UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIEROS INDUSTRIALES VALENCIA

TRABAJO FIN DE GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

DESARROLLO DE UN CONTROLADOR PID INDUSTRIAL DE BAJO COSTE MEDIANTE RASPBERRY PI PARA CONTROL DE TEMPERATURA

AUTORA: Esperanza García Martínez

TUTOR: Juan Manuel Herrero Durá

COTUTOR: Raúl Simarro Fernández

Curso Académico: 2015-16

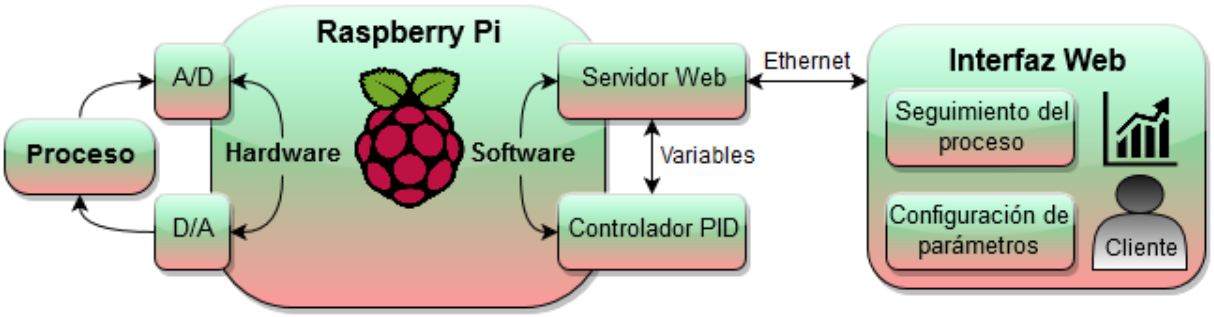
RESUMEN

Esta es la memoria escrita que presenta el Trabajo de Fin de Grado de la titulación Grado en Tecnologías Industriales.

El desarrollo del trabajo consiste en la programación e implementación de un controlador industrial de tipo PID, utilizando una Raspberry Pi y tarjetas de adquisición de datos como plataforma hardware. También se diseña una interfaz web mediante la cual el cliente puede interactuar y configurar el proceso a través de la red.

La documentación que se presenta es diversa. En la presente memoria se expone de forma generalizada el software y hardware utilizado, así como el proceso de desarrollo de cada una las partes. Además se adjuntan tres anexos donde se desarrolla al detalle un manual de programación, otro de usuario y un ensayo práctico del control de un horno.

Con el desarrollo de este trabajo se pretende crear una alternativa más económica y con una interfaz de usuario más completa y gráfica que los PIDs industriales convencionales. Permitiéndose a través de ésta diferentes opciones para la configuración del controlador al igual que de los sensores utilizados, proporcionando el seguimiento en tiempo real del proceso y el control, y ofreciendo facilidades al cliente, como el diseño automático de controladores a partir del modelo del proceso.



ÍNDICE

DOCUMENTOS CONTENIDOS EN EL TRABAJO:

- Memoria
- Presupuesto
- Anexos

Tabla de Contenido de la Memoria

Tabla de Figuras de la Memoria

Tabla de Contenido del Presupuesto

Tabla de Contenido de los Anexos

Tabla de Figuras de los Anexos

MEMORIA

1. INTRODUCCIÓN

En la actualidad el uso en la industria de los controladores de tipo PID está ampliamente extendido debido a su simplicidad y robustez. Son mecanismos versátiles, indicados para procesos cuya dinámica se rige por modelos de primer y segundo orden, pero también si se desconoce el proceso. Entre sus aplicaciones más comunes destacan el control de la temperatura, el nivel, la presión o el flujo.

Los algoritmos PID son mecanismos de control por realimentación, caracterizados por el uso de tres parámetros: uno proporcional que depende del valor actual, otro integral que depende de los errores pasados y uno derivativo que es una predicción de los errores futuros. La suma de las tres acciones asociadas a cada parámetro se utiliza para ajustar el proceso. El propósito del controlador es hacer que el error en régimen estacionario sea cero, es decir, que el valor deseado y el valor medido sean iguales.

En este trabajo se va a desarrollar un controlador de este tipo utilizando una plataforma de bajo coste, como es Raspberry Pi, y se va a aplicar al control de la temperatura de un horno. Aunque en el mercado se puede encontrar una amplia gama de controladores de este tipo, una de las principales diferencias de los mencionados con el que se va a desarrollar radica en la mejora de experiencia del cliente. Se va a diseñar una interfaz web que permitirá el seguimiento del proceso en tiempo real y la configuración del mismo de manera remota, a través de la red, con el objetivo de presentar la información de manera gráfica y clara para que cualquier usuario pueda utilizarla, independientemente de su nivel de especialización.

El alcance del presente documento abarca desde la programación del servidor web, el controlador y la aplicación, hasta la implementación y montaje de la Raspberry Pi y los sensores que han sido proporcionados para este fin.

En esta memoria se plasmará el proceso de diseño e implementación de cada una de las partes antes mencionadas de manera general, así como las características técnicas del hardware empleado. Finalmente, contará con tres anexos en los cuales se expondrá detalladamente el código empleado, la puesta en marcha y uso de la aplicación y, para concluir, se presentará un caso práctico de aplicación para el control de la temperatura de un horno.

No obstante, este proyecto está abierto a futuras mejoras como podría ser la incorporación de autotuning para la identificación del tipo de proceso que se va a controlar y autoajuste de los parámetros del controlador, o un registro de usuarios, para restringir quien puede manipular el controlador.

Finalmente, esta memoria puede ser considerada la antesala de un trabajo de final de máster que cubra, además de lo expuesto anteriormente, el diseño de la placa y los sensores así como la incorporación de un controlador más complejo tipo multi-lazo y el desarrollo de la correspondiente interfaz web. Creando así un producto final, compuesto por software y hardware propio, enfocado para su comercialización.

1.1. Conocimientos Previos

1.1.1 Conceptos Teóricos

Se procede a definir una serie de términos que serán utilizados en el desarrollo del presente documento con la finalidad de facilitar la comprensión del mismo (*Blasco)*:

Modelo: aproximación matemática del comportamiento de un proceso real de acuerdo con las leyes físicas que lo rigen.

Referencia: o setpoint, es el valor deseado de una variable.

Variable controlada: variable que describe el comportamiento del proceso y se trata de controlar.

Variable manipulada: variable sobre la que se puede actuar para influir en las variables controladas.

Controlador: dispositivo encargado de tomar las decisiones que afectan a la variable manipulada para que la variable controlada alcance la referencia.

Control en bucle abierto: el controlador toma las decisiones sin tener información de la variable controlada.

Control en bucle cerrado: o con recirculación, el controlador dispone de una medida de la variable controlada.

IDE: Integrated Development Environment, en castellano entorno de desarrollo integrado.

Front-end: parte de una aplicación web que interactúa con el cliente.

Back_end: parte de una aplicación web que procesa y ejecuta las peticiones realizadas por el cliente a través del front-end.

Raspberry Pi: es un ordenador de placa reducida, desarrollado con el objetivo de fomentar el aprendizaje de la computación en las escuelas.

1.1.2 Introducción al Software empleado

Se van a presentar las características principales y funciones que desempeñan cada uno de los lenguajes de programación utilizados en el desarrollo de este proyecto, con el fin de proporcionarle al lector los conocimientos básicos necesarios para la comprensión de la memoria:

- HTML: es un lenguaje de marcado utilizado en la elaboración de páginas web. Este lenguaje se encarga etiquetar y describir el contenido de la página web para que el navegador pueda interpretar esas marcas y mostrarlo correctamente. HTML puede diferenciar entre una gran

variedad de tipos de contenido, desde texto y links hasta video e imágenes, entre muchos otros.

En concreto la última versión, HTML5, incluye una serie de nuevas etiquetas, como por ejemplo <output> o <nav> que hacen referencia a la muestra de datos y a la barra de navegación respectivamente, así como atributos asociados a las mismas, o nuevos atributos asociados a etiquetas ya existentes. Todas estas actualizaciones se han desarrollado con el fin de adaptarse al uso de los sitios web actuales.

- CSS: también llamado hoja de estilos en cascada es un lenguaje utilizado para definir la apariencia y el estilo de los documentos HTML. Este lenguaje permite diseñar los estilos de los elementos contenidos en la aplicación web y aplicarlos a un conjunto de etiquetas.
- JavaScript: se utiliza para programar el comportamiento de las páginas web. Este lenguaje nos permite crear eventos que se ejecutan cuando se realizan determinadas acciones, como pulsar un botón. También es el medio a través del cual se establece la comunicación con el servidor, se encarga de organizar los datos para ser enviados al servidor y procesar las respuestas.
- Python: es un lenguaje muy potente pero a la vez simple y de fácil comprensión ya que su sintaxis favorece que el código sea legible. Python soporta diferentes tipos de programación: orientada a objetos, imperativa y funcional. Además cuenta con una amplia gama de librerías disponibles.

1.2. Motivaciones

Como ya se ha comentado, en el sector industrial está ampliamente arraigado el uso de algoritmos de tipo PID para el control de procesos. Desde su diseño y primeras aplicaciones a finales del siglo XIV hasta la actualidad este mecanismo de control ha probado su valía, y de momento no se ha desarrollado ningún otro igual de simple y eficaz.

Por otra parte es importante tener en cuenta la imparable revolución tecnológica, y todo el desarrollo que ello conlleva. Internet ha supuesto un antes y un después en la comunicación y el aprendizaje.

Teniendo en cuenta las dos consideraciones anteriores y la descripción del proyecto que se va a realizar, el hecho de utilizar dos tecnologías con tanto futuro supone una gran motivación para embarcarse en un proyecto donde se usen. Aún más teniendo en cuenta que lo que se pretende con este proyecto es proporcionar un enfoque más actual a una tecnología desarrollada hace más de un siglo modernizándola, conservando los principios básicos de su funcionamiento que tan bien han servido durante años, pero proporcionando una nueva forma de comunicación con el cliente.

La naturaleza multidisciplinar que presenta este proyecto también ha sido una gran motivación debido a que permite profundizar y ampliar los conocimientos que se han ido adquiriendo a lo largo de la carrera, en las ramas de control, automática e informática. Además de brindar la posibilidad de adquirir otros nuevos, como es el caso del desarrollo de aplicaciones web o la utilización de plataformas como Raspberry Pi.

1.3. Objetivo del Trabajo

El objetivo de este proyecto es el diseño e implementación, en una plataforma Raspberry Pi, de un controlador PID con interfaz de usuario basado en web. Para ello es necesario llevar a cabo, por una parte, la programación de un controlador de tipo PID de dos grados de libertad y un servidor web embebidos en una placa Raspberry Pi. Por otra, el desarrollo de un aplicación web que permita el seguimiento del proceso así como la configuración de los sensores y actuadores involucrados en el proceso.

El proceso de desarrollo ha sido el siguiente: primero se ha diseñado la aplicación web, a continuación se ha programado el servidor y el controlador para, finalmente, realizar el montaje y los ensayos para el control de la temperatura de un horno. Antes del ensayo experimental con el horno se han realizado varias simulaciones del proceso empleando el ordenador, así como pruebas con una red RC, con el fin de detectar posibles errores.

El desglose de objetivos es el siguiente:

- Diseñar y programar la interfaz web, permitiendo la configuración de parámetros, sensores y actuadores.
- Programar el controlador PID de dos grados de libertad.
- Programar el servidor web.
- Montar la placa Raspberry Pi y los sensores.
- Adaptar los programas para la lectura/escritura a través de las tarjetas de adquisición de datos.
- Realizar el control real sobre un horno.

1.4. Estructura de la Memoria

En este apartado se resume brevemente el contenido expuesto en cada una de las partes que componen esta memoria.

INTRODUCCIÓN

Da una visión general del proyecto, introduce los conceptos necesarios para la comprensión del trabajo y software utilizado. Describe las motivaciones que llevaron al desarrollo del proyecto y los objetivos para su realización.

ESTADO DEL ARTE DE LOS SISTEMAS DE CONTROL INDUSTRIAL

Se realiza un breve resumen de la historia de los sistemas de control para después definir los controladores PID.

INTERFAZ WEB

Se expone el proceso de diseño y los resultados del desarrollo de la interfaz web. Primero la parte relativa al front-end comentando cada uno de los HTMLs, y a continuación, los programas desarrollados para el back-end.

SISTEMA DE CONTROL

Se desarrolla y expone el sistema de control implementado. También se presentan los tres métodos para el diseño de parámetros empleados.

COMUNICACIÓN

Se proporcionan las especificaciones técnicas del hardware empleado. A continuación, se detallan las conexiones físicas y los protocolos de comunicación utilizados entre las distintas partes que componen el proyecto.

CONCLUSIONES

Se comentan los resultados de la aplicación del proyecto así como los posibles trabajos futuros para mejorarlo.

BIBLIOGRAFÍA Y REFERENCIAS

Bibliografía y referencias utilizadas.

2. ESTADO DEL ARTE DE LOS SISTEMAS DE CONTROL INDUSTRIAL

2.1. Reseña Histórica

Los sistemas de control son dispositivos encargados de regular el comportamiento de otro sistema con el fin de obtener los resultados deseados.

Los primeros sistemas de control desarrollados datan del siglo III a.C. en la Antigua Grecia y surgen de la necesidad de controlar de forma precisa la evolución del tiempo. Estos mecanismos denominados Clepsydras eran unos relojes de agua, cuyo funcionamiento se basaba en el llenado, a velocidad constante, de un depósito. Existen documentos en los que se plasma el uso de estos mecanismos en procedimientos judiciales, para controlar que ambas partes disponían del mismo tiempo *(Auslander, 1969)*.

Siglos después, en la Revolución Industrial y con el auge de las máquinas de vapor surge la necesidad de controlar la velocidad de rotación de las turbinas de vapor. La solución a este problema llegó de la mano del ingeniero e inventor escocés James Watt a finales del siglo XVIII, el cual desarrolló el péndulo centrífugo *(Dickinson, 1927)*. Este regulador empleaba dos esferas metálicas que, dependiendo de la velocidad, se separaban de su eje de giro debido a la fuerza centrífuga y, cuando se superaba la velocidad de diseño, accionaban un mecanismo que estrangulaba la válvula del flujo de vapor, controlando así el proceso y manteniéndolo a una velocidad de giro constante *(Bennett, 1979)*.

Durante el siglo XX la Ingeniería de Control se consolida de manera teórica mediante el uso de herramientas matemáticas como las Transformadas de Laplace y Fourier. Además, las guerras acontecidas durante este siglo, sobre todo la Segunda Guerra Mundial, supusieron un gran impulso del desarrollo tanto teórico como práctico en este campo. Finalmente, la aplicación del computador en el control de procesos supone un salto tecnológico enorme, permitiendo la implantación de nuevos sistemas de control en la Industria y posibilitando el desarrollo de la navegación espacial. Es en este contexto donde empezaron a desarrollarse los controladores de tipo PID, para la dirección automática de buques. Fue el ingeniero Nicolas Minorsky en 1922 el primero en realizar un análisis teórico de los controladores de este tipo *(Bennett, 1993)*.

2.2. El Controlador PID

Es un mecanismo de control que funciona en bucle cerrado cuyo objetivo es minimizar el error ajustando la entrada del sistema. Siendo el error la diferencia entre la referencia y la variable controlada.

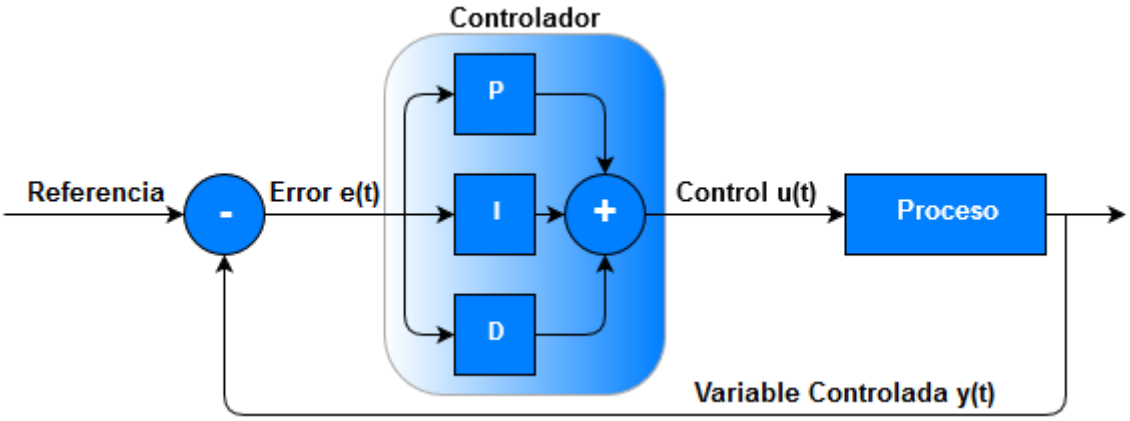


Figura 1: Controlador PID, estructura ISA-paralela.

