diseña a partir de una identificación del sistema a controlar, que compone el modelamiento de la planta, para así proceder a entrenar una red que imite su comportamiento.

Al obtener el modelo de referencia, según como se desee modificar el comportamiento de la planta, se lo entrena en conjunto con los pesos, estados y variaciones obtenidas de la identificación y el modelo de referencia, todo este proceso se lo puede resumir con el esquema que se muestra en la Figura 2.3.

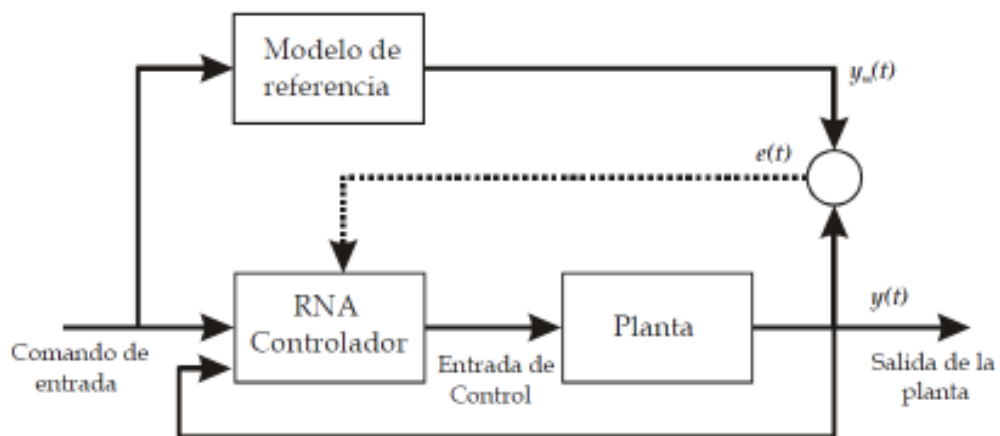


Figura 2.3 Red neuronal con modelo de referencia

Sin embargo, la calidad del control obtenido con esta aproximación depende crucialmente de la calidad del modelo del proceso. Si un modelo no es bastante exacto, el controlador neuronal entrenado es improbable que maneje satisfactoriamente el proceso real. Sin un componente adaptativo en línea, este controlador neuronal no podrá generar buenos resultados en la planta. Un controlador que es altamente optimizado para un proceso específico, no se puede esperar que tolere desviaciones del proceso nominal con ligereza.⁴

2.2 Control Difuso

Esta estrategia de control se basa en la lógica difusa, es decir, busca imitar la forma en que una persona tomaría decisiones según lo relativo de lo observado o experimentado. La lógica difusa proporciona la facilidad de resolver situaciones complejas y dinámicas, que son más fácilmente expresadas con palabras que con modelos matemáticos.

Un controlador difuso contiene un algoritmo que convierte una estrategia de control lingüística en una de control automático, mediante una base de reglas que genera un conjunto de expresiones del tipo:

Si <Condiciones> Entonces <Acciones>

En la figura 2.1 se muestra la estructura de funcionamiento de un controlador difuso.

⁴ Ochoa José, *Control Clásico y Control Inteligente*

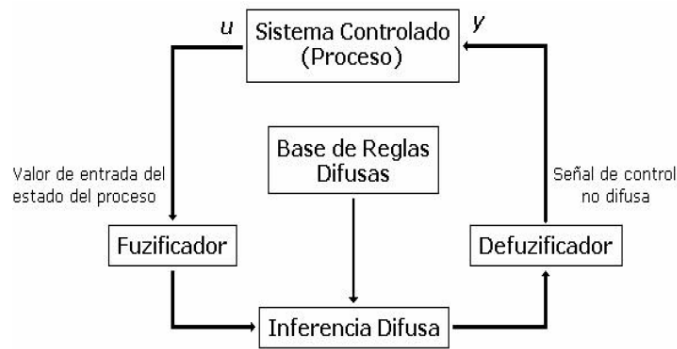


Figura 2.4 Arquitectura básica de un controlador difuso

- **Fusificación**

El objetivo de la fusificación es convertir los valores medidos del mundo real (entradas del estado del proceso) a un valor lingüístico. La fusificación se realiza utilizando las funciones de membresía de las variables lingüísticas para calcular el grado de pertenencia (grado de verdad) de cada término en los conjuntos difusos previamente definidos (Figura 2.2).

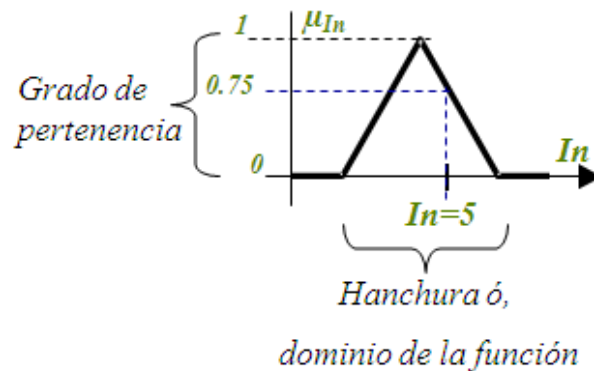


Figura 2.5 Asignación de grado de pertenencia en una función de membresía triangular

- **Base de Reglas**

Contiene todas las reglas lingüísticas que se utilizan para relacionar los conceptos imprecisos con el comportamiento del sistema a controlar, esta etapa se encarga de la toma de decisiones que gobernara la forma que debe actuar el sistema.

- **Inferencia**

La inferencia utiliza la informacion de la base de reglas para generar las condiciones del tipo: *Si caso1 y Caso 2 entonces Accion1.*

- **Defusificación**

Mediante las funciones de membrecía de la variable de salida se realiza el proceso de convertir los valores difusos generados, en los valores que posteriormente se utilizaran en el proceso de control.

2.3 Control Predictivo

El control predictivo basado en modelo o MPC, es una técnica de control que mediante la utilización de un modelo matemático interno del proceso, predice la evolución de las variables a controlar para calcular con un procedimiento de optimización, las señales de control que lleven al proceso a los objetivos deseados.

El intervalo de tiempo futuro que se considera en la optimización se denomina horizonte de predicción (Figura 2.3). El comportamiento futuro del sistema

depende de las actuaciones que se aplican a lo largo del horizonte de predicción, con lo que se logra un control en bucle abierto. El error debido a las diferencias entre el modelo y el comportamiento real del sistema crean la necesidad de incorporar realimentación del mismo. Esta realimentación se realiza mediante la técnica del horizonte deslizante, el horizonte de predicción se va deslizando a lo largo del tiempo resolviendo cada vez un nuevo problema de optimización.

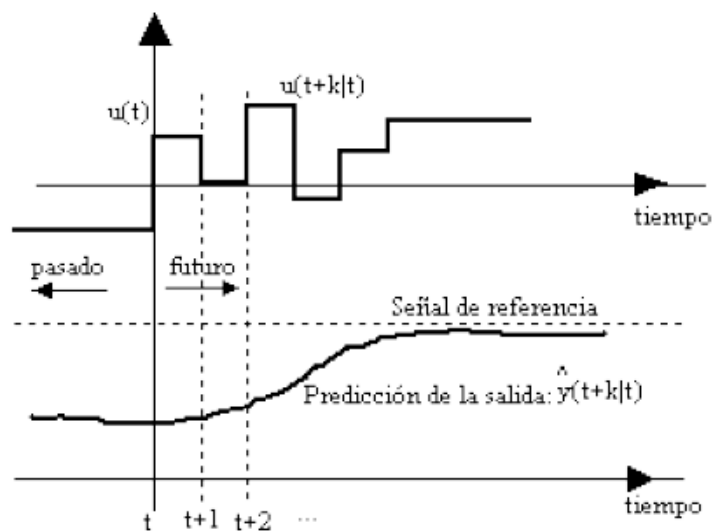


Figura 2.6 Principio básico del control predictivo basado en modelo

La figura 2.6 muestra el proceso básico de control predictivo que se explica a continuación:

- En cada instante t de tiempo, se predicen las salidas futuras para un horizonte de predicción N , usando un modelo del proceso a controlar. Estas salidas $\hat{y}(t+k|t)$ para $k = 1, 2, \dots, N$ dependen de las entradas y salidas pasadas, cuyos valores son conocidos y las señales de control futuras $u(t+k|t)$ para $k=1,2,\dots,N-1$ que son las que se calcularán y se enviarán al sistema.
- Las señales de control futuras serán calculadas por medio de la función de optimización, la cual se ajustará lo más cercano posible el proceso a la referencia dada $w(t+k|t)$.

- La señal de control $u(t|t)$ se envía al proceso, mientras que las demás señales de secuencia de control se eliminan. En el siguiente instante de muestreo se calcula la secuencia actualizada, esto se realiza porque ya se conoce el $y(t + 1)$ y se obtiene $u(t + 1|t + 1)$, haciendo uso del concepto de horizonte deslizante.

Una de las propiedades más atractivas del MPC es que permite la incorporación de distintos tipos de modelos de predicción, sean lineales o no lineales, monovariantes o multivariantes, y la consideración de restricciones sobre las señales del sistema. Esto hace que sea una estrategia muy utilizada en diversas áreas del control. El MPC es una de las pocas técnicas que permiten controlar sistemas con restricciones incorporando éstas en el propio diseño del controlador.

CAPITULO III

3 MODELAMIENTO DE LA PLANTA

3.1 Características de la unidad Air Flow Temperature Control System

La Figura 3.1 muestra un diagrama del módulo de temperatura, a continuación se describen sus componentes:

- **Elemento primario y de medición:** consta de un sensor-transmisor de temperatura tipo termistor. Toma la temperatura a la salida del calentador (20 – 70 [°C]), y envía una señal analógica de voltaje (0 – 5 [V]) hacia el controlador.
- **Conducto de flujo:** mediante un ventilador, renueva el aire que se encuentra en su interior.

- **Niquelina:** se encuentra en la entrada del conducto, es la encargada de calentar el aire que ingresa mediante una señal de voltaje de 0 a 10 [V]
- **Placa Metálica:** posee cuatro orificios, que determinan la cantidad de aire que ingresará al proceso.

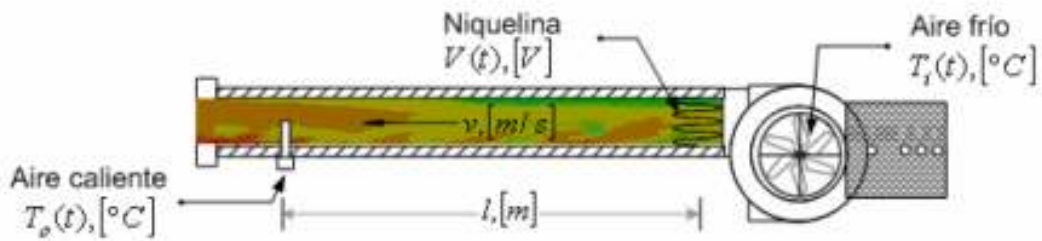


Figura 3.1 Unidad Air Flow Temperature Control System

Para asegurar que se mida una solución homogénea, el punto de supervisión se localiza a cierta distancia del punto de mezcla. El objetivo es conocer la forma en que responde la temperatura de salida, a los cambios en el voltaje de la niquelina en la entrada.

3.2 Modelo Matemático del Sistema

El flujo de calor en el sistema de temperatura se da por convección, es decir, se transfiere energía entre una superficie, que es la niquelina, y un fluido que se mueve sobre esta que en este caso es el aire.

Mediante el ventilador se fuerza al aire a fluir a través de la niquelina y el conducto, a esto se lo denomina convección interna forzada y la transferencia de calor viene descrita por la ley de Newton del enfriamiento:

$$Q = h A_s (T^\infty - T) \quad [\text{Ec 3.1}]$$

Donde:

h = Coeficiente de transferencia de calor por convección

A_s = Área superficial a través de la cual tiene lugar la transferencia de calor.

T^∞ = Temperatura de la niquelina

T = Temperatura del Aire

La transferencia de calor del sistema es en régimen transitorio, es decir, la temperatura no cambia sólo con la posición dentro del cilindro, sino que también cambia con el tiempo.

El calor acumulado en el cilindro viene dado por la siguiente fórmula:

$$Q = m C_p \frac{dT}{d\theta} \quad [\text{Ec 3.2}]$$

m = Masa del fluido

C_p = Calor específico del fluido (aire)

T = Temperatura del fluido

θ = tiempo

Igualemos las ecuaciones 3.1 con 3.2 y resolvemos para llegar a la ecuación que describe el sistema transiente de temperatura:

$$h As (T^\infty - T) = m Cp \frac{dT}{d\theta}$$

$$m Cp \frac{dT}{d\theta} - h As (T^\infty - T) = 0$$

$$t = T^\infty - T \quad ; \quad \frac{dT}{d\theta} = \frac{dt}{d\theta}$$

$$\frac{dt}{d\theta} + \frac{h As}{m Cp} t = 0$$

$$\frac{dt}{d\theta} = -\frac{h As}{m Cp} t$$

$$\int \frac{dt}{t} = - \int \frac{h As}{m Cp} d\theta$$

$$\ln t = -\frac{h As}{m Cp} \theta + \ln a1$$

$$\ln(T^\infty - T) = -\frac{h As}{m Cp} \theta + \ln a1$$

$$a1 = (T^\infty - T)$$

Condiciones Iniciales: $\theta = 0 ; T = T_0$

$$\ln(T^\infty - T) = -\frac{h As}{m Cp} \theta + \ln(T^\infty - T_0)$$

$$\ln\left(\frac{T^\infty - T}{T^\infty - T_0}\right) = -\frac{h As}{m Cp} \theta$$

$$\frac{T^\infty - T}{T^\infty - T_0} = e^{-\left(\frac{h As}{m Cp}\right)t}$$

$$T = T^\infty - (T^\infty - T_0)e^{-\left(\frac{h As}{m Cp}\right)t} \quad [\text{Ec. 3.3}]$$

La ecuación 3.3 describe el comportamiento de la temperatura en función del tiempo del sistema como se observa en la figura 3.2:

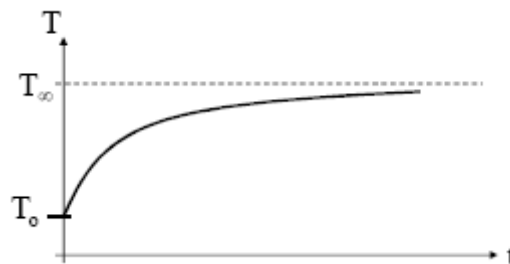


Figura 3.2 Comportamiento de la temperatura en función del tiempo.

Mediante el voltaje de entrada de la niquelina del sistema se manipula el voltaje AC que cae directamente en este elemento para generar calor:

$$Q = \frac{V^2}{R} \text{ [Ec. 3.4]}$$

Donde:

V = Voltaje AC que cae sobre la niquelina [V]

R = Resistencia eléctrica de la niquelina [Ohmios]

La constante que determina el retardo de la función de temperatura se calcula de:

$$\frac{h A_s}{m C_p}$$

Los valores de coeficiente de convección (h), área superficial (As), masa (m) y calor específico del aire (Cp) son conocidos, pero para un mejor ajuste del modelo matemático con la respuesta real del sistema, la constante de retardo se la determino mediante identificación como se explica a continuación.

3.3 Identificación

Para la identificación de la constante de retardo del modelo que es el parámetro desconocido es necesario conocer el comportamiento real de la planta de temperatura, en Simulink se diseñó un simple diagrama para que con la tarjeta de adquisición PCI 6221 guardar los datos de respuesta del sistema en lazo abierto como se muestra en la Figura 3.3

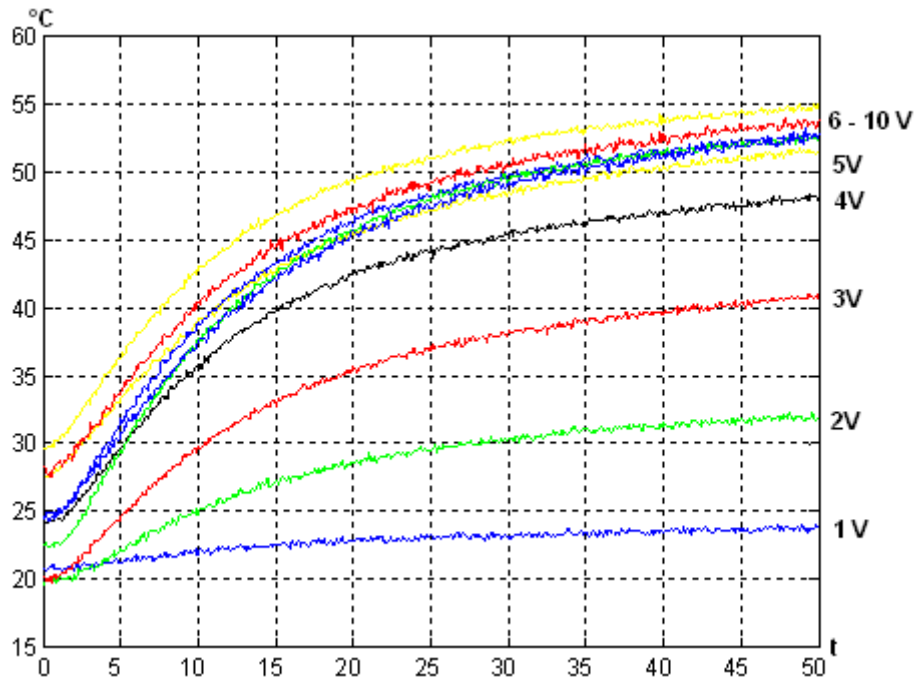


Figura 3.3 Respuestas de la planta de temperatura para diferentes voltajes en 50 segundos.

Se observó que para voltajes de entrada de 6 a 10 voltios la respuesta del sistema fue casi la misma, por lo que se redujo el rango de voltajes de 0 a 6 voltios.

La ecuación 3.3 que caracteriza al sistema expresada como función de transferencia en diagrama de bloques se aprecia en la Figura 3.6, donde la entrada es el voltaje de 0 a 6 voltios, la salida la temperatura en grados centígrados, K la ganancia del sistema y ζ la constante de retardo que será identificada.

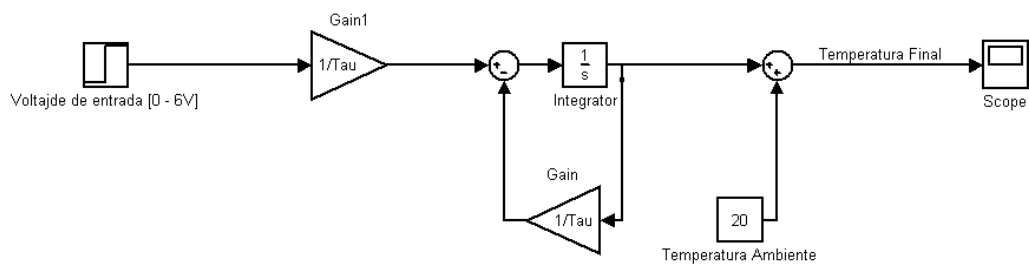


Figura 3.4 Diagrama de simulación del modelo matemático

La ganancia del sistema (K) relaciona el voltaje de entrada con la temperatura que es la salida del modelo. Por ejemplo para 2 voltios la temperatura alcanzó 31.7°C (Figura 3.5), la temperatura subió 11.7 grados desde la temperatura ambiente (20°C) por lo que la ganancia para este valor es de 5.85.

La ganancia para cada voltaje de entrada se obtuvo usando el máximo valor que alcanzó la temperatura para cada muestra que se tomó, estos datos se los muestra en la Tabla 3.1

Voltios	Ganancia (K)
1	3,57
2	5,87
3	7,06
4	6,94
5	6,29
6	5,43

Tabla 3.1 Valores de ganancia para cada voltaje de entrada

Se puede observar en la Figura 3.6, la variación de la ganancia con respecto al voltaje de entrada.