La característica principal de este tipo de controladores es que proporcionan tres acciones básicas de control (*Blasco)*:

- Proporcional (P): esta acción proporciona un control proporciona a la señal de entrada del regulador, es decir, proporcional al error.

$$u(t) = Kc\ e(t) \ \rightarrow U(s) = Kc\ E(s) \quad (2.1)$$

  Donde Kc es el parámetro característico de esta acción, el cual recibe el nombre de ganancia proporcional. Esta acción de control se basa en la información actual del error generando un valor proporcional.
- Integral (I): proporciona una acción de control proporcional a la integral del error medido.

$$u(t) = \frac{Kc}{Ti} \int_0^t e(\tau)d\tau \ \rightarrow U(s) = \frac{Kc}{Ti\ s} E(s) \quad (2.2)$$

  Ti es el tiempo integral. Se trata de una acción que tiene en cuenta los errores pasados.
- Derivada (D): produce una acción de control proporcional a la derivada del error.

$$u(t) = Kc \cdot Td\ \frac{de(t)}{dt} \ \rightarrow U(s) = Kc \cdot Td\ s\ E(s) \quad (2.3)$$

De la combinación de las tres acciones se obtiene el controlador PID:

$$u(t) = Kc\left(e(t) + \frac{1}{Ti} \int_0^t e(\tau)d\tau + Td\ \frac{de(t)}{dt}\right) \quad (2.4)$$

2.3. Modelo Teórico de un Proceso

Para abordar un problema de control el primer paso es la obtención de un modelo matemático del proceso que se va a controlar. El modelo debe tratar de reproducir fielmente la respuesta ante una misma entrada u(t), suponiendo las mismas condiciones iniciales, obteniendo y(t)≈ym(t).

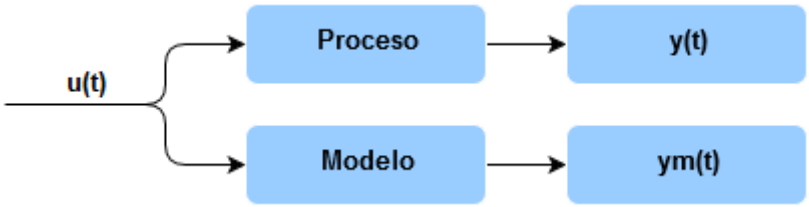


Figura 2: Modelo teórico y proceso real

Los modelos se generan por dos métodos principalmente:

- Modelado: Conociendo los valores de los parámetros de un sistema así como las leyes físicas mediante las cuales se rige.
- Identificación experimental: Se somete el sistema a varias pruebas y a partir de los resultados se construye un modelo matemático. Se utiliza este método en procesos complejos o de los cuales se desconocen partes.

Independientemente de la forma con la que se ha obtenido el modelo, tanto en el diseño de controladores como en la obtención de los modelos, se busca que sean lo más simples posible. Por este motivo, en este trabajo se van a usar modelos matemáticos de procesos de primer orden con retardo, que responden a la siguiente forma:

$$G(s) = \frac{Kp}{(1+\tau s)} e^{-\theta s} = \frac{\Delta Y(s)}{\Delta U(s)} \quad (2.5)$$

Donde las constantes son:

- Kp: Constante de proporcionalidad del proceso, adimensional.
- τ: Constante de tiempo.
- θ: Retardo.

Otra consideración importante es que, por muy bueno que sea el modelo obtenido, no deja de ser una aproximación y, por lo tanto, es una fuente de error.

3. INTERFAZ WEB

En este apartado se va a exponer de manera general el funcionamiento de la interfaz web desarrollada, presentando las funciones disponibles para el usuario, así como los lenguajes de programación y los recursos empleados en cada una de sus partes.

Este capítulo está dividido en dos grandes bloques: el primero se encarga de la comunicación entre el cliente y el sistema, realizándose la introducción de datos y selección de opciones, además de la visualización y seguimiento del proceso. El segundo bloque consiste en el procesado de esas entradas y la consiguiente actuación.

Para una explicación más detallada del código empleado o del arranque y uso de la aplicación que el que se presentan en este capítulo consultad los anexos: Manual de Usuario y Manual de Programación.

3.1 Front-end

El front-end o interfaz es el nombre que recibe la capa de presentación, la cual se encarga de la traducción de la información que es introducida por el usuario con el fin de adaptarla para su posterior procesado y viceversa.

El objetivo de diseño de esta parte ha sido crear una aplicación sencilla e intuitiva mediante la cual el usuario pueda configurar los parámetros pertinentes para el control PID, además de poder realizar el seguimiento real del proceso e interactuar con él.

Se ha pretendido que la web tenga una estética simple pero cuidada con el fin de presentar la información de la manera más clara y visual posible para que cualquier usuario, sin importar de su nivel de conocimiento en la materia, pueda comprender y utilizar la aplicación.

3.1.1. Software Empleado

Para la programación de las diferentes partes que componen este trabajo, independientemente del lenguaje de programación empleado, se ha utilizado la plataforma de software libre Eclipse Java Mars como editor de texto.

Además, para facilitar la programación relativa a esta sección se ha instalado el complemento: Eclipse IDE for JavaScript Developers, el cual contiene una serie de herramientas muy útiles como la construcción automática de texto o el depurador, entre muchas otras, que permiten agilizar el

proceso de escritura en lenguajes cuyas sentencias son repetitivas como las que se utilizan en HTML o CSS.

Se van a enumerar los programas empleados y algunas consideraciones respecto a su utilización, también se van a mostrar algunos ejemplos de utilización de los mismos en este proyecto:

- HTML: Se ha utilizado la versión 5 ya que es la más reciente. A continuación se muestra un ejemplo de visualización de código de la cabecera y la barra de navegación en el navegador Google Chrome, y otro de parte del mismo código utilizado en HTML5.

```
<body onload="GetMain();">
      <header id="cabecera">
          <h1>Controlador PID</h1>
          <h3>Para un proceso térmico</h3>
      </header>

      <nav id="selector">
           <ul>
               <li><a class="active" href="Untitledm.html"><b>Main</b></a></li>
               <li><a href="Untitledc.html"><b>Configuración</b></a></li>
               <li><a href="Untitledp.html"><b>Proceso</b></a></li>
          </ul>
     </nav>
```

Figura 3: HTML5. Barra de navegación

Figura 4: Visualización de HTML5 en Google Chrome

- CSS: existen dos formas de añadir el código CSS a los documentos HTML. La primera es de forma interna, contenido todo en el mismo HTML mediante la etiqueta <style> o en una etiqueta concreta mediante el atributo style. La segunda es incluir el estilo de manera externa, creando la plantilla de estilo en un documento a parte que se añade en apartado <head> del documento HTML.
  En este trabajo se ha realizado una hoja de estilo externa que se ha incluido en los tres documentos HTML creados. A continuación se muestran unos ejemplos de programación en CSS y, por último, la aplicación del mismo al ejemplo HTML anterior.

```
<link rel="stylesheet" type="text/css" href="PIDweb.css">
```

Figura 5: Incluir CSS en HTML

```
p{
color:white;
border:solid #404040;
border-width:2px;
width:150px;
height:20px;
text-align:center;
background-color:#404040;
margin:auto;
font-weight:500;
margin-top:20px;
padding:0.5em 0.5em;
 }
```

Figura 6: Estilo CSS definido para los elementos párrafo <p>

Figura 7: Resultado de aplicación de CSS en HTML. Barra de navegación

- JavaScript: Al igual que ocurre con el código CSS, JavaScript también puede ejecutarse de manera externa o interna. En este caso también se ha realizado en un documento a parte que se ha incluido en la cabecera del HTML.
  A continuación se muestra un ejemplo en el cual según el valor de la temperatura se activan las acciones de la alarma, que constan de una salida de texto y un indicador de colores.

```
if(temperatura<tmin){
    document.getElementById("alarma").value="Tª demasiado baja";
    document.getElementById("alarm").style.backgroundColor="#0099ff";
    setTimeout("frio()",periodo*500);
}
```

Figura 8: JavaScript. Alarma fría

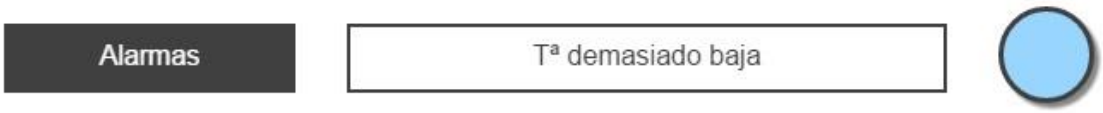


Figura 9: Resultado acción Alarma Fría

3.1.1. Diseño y Funciones

Como resultado de la utilización conjunta del software presentado en el aparatado anterior se va a proceder a explicar en los siguientes puntos las opciones disponibles en cada una de las pestañas de la aplicación y las necesidades que cubren.

La secuencia de acciones para las cuales se ha diseñado la aplicación es la siguiente: al inicio se muestra la pantalla principal, Main, y por defecto el controlador se encuentra en modo manual. El usuario debe fijar una referencia y modificar el tanto por ciento de actuación para que se estabilice la temperatura. A continuación es el momento de introducir o modificar los parámetros deseados en la pestaña de Configuración. Finalmente, si se desea obtener los parámetros del controlador de manera automática se ha creado una pestaña auxiliar donde se ofrecen distintas alternativas. Una vez está todo configurado es el momento de volver a la pantalla principal y conmutar a modo automático, cambiar la referencia y observar la evolución.

3.1.1.1. Main

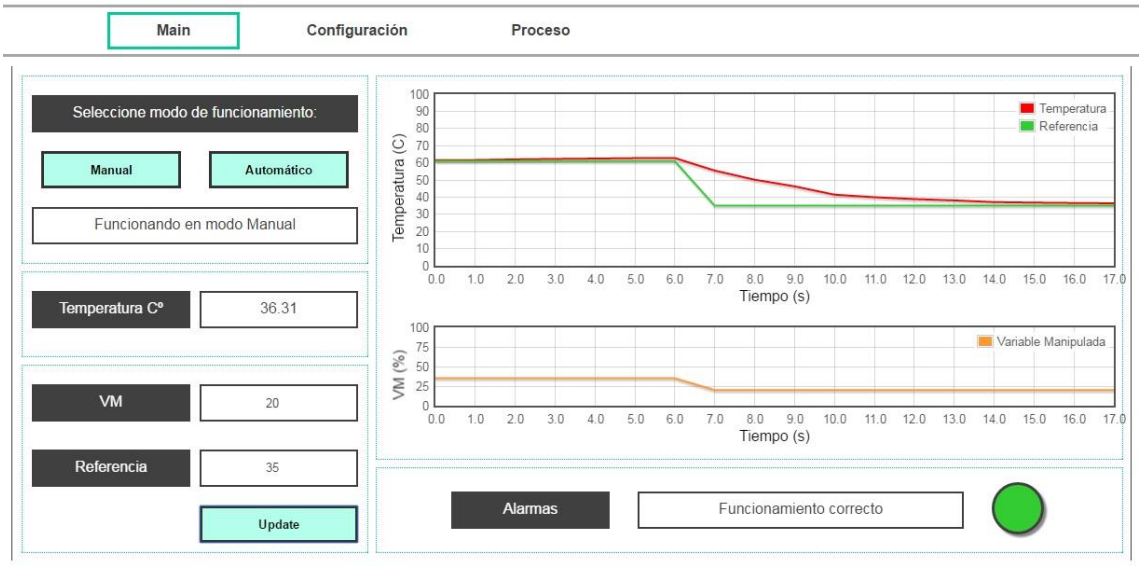


Figura 10: Pantalla Main del interfaz web

Esta es la pantalla principal de la aplicación, la cual se ha diseñado de manera que se pueda tener una visión general del estado del proceso y del tipo control que se está llevando a cabo en tiempo

real. Aquí se puede realizar la conmutación de modo manual a automático, así como fijar la referencia de la temperatura y el valor de la variable manipulada.

Se han incorporado dos gráficas dinámicas que muestran el estado del proceso. En la superior se plasma la medida de la temperatura y la referencia. En la inferior se muestra la acción de control en cada instante. El eje temporal es el mismo en ambas, las gráficas se actualiza según el periodo especificado por el cliente y se van completando hasta albergar cien puntos, momento en el cual se muestra solo los últimos cien valores. El eje y de la gráfica superior es configurable.

Finalmente hay que destacar los indicadores asociados a las Alarmas del proceso, uno de tipo texto el cual indica si el funcionamiento es correcto o no, y otro que emula un indicador luminoso, con un código de colores.

3.1.1.2. Configuración

Figura 11: Pantalla Configuración del interfaz web

En esta sección se ofrece una serie de opciones referentes al control, la seguridad y al seguimiento del proceso. Los parámetros sólo se pueden actualizar en modo manual.

Relativo al control del proceso, a parte de la configuración de los parámetros básicos de un controlador, se ofrece la posibilidad de utilizar ponderación de la referencia, así como filtrado de la señal de entrada. No se indican unidades de las constantes de tiempo o del periodo, queda a elección del cliente, pero deben estar todas en las mismas unidades.

Para la configuración de la recta del sensor se ofrece la posibilidad de introducir hasta 4 puntos para definirla.

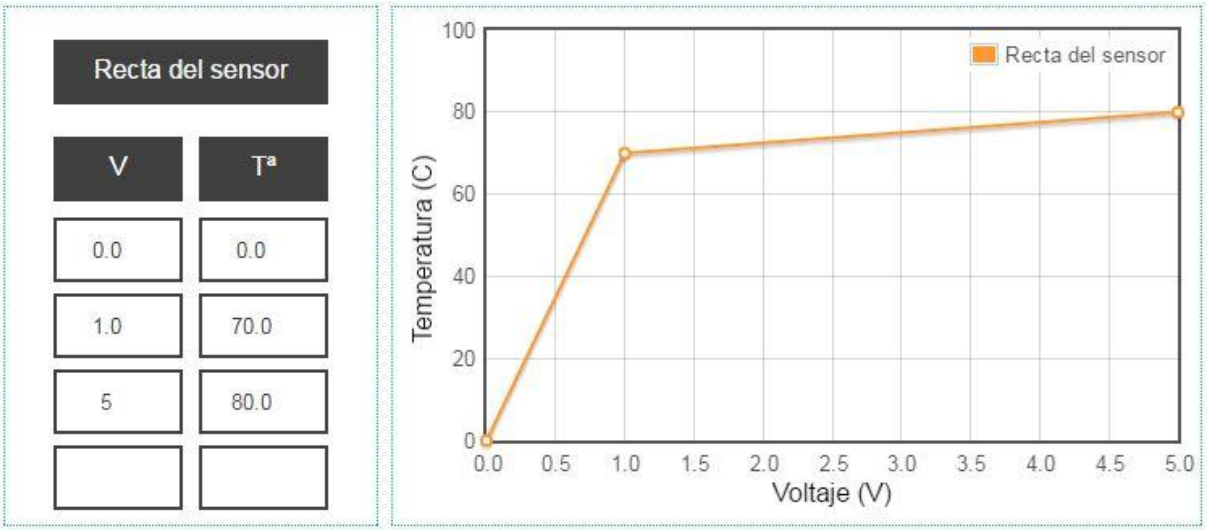


Figura 12: Pantalla Configuración. Recta del sensor

En la parte de seguridad, el usuario puede configurar las temperaturas para las alarmas y el rango de la variable manipulada. Por último, para la mejor visualización y seguimiento del proceso se ofrece la posibilidad de modificar el rango de temperaturas a mostrar en la gráfica principal.

3.1.1.3. Proceso

Figura 13: Pantalla Proceso del interfaz web

Esta es una ventana auxiliar, creada para facilitar la labor del cliente, cuya función es el diseño de los parámetros del controlador, según el método deseado, para el modelo de primer orden introducido. Sólo se pueden actualizar los parámetros del controlador si el modo de funcionamiento es manual, al igual que ocurre en la ventana de configuración.

3.2 Back-end: Servidor

El back-end o motor es el nombre que recibe la capa de acceso a datos. Esta parte es la encargada de procesar todos los datos y las entradas que el cliente ha proporcionado a través del front-end y actuar en consecuencia.

En los siguientes apartados se va a exponer el software utilizado y los programas creados para este fin.

3.2.1. Software Empleado

Se ha utilizado el editor de texto Eclipse Java Mars, pero en este caso se ha instalado el IDE PyDev. El lenguaje utilizado para la programación de esta parte ha sido Python complementado con la librería SimpleHTTPServer, específica para la creación de servidores web.

A continuación se muestra un ejemplo del entorno de programación PyDev en Eclipse.

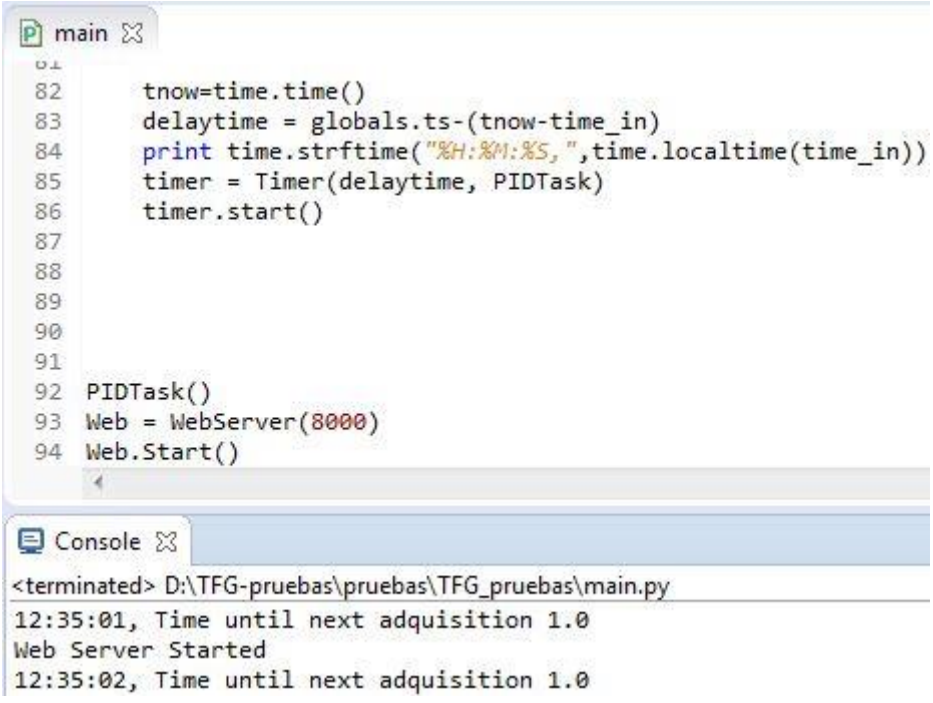


Figura 14: PyDev en Eclipse

3.2.2. Programas Desarrollados

Se va a proceder a presentar el desarrollo de los programas utilizados para la realización del servidor web. Se va a proporcionar una breve descripción de las funciones de cada programa implementado:

- Main: Este es el programa principal, donde se lanza el servidor web y se ejecuta de manera periódica el controlador.
- Globals: En este programa se encuentran todas las variables que se van a utilizar en los programas a lo largo de la ejecución. Se les asigna un valor inicial.
- Webserver: se encarga de escuchar las peticiones que se realizan desde la web y dirigirlas para ser procesadas.
- Events: ejecuta las peticiones que han sido escuchadas por el servidor, las procesa y, finalmente, las prepara la devolver una respuesta en el formato adecuado.

En el diagrama de flujo se muestra el funcionamiento conjunto de los programas y la interacción entre ellos, siendo: Main-amarillo, Globals-rojo, Webserver-verde y Events-morado.

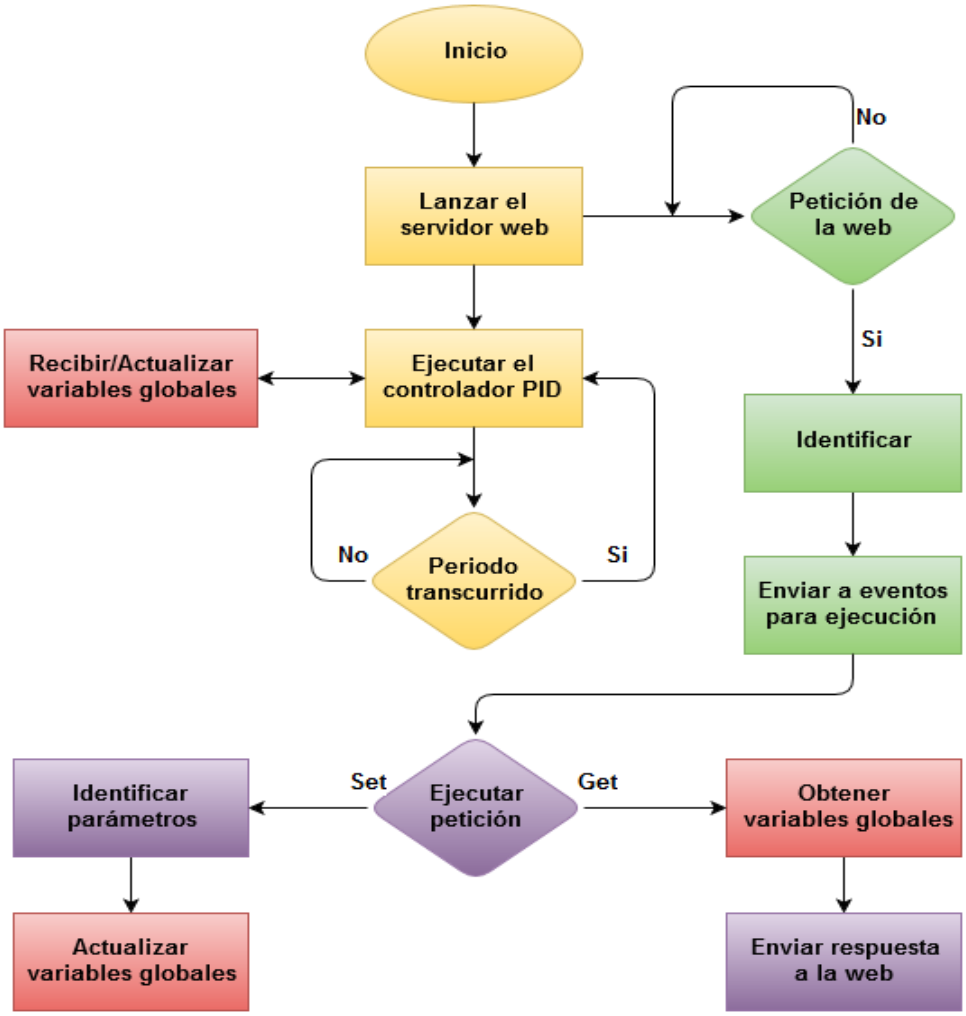


Figura 15: Diagrama funcionamiento back-end

4. SISTEMA DE CONTROL

4.1 Implementación

Para lograr el control de la temperatura, se ha implementado el algoritmo representado en el siguiente diagrama de flujo. En los siguientes apartados se detallará su funcionamiento. Para una explicación detallada del código empleado consultad el Manual de Programación.

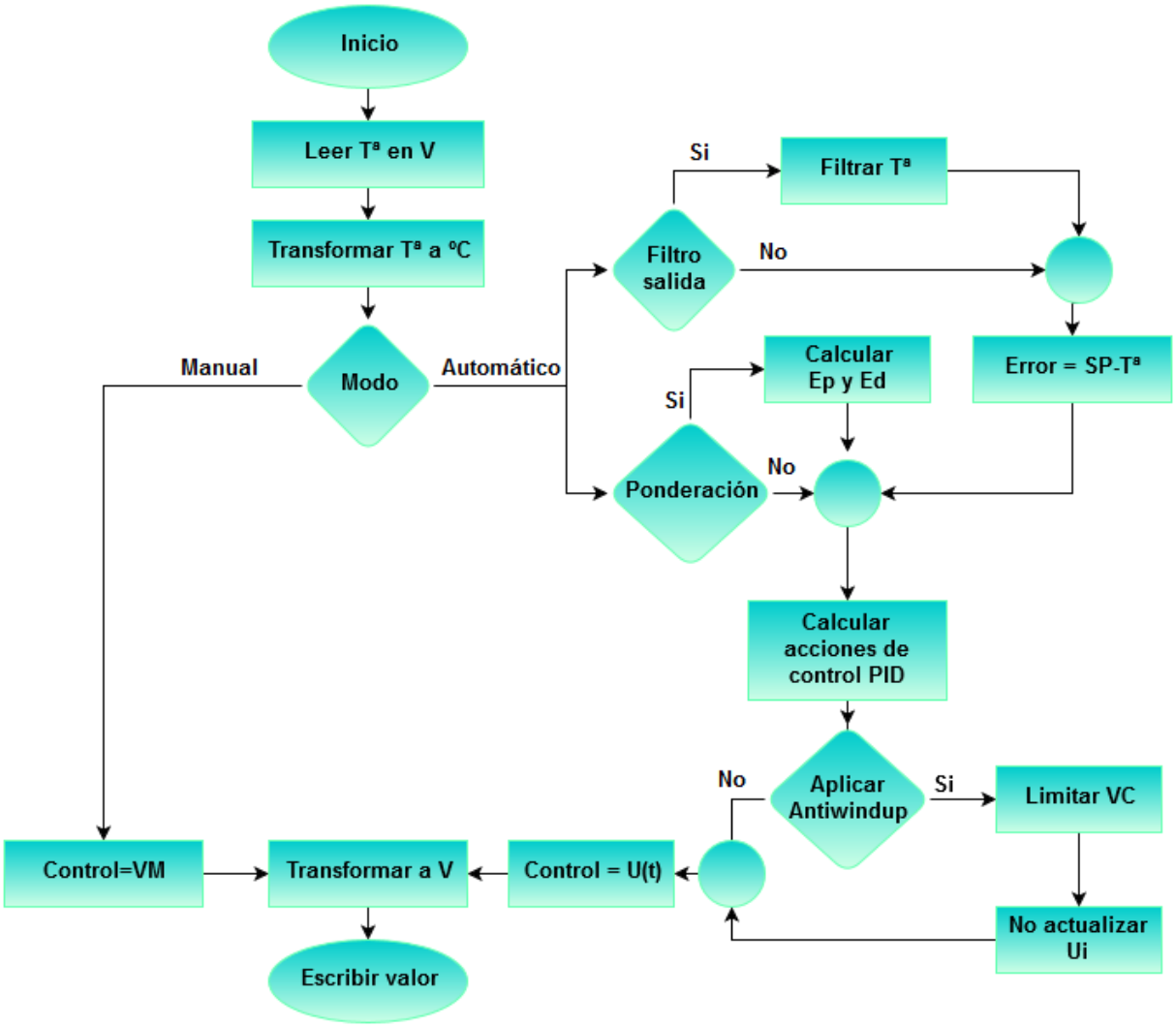


Figura 16: Diagrama controlador PID

4.1.1 Modo Manual

En este modo el usuario es el encargado de hacer el control en función de la respuesta que observe del sistema. La acción de control que se introduce es la que se aplica directamente, restringiendo de manera interna esta actuación para la protección de los equipos entre 0-100% de actuación.

La aplicación por defecto arranca en modo manual por dos motivos, primero para inicializar los parámetros necesarios a la hora de conmutar al modo automático, ya que requiere valores de algunas variables en instantes anteriores, y segundo, para que el usuario realice la configuración pertinente.

Siendo U(t) la acción de control y VM(t) la variable manipulada, en este caso el tanto por ciento de aplicación del voltaje de salida del que dispone la Raspberry Pi el cual se encuentra en el rango [0,5]V. El algoritmo de control manual es el siguiente:

$$U(t) = VM(t) \quad (4.1)$$

4.1.2 Modo automático

Este modo permite el ajuste automático ante cambios en la referencia. El algoritmo básico de control que se utiliza en los desarrollos teóricos se corresponde con (*Blasco)*:

$$U(t) = Kc\left(E(t) + \frac{1}{Ti}\int_0^t E(\tau)d\tau + Td\,\frac{dE(t)}{dt}\right) \quad (4.2)$$

Discretizando cada término se obtiene:

$$Up(k) = Kc\,E(k) \quad (4.3)$$

$$Ui(k) = Ui(K-1) + \frac{Kc\,Ts}{Ti}\,E(k) \quad (4.4)$$

$$Ud(k) = Kc\,Td\,\frac{E(k)-E(k-1)}{Ts} \quad (4.5)$$

$$U(t) = Up(k) + Ui(k) + Ud(k) \quad (4.6)$$

Se ha programado de manera que al controlador básico, el cual incluye antiwindup, se le pueden añadir diferentes funcionalidades como el filtrado de la señal de entrada o la opción de hacer ponderación de la referencia. Se procede a exponer cada una de estas funciones:

- Antiwindup: Uno de los principales problemas asociados a controladores con acción integral es el windup del integrador. Este problema aparece cuando el error no se anula lo suficientemente rápido y el regulador suministra una acción de control elevada. Debido a las limitaciones de los actuadores, no se puede permitir que la acción de control sea superior a los límites del sistema. Si la acción de control no se transmite en su totalidad se retrasa la atenuación del error, además una vez que el sistema alcanza el valor deseado en la variable de salida en estas condiciones, el término integral es muy elevado puesto que el integrador no deja de operar y sería necesario que el error fuera negativo durante un tiempo para reducir el valor de la integral. La solución es hacer que el regulador deje de integrar si la acción de control se satura. La condición antiwindup es la siguiente:

$$u_{mín} \leq U(t) \leq u_{máx}$$

Si se cumple la condición el valor de Ui(k) es válido, en caso contrario :

-No se actualiza el valor Ui(k)= Ui(k-1)
-La acción de control calculada se ajusta al valor máximo o mínimo.

$$U(t) > u_{máx} \rightarrow U(t) = u_{máx}$$

$$U(t) < u_{mín} \rightarrow U(t) = u_{mín}$$

El objetivo es evitar que la integral crezca demasiado si la acción de control está saturada.

- Filtrado de la salida: Se ha implementado un filtrado de la señal de salida mediante el cual es posible ajustar el nivel de ruido modificando el parámetro $\alpha \in [0,1)$. Obteniendo para $\alpha=0$ la señal sin filtrar y teniendo en cuenta que para valores elevados de α podemos hacer que la respuesta del sistema sea muy lenta, alterando el comportamiento original del sistema sin filtro. Las ecuaciones de implementación del filtro son las siguientes:

$$Yf(k) = (1-\alpha)Y(k) + \alpha\, Yf(k-1) \quad (4.7)$$
$$E(k) = R(k) - Yf(k) \quad (4.8)$$

Siendo Yf(k) la temperatura filtrada, Y(k) la temperatura sin filtrar, E(k) el error y R(k) la referencia.

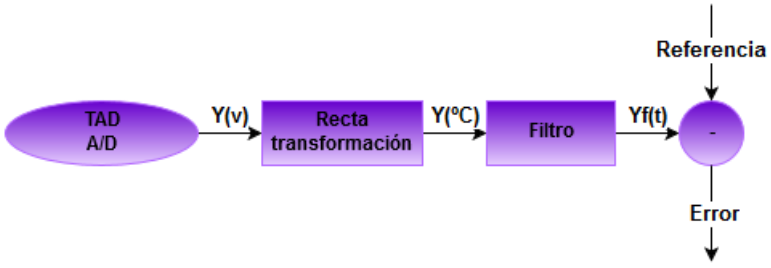


Figura 17: Diagrama filtro

- Ponderación de la referencia: consiste en tratar por separado la referencia y la medida de la variable controlada en la acción proporcional y en la derivada. La expresión del PID es la siguiente, siendo *b* y *c* los parámetros de ponderación en la acción proporcional y en la derivada respectivamente:

$$U(t) = Kc\left(bR(t) - Y(t) + \frac{1}{Ti}\int_0^t e(\tau)d\tau + Td\left(c\frac{dR(t)}{dt} - \frac{dY(t)}{dt}\right)\right) \quad (4.9)$$

Las configuraciones más comunes a la hora de realizar ponderación de la referencia son:
- I-PD con los valores b=c=0.
-PI-D con los valores b=1 y c=0.

4.2 Métodos de Diseño de Controladores

Se van a presentar los tres métodos implementados para la obtención de los controladores, todos ellos a partir de procesos con modelos de primer orden.

4.2.1 Ciancone

Este método se ajusta teniendo en cuenta (*Blasco)*:

- La minimización de la integral del error absoluto (IAE).
- Limitación de la variable manipulada
- Los errores de modelado. Funciona con variaciones de hasta un 25% en los parámetros del modelo

El procedimiento para el cálculo de los parámetros es el siguiente:

1. Calcular la fracción de retardo a partir del modelo:

$$fr = \frac{\theta}{\theta+\tau} \quad (4.10)$$

2. Obtener de los gráficos:
   a. Constante de proporcionalidad Kc conociendo Kp.
   b. Tiempo integral Ti conociendo $(\theta + \tau)$.
   c. Tiempo derivativo Td conociendo $(\theta + \tau)$.