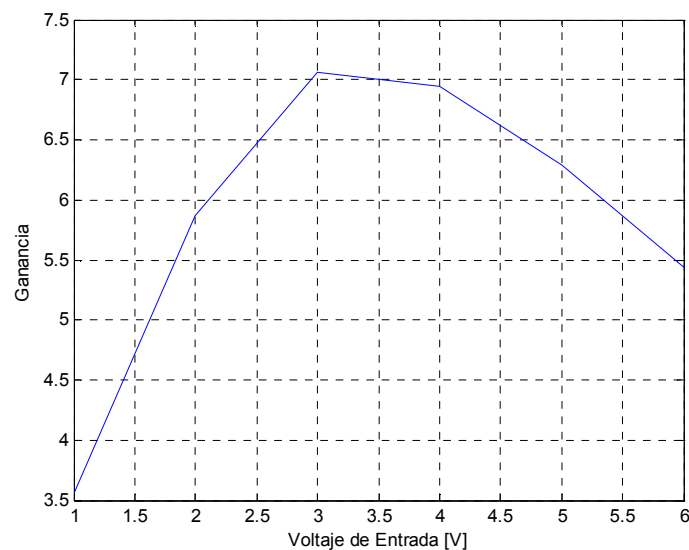


Figura 3.5 Gráfica de ganancia del Modelo con respecto al voltaje de entrada

Para introducir la variación de la ganancia en el modelo, se ajustó esta curva a un polinomio de segundo orden y se creó una función en Matlab que recibe el voltaje de entrada y nos devuelve el valor de la ganancia para este valor.

El polinomio de ajuste es el siguiente: $-0.414 x^2 + 3.215 x + 0.885$

Para identificar el parámetro desconocido que es la constante de retardo, se desarrollo un programa que mediante el cálculo integral del error cuadrático, nos entrega el valor de la constante que mas ajusta las respuestas del modelo de simulación con los datos guardados de la planta. Así por ejemplo para 3 voltios de entrada, la constante de retardo resulto 13, con este valor en la Figura 3.8 podemos observar un comportamiento similar entre respuesta de la planta (línea continua), con la gráfica que resulta del modelo de simulación (línea cortada).

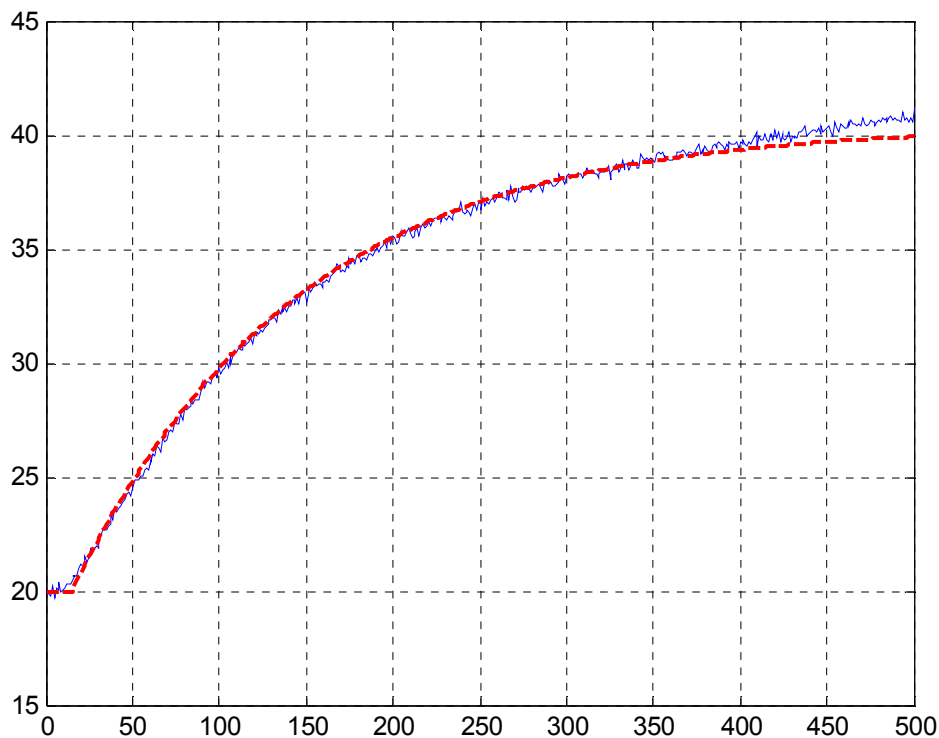


Figura 3.6 Comparación de respuestas del modelo de simulación con la planta para un voltaje de entrada de 3V.

Este procedimiento se lo realizó para cada muestra obtenida en la recolección de datos, la constante de retardo es el promedio de la identificada para cada voltaje de entrada como se indica en la Tabla 3.2:

Vin [V]	ζ
1	14
2	14
3	14
4	12
5	13
6	14
Promedio	13.5

Tabla 3.2 Valores de la constante de retardo para cada valor de voltaje de entrada

De esta manera completamos el modelo de simulación como se muestra en la Figura 3.9:

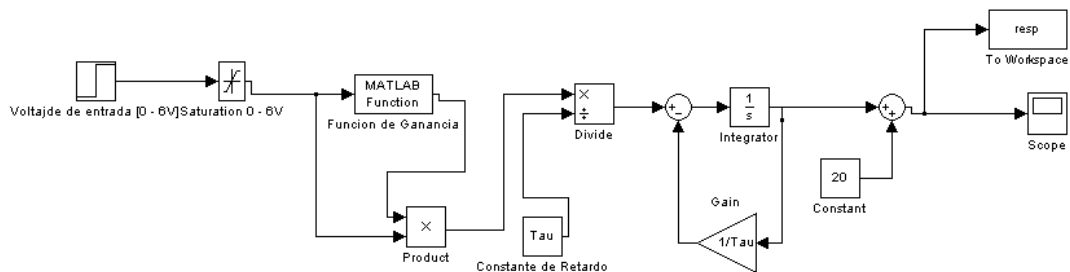


Figura 3.7 Diagrama de simulación del modelo matemático con la función de la ganancia y la constante de retardo identificada.

Este modelo se usó en este proyecto para la simulación de los controladores desarrollados como se puede apreciar en los siguientes capítulos.

CAPITULO IV

4 DISEÑO DE LOS CONTROLADORES INTELIGENTES

4.1 Controlador Difuso

4.1.1 Especificaciones del Problema

En este tema se describe el diseño de un controlador difuso en Matlab, el objetivo es estabilizar la temperatura del sistema a los valores deseados utilizando descripciones lingüísticas en el rango de 20 a 70°C, este principio es mucho más sencillo que los métodos de control clásico.

4.1.2 Solución del Problema

El control difuso para el sistema de temperatura funciona en base a un voltaje de referencia, a partir de este voltaje el controlador lo aumenta o disminuye según la respuesta de la planta, es decir, el voltaje de referencia con la salida del controlador forman la señal de control (Figura 5.1).

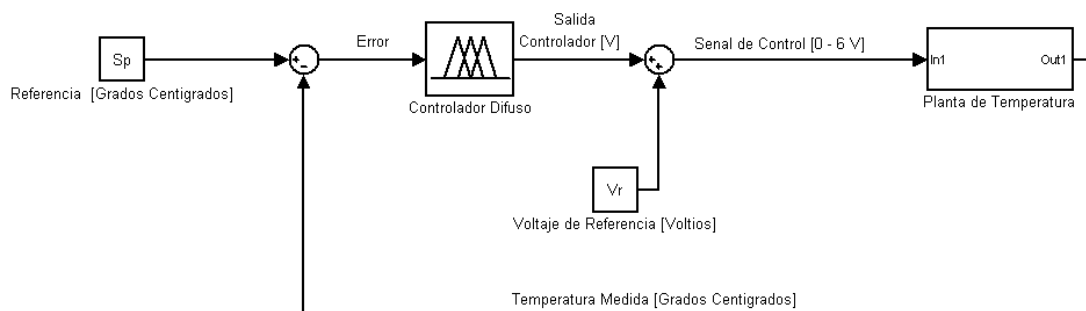


Figura 4.1 Diagrama del Controlador Difuso

En principio el voltaje de referencia que se usó fue el resultado de la siguiente relación lineal:

$$\begin{aligned} 0 &- 10 [V] \\ 20 &- 70 [^{\circ}C] \end{aligned}$$

De lo que resulta la ecuación:

$$\text{Voltaje de referencia} = \frac{(\text{Temperatura} - 20)}{5} [v] \quad [\text{Ec 5.1}]$$

Al usar esta relación, el voltaje de referencia resultante generaba una temperatura mucho mayor a la deseada, esto se debe a que la planta de temperatura describe un sistema no lineal, como se explicó en el Capítulo 3, por lo que se restó 2 voltios a esta relación para disminuir la temperatura generada, así la relación usada nos queda:

$$\text{Voltaje de referencia} = \frac{(\text{Temperatura}-20)}{5} - 2 \quad [v] \quad [5.2]$$

Con este valor se logra un control de lazo abierto adecuado.

Para diseñar el controlador difuso se utilizó el FIS Editor de Matlab (Figura 5.2), a continuación se describen los pasos para obtener el controlador.

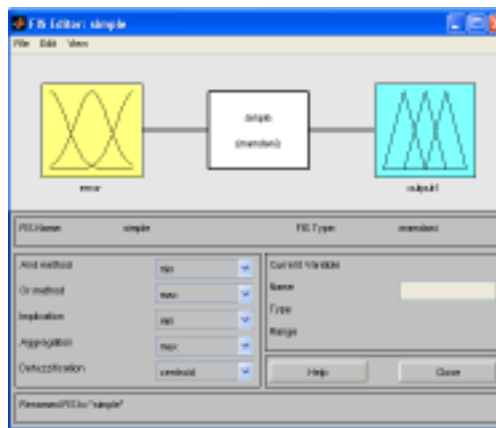


Figura 4.2 FIS Editor de Matlab

- **Rango de Operación de las Variables de Entrada y Salida:**

En este controlador solo se tiene una variable de entrada que es el error, los límites de operación: -5 a 5 grados. La variable de salida que se sumará con el voltaje de referencia es de -5 a 5 Voltios.

- **Definición de las Funciones de Pertenencia Fuzzificación:**

El controlador tiene dos funciones de pertenencia, el error y el voltaje de salida.

El error se define en cinco valores lingüísticos:

- Grande positivo (GP)
- Pequeño positivo (PP)
- Cero (Z)
- Pequeño negativo (PN)

➤ Grande negativo (GN)

Cuando la temperatura medida es igual a la deseada, el error tendrá un valor de 0, correspondiente a la variable lingüística Cero.

Cuando la temperatura medida supere a la deseada, el error será negativo, y podría pertenecer a las variables lingüísticas: Cero, Pequeño negativo o Grande negativo según su valor.

Si la temperatura medida es menor a la deseada, el error será positivo, y podría pertenecer a las variables lingüísticas: Cero, Pequeño positivo o Grande positivo según su valor.

En la Figura 5.2 se puede apreciar la función de pertenencia del error.

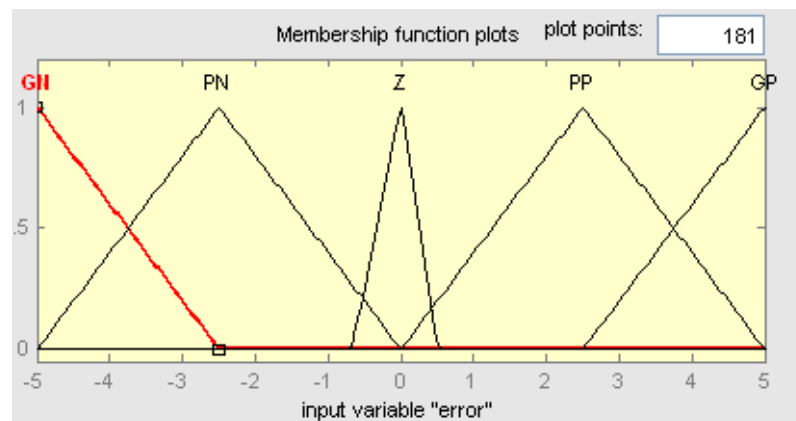


Figura 4.3 Función de pertenencia del error

La variable de salida del controlador se define en cinco valores lingüísticos:

- Calentar mucho (CM)
- Calentar poco (CP)
- Cero (Z)

- Enfriar poco (EP)
- Enfriar mucho (EM)

La variable de salida del controlador modificará el voltaje de referencia para controlar la temperatura del sistema.

En la Figura 5.3 se puede apreciar la función de pertenencia de la salida del controlador.

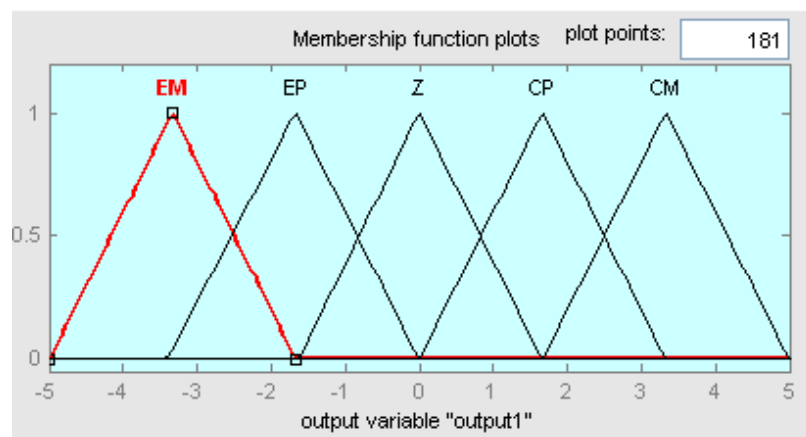


Figura 4.4 Función de pertenencia de la salida del controlador.

- **Desarrollo de la base de reglas**

La base de reglas consta de cinco reglas donde se relacionan las premisas de las variables de entrada con las premisas de la variable de salida, como por ejemplo: *Si el error es grande positivo, la salida del controlador será Calentar mucho.*

En la tabla 5.1 se muestra la base de reglas

N°	Regla
1	Si (Error es grande positivo) entonces (Salida es Calentar mucho)
2	Si (Error es pequeño positivo) entonces (Salida es Calentar poco)
3	Si (Error cero) entonces (Salida es cero)
4	Si (Error es pequeño negativo) entonces (Salida es Enfriar poco)
5	Si (Error es Grande Negativo) entonces (Salida es Enfriar mucho)

Tabla 4.1 Base de reglas del controlador difuso

4.1.3 Simulación

Para la simulación del control difuso, se necesita exportar el controlador desarrollado en el FIS Editor hacia el espacio de trabajo de Matlab, en el diagrama de Simulink insertar el bloque del controlador difuso ubicado en la librería Fuzzy Logic Control y cargar el nombre del controlador exportado. El diagrama de simulación resultante del controlador difuso diseñado se puede apreciar en la Figura 5.5

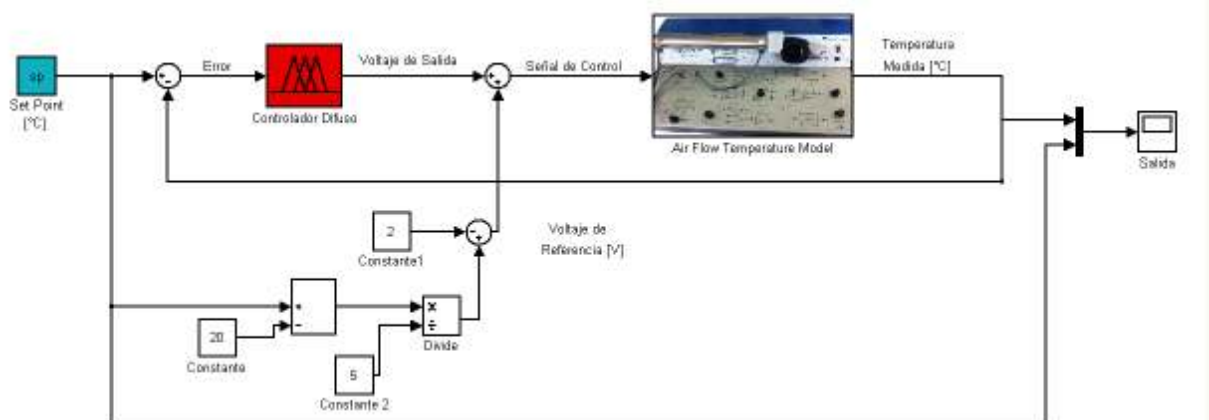


Figura 4.5 Diagrama de simulación del controlador Difuso

En la Figura 5.6 podemos ver la respuesta del controlador para diferentes valores de referencia en 350 segundos.

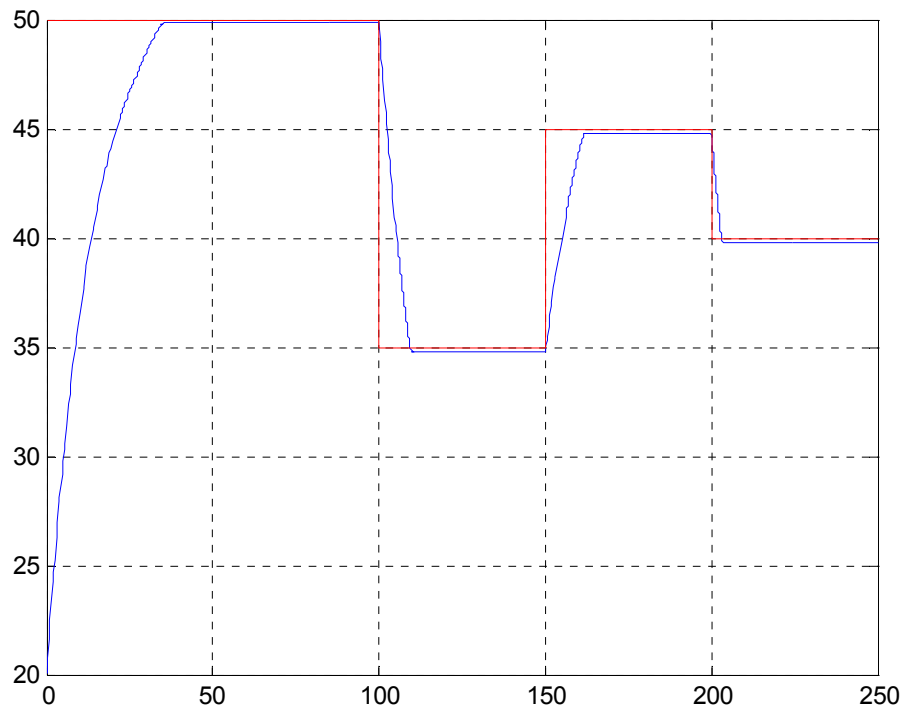


Figura 4.6 Respuesta de la simulación del controlador difuso para diferentes valores de referencia

En la Figura 4.7 se observa la señal de control del controlador difuso para un set point de 35°C con una temperatura inicial de 20°C en función del tiempo.

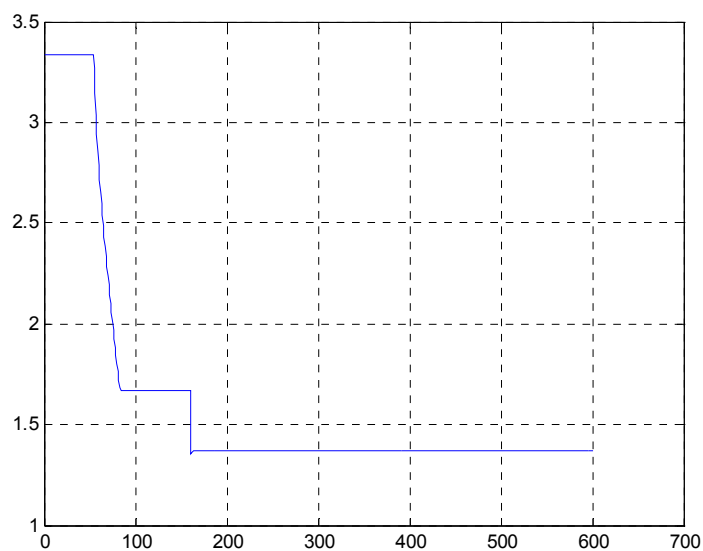


Figura 4.7 Señal de control del controlador difuso para un set point de 35°C con una temperatura inicial de 20°C

La Figura 4.8 muestra la señal de control del controlador difuso para un set point de 40°C con una temperatura inicial de 55°C en función del tiempo.