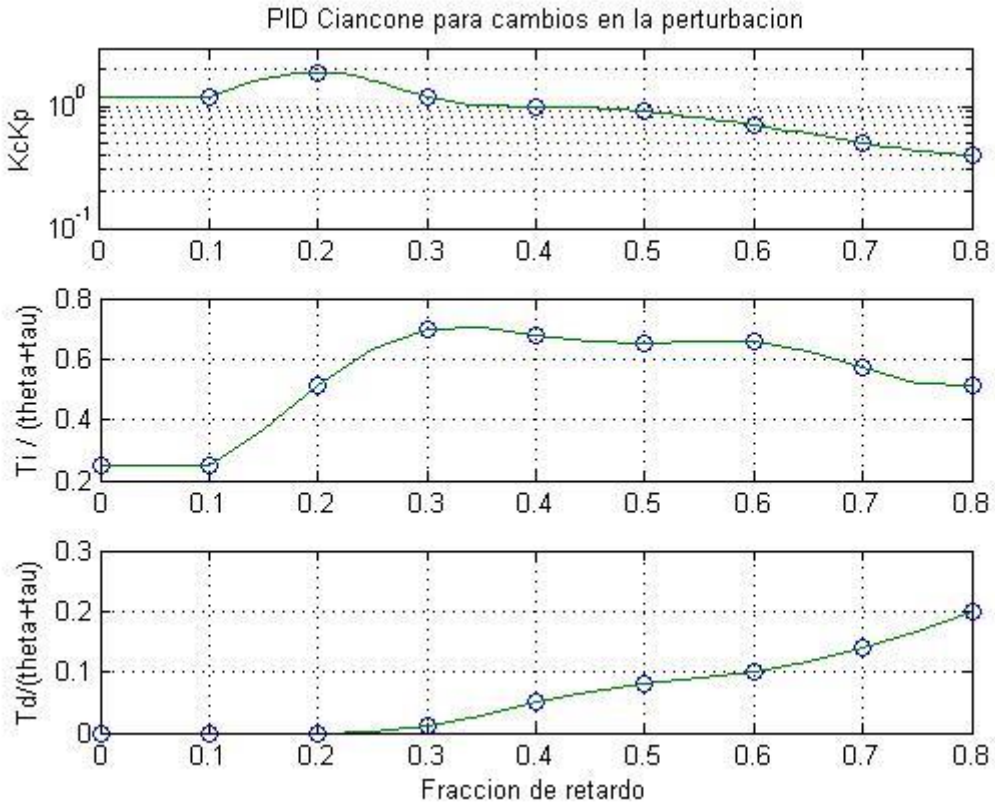


Figura 18: Ciancone

4.2.2 Cohen-Coon

El criterio de diseño para este método es el rechazo de perturbaciones e intenta posicionar los polos dominantes que proporcionan una razón de decaimiento de amplitud ¼ *(Shahrokhi)*.

Se basa en un modelo de primer orden con retardo:

$$P(s) = \frac{K}{1+sT}\, e^{-sL} \quad (4.11)$$

Del cual se obtienen los siguientes parámetros:

$$a = \frac{K\,L}{T} \quad (4.12)$$

$$\tau = \frac{L}{L+T} \quad (4.13)$$

Finalmente se introducen en la siguiente tabla, según el controlador deseado:

Controlador	Kc	Ti	Td
P	$\frac{1+\frac{0.35\,\tau}{1-\tau}}{a}$		
PI	$\frac{0.9\left(1+\frac{0.092\tau}{1-\tau}\right)}{a}$	$\frac{3.33-3\tau}{1+1.2\tau}\,L$	
PD	$\frac{1.25\left(1+\frac{0.013\tau}{1-\tau}\right)}{a}$		$\frac{0.27-0.36\tau}{1-0.87\tau}\,L$
PID	$\frac{1.35\left(1+\frac{0.18\tau}{1-\tau}\right)}{a}$	$\frac{2.5-2\tau}{1-0.39\tau}\,L$	$\frac{0.37-0.37\tau}{1-0.81\tau}\,L$

El problema principal de este método es que tiene poca robustez debido a su criterio de diseño y suele ser inestable, ya que los tiempos integrales obtenidos además de la ganancia proporcional, suelen ser muy elevados en comparación con los otros métodos aquí citados.

4.2.3 SIMC

Este método desarrollado en el año 2003 por Skogestad sirve para obtener los parámetros de controladores tipo PI o PID, evitando la cancelación de los polos mediante una redefinición del modo integral, para los casos de sistemas dominados por constantes de tiempo grandes *(Skogestad, 2003)*.

El procedimiento de cálculo es el siguiente, para modelos de primer orden, primero se establece el parámetro τc, teniendo en cuenta la siguiente consideración:

$$\tau c > \theta$$

A continuación se obtienen los parámetros del controlador según las fórmulas:

$$Kc = \frac{1}{Kp\,(\tau c+\theta)} \quad (4.14)$$

$$Ti = min(\tau, 4\,(\tau c+\theta)) \quad (4.15)$$

5. COMUNICACIÓN

En este capítulo se van a exponer los diferentes tipos de comunicación utilizados, así como el montaje del hardware empleado en la realización de este proyecto.

Si se desea ver el resultado de aplicación de todo el contenido desarrollado en esta memoria consultad el anexo: Ejemplo Práctico.

5.1 Hardware Empleado

5.1.1 Raspberry Pi

Raspberry Pi es un ordenador de placa simple, del tamaño de una tarjeta de crédito, desarrollado en Reino Unido por la fundación Raspberry Pi, con el objetivo de llegar al máximo número de usuarios siendo lo más barato posible. Una de las aplicaciones para las que fue diseñada es fomentar la enseñanza de ciencias de la computación en las escuelas.

El primer lote de placas se puso a la venta en febrero de 2012, desde entonces se han desarrollado varios modelos. El utilizado en este proyecto ha sido Raspberry Pi 3 Model B V1.2 cuyas características son *(Element14)*:

- Procesador Broadcom BCM2837 a 1.2GHz
- ARM Cortex-A53 de 64 bits y cuatro núcleos
- 1 GB de memoria RAM
- SO: Raspbian.
- Conectividad: Ethernet, Bluetooth 4.1, 4 puertos USB 2.0, HDMI, conector video/audio RCA, conector GPIO de 40 pines.
- Puerto micro USB para alimentación de 2.5A
- Dimensiones: 85 x 56 x 17 mm

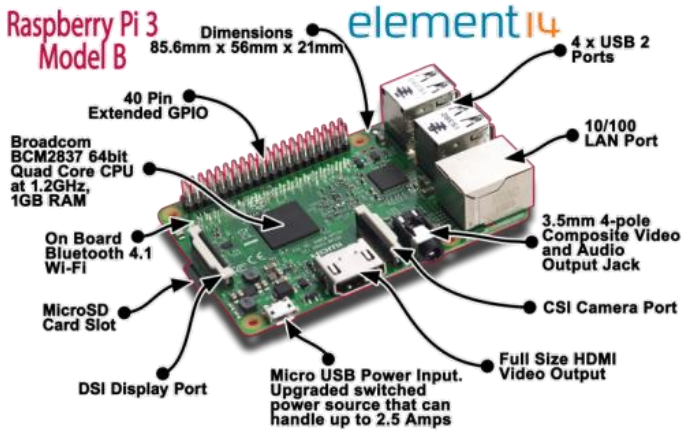


Figura 19: Raspberry Pi Model B v1.2. Imagen obtenida de www.element14.com

5.1.2 Conversor Analógico/Digital

Se ha incorporado el complemento ADC Pi (8channel) V2.0, el cual permite la conversión analógico/digital de las señales de entrada. Este conversor es compatible con distintos modelos de Raspberry Pi, incluido el utilizado en este proyecto. *(ABelectronics)*

Especificaciones:

- 8 entradas analógicas
- Vdd: 5V
- Todas las entradas y salidas A/D: VSS-0.4V hasta VDD+0.4V
- Corriente en los pins de entrada: ±2 mA
- I2C SDA/SCL voltaje 5V
- Corriente del puerto I2C 100mA

Se conecta mediante los pins GPIO y el control se realiza mediante el puerto I2C de la Raspberry Pi.

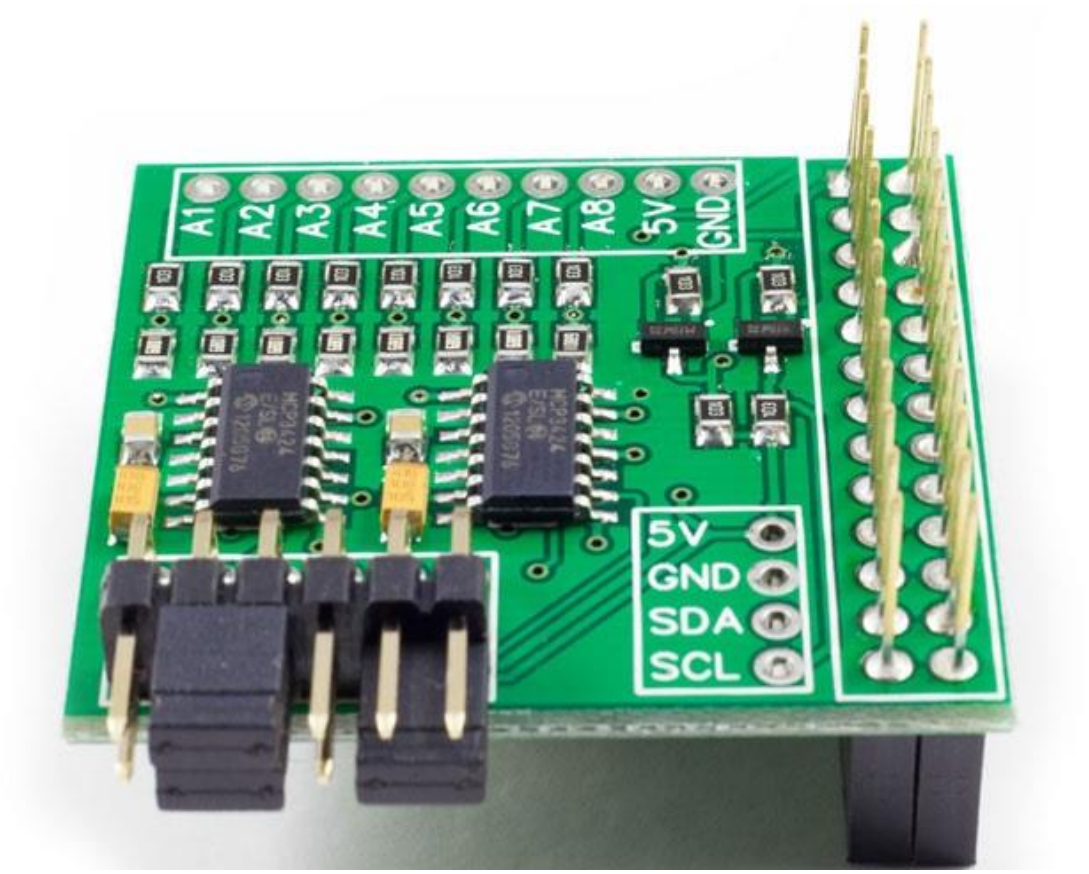


Figura 20: ADC Pi v2.0. Imagen obtenida de https://www.abelectronics.co.uk/

5.1.3 Conversor Digital/Analógico

Para la conversión de las señales de salida se ha utilizado el modelo adafruit MCP4725, el cual permite realizar la conversión de digital a analógico, controlado via I2C. El chip utiliza direcciones I2C de 7 bits entre la 0x62-0x63, seleccionadas mediante jumpers. Tiene una resolución de 12 bits por lo tanto sus valores de entrada se encuentras entre 0 y 4096 bits, y se configurará de manera que la salida esté comprendida entre 0 y 5V. *(Adafruit)*

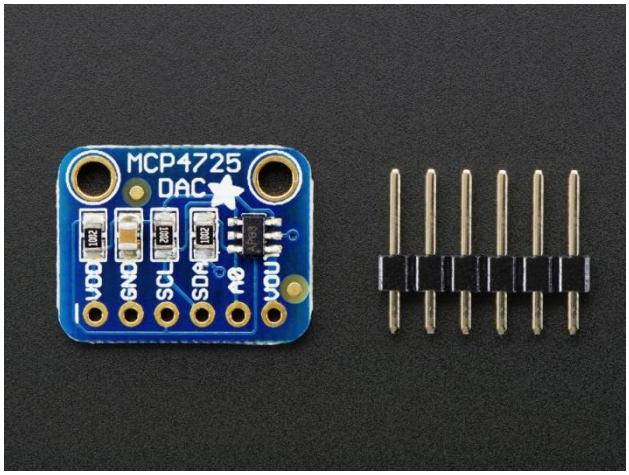


Figura 21: MCP4725. Imagen obtenida de https://www.adafruit.com/

5.1.4. Montaje

Se va a mostrar el resultado del montaje del hardware expuesto en los tres puntos anteriores. De acuerdo con la siguiente imagen, la alimentación de la placa Raspberry Pi 3 se realiza a través de dos jumpers (cables rojo y negro). Sobre la Raspberry se ha insertado la placa ADC Pi, la cual es verde y se encuentra en la parte inferior derecha en la imagen, conectada mediante los pins GPIO. Sobre esta placa, de color azul, se encuentra el chip MCP4725. La entrada, salida y neutro del conjunto son los cables verde, naranja y amarillo respectivamente.

Para la conexión a red existen dos opciones: por WiFi o a través de Ethernet. Una vez se encuentre en red es posible acceder a la web diseñada, utilizando cualquier ordenador o dispositivo conectado a la misma. Para la conexión, es necesario introducir en el navegador la dirección IP de la Raspberry Pi y seleccionar el puerto 8000.

Para el desarrollo de este proyecto se ha utilizado la placa Raspberry Pi descrita anteriormente, la cual dispone de una pantalla táctil, como puede observarse en la imagen. Es preciso aclarar que para el desarrollo de este trabajo la pantalla táctil es prescindible.

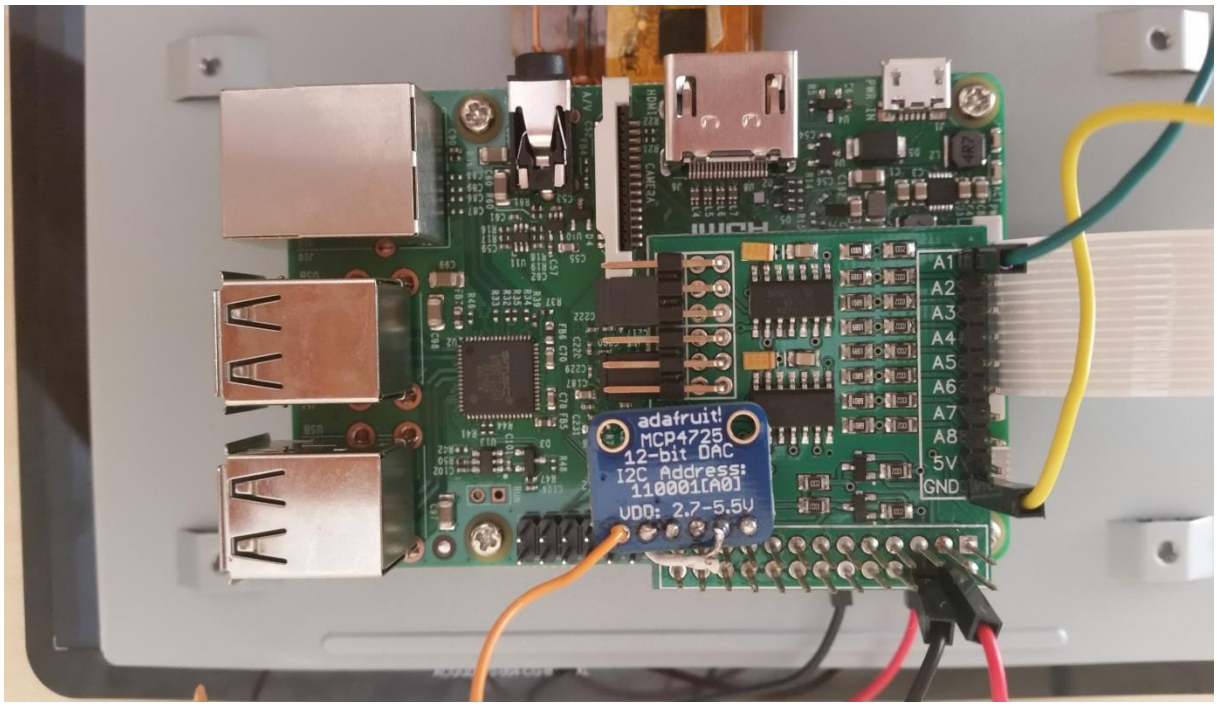


Figura 22: Montaje

5.2 Protocolos de Comunicación

En los siguientes apartados se van a comentar los protocolos utilizados para establecer la comunicación de datos entre las diferentes partes de este proyecto. Para conocer de manera detallada las librerías empleadas o los comandos utilizados consultad el anexo: Manual de programación.