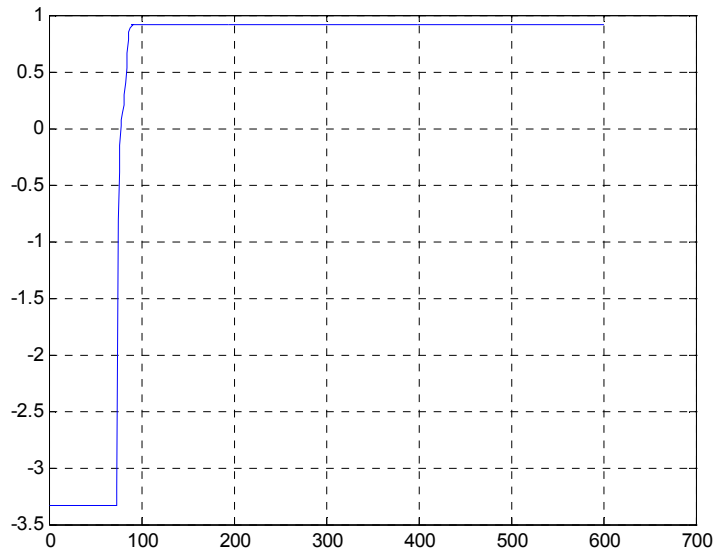


Figura 4.8 Señal de control del controlador difuso para un set point de 40°C con una temperatura inicial de 55°C

4.1.4 Implementación

Para la implementación del controlador difuso se utiliza el modulo real de la planta de temperatura, esta es la única diferencia con el controlador simulado, en vez de utilizar el modelo matemático se utiliza la entrada analógica de la tarjeta de adquisición para obtener la temperatura del sistema y la salida analógica para enviar la señal de control, la temperatura del sistema se recibe en valores de 0 a 5 voltios por lo que se realiza su respectiva transformación a grados centígrados. Este diagrama se lo puede observar en la Figura 5.7

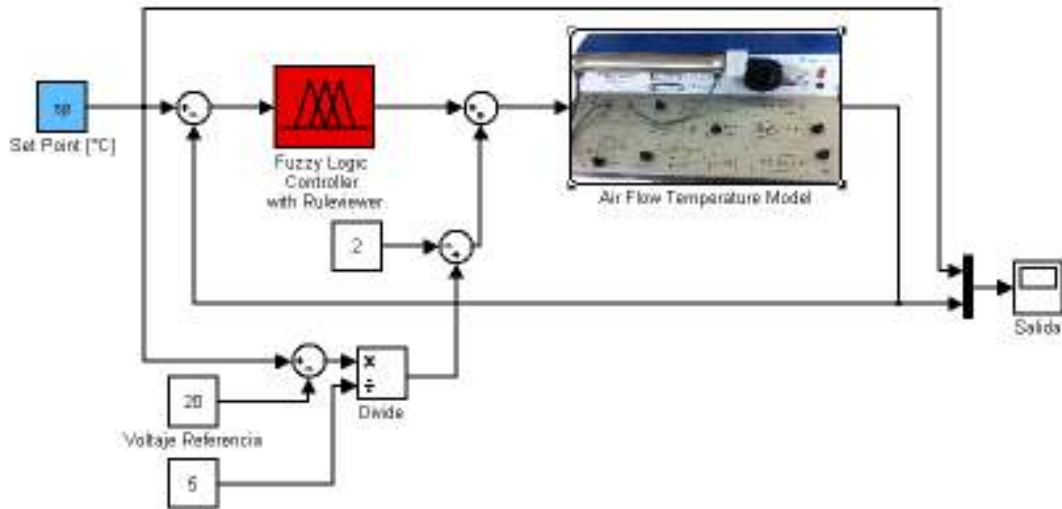


Figura 4.9 Diagrama del Controlador Difuso Implementado

La figura 5.8 nos muestra el desempeño del controlador difuso para tres diferentes valores de referencia en 350 segundos.

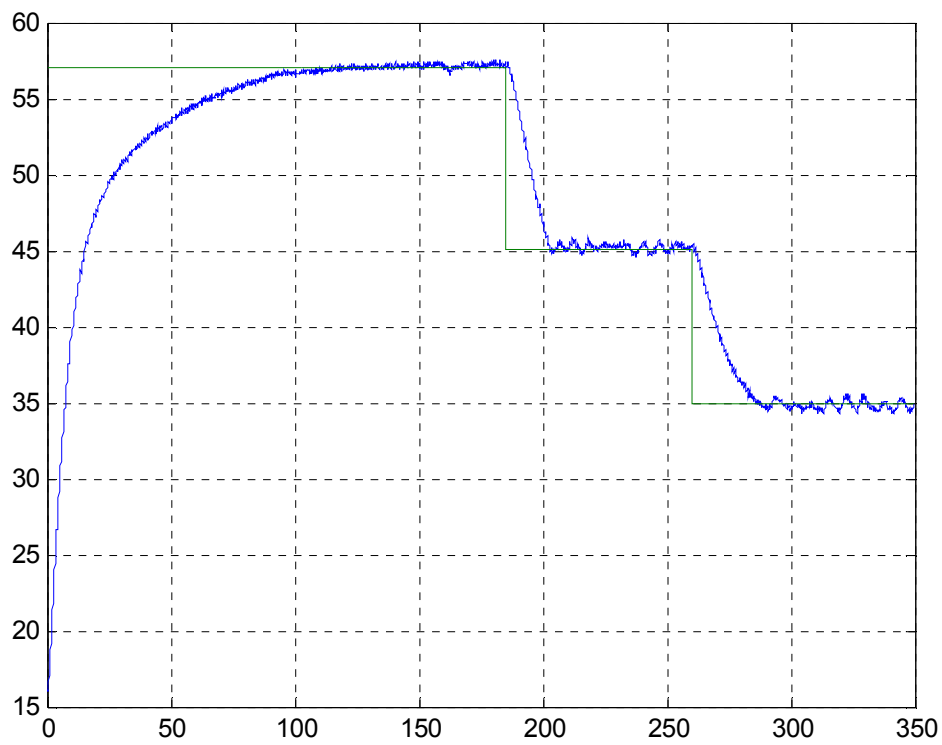


Figura 4.10 Desempeño del controlador difuso con la planta de temperatura real para diferentes valores de referencia

4.2 Controlador Predictivo

4.2.1 Especificaciones del problema

El objetivo de este capítulo es detallar el procedimiento de diseño de un controlador predictivo en Matlab para la unidad *air flow temperature control system*. Con esta técnica de control se pretende optimizar las señales de control que lleven al proceso a los objetivos deseados, mediante el concepto predicción del comportamiento de la planta.

4.2.2 Solución del Problema

El control predictivo funciona en base a la función de transferencia del proceso, por lo que se debe linealizar el sistema a un punto de operación. Para obtener la función de transferencia de la planta de temperatura se utilizó la herramienta Ident de Matlab con los valores de respuesta guardados del modelo matemático para un voltaje de entrada de 5 V con un tiempo de muestreo de 0.1 segundos, el diagrama para la obtención de estos datos se puede ver en la Figura 6.1.

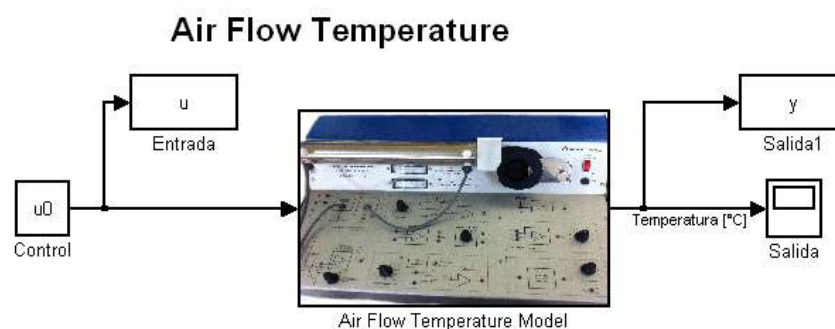


Figura 4.11 Diagrama para registrar la respuesta del modelo matemático a 5 V.

Para importar los datos obtenidos a la herramienta Ident realizamos los siguientes pasos (Figura 6.2):

- En la ventana de comandos de Matlab digitamos:

$$data=iddata(y,u,Ts)$$
 Donde y es la salida del modelo, u la entrada y Ts el tiempo de muestreo en segundos.
- En la herramienta Ident en la Pestaña “Import data” seleccionamos: “Time domain data”
- Posteriormente se abre la ventana “Import Data” y en la opción “Object” escribimos el nombre de la variable que en nuestro caso es “data”.
- Damos click en el botón Import y luego Close.

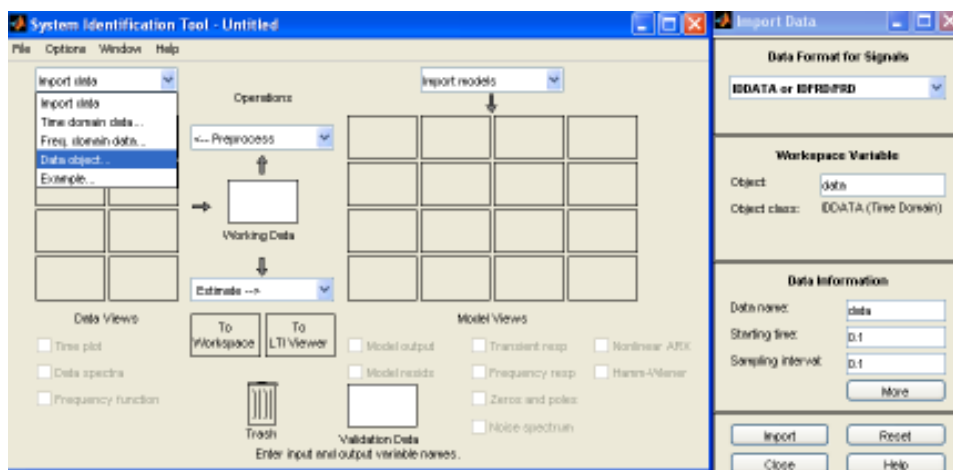


Figura 4.12 Carga Del Datos En La Herramienta Ident

Con los datos importados se utilizó la opción de estimación: “Process Model”, exportamos la función de transferencia identificada al espacio de trabajo arrastrando al cuadro “To Worksapce”.

La función transferencia identificada es:

$$G_s = \frac{10.614}{1 + 13.5 s}$$

4.2.2.1 Diseño del Controlador Mediante la Herramienta MPCTOOL

Para diseñar el controlador en la herramienta MPCTOOL es necesario realizar los siguientes pasos:

- **Importar la función de transferencia de la planta de temperatura**

Para importar función de transferencia anteriormente identificada digitamos en la ventana de comandos de Matlab:

$$Gs = tf([10.614],[1 13.5])$$

- **Ingresar datos de la planta de temperatura**

En la tabla 6.1 se muestra las características de la planta que se deben ingresar:

Nombre	Tipo	Descripción	Unidad	Valor Nominal
Voltaje	Variable Manipulada	Entrada	V	0
Temperatura	Variable Medida	Salida	°C	0

Tabla 4.2 Datos de la entrada y salida de la planta de temperatura

- **Diseñar el control predictivo**

Los parámetros del control predictivo que se deben ingresar en la herramienta MPCTOOL se describen en la tabla 6.2

Parametro	Detalle	Valor
Modelo y Horizonte	Intervalo de Control	0.1
	Horizonte de Prediccion	10
	Horizonte de Control	2
Restricciones	Entrada : Voltaje	Min: 0 Max: 6
	Salida: Temperatura	Min: 20 Max: 70
Pesos	Overall	0.8
	Varianza del peso	0.1
	Temperatura	1

Tabla 4.3 Datos de los parámetros del controlador Predictivo

4.2.3 Simulación

Para simular con el modelo matemático de la planta es necesario exportar al espacio de trabajo de MATLAB el controlador diseñado en la herramienta MPCTOOL, en el diagrama de Simulink se debe insertar el bloque del controlador predictivo ubicado en la librería Model Predictive Control y cargar el nombre del controlador exportado. El diagrama de simulación resultante del controlador predictivo diseñado se puede apreciar en la Figura 6.3.

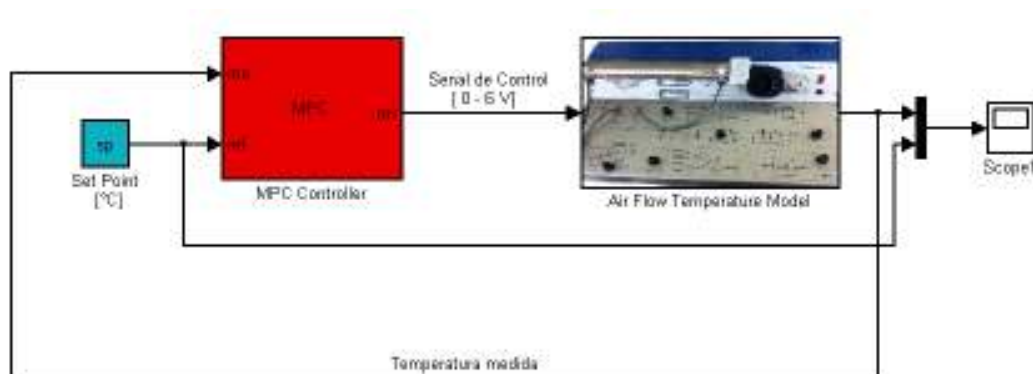


Figura 4.13 Diagrama de simulación del controlador predictivo

En la Figura 6.4 podemos ver la respuesta del controlador para diferentes valores de referencia en 350 segundos.

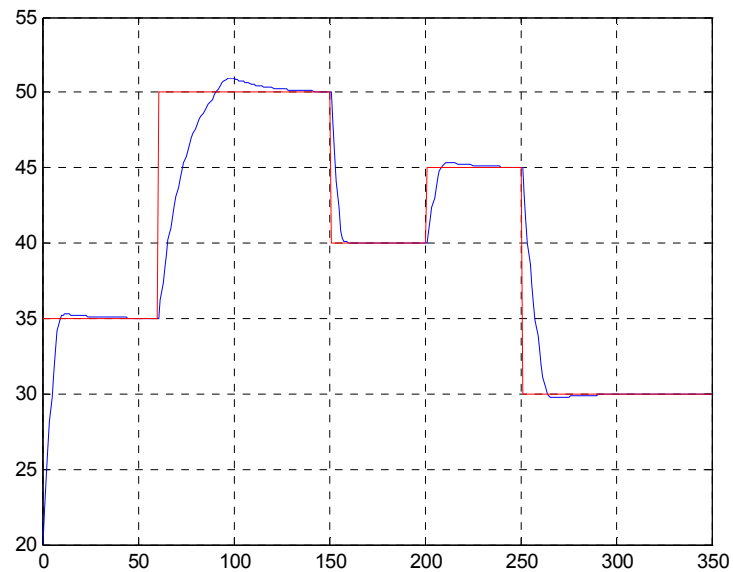


Figura 4.14 Respuesta de la simulación del controlador predictivo para diferentes valores de referencia

En la Figura 4.7 se observa la señal de control del controlador difuso para un set point de 35°C con una temperatura inicial de 20°C en función del tiempo.

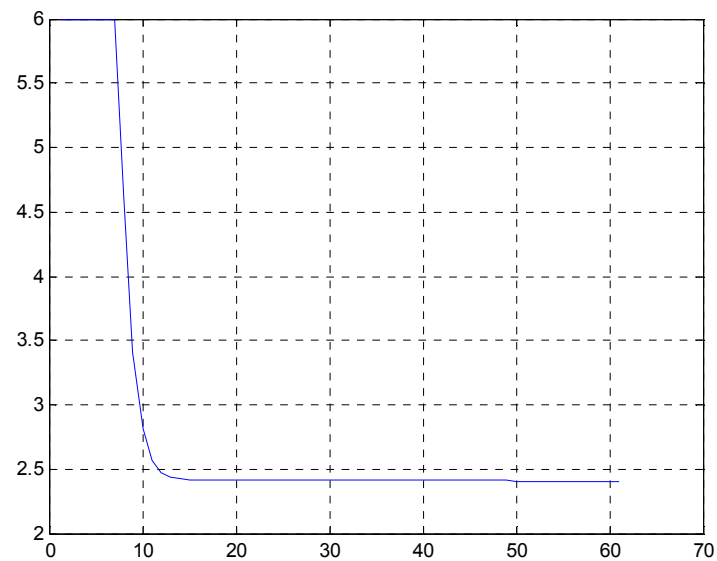


Figura 4.15 Señal de control del controlador predictivo para un set point de 35°C con una temperatura inicial de 20°C

La Figura 4.8 muestra la señal de control del controlador difuso para un set point de 40°C con una temperatura inicial de 55°C en función del tiempo.

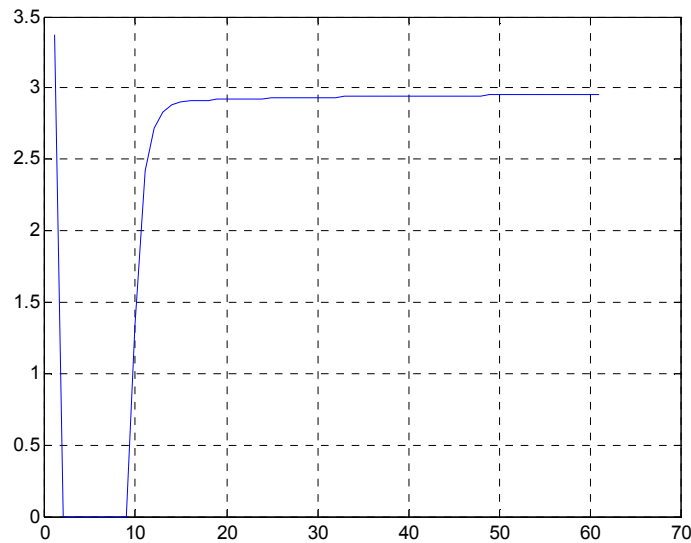


Figura 4.16 Señal de control del controlador predictivo para un set point de 40°C con una temperatura inicial de 55°C

4.2.4 Implementación

Para la implementación del controlador predictivo se utiliza el módulo real de la planta de temperatura, esta es la única diferencia con el controlador simulado, en vez de utilizar el modelo matemático se utiliza la entrada analógica de la tarjeta de adquisición para obtener la temperatura del sistema y la salida analógica para enviar la señal de control, la temperatura del sistema se recibe en valores de 0 a 5 voltios por lo que se realiza su respectiva transformación a grados centígrados. Este diagrama se lo puede observar en la Figura 6.4

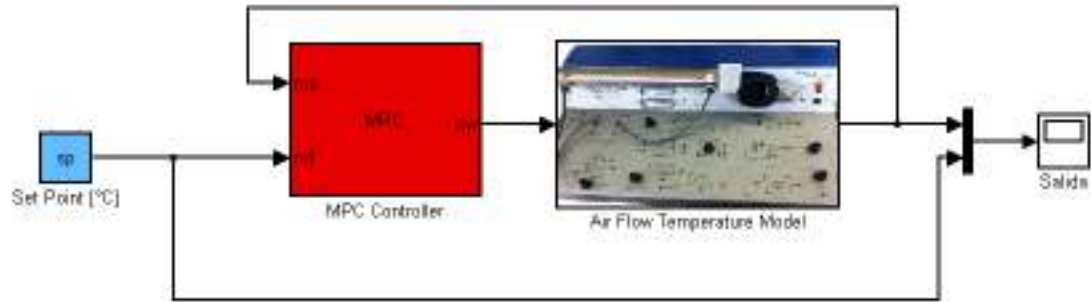


Figura 4.17 Diagrama del controlador predictivo implementado

En la figura 6.6 podemos ver el desempeño del controlador predictivo implementado en la planta real para diferentes valores de referencia en 350 segundos.