5.2.1. Web-Servidor

La comunicación entre la web y el servidor se realiza mediante el protocolo denominado CGIs o Interfaz de Entrada Común. CGI es una importante tecnología de la World Wide Web que establece una comunicación entre el cliente, a través del navegador, y el programa, ejecutado en un servidor.

Desde las diferentes funciones programadas en JavaScript que rigen el comportamiento de la aplicación web, cada vez que era necesario actualizar o recibir datos del servidor, se ha llamado a la siguiente función, asignando los argumentos pertinentes, la cual se encargaba de crear el CGI.

```
function getData(query,get,action){

    $('#loading').css("display", "block");
    $.get(query+'.cgi?'+get,function(data){
        $('#loading').css("display", "none");

eval(action+"('"+data+"')");

    }, "text");
}
```

Figura 23: JavaScript CGI

Se procede a analizar los argumentos de la función getData:

- Query: es la consulta realizada. Según el tipo de acción que queramos ejecutar introducimos un dato de tipo cadena de caracteres que será identificado en el servidor, en concreto en el programa webserver, a través del cual se direccionará al programa events, donde se realizarán las acciones pertinentes y se actualizarán o se prepararán los datos para la respuesta.
- Get: son los datos que se intercambian. Se introducen en forma de cadena de caracteres con un formato específico para que sean procesados correctamente. Mediante una función de Python la cadena que contiene los datos se separa y almacena en un objeto tipo diccionario.
- Action: en los casos en los que se reciben datos del servidor, este argumento hace referencia al programa implementado en JavaScript que procesa esos datos.

5.2.2. Servidor-Proceso

El protocolo utilizado para la comunicación entre la Raspberry Pi y el proceso es el denominado I^2C, en inglés Inter-Integrated Circuit. Es un bus de comunicación desarrollado en 1982 por Philips Electronics. El objetivo es establecer la comunicación entre microcontroladores y periféricos a una velocidad aceptable, cuyo rango abarca desde 100Kbps hasta 3.4Mbps *(Verdú, 2014)*.

El método de comunicación de este protocolo es en serie y síncrono, es decir, se necesitan dos líneas para desencriptar la información. Estas líneas de comunicación son:

- SCL: pulsos de reloj que sincronizan el sistema.
- SDA: línea de datos.

Esta comunicación se caracteriza por la utilización de dos líneas, pero son necesarias dos más: una que proporcione la alimentación, a tensión continua específica según el dispositivo, y otra a masa.

Se pueden conectar tantos dispositivos como se desee, siempre y cuando no se supere su capacidad máxima, 400pF.

En este bus se pueden diferenciar dos tipos de dispositivos:

- Maestro: suele haber uno pero pueden haber más, son os que pueden iniciar o parar la comunicación y son los encargados de generar la señal de reloj.
- Esclavo: contienen su propia dirección dentro del bus a través de la cual se envían o reciben datos.

El funcionamiento del sistema se resume a continuación. En reposo las dos líneas de comunicación están en nivel alto. Existen marcas de inicio y parada asociadas a determinadas posiciones de SCL y SDA. Una vez el maestro envía la marca de inicio, se procede a enviar 7 bits que conforman la dirección del esclavo, seguidos de un bit que indica la acción, lectura o escritura. Si la dirección enviada coincide con alguno de los dispositivos conectados al bus, el dispositivo envía una marca de reconocimiento, que significa que acepta la solicitud. La transmisión de datos se realiza en paquetes de 8 bits hasta que el maestro corte la comunicación mediante la marca de parada mencionada anteriormente.

En el desarrollo de este proyecto se ha utilizado este protocolo para la comunicación entre la Raspberry Pi (maestro) y los dos conversores (esclavos).

6. CONCLUSIONES

Las conclusiones que se extraen a lo largo del desarrollo del proyecto son varias.

En primer lugar, teniendo en cuenta los resultados obtenidos de aplicar el controlador PID desarrollado en este proyecto a una situación real, como es el horno utilizado para el ejemplo práctico que se encuentra en los anexos de esta memoria, se puede concluir que los resultados son satisfactorios ya que es posible realizar el control de la temperatura, su seguimiento en tiempo real y la configuración de todos los parámetros del controlador a través de la interfaz diseñada. Además, se cumplen todos los objetivos y requerimientos para los cuales este proyecto fue diseñado en un primer momento e incluso se han llegado a desarrollar mejoras, como la pestaña auxiliar de Proceso, en la cual se ofrece la posibilidad de diseñar los controladores.

En segundo lugar, considerando el nivel de conocimiento adquirido durante el desarrollo de este proyecto, sobre todo en la parte relacionada con el diseño web, si se tuviera que realizar de nuevo en estos momentos, muchos de los programas que lo conforman serían modificados e implementados de una forma más eficiente. Los errores cometidos en la programación o los desarrollos poco eficientes han sido fruto de la falta de experiencia, que se ha ido compensando a lo largo del tiempo, pero en proyectos de esta índole nada queda exento de mejoras.

Seguidamente es importante remarcar el carácter y la motivación que ha guiado el desarrollo de este trabajo. En todo momento se ha tenido presente que no se estaba inventando nada nuevo, con este trabajo se ha pretendido adaptar a las necesidades y demandas actuales una tecnología ya existente, proporcionando al cliente una nueva forma de interactuar con el proceso, mucho más versátil, gráfica y sencilla. A esto se le suma el hecho de que ha sido desarrollado empleando plataformas de bajo coste y utilizando la red, para permitir una mejor accesibilidad, tanto física como económicamente hablando.

Como se ha comentado con anterioridad, este proyecto está sujeto a modificaciones y mejoras como podrían ser:

- Registro de usuarios: para limitar el acceso y la manipulación de los parámetros configurables a través de la interfaz. La aplicación de esta mejora a nivel industrial serviría para diferenciar entre operarios de alto y bajo nivel. Permitiendo a los primeros, con mayor nivel de conocimiento, realizar la modificación de los parámetros del controlador. En cambio, para los segundos solo sería posible la manipulación en modo manual o el mero seguimiento del proceso.
- Autotuning: esta mejora sustituiría a la implementada que desarrolla el controlador a partir del modelo de primer orden, ya que con la incorporación del autotuning la identificación del proceso, así como el desarrollo del controlador, se realizarían de manera automática.

Finalmente también son remarcables las futuras aplicaciones o desarrollos que se podrían realizar tomando este trabajo como base. Permitiría desarrollar un proyecto mucho más ambicioso, en el cual, no solo se diseñara la interfaz web y el controlador, si no la placa y los sensores necesarios. Esto

permitiría ampliar el nivel de aplicación de uno a varios ejecutándose a la vez, implementando un controlador multi-lazo, con sus correspondientes sensores y actuadores.

De esta forma se crearía un producto original, compuesto por software y hardware propio, que permitiría su comercialización.

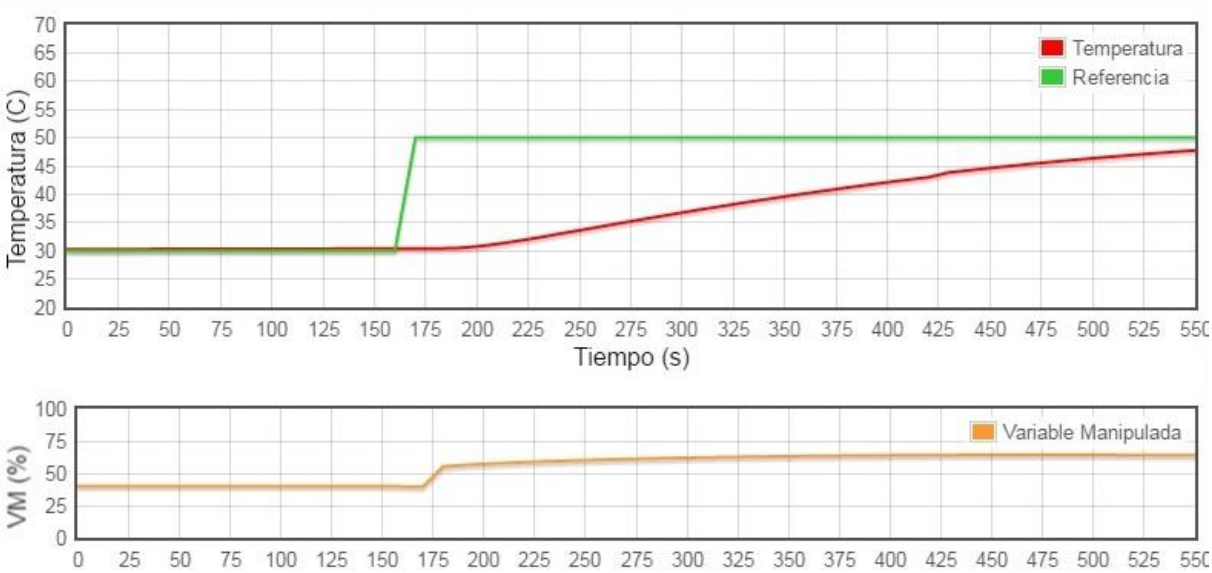


Figura 24: Modo Automático. Ajuste ante cambio en la referencia a 50ºC

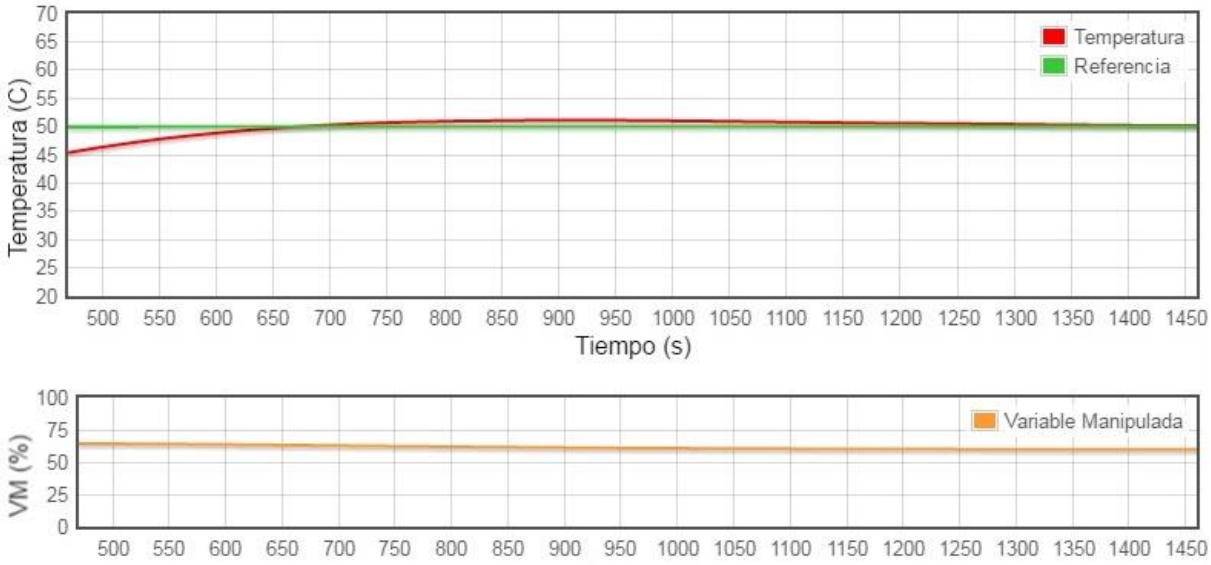


Figura 25: Modo Automático. Ajuste en régimen permanente

7. BIBLIOGRAFÍA Y REFERENCIAS

ABelectronics. (21/06/2016). Obtenido de https://www.abelectronics.co.uk/p/56/ADC-Pi-Plus-Raspberry-Pi-Analogue-to-Digital-converter.

Adafruit. (22/06/2016). Obtenido de https://www.adafruit.com/products/935

Alfaro, V., Vilanova, R. (2011). *Control PID Robusto: Una visión panorámica.* Revista Iberoamericana de Automática e Informática Industrial 8 141-158.

Auslander, D.E. (1971). *Evolutions in Automatic Control. Journal of Dynamic Systems, Measurement and Control*. ASME Transitions.

Bennet, S.(1979). *A history of control engineering: 1800-1930.* Editorial Peter Peregrinus, Londres.

Bennet, S.(1993). *A history of control engineering: 1930-1955.* Editorial Peter Peregrinus, Londres.

Blasco, F., Martínez, M., Senent, J., Sanchis, J (s.f.). *Sistemas Automáticos.* Editorial UPV Ref.: 2000.4186.

Dickinson, H. W., Jenkins, R. (1927) *James Watt and the steam engine*. Oxford University Press.

Element14 (20/06/2016). Obtenido de https://www.element14.com/community/docs/DOC-80899/l/raspberry-pi-3-model-b-technical-specifications

Flot (s.f). Obtenido de http://www.flotcharts.org/

Raspberry Pi (20/06/2016). Obtenido de https://www.raspberrypi.org/

Sancho, A. (2015). *Diseño de un controlador PID para un proceso térmico basado en las plataformas Arduino y Raspberry Pi.* Valencia: Trabajo de Fin de Grado ETSII UPV.

Shahrokhi, M., Zomorrodi, A. (s.f). *Comparison of PID Controller Tuning Methods.* Sharif University of Technology.

Skogestad, S. (2003). *Simple analytic rules for model reduction and PID controller tuning.* Journal of Process Control 13, 291-309.

Verdú, D. (2014). *Desarrollo integral de un quadrotor: control de orientación basado en una IMU de bajo coste y control de altura mediante un sensor barométrico*. Valencia: Trabajo de Fin de Grado ETSII UPV.

PRESUPUESTO

PRESUPUESTO

A continuación se procede a describir el presupuesto global estimado del proyecto realizado, con el objetivo de mostrar la aportación económica necesaria para su realización.

1. Cuadro de Mano de Obra

Código	Empleado	Salario mensual (€/mes)	Salario (€/h)
MO.GITI	Graduado en Tecnologías Industriales	3200	20

2. Cuadro de Materiales

El cambio de moneda empleado ha sido 1 £=1.20299€ (consultado el 28/06/2016).

Código	Ud	Denominación	Precio (€)	Precio (£)	Rend	Total (€)
MT.RPI	u	Raspberry Pi 3 Model B v1.2	45.9		1	45.90
MT.ADC	u	ADC Pi v2.0		14.99	1	18.03
MT.MCP	u	MCP4725		4.5	1	5.41
		Precio total				**69.35**

3. Cuadro de Maquinaria

Para el cálculo de la amortización de la maquinaria se ha supuesto un periodo de amortización de 1 año, en el cual la vida útil es de 1800h/año, equivalente a 225 días laborales con jornadas de trabajo de 8 horas.

Código	Unidades	Denominación	Precio (€)
M.ORDP	1	Ordenador Portátil	700

4. Cuadro de Unidades de Obra

Capítulo 1: Implementación del front-end.

1.01		Diseño de la interfaz web			
Código	Unidades	Descripción	Precio (€)	Rendimiento	Importe (€)
		Diseño de la apariencia y el estilo de la interfaz web			
MO.GITI	h	Graduado en Tecnologías Industriales	20	120	2400
M.ORDP	h	Ordenador Portátil	0.36	120	43.2
	%	Costes directos complementarios		2	48.864
		Total unidad de obra			**2492.064**

1.02		Programación de la interfaz web			
Código	Unidades	Descripción	Precio (€)	Rendimiento	Importe (€)
		Programación de las funciones dinámicas de la interfaz web			
MO.GITI	h	Graduado en Tecnologías Industriales	20	50	1000
M.ORDP	h	Ordenador Portátil	0.36	50	18
	%	Costes directos complementarios		2	20.36
		Total unidad de obra			**1038.36**

Capítulo 2: Implementación del back-end.

2.01		Programación del Controlador PID			
Código	Unidades	Descripción	Precio (€)	Rendimiento	Importe (€)
		Programación del algoritmo necesario para realizar el control de la temperatura			
MO.GITI	h	Graduado en Tecnologías Industriales	20	50	1000
M.ORDP	h	Ordenador Portátil	0.36	50	18
	%	Costes directos complementarios		2	20.36
		Total unidad de obra			**1038.36**

2.02		Programación del Servidor web			
Código	Unidades	Descripción	Precio (€)	Rendimiento	Importe (€)
		Implementación de los programas necesarios para el funcionamiento del servidor web			
MO.GITI	h	Graduado en Tecnologías Industriales	20	70	1400
M.ORDP	h	Ordenador Portátil	0.36	70	25.2
	%	Costes directos complementarios		2	28.504
		Total unidad de obra			**1453.704**

Capítulo 3: Montaje del hardware.

3.01		Montaje del hardware			
Código	Unidades	Descripción	Precio (€)	Rendimiento	Importe (€)
		Montaje de la Raspberry Pi y las TAD			
MO.GITI	h	Graduado en Tecnologías Industriales	20	1	20.00
MT.RPI	u	Raspberry Pi 3 Model B v1.2	45.9	1	45.90
MT.ADC	u	ADC Pi v2.0	18.03	1	18.03
MT.MCP	u	MCP4725	5.41	1	5.41
	%	Costes directos complementarios		2	1.79
		Total unidad de obra			**91.13**

5. Presupuesto Total

Código	Descripción	Subtotal (€)	Importe(€)
1	Implementación del front-end	3530.42	
2	Implementación del back-end	2492.06	
3	Montaje del hardware	91.13	
	Presupuesto de ejecución material		**6113.61**
	Gastos generales 13%		**794.77**
	Beneficio industrial 6%		**366.82**
			7275.20
	IVA 21%		**1527.79**
	Total presupuesto		**8802.99**

ANEXOS

MANUAL DE USUARIO

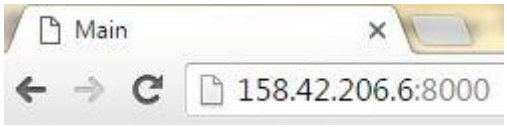
En los siguientes puntos de este anexo se van a detallar los pasos a seguir y las opciones disponibles para el correcto funcionamiento de la interfaz web desarrollada.

1. Arranque

En caso de disponer de una Raspberry Pi con pantalla táctil existen dos opciones a la hora de acceder a la interfaz web, dependiendo de si el servidor se está ejecutando en el dispositivo en el que nos encontramos. Como se ha mencionado en la memoria de este proyecto, para el desarrollo del mismo se contaba con una pantalla táctil acoplada a la Raspberry Pi, por lo tanto existía la opción de acceder desde el mismo dispositivo a la aplicación, pero este suceso debe considerarse una excepción.

Anexo Figura 1: local host

La otra opción, la cual es la usual para el cliente, es acceder conectándose a la dirección IP de la Raspberry Pi y especificando el puerto 8000 al igual que antes.

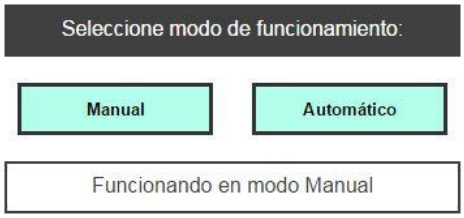


Anexo Figura 2: Arranque

2. Main

En este apartado se van mostrar todos los elementos que componen esta página, así como sus funciones y limitaciones.

- Conmutador de modo de funcionamiento: se ha limitado de manera que sólo se pueden modificar los parámetros se encuentra el modo manual activo. Por el contrario, solo se podrá manipular la referencia.



Anexo Figura 3: Modo de funcionamiento

- Indicador de la temperatura: este elemento actualiza la temperatura con una frecuencia igual al periodo introducido por el cliente.

Anexo Figura 4: Indicador Tª

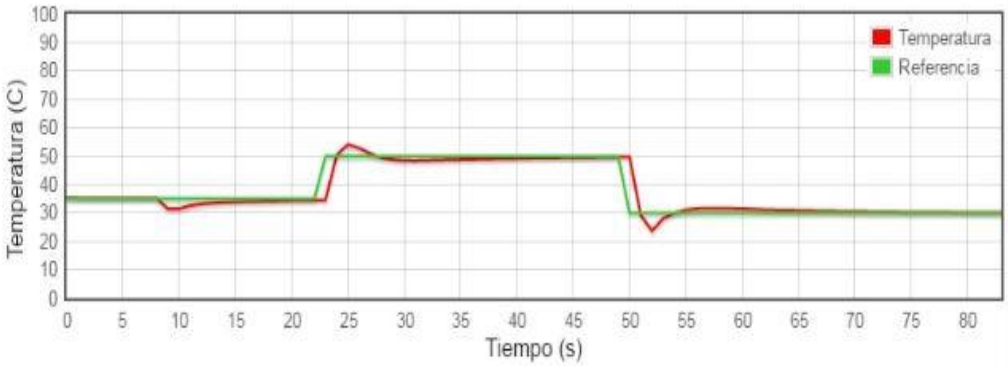
- Variable manipulada y referencia: Estos dos campos se pueden actualizar si el modo de funcionamiento es manual, en este modo el valor de VM está limitado a ser un número entero comprendido entre 0 y 100, por el contrario no es posible actualizar los valores. En modo automático es posible modificar la referencia, la variable manipulada se vuelve un indicador, actualizando su valor con una frecuencia igual al periodo.

Anexo Figura 5: VM y SP

- Alarmas: nos indica el estado del sistema en función de las temperaturas que se hayan configurado como límite. Consta de un indicador de tipo texto y otro gráfico, el cual emula un indicador luminoso parpadeando si se sobrepasan los límites.

Anexo Figura 6: Alarma

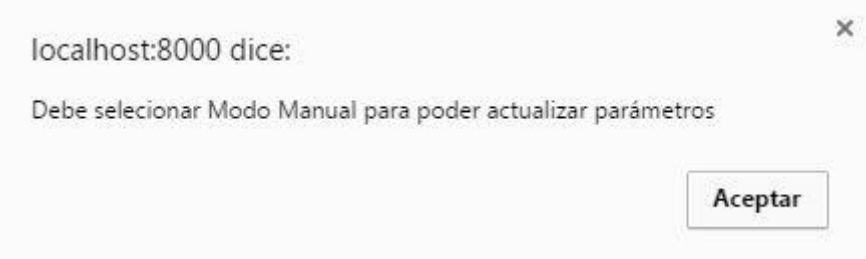
- Gráficas: contienen hasta cien puntos cada una, se dibuja un nuevo punto con frecuencia igual al periodo. El eje y de la gráfica de la temperatura es configurable.



Anexo Figura 7: Gráfica

3. Configuración

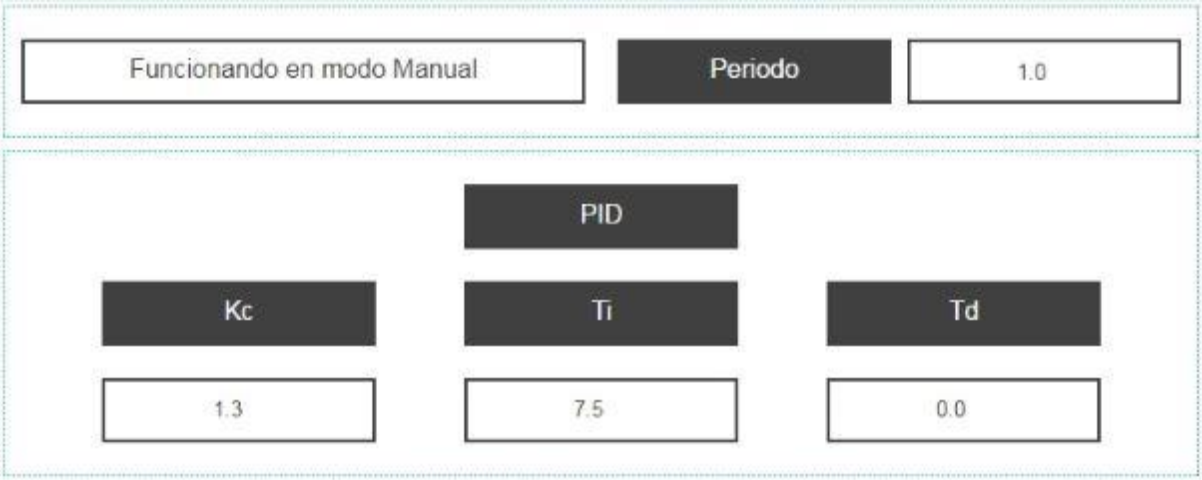
Como se ha mencionado anteriormente, no es posible la modificación de los parámetros si el controlador se encuentra en modo automático. En caso de estar es ese modo y acceder, tanto a la pestaña de configuración como a la de proceso, aparece una alerta que indica que se debe realizar el cambio de modo y además se ocultan los botones mediante los cuales se actualizan los parámetros.



Anexo Figura 8: Alerta

En esta ventana también hay parámetros que se encuentran restringidos a determinados valores, al igual que en el apartado anterior ocurría con la variable manipulada. Cada vez que se introduce un valor incorrecto, como un tiempo negativo por ejemplo, aparece una alerta indicando el parámetro incorrecto y la restricción a la cual está sometido y, por supuesto, no se realiza la actualización de ningún parámetro hasta que todos se encuentran en el rango correcto. Debido a la cantidad de parámetros que se pueden configurar en este apartado, no se van a exponer uno a uno como se ha realizado en el apartado anterior debido a que su funcionamiento es el mismo.

Es conveniente tener en cuenta que los parámetros temporales no están restringidos a tener unas unidades específicas, esto es debido a que es el usuario el que decide en qué unidades introducirlos, teniendo en cuenta que todos deben encontrarse en las mismas unidades. Es recomendable introducir los parámetros en segundos.



Anexo Figura 9: PID y periodo

4. Proceso

Esta pestaña contiene la antes mencionada restricción en función del modo de operación del controlador.

Es importante destacar que en esta pestaña auxiliar, una vez introducido el modelo, cada vez que se pulsa un método de diseño de controlador se muestran los valores del mismo pero para seleccionar el controlador hay que pulsar otro botón. Se ha realizado de esta forma para que el usuario pueda comparar los parámetros obtenidos por cada método.

Anexo Figura 10: Diseño de controladores

MANUAL DE PROGRAMACIÓN

Este anexo contiene parte del código empleado en el desarrollo de este proyecto y se procederá a su exposición de manera detallada. No se incluye el código completo ya que se ha considerado que es demasiado extenso. Han sido seleccionadas las partes más relevantes.

1. HTML

Se considera importante comentar en este apartado el código para incluir archivos externos y otras opciones que se pueden incluir en la cabecera de los documentos de este tipo.

```
3⊖  <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
5
6     <title>Main</title>
7
8     <script type="text/javascript" src="jquery-1.7.2.min.js"></script>
9     <script type="text/javascript" src="todo.js"></script>
10    <script type="text/javascript" src="jquery.flot.min.js"></script>
11    <script type="text/javascript" src="jquery-ui-1.10.1.custom.min.js"></script>
12    <script type="text/javascript" src="jquery.flot.axislabels.js"></script>
13    <script type="text/javascript" src="jquery.flot.time.mim.js"></script>
14
15    <link rel="stylesheet" type="text/css" href="PIDweb.css">
16
```

Anexo Figura 11: Cabecera HTML Main

El código de la línea 4 es el que hace posible la correcta utilización de los acentos en la aplicación.

De la línea 8 a la 13 se incluyen los programas para la utilización de las gráficas además del programa que rige el comportamiento de toda la web (línea 9). Finalmente en la línea 15 se incluye el archivo que contiene todo el estilo.

Debido a la cantidad de elementos que se manejan, es importante ser ordenado a la hora de asignar las IDs de cada elemento ya que serán utilizadas posteriormente en el programa implementado en JavaScript.

2. CSS

Para hacer la aplicación más atractiva se han utilizado diversos elementos para ofrecer dinamismo. A continuación se muestra un ejemplo de un botón el cual cambia de color cuando el cursor sobrepasa el elemento. Además, como ejemplo, se van a comentar los atributos que contiene.

El atributo color y Font-weight hacen referencia al texto que contiene el botón, el resto son atributos asociados al fondo y el borde del elemento. Se puede apreciar que en CSS es posible utilizar distintos formatos para introducir los colores.

```
input.but1:hover {
background-color: white;
color:#00cc99;
border:solid;
border-color: #595959;
font-weight:bold;
}
```

Anexo Figura 12: CSS

Finalmente es importante comentar las unidades que acepta este formato, referentes al tamaño. Se distingue entre unidades absolutas y relativas:

- Absolutas:
  - in: pulgadas
  - cm: centímetros
  - mm: milímetros
  - pt: puntos
  - pc: picas
- Relativas:
  - em: relativa al tamaño en puntos de la tipografía utilizada. Equivale a la anchura de la letra M
  - ex: al igual que la anterior pero equivale a la altura de la letra x
  - px: píxel. Relativa a la resolución de la pantalla del dispositivo en el que se visualiza la página HTML

3. JavaScript

3.1. Petición SET/GET

Se va a comentar un ejemplo de petición en la cual se realizan las dos acciones básicas y se comentarán diversos aspectos importantes.

```
354 function Ciancone(){
355     kpp=parseFloat(document.getElementById("k_write").value);
356     z=parseFloat(document.getElementById("z_write").value);
357     r=parseFloat(document.getElementById("r_write").value);
358     document.getElementById("modpid_read").value="Modelo Cianconne";
359     getData('setMpid','kp='+kpp.toString()+'&tau='+z.toString()+'&teta='+r.toString()+'&ajuste=1', 'procesa_modelos');
360
361 }
362 function procesa_modelos(data){
363
364      a=data.split('&');//orden de los elementos en eventos
365      document.getElementById("kc_read").value=a[0];
366      document.getElementById("ti_read").value=a[1];
367      document.getElementById("td_read").value=a[2];
368
369 }
```

Anexo Figura 13: Petición SET/GET

En la primera línea se crea la función correspondiente a la implementación del método de Ciancone. A continuación se leen los parámetros correspondientes al modelo y se escribe en el indicador que se ha seleccionado el método de diseño correspondiente.

La comunicación con el servidor se hace mediante la función getData, la cual está explicada en la memoria en el apartado 5.2.1.

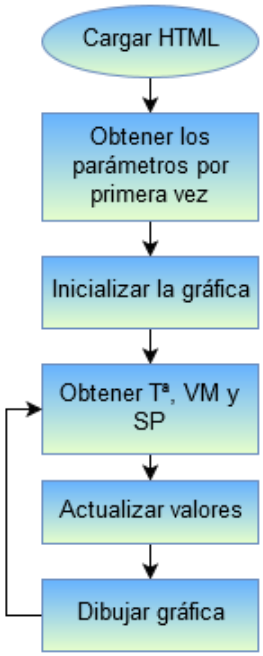
Las peticiones en las que solo se actualizan parámetros contienen como último argumento una función vacía, ya que no reciben datos de respuesta. También es importante el formato en el que se introducen los datos sea específicamente el mostrado para el posterior procesado en el servidor.

Finalmente, como esta petición también recibe datos en respuesta, éstos se tratan en la función procesa_modelos, la cual se encarga de separar la cadena de caracteres que se recibe mediante la función Split y asociarlos al elemento correspondiente. El orden en el que llegan los parámetros es el que se haya especificado en el programa eventos.

Con este ejemplo se pretende ilustrar el proceso realizado en la programación referida a esta parte. Todas las peticiones se realizan de la misma forma, realizando modificaciones pertinentes en función del objetivo.