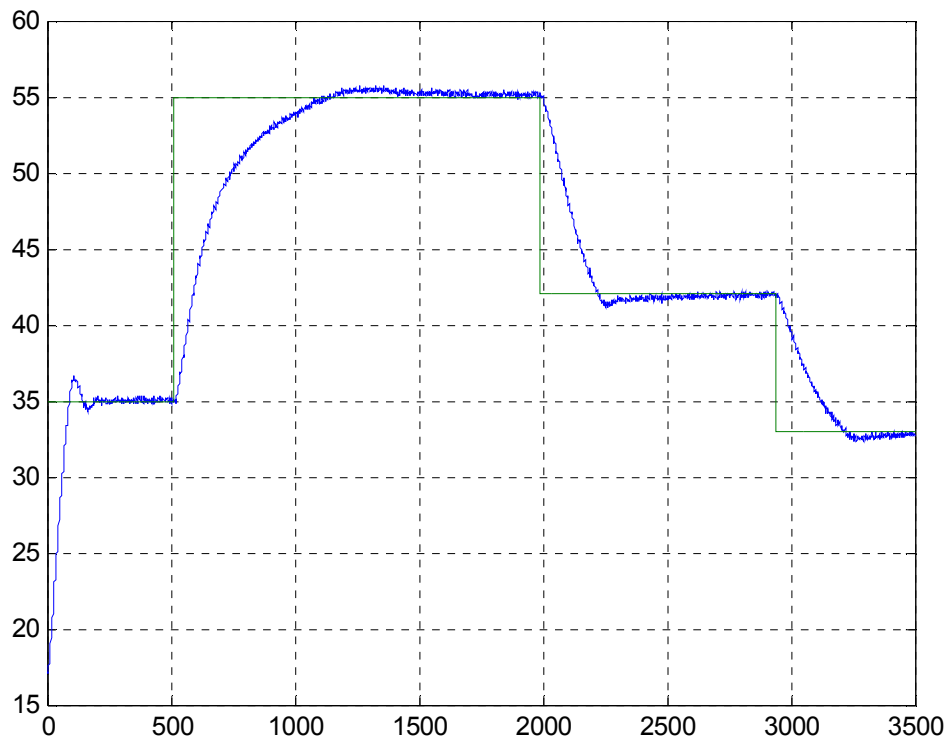


Figura 4.18 Desempeño del controlador predictivo en la planta real de temperatura para diferentes valores de referencia

4.3 Controlador Neuronal Indirecto por Modelo Inverso

4.3.1 Especificaciones del problema

El objetivo es generar un sistema de control mediante una red neuronal inversa. Para generar el controlador, se debe entender que dado un estado inicial T_0 (Temperatura inicial) se genera una señal de control u para con esto obtener las variaciones de estado deseadas.

4.3.2 Solución del Problema

Toda red neuronal comprende dos etapas importantes: operación y aprendizaje.

Operación: Se desea una red neuronal con dos entradas, la temperatura o el estado inicial o actual del sistema y la variación de estado deseada de la temperatura. La salida es la señal de control.

Aprendizaje: Para el aprendizaje (entrenamiento) los patrones se obtienen mediante la simulación del procesos mediante un tiempo de procesamiento con un conjunto de entradas que operan todas las posibilidades del estado inicial y la señal de entrada. Como resultado se obtiene la variación del estado del sistema.

El controlador de red neuronal inverso actúa en forma muy similar a la planta pero en forma inversa, es por esto que en el controlador tenemos como entrada la variación de estado y como salida la señal de control.

Lo que se desea es que la entrada T_0 , que es el estado o temperatura inicial coincida después de un poco de procesamiento con la temperatura de referencia, es decir que la variación de estado deseada será:

$$\Delta t = t_{ref} - t_0 \quad [\text{Ec 4.1}]$$

Donde:

Δt	Variación de estados
t_{ref}	Temperatura de referencia
t_0	Estado inicial

- **Obtención de los patrones de entrenamiento**

Previo a realizar la red de control se debe identificar la planta, para esto se debe utilizar el modelo matemático en lazo abierto para obtener los valores objetivos.

La red neuronal de control tiene como objetivo mantener la temperatura, manteniendo la variable de estado en el valor deseado, para esto se debe describir posibles valores que puede tomar tanto el estado como el valor de referencia:

%RANGOS DE LAS VARIABLES DEL SISTEMA

entrada=0:0.2:6; %Temperatura de referencia

estado=20:1:60; %Estado inicial

Lo siguiente será crear una variable Pm que tendrá todas las posibles combinaciones entre el valor de referencia y el estado para así simular en el sistema dinámico con una variación de tiempo de simulación (tiempo de paso) igual a 2.

```
Pm = combvec(estado,entrada);
```

```
timestep = 2;
```

A continuación se debe crear la variable Tm, este vector contiene la variación que se produce en la variable de estado ante las condiciones de entrada de los estados actuales. Cabe recalcar que este vector Tm no es el vector objetivo.

```
Q = length(Pm);
Tm = [];
%Identificacion
for i=1:Q
    entrada = Pm(2,i);
    to = Pm(1,i)-20;
    to1 = Pm(1,i);
    sim('subingpro',[0 timestep]); %modelo no lineal
    dt = tf-to1;
    Tm=[Tm [dt]];
end
```

- **Entrenamiento de la red neuronal**

Los patrones de entrenamiento para la red de control deben tener en consideración que el controlador debe funcionar en forma inversa a la planta. En el modelo matemático de la planta se tiene como entradas el estado inicial y el voltaje de entrada o señal de control, mediante estos valores de entrada obtenemos una variable de salida que es temperatura o una variación de temperatura.

En el controlador teniendo en cuenta que es de forma inversa, se tiene como entradas el estado inicial y la variación de temperatura y como salida una señal de control.

Para empezar el aprendizaje, se debe volver a ordenar los vectores Pm y Tm, así el vector Pm deberá contener las variables que se desea que sean entradas y el vector Tm contendrá las variables que se desea que sean salidas en el controlador, y se las describe mediante código de la siguiente manera:

```
Targ=Pm(2,:);
Pm=Pm(1,:);
Pm=[Pm;Tm];
Tm=Targ;
```

4.3.3 Simulación



Figura 4.19 Diagrama de simulación del controlador neuronal inverso

Se realizó varias pruebas del controlador neuronal inverso con varias temperaturas de entrada y siempre con un estado inicial de 20°C.

Prueba 1

Temperatura = 35°C

Salida:

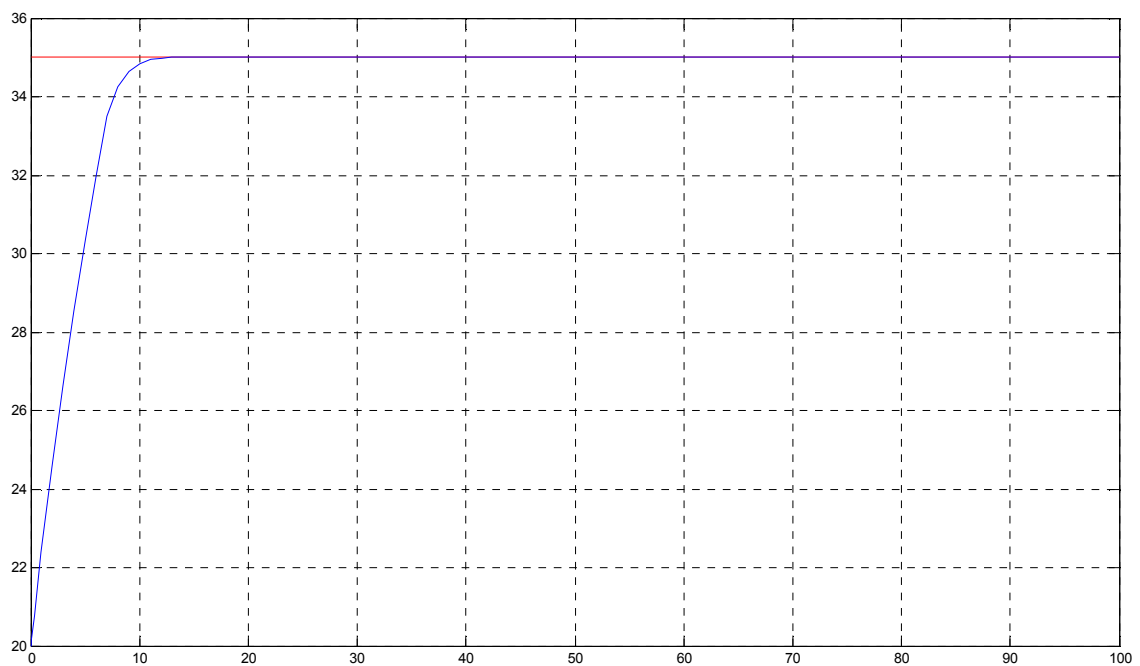


Figura 4.20 Respuesta en el tiempo de la simulación del controlador neuronal inverso con entrada de 35°C.

Señal de control:

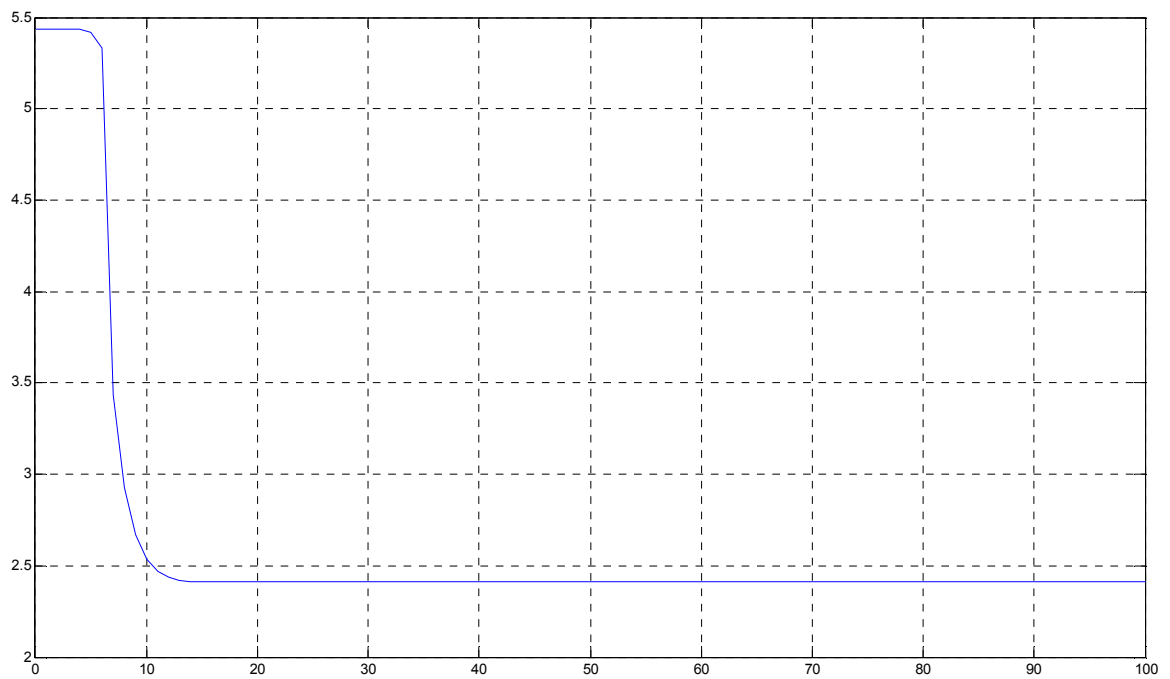


Figura 4.21 Señal de control del controlador neuronal inverso en simulación.

Prueba 2

$T_0=20^{\circ}\text{C}$

Temperatura 45°C

Salida



Figura 4.22 Respuesta en el tiempo de la simulación del controlador neuronal inverso con entrada de 45°C .

Señal de control

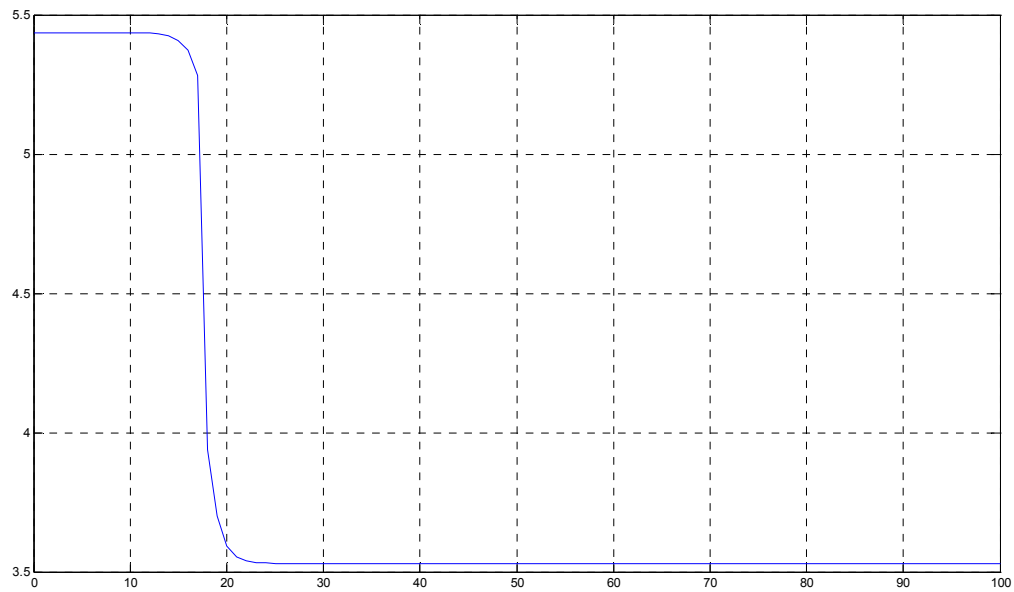


Figura 4.23 Señal de control del controlador neuronal inverso en simulación.

Prueba 3

$T_0=40^{\circ}\text{C}$

Temperatura= 28°C

Salida

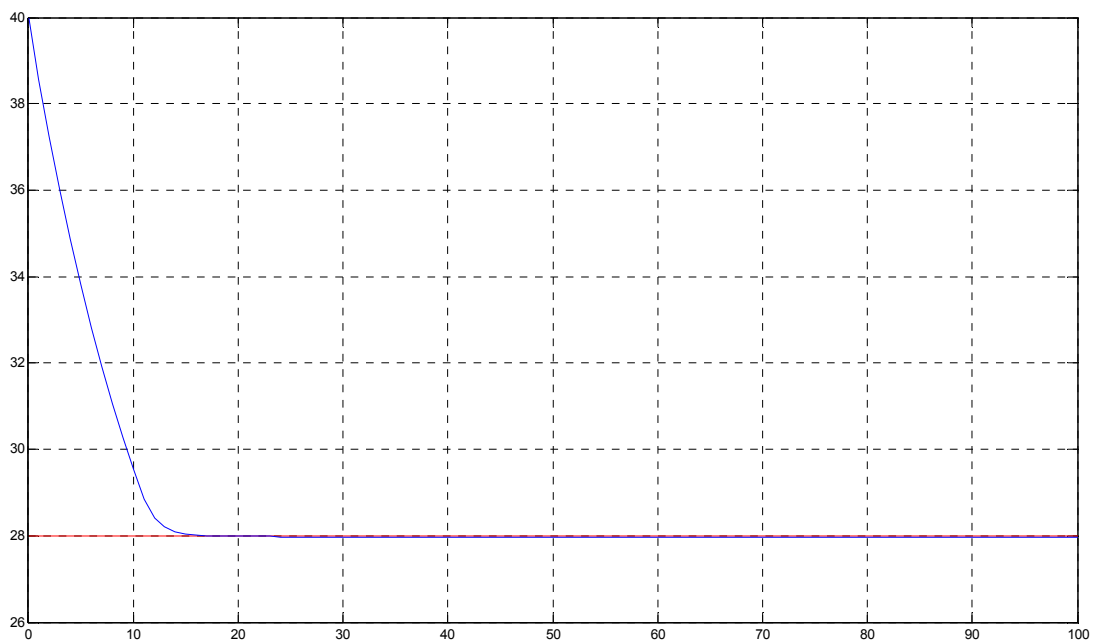


Figura 4.24 Respuesta en el tiempo de la simulación del controlador neuronal inverso con entrada de 28°C .

Señal de control

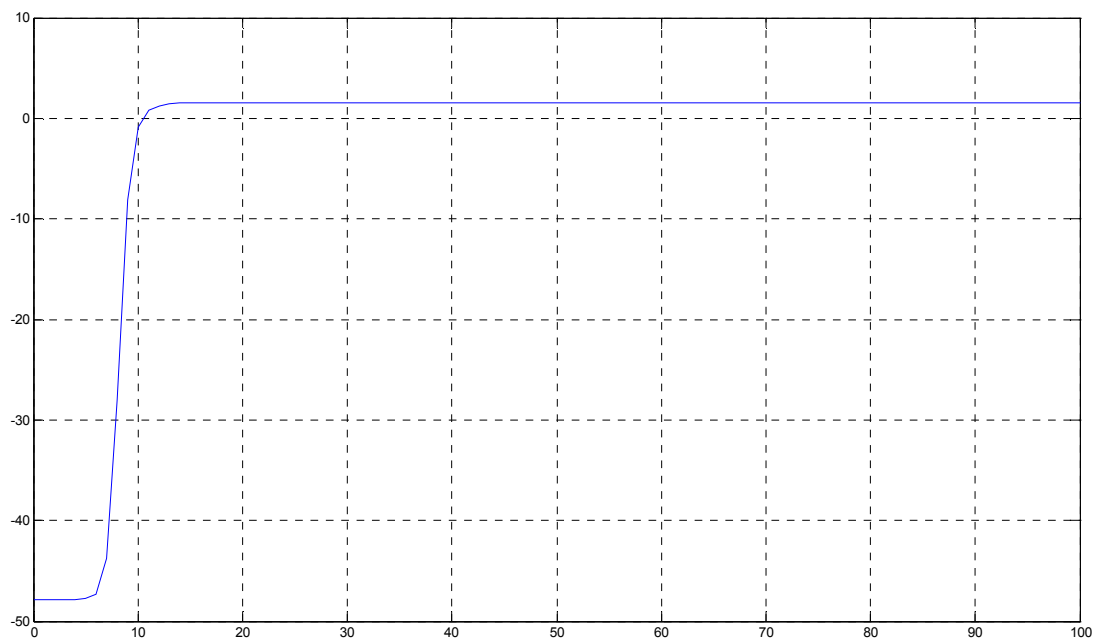


Figura 4.25 Señal de control del controlador neuronal inverso en simulación

4.3.4 Implementación

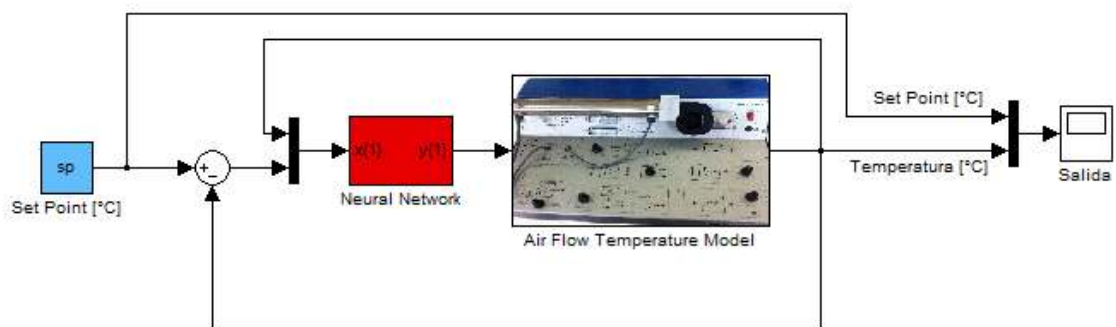


Figura 4.26 Diagrama para la implementación del control neuronal inverso en la planta

Se realizó varias pruebas usando Simulink con nuestro modelo de implementación como se muestra en la figura, cada una de ellas con diferentes temperaturas de referencia, pero además con un diferente tiempo de paso, con valores de tiempo de paso de 1, 2, 5, 10 y 20 segundos. Con estos resultados podremos ver cuál fue la mejor identificación de nuestro sistema y si nuestro controlador inverso fue el más adecuado.