3.2. Gráficas

Toda la información necesaria para la utilización de las gráficas dinámicas, el manual de funcionamiento y opciones de configuración, además de las librerías que las contienen se encuentra disponible en la red *(Flot).* Se va a describir la secuencia de acciones utilizada para el funcionamiento de las gráficas mediante un diagrama:



Anexo Figura 14: Secuencia datos para las gráficas

A continuación se van a mostrar las partes más destacadas de este proceso *(Sancho, 2015)*:

```
var T= new Array(100), Ti=new Array(100), VM=new Array(100), Ref=new Array(100);

function IniGraf(){

    for(i=0;i<100;i++){
        T[i]=temperatura;
        Ti[i]=0;
        VM[i]=variableM;
        Ref[i]=referencia;
    }

    var dat=[[Ti[0],T[0]],[Ti[1],T[1]],[Ti[2],T[2]],[Ti[3],T[3]],[Ti[4],T[4]],[Ti

    var dat1=[[Ti[0],Ref[0]],[Ti[1],Ref[1]],[Ti[2],Ref[2]],[Ti[3],Ref[3]],[Ti[4],

    var dat2=[[Ti[0],VM[0]],[Ti[1],VM[1]],[Ti[2],VM[2]],[Ti[3],VM[3]],[Ti[4],VM[4
```

Anexo Figura 15: Inicializar gráfica

Se declaran como variables globales los vectores que contendrán los cien valores de cada parámetro. A continuación, cuando se ejecuta la función para inicializar la gráfica se les asigna a todos los puntos el mismo valor. A continuación se crean las variables en el formato específico que acepta la librería flot. Debido a que es una tarea tediosa escribir a mano tres vectores con cien parejas de valores se ha desarrollado el siguiente programa en Python con el cual se han generado los vectores que se han copiado aquí.

```
1  dat="["
2  dat1="["
3  dat2="["
4
5  for i in range(0,100):
6      dat=dat+"[Ti["+str(i)+"],T["+str(i)+"]],"
7      dat1=dat1+"[Ti["+str(i)+"],Ref["+str(i)+"]],"
8      dat2=dat2+"[Ti["+str(i)+"],VM["+str(i)+"]],"
9
10 dat=dat+"];"
11 dat1=dat1+"];"
12 dat2=dat2+"];"
13
14 print dat
15 print dat1
16 print dat2
```

Anexo Figura 16: Creación de vectores para las gráficas

Seguidamente se realiza la configuración de los ejes y las opciones para las gráficas:

```
var options1 = {
            series: {
                lines: { show: true },
                points: { show: false },
            },
            xaxis:{
                ticks: 20,
                min:0,
                axisLabel: 'Tiempo (s)',
            },
            yaxis:{
                ticks: 10,
                min:yminM,
                max:ymaxM,
                axisLabel: 'Temperatura (C)',
            },
        };
```

Anexo Figura 17: Configuración ejes

Finalmente se dibuja por primera vez la gráfica:

```
/*Llamamos a la funcion plot*/
$.plot($("#placeholderT"),[
                        {label:"Temperatura", data:dat, color: '#ff0000'},
                        {label:"Referencia", data:dat1, color: '#33cc33'}], options1);

$.plot($("#placeholderVM"),[{label:"Variable Manipulada", data:dat2, color: '#ff9933'}], options2);
```

Anexo Figura 18: Plot gráfica inicial

El proceso para dibujar de nuevo la gráfica es el mismo, salvo la parte añadida, en la que se desplazan los datos de los vectores y se añaden los valores nuevos recibidos del servidor.

```
function Plot(){

    for(i=0;i<99;i++){
        T[i]=T[i+1];
        Ti[i]=Ti[i+1];
        VM[i]=VM[i+1];
        Ref[i]=Ref[i+1];
    }

    T[99]=temperatura;
    Ti[99]=Ti[98]+periodo;
    VM[99]=variableM;
    Ref[99]=referencia;
```

Anexo Figura 19: Desplazamiento de valores en los vectores de la gráfica

4. Python

4.1. Main

4.1.1. Lectura y escritura de datos. Protocolo I2C

Para la realización de esta parte es necesario descargar de la web oficial de los componentes las librerías necesarias para realizar las operaciones de lectura y escritura. En el apartado 5.1 de la memoria aparece la descripción de las TAD así como la referencia a sus webs oficiales.

Los pasos son los siguientes:

```
11  #Codigo tarjeta
12  from smbus import SMBus
13  import re
14
15  def changechannel(bus,address, adcConfig):
16      tmp= bus.write_byte(address, adcConfig)
17
18  def getadcreading(bus,address, adcConfig):
19      adcreading = bus.read_i2c_block_data(address,adcConfig)
20      h = adcreading[0]
21      m = adcreading[1]
22      l = adcreading[2]
23      s = adcreading[3]
24      varDivisior = 64 # from pdf sheet on adc addresses and config
25      varMultiplier = (2.4705882/varDivisior)/1000
26      # wait for new data
27      while (s & 128):
28          adcreading = bus.read_i2c_block_data(address,adcConfig)
29          h = adcreading[0]
30          m = adcreading[1]
31          l = adcreading[2]
32          s = adcreading[3]
33
34      # shift bits to product result
35      t = ((h & 0b00000001) << 16) | (m << 8) | l
36      # check if positive or negative number and invert if needed
37      if (h > 128):
38          t = ~(0x020000 - t)
39      return t * varMultiplier
```

Anexo Figura 20: Librerías y funciones lectura/escritura TAD

```
149 bus=SMBus(1)
150 adc_address1 = 0x68
151 changechannel(bus,adc_address1, 0x9C)
```

Anexo Figura 21: Creación objeto bus y selección del canal

```
41 def getValue(bus):
42     return getadcreading(bus,adc_address1,0x9C)
43
44 def setValue(bus,data):#data [0...5]
45     data=int(data/5.0*4095.0)
46     value=[(data>>4) &0xFF, (data<<4)&0xFF ]
47     bus.write_i2c_block_data(0x62,0x40,value)
```

Anexo Figura 22: Funciones para lectura y escritura

Estas funciones definidas serán llamadas durante la ejecución del controlador para leer la temperatura y escribir la actuación, como veremos en el siguiente apartado.

4.1.2. Controlador

```
99 def PIDTask():
100     time_in = time.time()
101     globals.temp=SensorT() #obtenemos la temperatura de la TAD
102
103     if(globals.auto==0): #modo manual
104         globals.u0=globals.vm
105         globals.control=globals.vm
106         globals.t_ant=globals.temp
107
108     else:#modo auto
109         Tfiltrada=(1-globals.alpha)*globals.temp+globals.alpha*globals.t_ant #FILTRO
110         globals.err=globals.sp-Tfiltrada #Error real
111         ep=globals.sp*globals.b-Tfiltrada#Error proporcional ponderacion b
112         ed=globals.sp*globals.c-Tfiltrada#ponderacion c
113
114         up=globals.kc*ep
115         ui=globals.u_iant+(globals.kc*globals.ts*globals.err)/globals.ti
116         ud=globals.kc*globals.td*(ed-globals.e_ant)/globals.ts
117
118         globals.control=up+ui+ud+globals.u0
119
120         if(globals.control < globals.vmin): #antiwindup
121             globals.control=globals.vmin
122             ui=globals.u_iant
123         if(globals.control > globals.vmax): #antiwindup
124             globals.control=globals.vmax
125             ui=globals.u_iant
126
127         globals.u_iant=ui
128         globals.e_ant=ed
129         globals.t_ant=Tfiltrada
130
131     #Enviamos a la tarjeta el valor del control VM
132     SensorVM(bus,globals.control)
```

Anexo Figura 23: Implementación controlador PID

A continuación se va a comentar el código empleado:

- Línea 101: Se llama a la función SensorT() la cual se encarga de leer la temperatura en voltios y transformarla según la recta especificada.

```
52 def SensorT():
53
54     #TRANSFORMAR TEMP, leemos t_v de la tarjeta en V pasamos a C
55     t_v=getValue(bus)
56     print 'voltaje leido',t_v
57
58
59     v=(globals.v0,globals.v1,globals.v2,globals.v3)
60     t=(globals.t0,globals.t1,globals.t2,globals.t3)
61
62
63
64     if (globals.v0<=t_v and t_v<globals.v1):
65
66         a=(globals.t0-globals.t1)/(globals.v0-globals.v1)
67         b=globals.t0-a*globals.v0
68         t_c=a*t_v+b
```

Anexo Figura 24: Transformación Tª

- Modo Manual:
  - Línea 103: Se comprueba el modo de funcionamiento.
  - Línea 105: Asignación del valor *VM* a la acción de control.
  - Líneas 104 y 106: inicialización/actualización de variables. Siendo u0 el punto de funcionamiento, el cual se utiliza en el modo automático, ya que el control se realiza mediante incrementos. En la variable t_ant se almacena la temperatura, la cual es necesaria más adelante en el filtrado.
- Modo Automático (línea 108):
  - Filtrado
  - Cálculo del error real, proporcional y derivativo (110-112)
  - Acciones de control (114-118)
  - Antiwindup (120-125)
  - Almacenamiento de parámetros para la siguiente iteración (127-129)
  - Escribir el valor de la acción de control (132)

4.2. Webserver

En este apartado se va a mostrar un ejemplo de identificación de una acción en el servidor web. Es la siguiente:

```
123     if self.path[:12]=='/getMain.cgi':
124         print "Estoy en el webservcer GET MAIN"
125         #Enviamos cabeceras:
126         self.Headers_OK()
127         #Parseamos los parametros
128         params = self.ParseGETParams(self.path)
129         print params
130         # Calling Function
131         self.Ev.getMain(params,self.wfile)
132         return
```

Anexo Figura 25: getMain

Cuando se realiza una petición a través de la web, el servidor se encarga de identificarla (línea 123) parsear los parámetros y llamar a la función correspondiente en eventos para su procesado (línea 131).

Para completar este ejemplo, en el siguiente apartado referente a las funciones desarrolladas en el programa eventos se va a exponer el código que procesa la petición aquí expuesta.

4.3. Eventos

```
19    def getMain(self, params, writer):
20        print "Estoy en el GET MAIN"
21
22        writer.write(str(globals.vm)+'&'+str(globals.sp)+'&'+str(globals.auto)+'&'+str(globals.ts))
23
24
25        return
```

Anexo Figura 26: Eventos getMain

Cuando el servidor llama a esta función se devuelve una cadena a la web con los parámetros correspondientes separados por &.

Finalmente se va a mostrar el código empleado en una función en la que se actualiza el valor de algunos parámetros. Para mostrar así un ejemplo de cada tipo de acción.

```
89    def setConf(self, params):
90        print "Estoy en el SET"
91
92        if ("kc" in params and params.get("kc")[0]!='NaN'):
93            globals.kc=float(params.get("kc")[0])
94
95        if ("ti" in params and params.get("ti")[0]!='NaN'):
96            globals.ti=float(params.get("ti")[0])
97
98        if ("ts" in params and params.get("ts")[0]!='NaN'):
99            globals.ts=float(params.get("ts")[0])
```

Anexo Figura 27: Actualizar configuración

4.4. Globals

Este fichero contiene todas las variables que se manejan en los diferentes programas. Es donde se establecen los valores iniciales.

```
6  #fichero con las variables globales
7  kc=0.0
8  ti=0.0
9  ts=1
10 sp=30.0
```

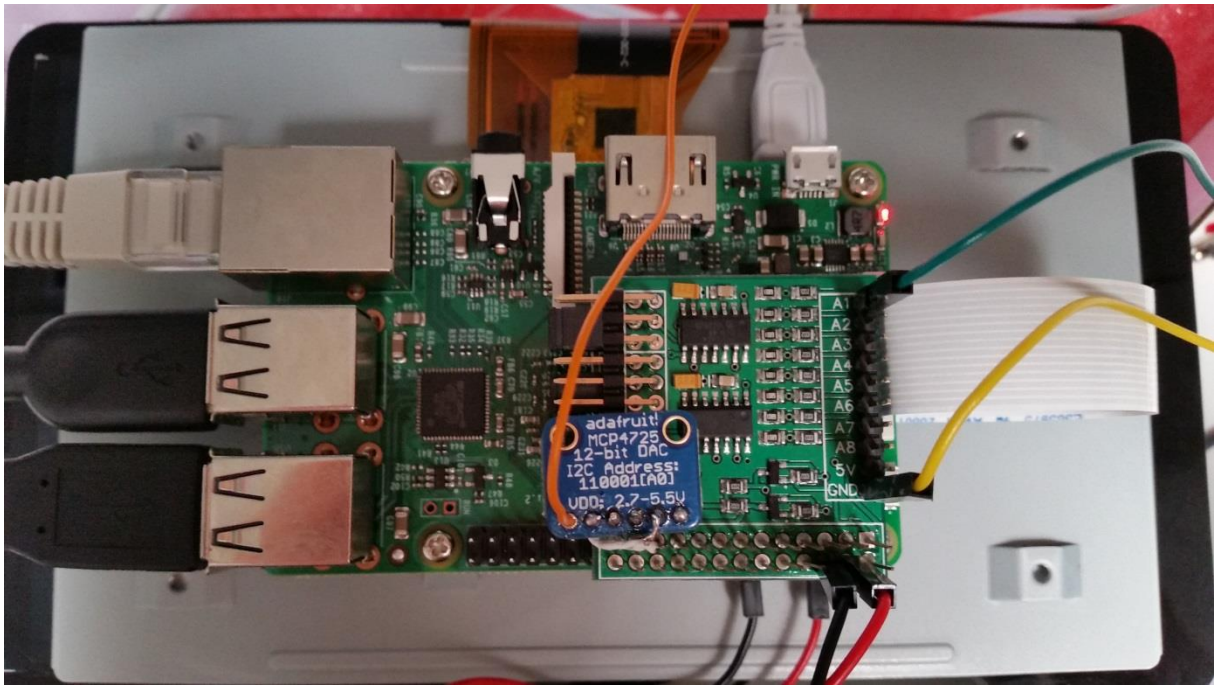
Anexo Figura 28: Variables globales

EJEMPLO PRÁCTICO: CONTROL DE LA TEMPERATURA DE UN HORNO

Se procede a presentar el desarrollo y resultados obtenidos de la aplicación del controlador PID implementado. El ensayo se ha realizado en los laboratorios de control del departamento de Ingeniería de Sistemas y Automática de la Universidad Politécnica de Valencia.

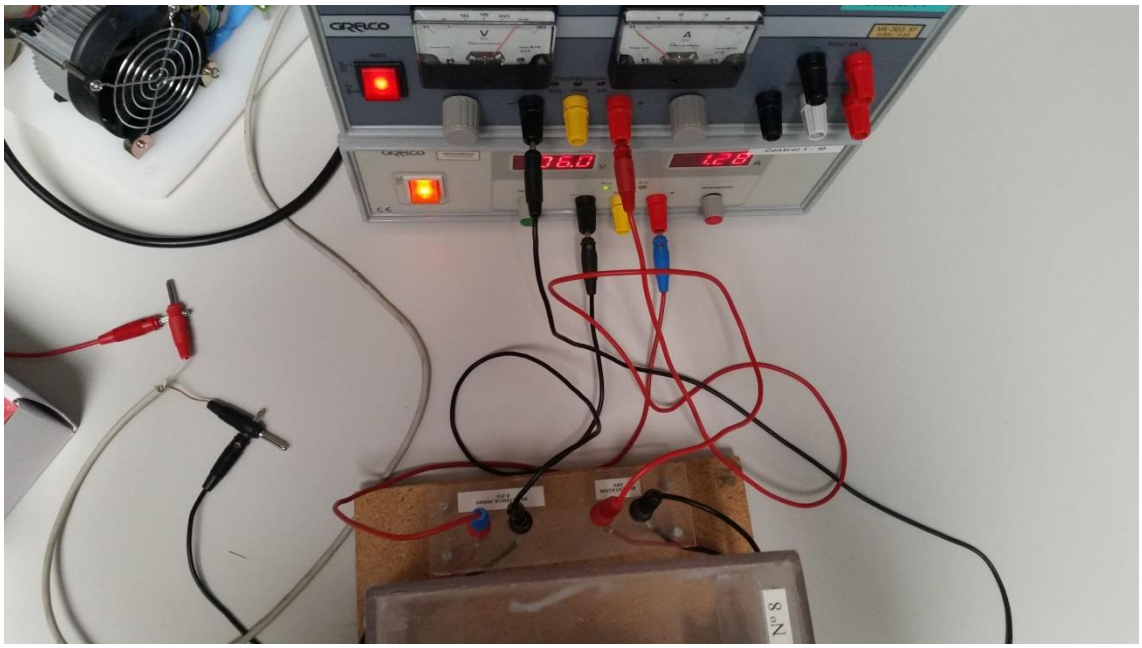
1. Montaje del conjunto

Recordando el resultado del montaje de la placa Raspberry Pi con los sensores (Memoria, apartado 5.1.4), la salida se corresponde con el cable naranja, la entrada con el verde y el amarillo con el neutro. Además, en la imagen se ve que la conexión de red se realiza mediante Ethernet. Como se disponía de una pantalla táctil, la alimentación se realiza con el micro USB blanco a la pantalla, y desde ella se alimenta la placa mediante los jumpers rojo y negro.



Anexo Figura 29: Montaje Raspberry Pi

La señal de salida (naranja) la cual se corresponde con el tanto por ciento de actuación de la variable manipulada, que en este caso es el voltaje. La Raspberry puede proporcionar de 0 a 5 voltios. Esta salida se pasa por un amplificador de ganancia 3, y se aplica al horno.



Anexo Figura 30: Conexiones del horno. Amplificador y fuente de alimentación.

La salida (naranja) está conectada a la entrada del amplificador, se corresponde con el cable rojo a la izquierda de la imagen. El neutro (amarillo) está conectado a la masa del amplificador y se corresponde con el cable negro también a la izquierda.

Además se utiliza una fuente de alimentación la cual se fija a 24V y se encarga de alimentar el sensor y el ventilador.