Prueba con tiempo de paso de segundo:

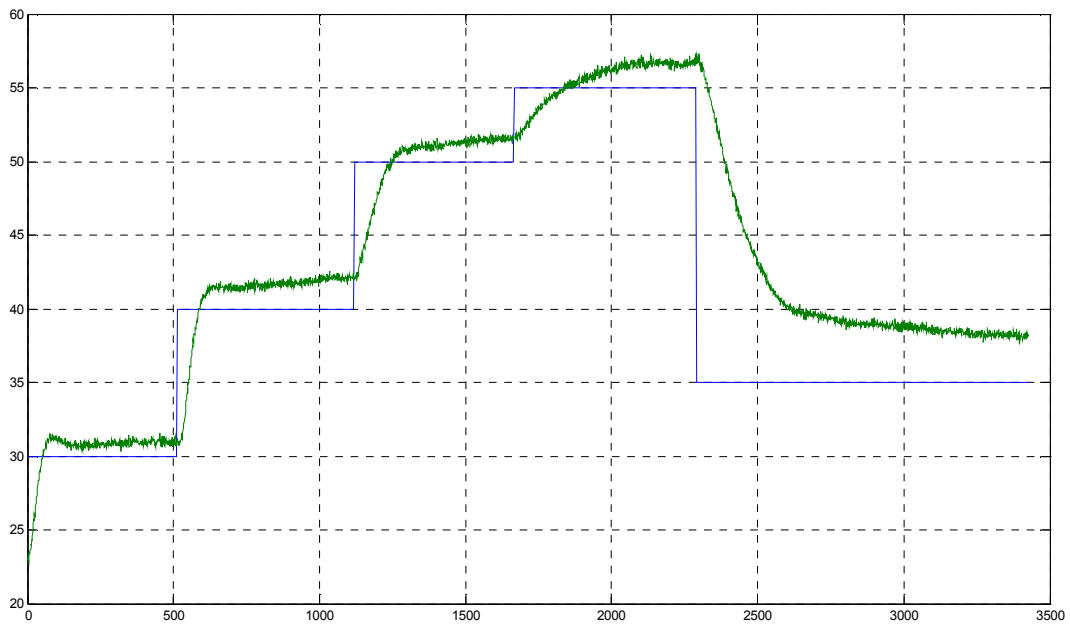


Figura 4.27 Prueba de la implementación del controlador neuronal inverso con un tiempo de paso de un segundo.

Prueba con tiempo de paso de dos segundos:

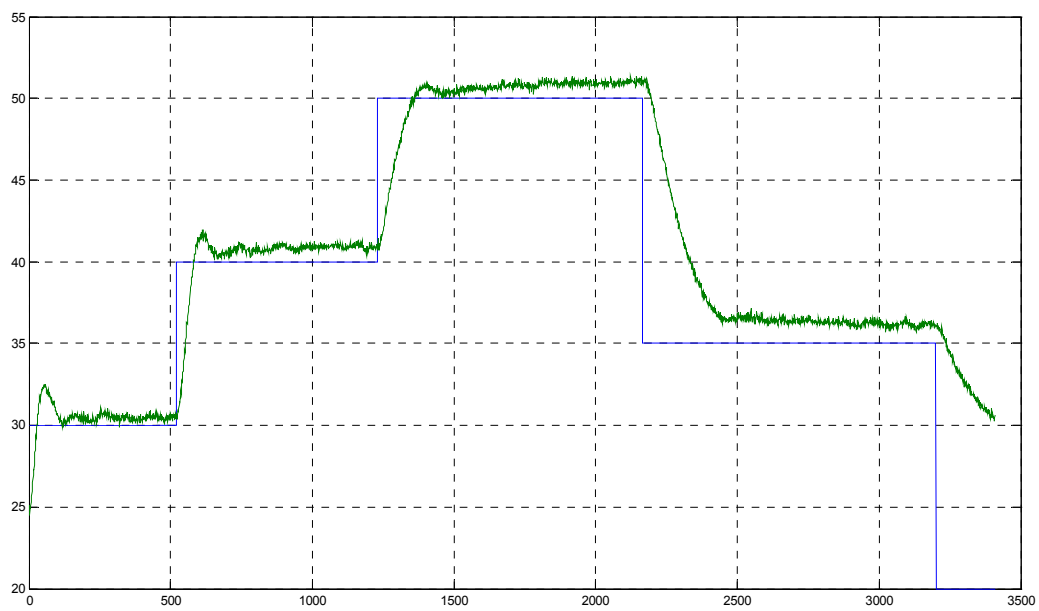


Figura 4.28 Prueba de la implementación del controlador neuronal inverso con un tiempo de paso de dos segundos.

Prueba con tiempo de paso de 5 segundos:

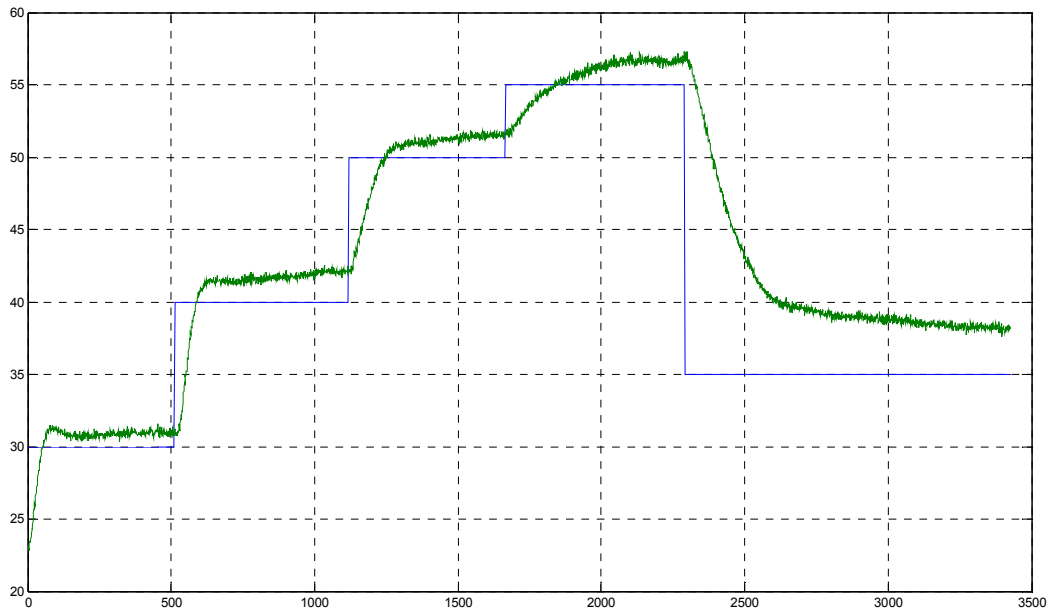


Figura 4.29 Prueba de la implementación del controlador neuronal inverso con un tiempo de paso de cinco segundos.

Prueba con tiempo de paso de 10 segundos:

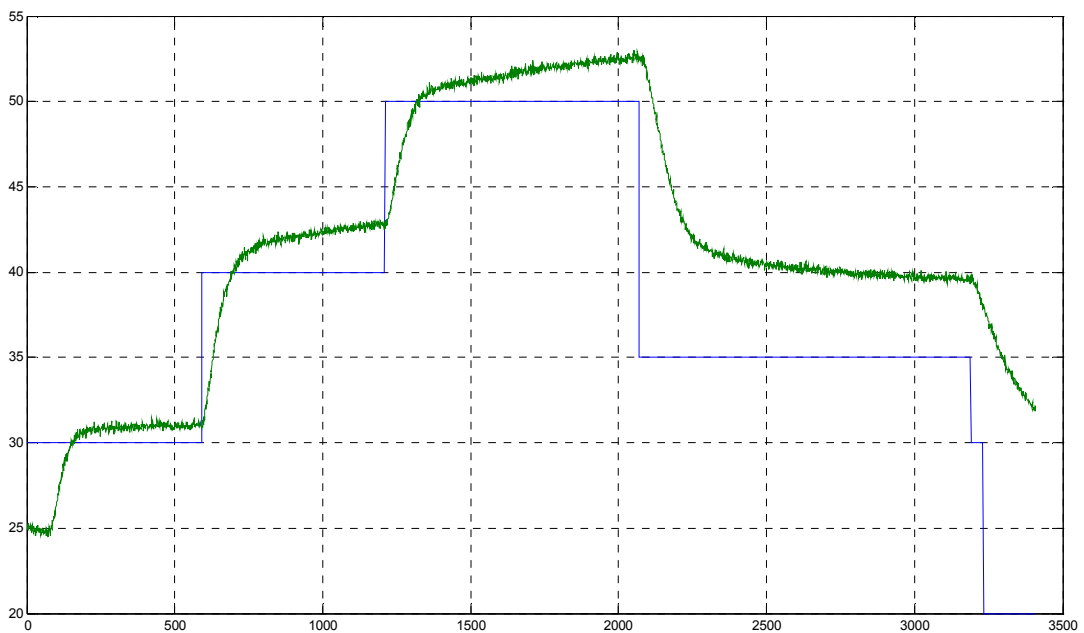


Figura 4.30 Prueba de la implementación del controlador neuronal inverso con un tiempo de paso de diez segundos.

Prueba con tiempo de paso de 20 segundos:

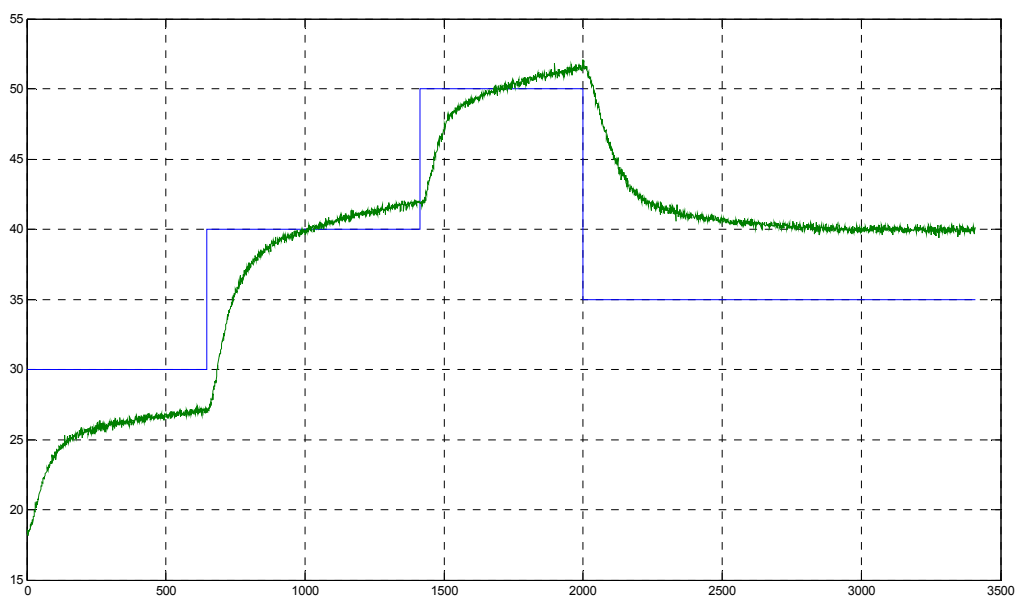


Figura 4.31 Prueba de la implementación del controlador neuronal inverso con un tiempo de paso de veinte segundos

4.4 Controlador Neuronal Directo Basado en Modelo de Referencia con Acción integral

4.4.1 Especificaciones del problema

El objetivo es generar un control por medio de redes neuronales usando como referencia un modelo propio.

El control neuronal por modelo de referencia consiste en un sistema que controla el comportamiento natural de la planta con el objetivo de que se aproxime a la respuesta que tiene un modelo de referencia establecido.

4.4.2 Solución del Problema

Para encontrar el controlador, antes se debe identificar la planta, como lo antes realizado en el control neuronal inverso, así esta identificación formará parte de la red de control total.

En el entrenamiento del control neuronal los pesos y las bias se ajustarán gracias al error que se produce entre el modelo de referencia lineal y el sistema de control representado por la red neuronal de control e identificación. De este proceso se extrae la red neuronal de control, que se encargara de controlar la planta de temperatura.

La red de control viene junto a una acción integral que decrementa el tiempo de subida y tiene el efecto de eliminar el error de estado estable; es una acción totalmente necesaria para nuestra red para poder tener un control completo, aunque se tienen desventajas ya que también se incrementa el sobre impulso y puede empeorar la respuesta transiente.

Es decir se obtendrá una mayor desviación del set point, la respuesta será más lenta y el periodo de oscilación será mayor, pero tendremos un error bastante bajo.

- **Obtención de la red neuronal identificación**

En este paso se realiza la identificación de la planta de temperatura en lazo abierto.

La red de identificación debe seguir la misma estructura de la red de control, por esto se tiene como entrada el voltaje $V1$ y su estado inicial t_0 . Esta red nos entregara como salida la variación de temperatura dt .

```
entrada=0:0.2:6;  
estado=20:1:60;  
Pm = combvec(estado,entrada);
```

Los patrones de entrenamiento están conformados por la matriz de patrones Pm y la matriz objetivo Tm , que representan la entrada y la salida de la red neuronal respectivamente.

```
timestep = 1;  
Q = length(Pm);  
Tm = [];  
for i=1:Q  
    entrada = Pm(2,i);  
    to = Pm(1,i)-20;  
    to1 = Pm(1,i);  
    sim('subingpro',[0 timestep]);  
    dt = tf-to1;  
    Tm=[Tm [dt]];  
end
```

Gracias a las múltiples variaciones de los valores patrones, se pudo obtener la red neuronal de identificación, donde a continuación se muestra la comparación con

el modelo matemático (Figura 4.28), cabe recalcar que los dos sistemas estarán con iguales condiciones de estado inicial y entradas.

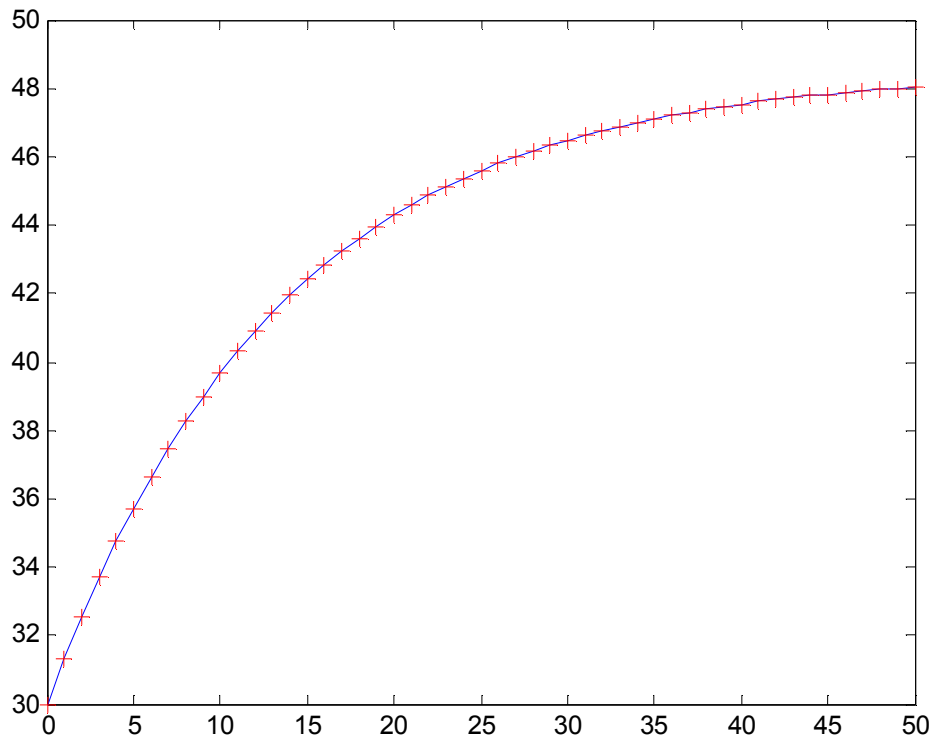


Figura 4.32 Comparación de la red neuronal identificación con el modelo matemático propuesto.

- **Obtención de los patrones de entrenamiento con el modelo de referencia lineal**

El modelo de referencia lineal escogido es definido por la siguiente ecuación:

$$T = \frac{1}{1 + \tau s}$$

Que imita un comportamiento similar a un sistema de primer orden como el de la planta, donde podemos controlar su tiempo de establecimiento.

Se definió una constante de retardo de tiempo en este modelo de referencia de Tau igual a 10 segundos; después de varias pruebas con diferentes valores, se determinó este valor.

El modelo de referencia ayuda a obtener los parámetros P_c y T_c para la red total “tnet”

```

demanda=25:1:40;
estado=25:1:40;
Pc=combvec(estado,demanda);
timestep = 1;
tao=10;
Q = length(Pc);
Tc = [];      %matriz de objetivos "Tc".
for i=1:Q
    demanda = Pc(2,i);
    to = Pc(1,i);
    sim('mmref',[0 timestep]);    %Modelo para el aprendizaje
    dt = tf-to;
    Tc=[Tc [dt]];
end

```

- **Obtención de la red neuronal de control**

El control neuronal por modelo de referencia para el sistema de temperatura es una red tansig, purelin, que tiene nueve neuronas en la capa oculta y una salida. El algoritmo de entrenamiento es de tipo trainlm.

```

S1 = 9;
cnet = newff(minmax(Pc),[S1 1],{'tansig' 'purelin'});

```

- **Entrenamiento de la red neuronal**

La red total que se obtiene al final está compuesta por la red del modelo y la red de control.

La red se compone por cuatro capas, dos capas para la identificación (“mnet”) y las otras dos para el control (“cnet”).

A continuación se muestra la asignación y aprendizaje de los pesos y bias de las entradas de la red de control “cnet” a la red total “tnet”

```

tnet.IW{1,1} = cnet.IW{1,1}(:,1);
tnet.inputWeights{1,1}.learn = 1;
tnet.IW{1,2} = cnet.IW{1,1}(:,2);
tnet.inputWeights{1,2}.learn = 1;
tnet.b{1} = cnet.b{1};
tnet.biases{1}.learn = 1;
tnet.b{2} = cnet.b{2};
tnet.biases{2}.learn = 1;
tnet.LW{2,1} = cnet.LW{2,1};
tnet.layerWeights{2,1}.learn = 1;

```

Y finalmente la asignación de los pesos y bias de las entradas de la red de identificación “mnet” a la red total “tnet”.

```

tnet.IW{3,1} = mnet.IW{1,1}(:,1);
tnet.inputWeights{3,1}.learn = 0;
tnet.LW{3,2} = mnet.IW{1,1}(:,2);
tnet.layerWeights{3,2}.learn = 0;
tnet.b{3} = mnet.b{1};
tnet.biases{3}.learn = 0;
tnet.LW{4,3} = mnet.LW{2,1};
tnet.layerWeights{4,3}.learn = 0;
tnet.b{4} = mnet.b{2};
tnet.biases{4}.learn = 0;

```

Una vez conformada la red total “tnet” con sus respectivos pesos y bias se procede al entrenamiento de la red total.

```
[tnet,tr] = train(tnet,Pc,Tc);
```

Finalmente se extrae la red de control “cnet” de la red total, es decir actualizando los pesos y bias entrenados.

```

cnet.IW{1,1}(:,1) = tnet.IW{1,1};
cnet.IW{1,1}(:,2) = tnet.IW{1,2};
cnet.b{1} = tnet.b{1};
cnet.b{2} = tnet.b{2};
cnet.LW{2,1} = tnet.LW{2,1};

```

4.4.3 Simulación

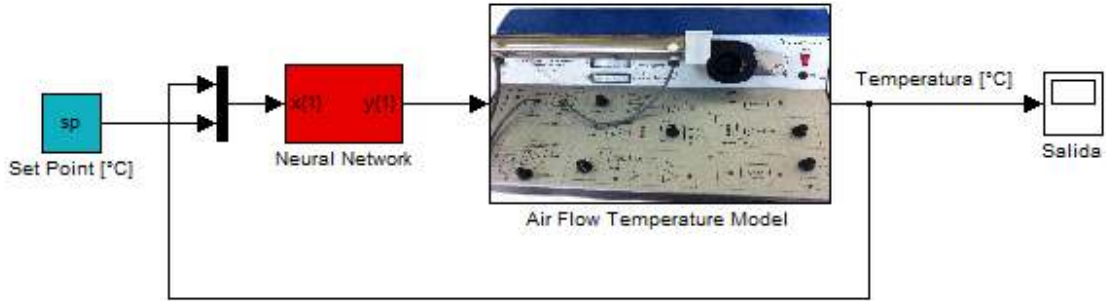


Figura 4.33 Modelo en Simulink para simular el controlador neuronal por modelo de referencia

Una vez creada la red neuronal de control, se procede a comprobar su adecuado funcionamiento, cabe recalcar que se debe realizar la simulación con el lazo de control propuesto, como se muestra en la Figura 4.29 el lazo de control realizado en Simulink.

Para la prueba de simulación se realizó con varias temperaturas de entradas así como diferentes estados iniciales.

Prueba 1

La primera prueba consiste en probar el controlador en la simulación con una temperatura de entrada de 35°C y una temperatura inicial de 20°C.

Planta simulada con 35°C de entrada y 20°C de estado inicial

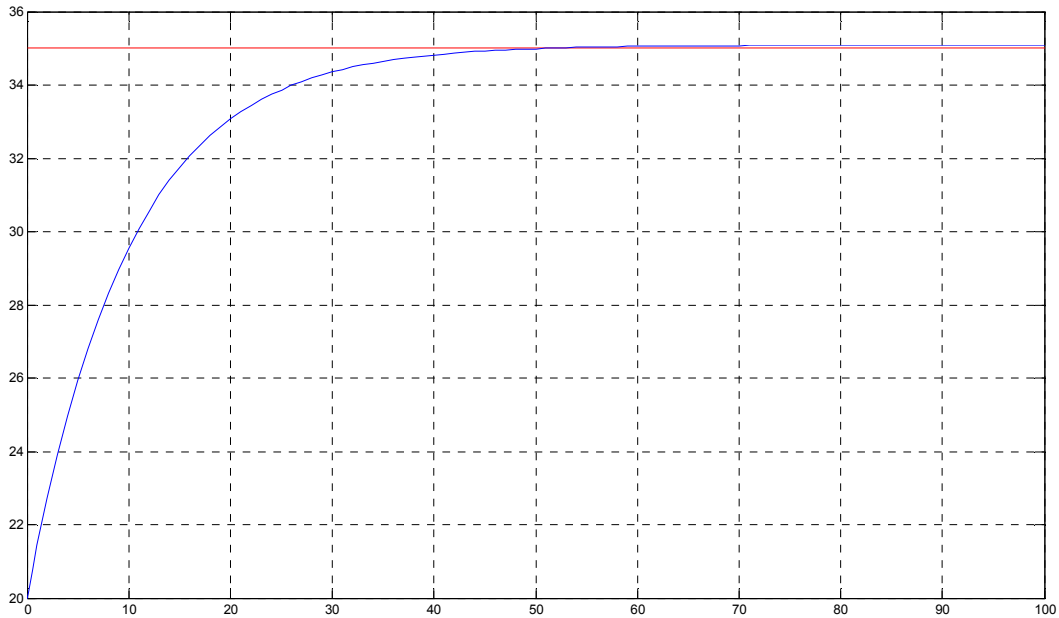


Figura 4.34 Respuesta de la simulación del controlador neuronal con modelo de referencia con entrada de 35°C.

Señal de control:

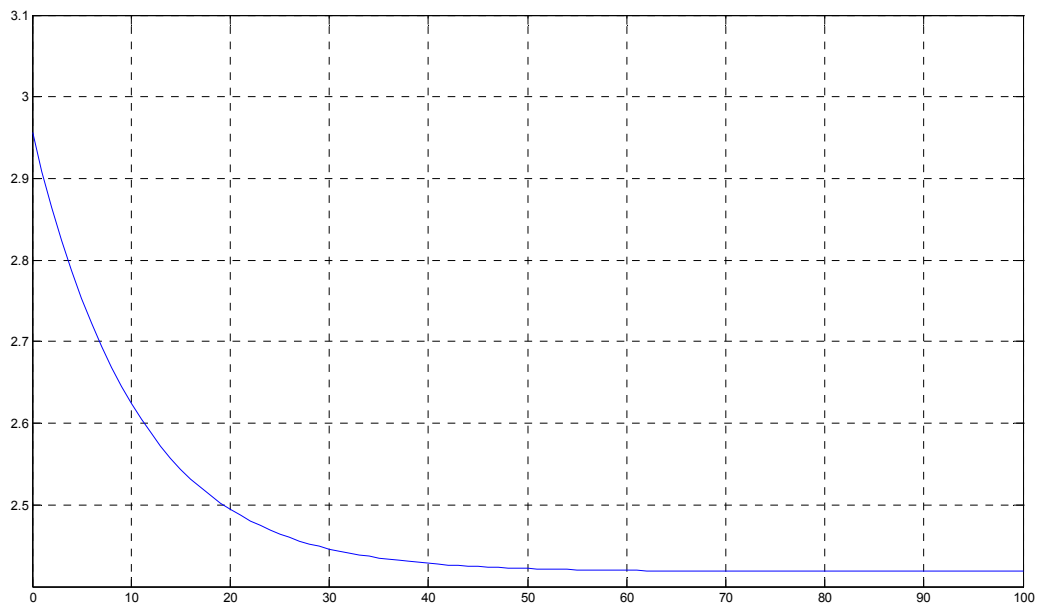


Figura 4.35 Señal de control de la simulación del controlador neuronal con modelo de referencia con entrada de 35°C.

Prueba 2

La segunda prueba consiste en probar el controlador en la simulación con una temperatura de entrada de 45°C y una temperatura inicial de 25°C. Planta simulada con 45°C de entrada y 25°C de estado inicial

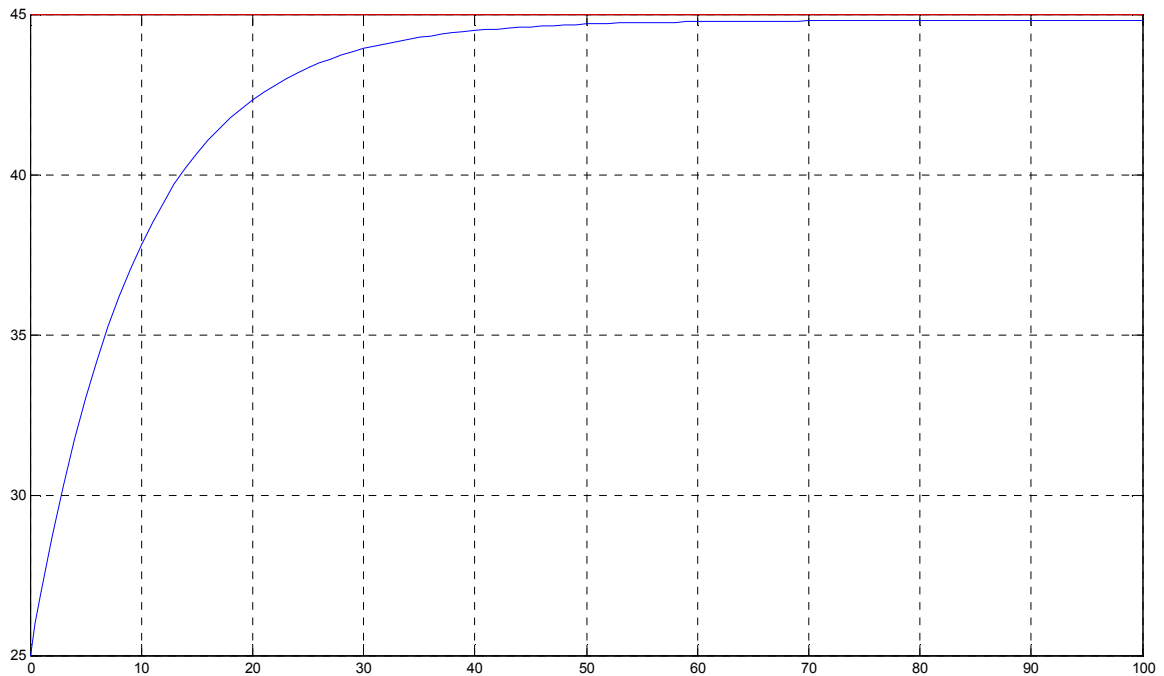


Figura 4.36 Respuesta de la simulación del controlador neuronal con modelo de referencia con entrada de 45°C.

Señal de control:

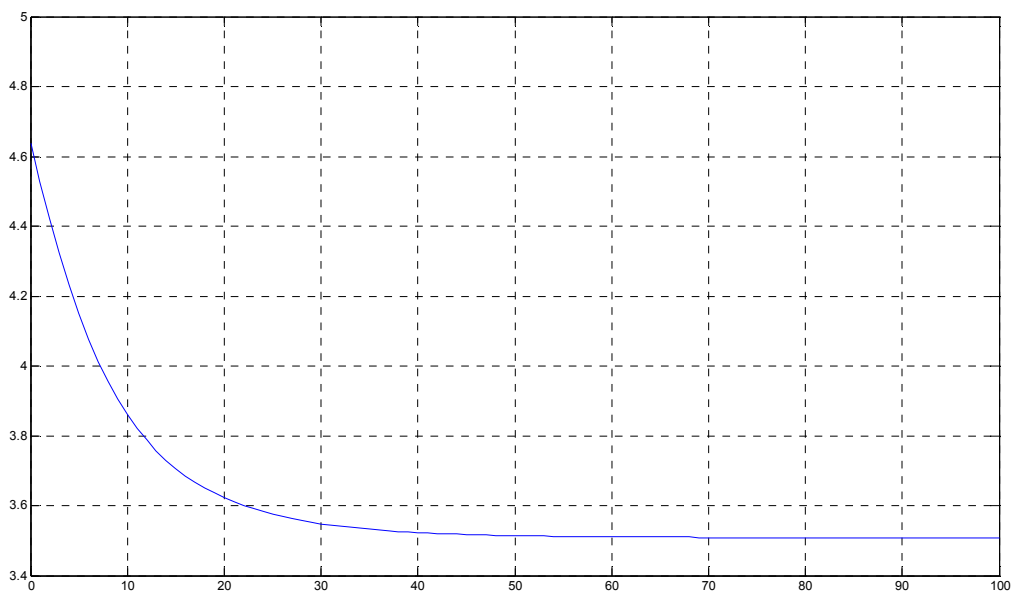


Figura 4.37 Señal de control de la simulación del controlador neuronal con modelo de referencia con entrada de 45°C.

Prueba 3

La tercera prueba consiste en probar el controlador en la simulación con una temperatura de entrada de 30°C y una temperatura inicial de 50°C. Planta simulada con 30°C de entrada y 50°C de estado inicial

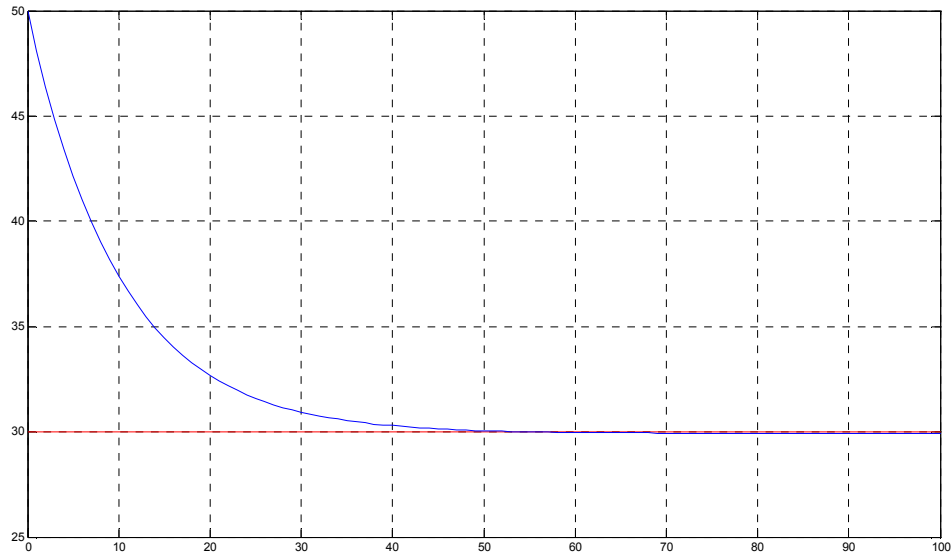


Figura 4.38 Respuesta de la simulación del controlador neuronal con modelo de referencia con entrada de 30°C.

Señal de control:

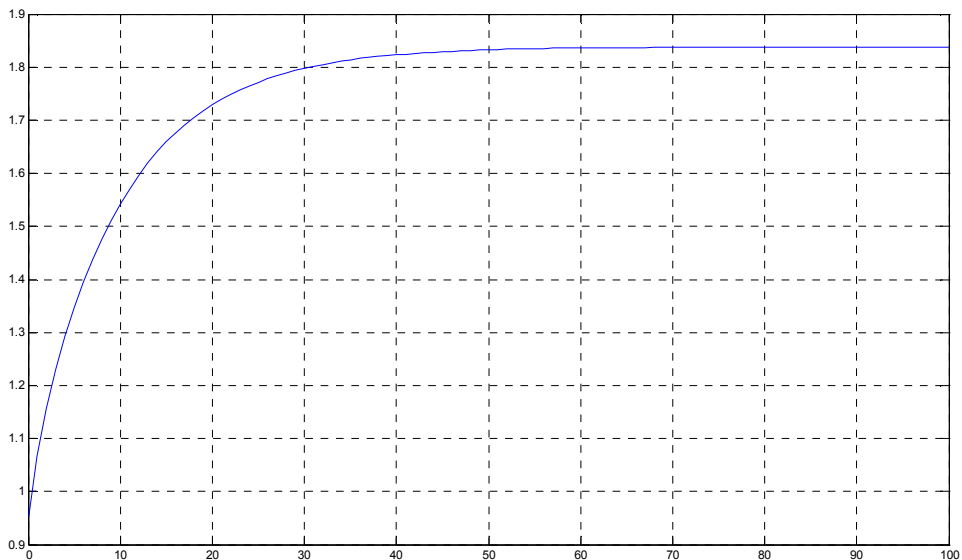


Figura 4.39 Señal de control de la simulación del controlador neuronal con modelo de referencia con entrada de 30°C.

4.4.4 Implementación

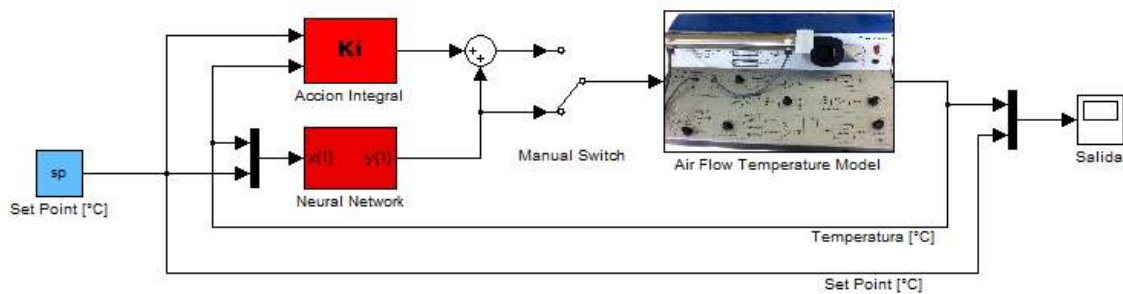


Figura 4.40 Modelo en Simulink para la implementación del control neuronal por modelo en referencia sin acción integral en la planta

Para la implementación del control se realizó pruebas con y sin la acción integral, ya que después de obtener el controlador lo último es ajustar y sincronizar la acción integral.

Se realizó la implementación de la red neuronal por modelo de referencia sin acción integral usando el modelo en Simulink del software como muestra la Figura 4.36, para esto se realizó pruebas con diferentes entradas de temperatura 35°C, 40°C y 50°C, a continuación la Figura 4.37 respuesta del controlador sin acción integral.

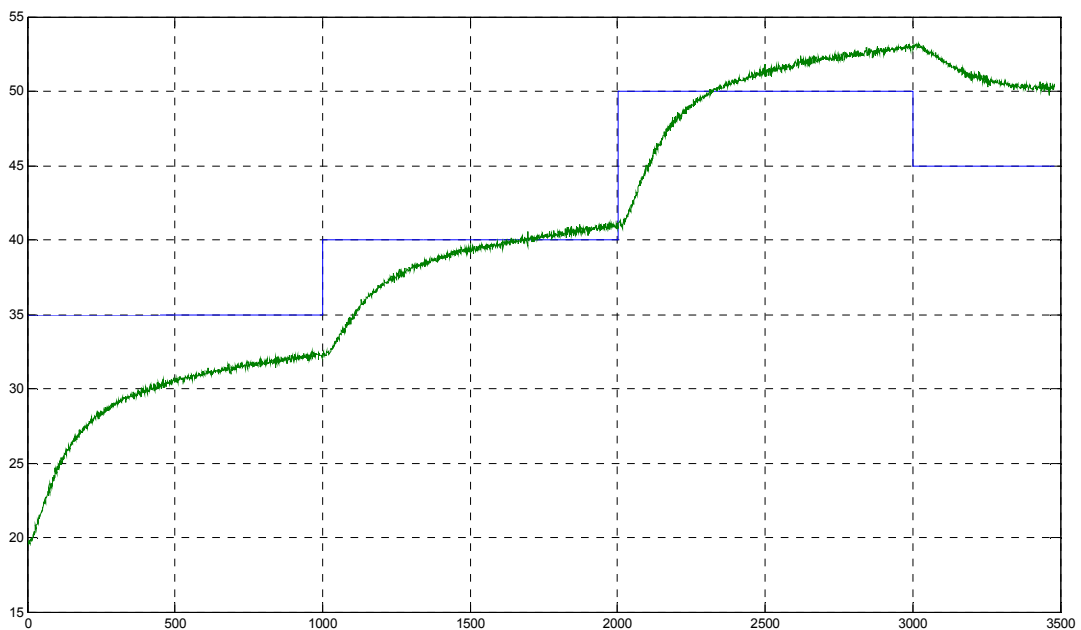


Figura 4.41 Respuesta del controlador neuronal con modelo de referencia sin acción integral

Como se observa es totalmente necesaria una corrección, ya que se tiene un error grande en este control y esto se compensa con la acción integral.

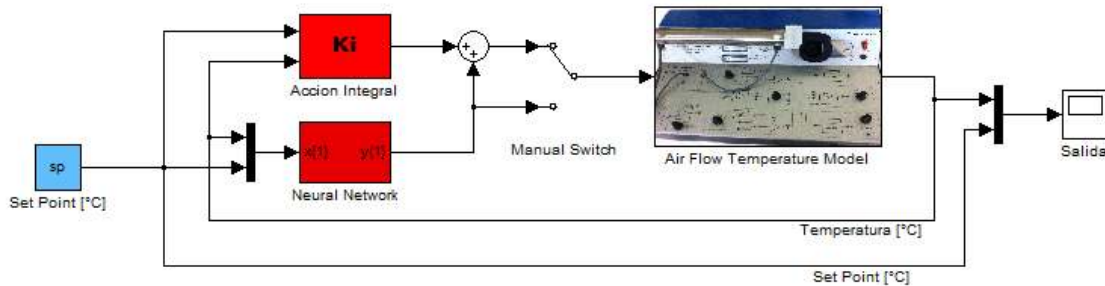


Figura 4.42 Modelo en Simulink para la implementación del control neuronal por modelo en referencia con acción integral en la planta

La siguiente prueba se realiza con el control neuronal por modelo de referencia y con la acción integral usando el modelo en Simulink como se muestra en la Figura 38, para empezar, se necesita sincronizar la acción integral.

Se realizó cuatro pruebas con diferentes valores de la ganancia integral, en la siguiente Figura, se puede observar la respuesta del sistema.

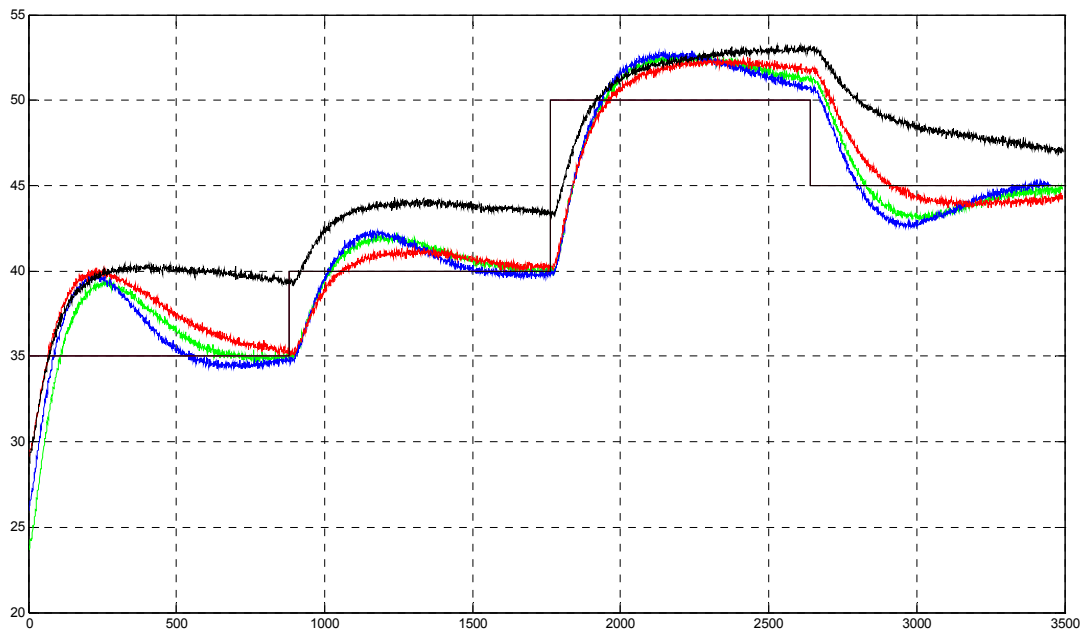


Figura 4.43 Respuestas de la implementación con diferentes ganancias de la acción integral.

Color	Ganancia Ki
Verde	0.008
Azul	0.01
Rojo	0.005
Negro	0.001

Tabla 4.4 Diferentes valores en la ganancia Ki para la sintonización de la acción integral en el controlador neuronal por modelo de referencia.

Al observar los resultados del comportamiento de la planta con la acción integral en la Tabla 4.4 y en la Figura 4.39, se puede concluir que el valor más adecuado para la acción integral es 0.008.

CAPITULO V

5 PRUEBAS Y RESULTADOS

En este capítulo se muestra el desempeño en tiempo real de los controladores diseñados, su respuesta para varios valores de set point, diferentes perturbaciones y su correspondiente análisis.

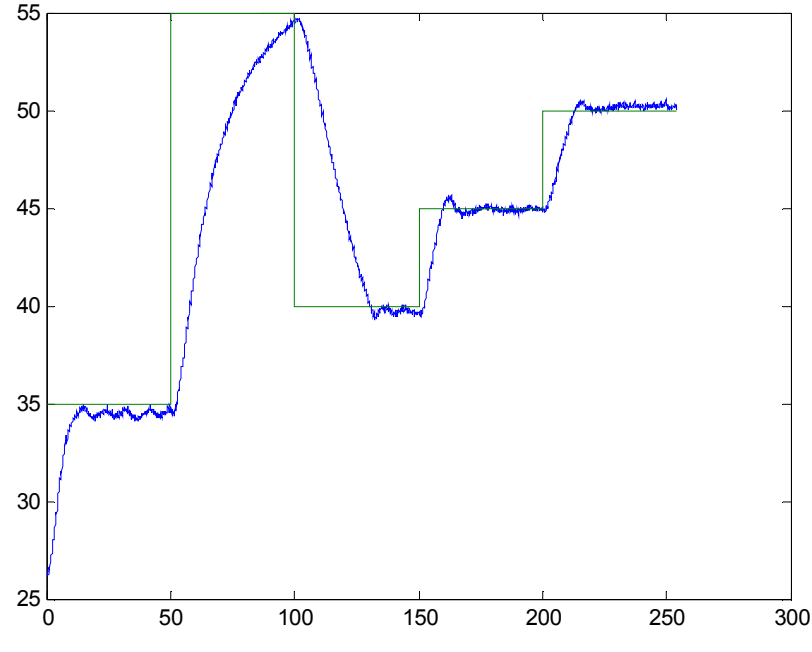
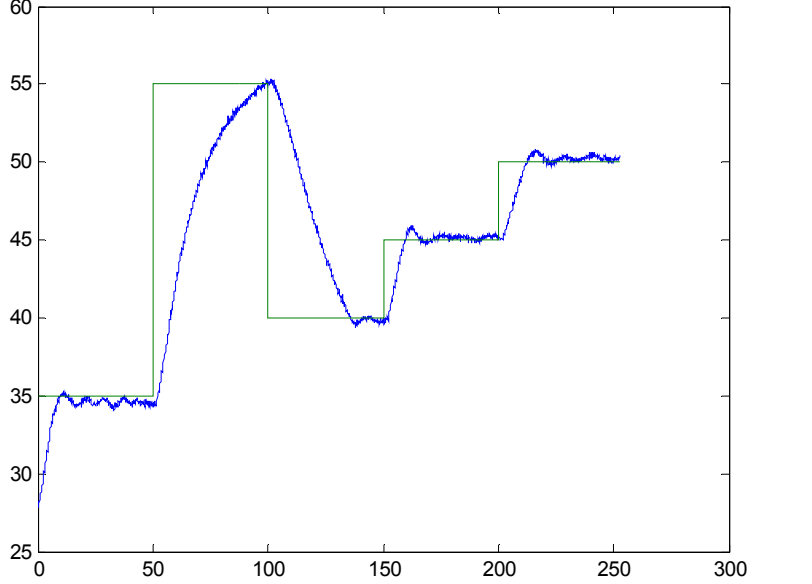
En la Tabla 7.1 se observa cómo está denominada cada una de las perturbaciones, estas consisten en mover la placa metálica que determina la cantidad de flujo de aire que entra en el conducto de la planta de temperatura.

<p>Perturbación 1</p>	
<p>Perturbación 2</p>	
<p>Perturbación 3</p>	
<p>Perturbación 4</p>	

Tabla 5.1 Perturbaciones de la Planta de Temperatura

5.1 Controlador Difuso

A continuación en la Tabla 7.2 se muestra el desempeño del controlador difuso.

Perturbación	Respuesta
1	
2	

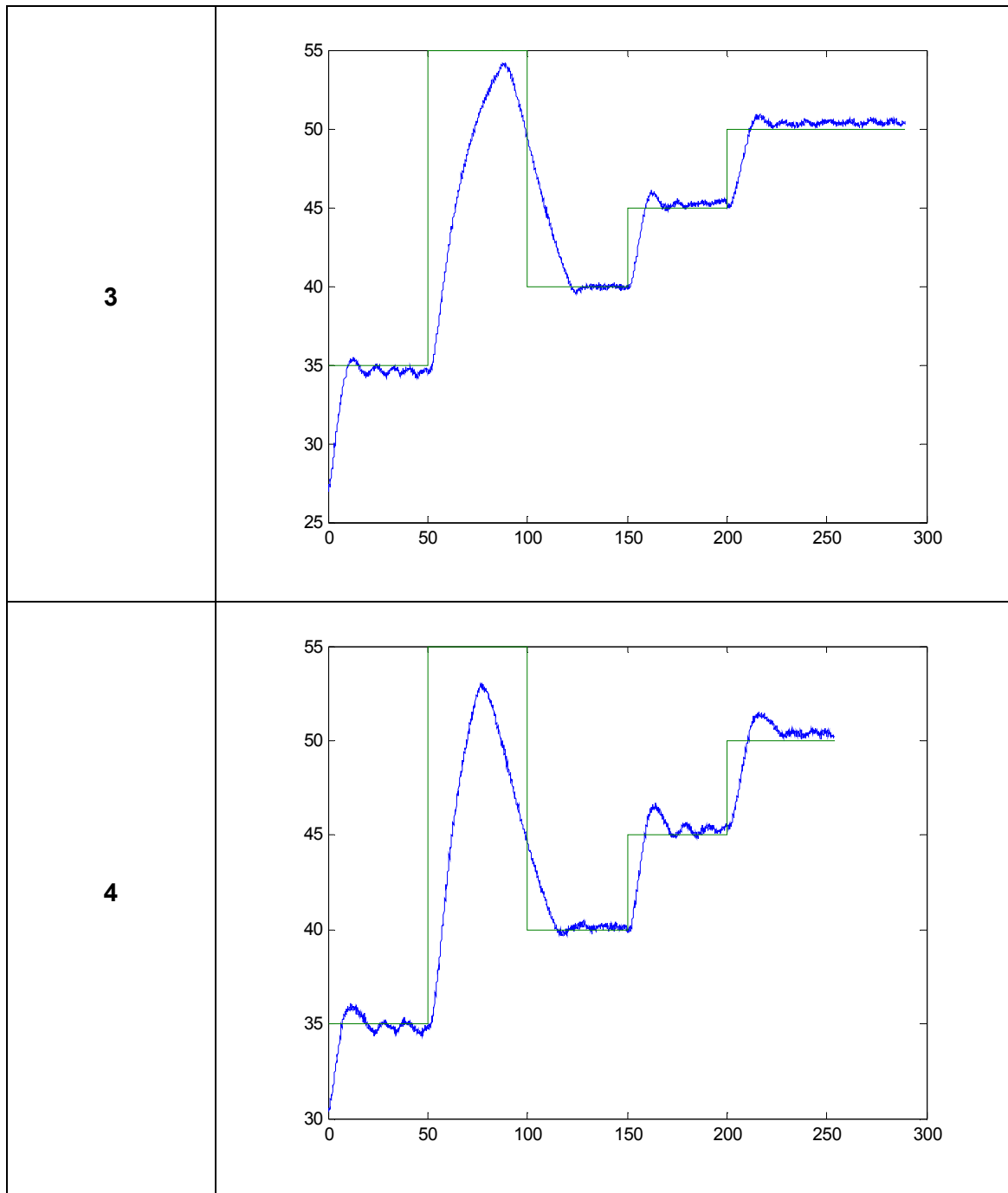


Tabla 5.2 Desempeño del Controlador Difuso para cada una de las perturbaciones

Como se puede apreciar en la Tabla 7.2 el controlador difuso con el conducto totalmente abierto (Perturbacion 1) presenta pequeñas oscilaciones para valores menores a 40°C, esto porque para este rango de valores la planta es mucho más sensible, un pequeño cambio en el voltaje de entrada genera un cambio considerable de temperatura y al ser un sistema no lineal lo contrario sucede para

voltajes mayores, se requiere un mayor cambio en los voltajes de entrada para provocar un cambio en la temperatura por lo que en el rango mayor a 40°C muestra un comportamiento más estable.

También se puede observar que mientras menos flujo de aire existe, las oscilaciones aumentan, esto sucede porque al disminuir el flujo los cambios de temperatura se demoran mucho mas, el aire caliente o frio se moverá mucho más lento a través del conducto aumentando las oscilaciones y provocando sobrepicos mayores.

En las perturbaciones 3 y 4 donde existe el menor flujo de aire, para el setpoint de 55°C se puede observar que no alcanza este valor y la temperatura empieza a descender, esto sucedió porque se activó la protección de la planta, la temperatura en la niquelina ubicada al inicio del conducto sobrepaso los 70°C pero sin flujo de aire el calor se transfiere demasiado lento hacia el final del conducto donde se encuentra el sensor.

5.2 Controlador Predictivo

El desempeño del controlador predictivo se observa en la Tabla 7.3

Perturbación	Respuesta
1	
2	

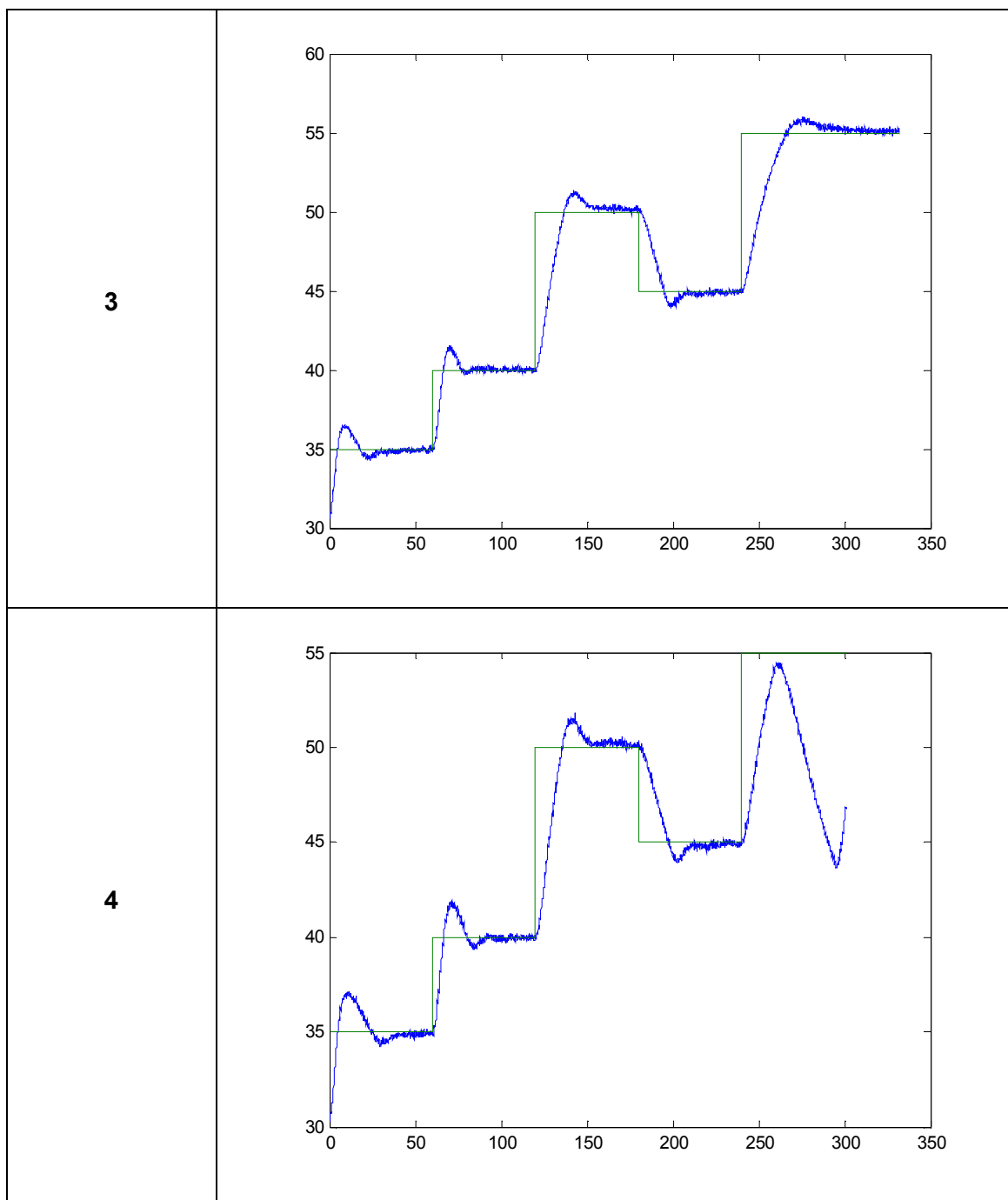


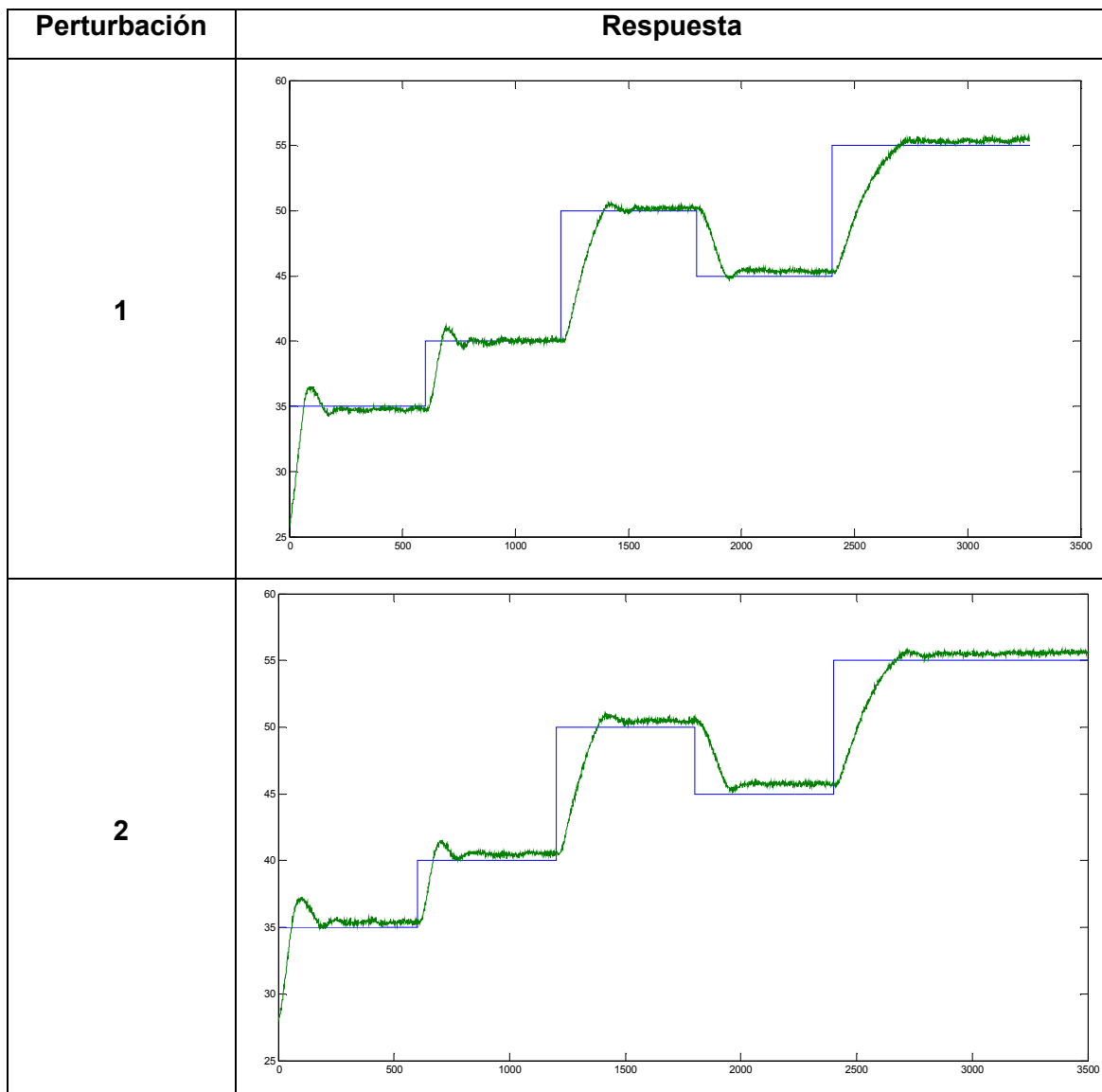
Tabla 5.3 Desempeño del Controlador Predictivo para cada una de las perturbaciones

Como se observa el controlador predictivo presenta un sobrepico mayor para temperaturas bajas y disminuye conforme la temperatura aumenta, este sobrepico también es inversamente proporcional al flujo del aire, con la perturbación 4 se observa mayores sobrepicos y en el setpoint de 55°C se activa la protección de la planta, la niquelina calienta pero al existir escaso flujo de aire no lo transmite

hasta el sensor ubicado al final del conducto. Para todas las perturbaciones se logró el control con un error de estado estable despreciable.

5.3 Controlador Neuronal Indirecto por Modelo Inverso

A continuación en la Tabla 5.4 se muestra el desempeño del controlador neuronal Inverso.



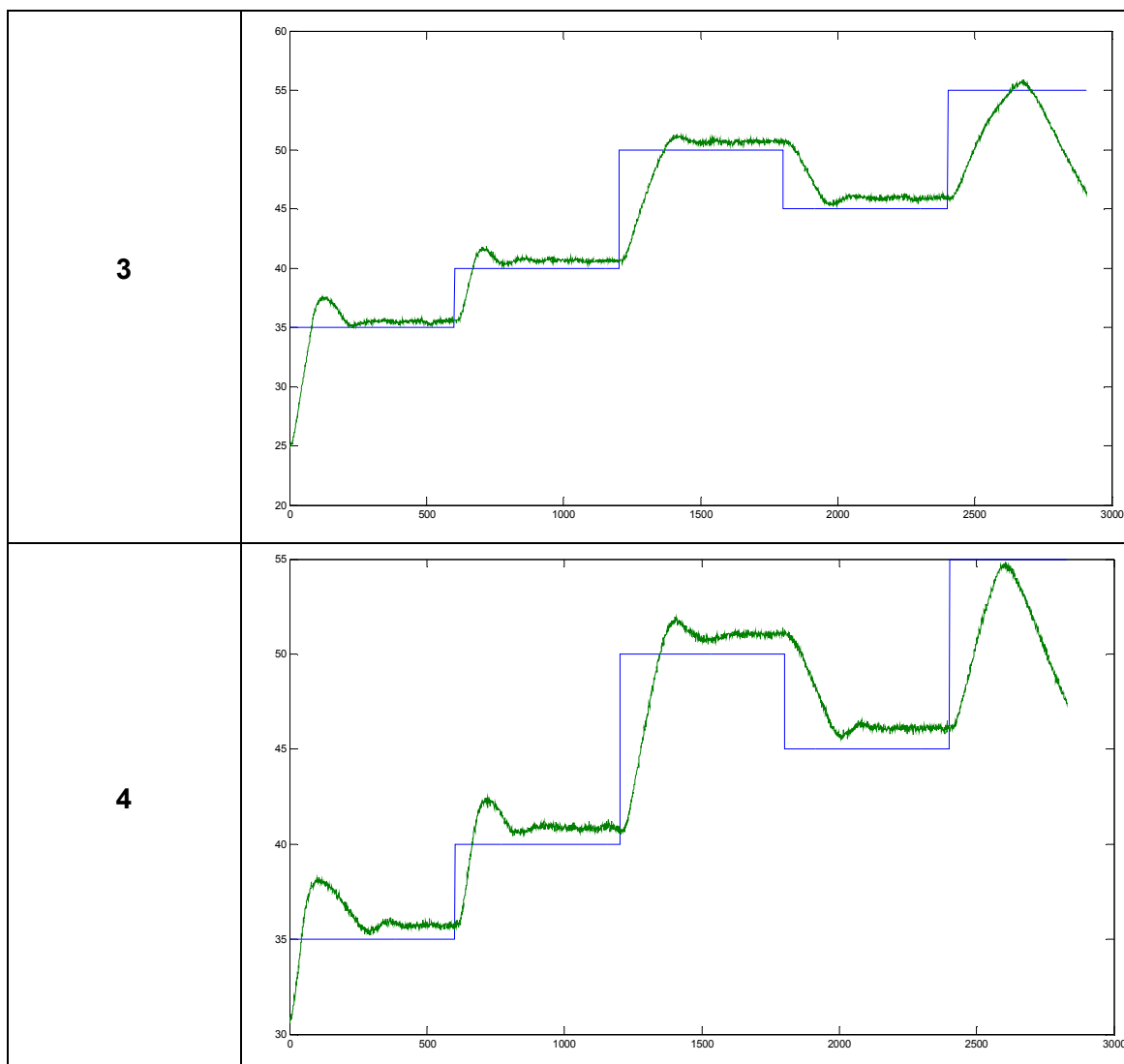


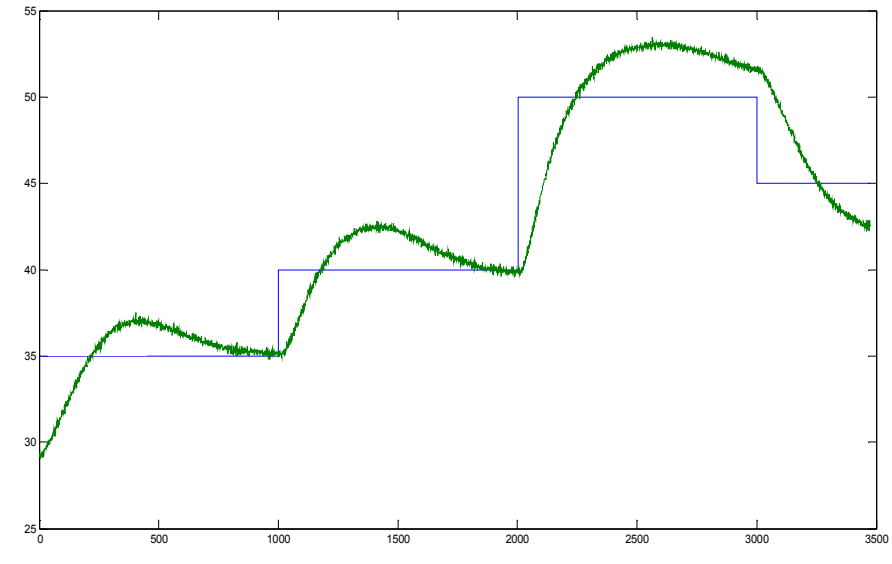
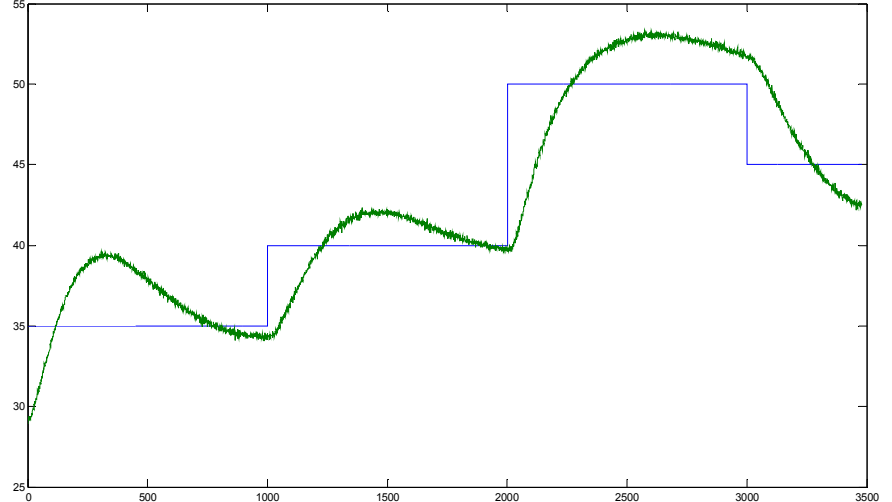
Tabla 5.4 Desempeño del Controlador Neuronal Inverso para cada una de las perturbaciones

Se puede observar que en la primera perturbación, donde el conducto está totalmente abierto, existe un sobre pico muy pequeño que es parte del controlador y un error de estado estable imperceptible que mientras más se cierra el ducto se nota que aumenta.

Adicionalmente en las perturbación 3 y 4 se puede observar que en el set point de 55°C activa la protección de la niquelina, donde el límite es 70°C, pero por lo falta de flujo de aire no fluye esta temperatura de 70°C a nuestro sensor.

5.4 Controlador Neuronal Directo Basado en Modelo de Referencia

A continuación en la Tabla 5.5 se muestra el desempeño del controlador neuronal con acción integral.

Perturbación	Respuesta
1	 <p>The graph for perturbation 1 shows a blue step function and a green response curve. The x-axis ranges from 0 to 3500, and the y-axis ranges from 25 to 55. The perturbation starts at 30, steps up to 35 at x=500, to 40 at x=1000, to 50 at x=2000, and to 45 at x=3000. The response curve follows these steps, showing overshoot and oscillation, especially at the 2000 and 3000 marks.</p>
2	 <p>The graph for perturbation 2 shows a blue step function and a green response curve. The x-axis ranges from 0 to 3500, and the y-axis ranges from 25 to 55. The perturbation starts at 30, steps up to 35 at x=500, to 40 at x=1000, to 50 at x=2000, and to 45 at x=3000. The response curve follows these steps, showing overshoot and oscillation, especially at the 2000 and 3000 marks.</p>

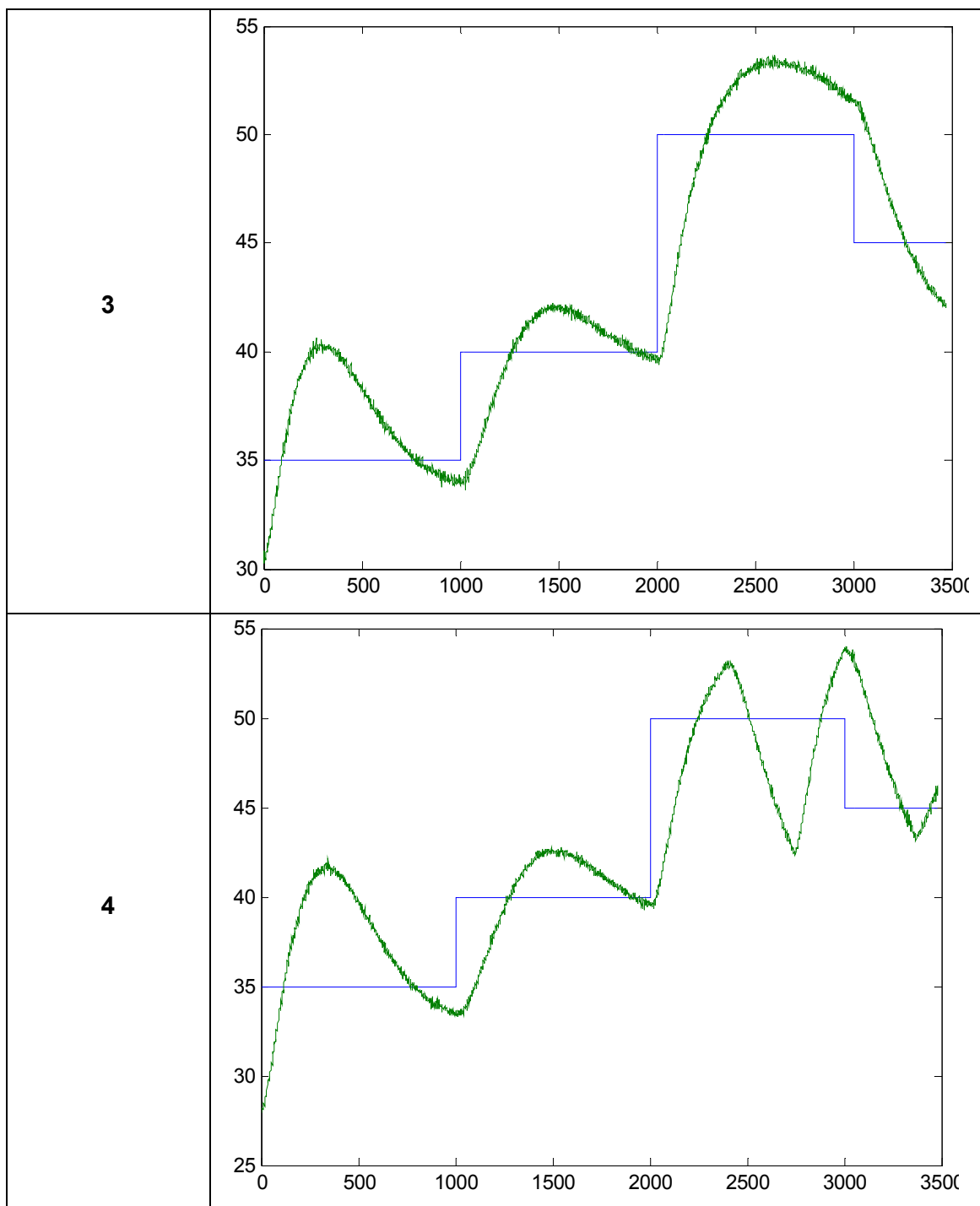


Tabla 5.5 Desempeño del Controlador Neuronal con Acción Integral para cada una de las perturbaciones

Como se puede apreciar en la Tabla 5.5 el controlador neuronal con el conducto totalmente abierto (Perturbacion 1) es lento en el control y el error se corrige gracias a la acción integral, se tiene un sobre pico porque el controlador junto a la

acción integral actúan lento, al finalizar podemos ver un comportamiento más estable.

En la perturbación 4 donde existe el menor flujo de aire, para el setpoint de 50°C se puede observar que alcanza este valor pero no logra controlarlo ya que llega a la temperatura límite de la planta y empieza a descender, esto sucedió porque se activó la protección de la planta, la temperatura en la niquelina ubicada al inicio del conducto sobrepaso los 70°C pero sin flujo de aire el calor se transfiere demasiado lento hacia el final del conducto donde se encuentra el sensor.

CAPITULO VI

6 CONCLUSIONES Y RECOMENDACIONES

6.1 Conclusiones

- Mediante el presente proyecto se ha logrado satisfactoriamente el control en tiempo real de la unidad *AIR FLOW TEMPERATURE CONTROL SYSTEM* mediante la herramienta RTW de Matlab, implementando los métodos de control inteligente planteados.
- Para la implementación de los controladores neuronales y predictivo es necesario un modelo matemático que describa la dinámica del sistema real. En el caso de las redes neuronales se utiliza este modelo en la etapa de entrenamiento de los controladores y en el caso del control predictivo

se requiere del modelo para optimizar los valores de la señal de control que ajuste la salida del proceso a la referencia ingresada.

- Para el controlador difuso el modelo matemático solo se utiliza en la simulación, una ventaja de este control difuso es que simplemente con el conocimiento del funcionamiento de la planta se puede implementar un controlador.
- El control predictivo requiere una gran capacidad de procesamiento de datos para el cálculo de optimización de la señal de control que se realiza para cada intervalo de control.
- El control difuso de un sistema de temperatura requiere de una temperatura de referencia, sobre la cual se comandan las acciones de control para llevar al sistema a los objetivos deseados.
- En el diseño de los controladores predictivo y difuso es necesario incorporar restricciones a las señal de control resultantes, esta es una ventaja importante que carecen los métodos de control convencional.
- En el control por redes neuronales se utilizaron dos métodos de entrenamiento: modelo de referencia y red neuronal inversa, la diferencia de estos controladores principalmente se nota en los diferentes procedimientos de entrenamiento y desempeño en tiempo real. El controlador neuronal más eficiente fue la red neuronal inversa. Esto se encuentra con una explicación detallada en el Capítulo 4.

- Un inconveniente que se pudo observar en el diseño de los controladores neuronales, es la necesidad de modelos matemáticos que contengan la dinámica exacta del sistema a controlar para obtener un control adecuado.
- Para el control neuronal por modelo de referencia se logro el objetivo de controlar el sistema con la ayuda de una acción integral K_i , la planta contiene demasiados parámetros que no pueden incorporarse en el modelo matemático, es por esto que la red neuronal no logró controlar totalmente al sistema, pero este problema se compensó con el efecto de una acción integral.

6.2 Recomendaciones

- Este proyecto se desarrolló con Matlab R2010a, se recomienda utilizar esta versión del software para el buen funcionamiento de los controladores obtenidos.
- Después de la instalación de la tarjeta de adquisición (Anexo 1), verificar las conexiones del sistema y por seguridad verifique los valores de los voltajes de entrada y salida en la tarjeta de adquisición de datos para una correcta implementación.
- Antes de implementar un controlador, comprobar su funcionamiento en las ventanas de simulación del software desarrollado y que se describe en el Anexo 4

- En la simulación e implementación del controlador predictivo se recomienda no tener otros programas en ejecución ya que este controlador requiere de una gran capacidad de procesamiento, esto también depende de la capacidad del hardware que se esté utilizando.
- Utilizar la técnica del control predictivo en procesos de respuesta lenta ya que se necesita de un tiempo de cálculo para obtener las señales de control resultantes de la optimización.
- Se recomienda utilizar los controladores diseñados a una temperatura ambiente de 20 a 25 grados centígrados y permitiendo una buena circulación del aire alrededor del sistema de temperatura.

REFERENCIAS BIBLIOGRÁFICAS

1. **Bonifacio Martin del Brio, Alfredo Sanz Molina.** *Redes Neuronales y Sistemas Difusos*, 2da Edicion, Alfaomega Ra-Ma
2. **Burns Roland S.,** *Advanced Control Engineering*, Butterworth Heinemann
3. **Rodriguez Ramirez Daniel, Bordóns Alba Carlos,** *Apuntes de Ingeniería de Control.*
4. **Elena López Guillen Ma.,** *Identificación de Sistemas*
5. **Yoshikuni Okawa,** *Introduction to Embedded Programming Using MATLAB, Simulink.*
6. **Ochoa José Bernardo,** *Control Clásico y Control Inteligente.*
7. *MATLAB & Simulink Based Books.*
8. **Ibarra Alexander, Acuña Alejandro,** *Diseño e Implementacion de un Sistema Controlador de Temperatura PID para la unidad Air Flow Temperature Control System mediante la utilización de la herramienta RTW (Real Time Workshop) de Matlab, 2010.*
9. **Yunus A., Cengel,** *Transferencia de Calor*, Segunda edición, Mc Graw Hill, 2004.
10. **Ati Patricia,** *Análisis, Diseño e Implementación de Controladores Predictivos*, 2011.

11. **Armijos Luis, Chicaiza Dario**, *Desarrollo e Implementación de Controladores Inteligentes en la Planta Multi-Tanque del Laboratorio de Servomecanismos*.
12. **Incropera Frank, DeWitt David**, *Fundamentos de Transferencia de Calor*, 1996
13. **Ortiz Paula, Arias Alexander**, *Control de una Planta No Lineal de Temperatura con Redes Neuronales*, 2009.