Anexo Figura 31: Entrada, sensor y ventilador

Finalmente tenemos el sensor de temperatura, el cual proporciona una salida entre 2 y 10V. Dicha señal se ha adaptado al rango aceptable para la Raspberry Pi mediante la incorporación de dos resistencias. Obteniéndose así una salida de 1 a 5V correspondiente a la entrada analógica (verde). Finalmente, es recomendable mantener el ventilador en la velocidad 3.

2. Ensayo

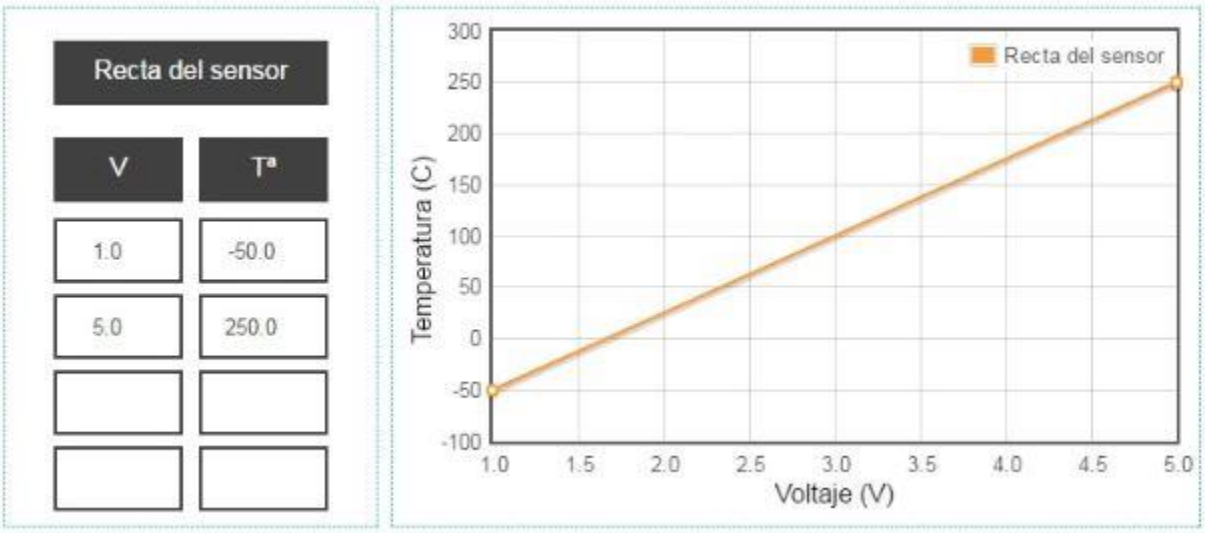
Utilizando un ordenador conectado a la misma red se procede a la realización del control de la temperatura del horno y su seguimiento a través de la interfaz web desarrollada. A continuación se van a mostrar imágenes de este proceso.

Anexo Figura 32: Main

Se accede poniendo la dirección IP de la Raspberry Pi y especificando el puerto 8000. Por defecto la aplicación arranca con los valores iniciales especificados y ejecutándose el controlador con una frecuencia igual a un segundo.

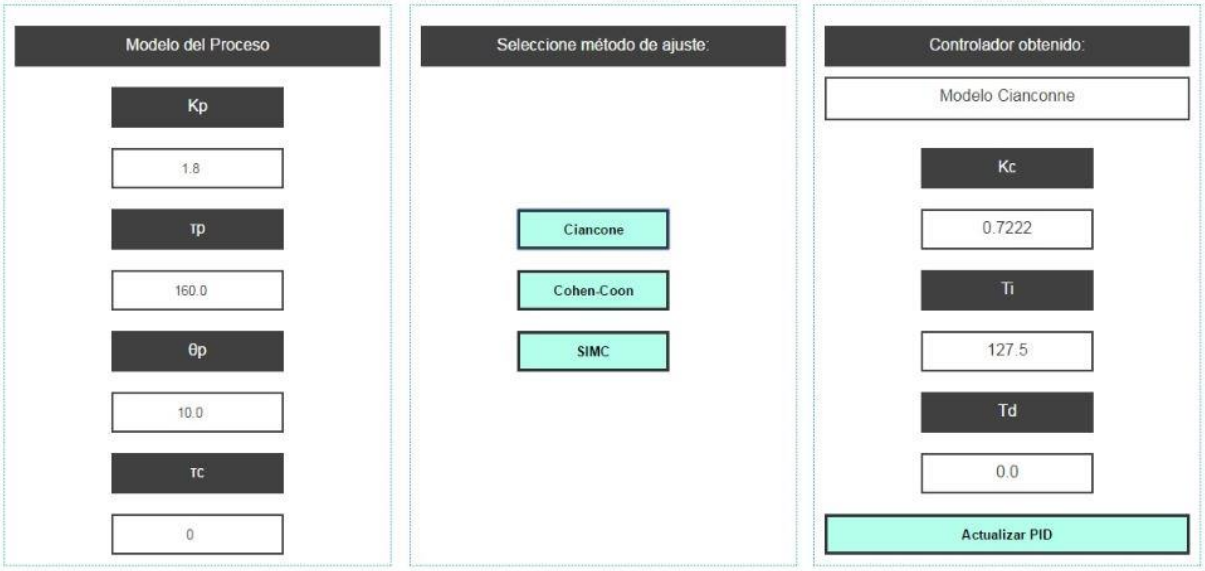
Anexo Figura 33: Configuración

Se ha fijado el periodo en 10 segundos y se han configurado los ejes de la gráfica principal así como las alarmas. También se ha configurado la recta del sensor.



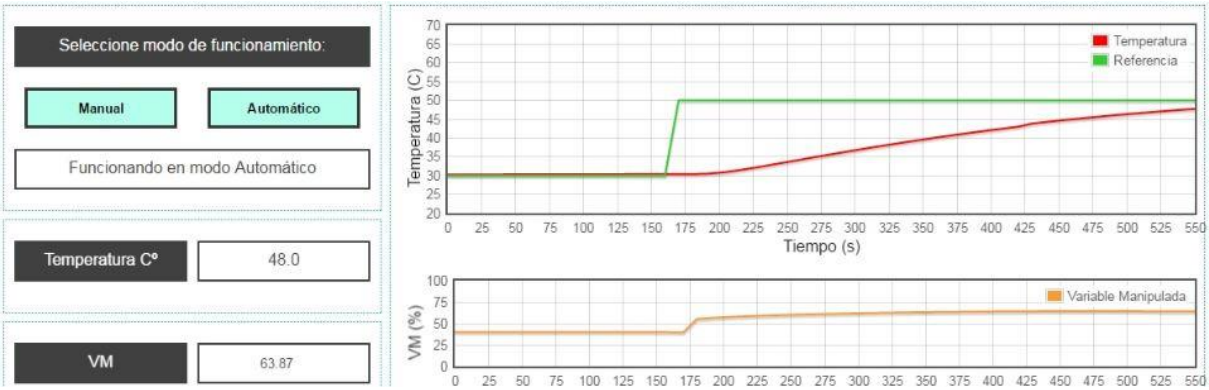
Anexo Figura 34: Detalle recta del sensor

Los parámetros del controlador se han escogido utilizando la pestaña auxiliar Proceso, en la cual se ha introducido el modelo del proceso y se ha seleccionado el método de Ciancone.

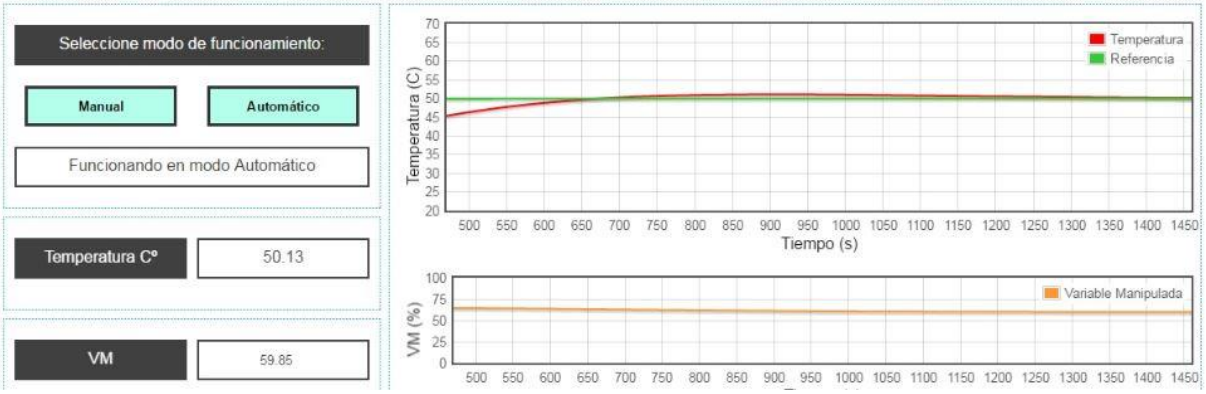


Anexo Figura 35: PID Ciancone

Finalmente, en la pantalla principal se ha conmutado a modo automático y se ha fijado la referencia a 50 grados, y el resultado es el siguiente:



Anexo Figura 36: Control temperatura. Cambio en la referencia



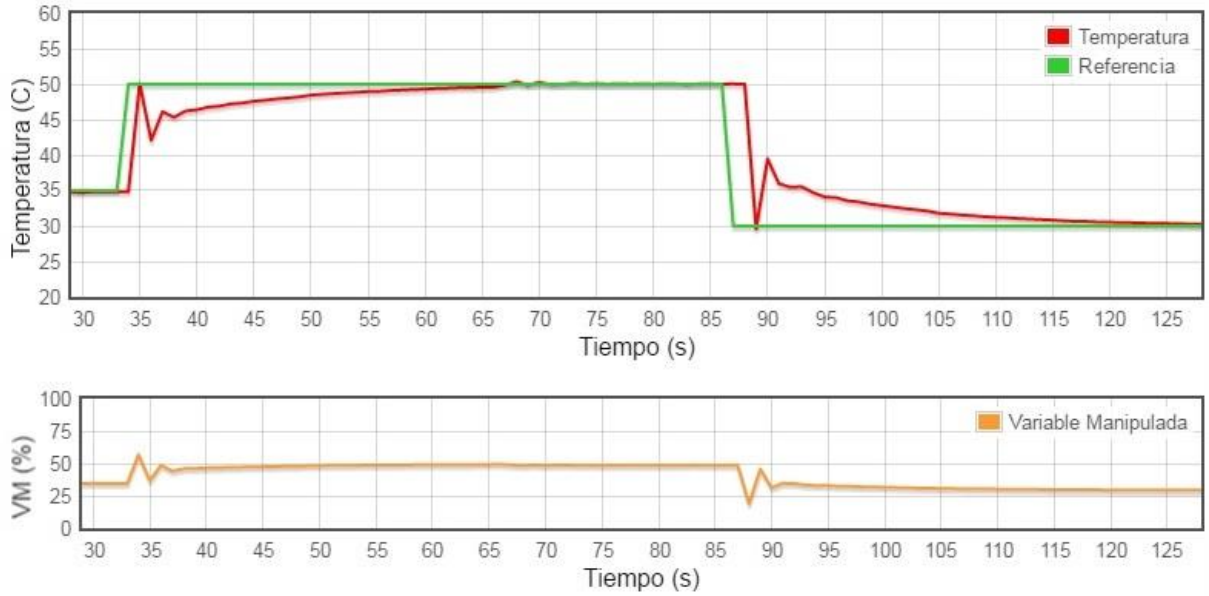
Anexo Figura 37: Control en régimen permanente

El tiempo de establecimiento de este proceso es de unos 10 minutos y como se puede observar, transcurrido ese tiempo desde el cambio de referencia la temperatura ya se ha estabilizado.

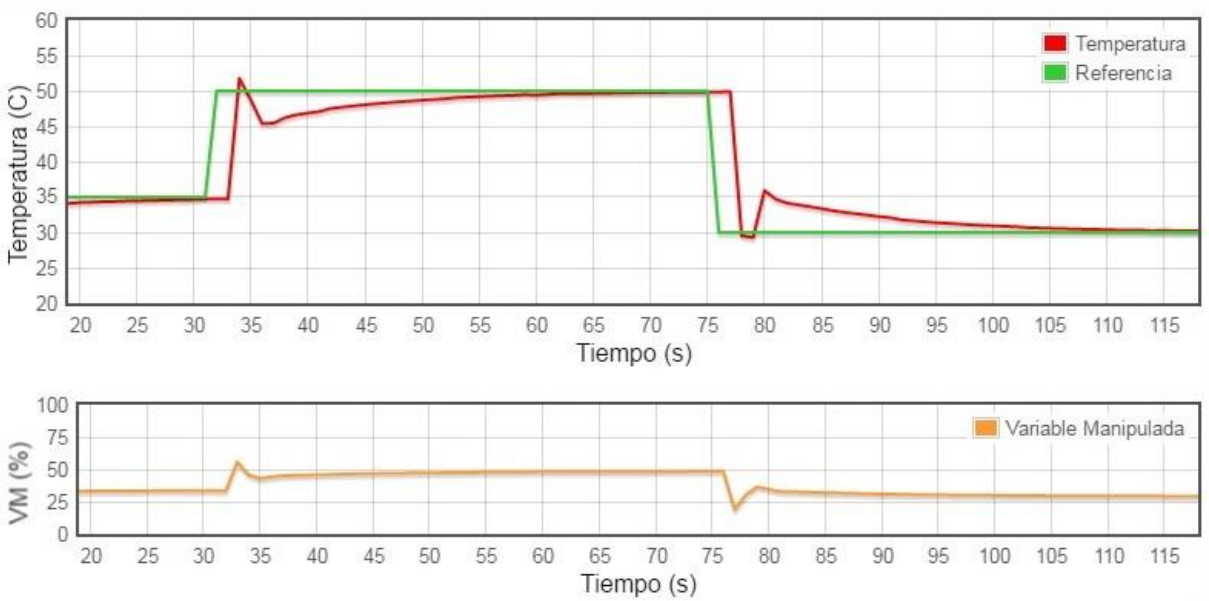
3. Otros ensayos

En este apartado se va a mostrar el resultado de experimentos realizados con una red RC, simulando un proceso térmico. Hay que tener en cuenta que la respuesta en este sistema es muy rápida.

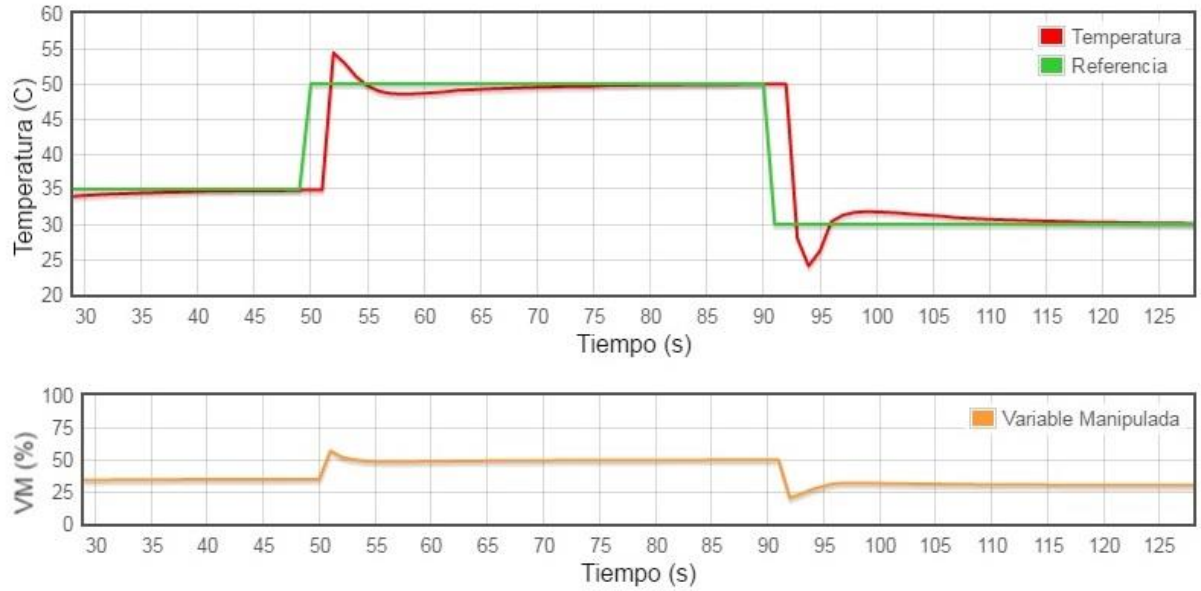
Se incluyen estos ensayos con el fin de mostrar los resultados de aplicación de las distintas opciones disponibles para el controlador implementado, como es el filtro o la ponderación de la referencia.



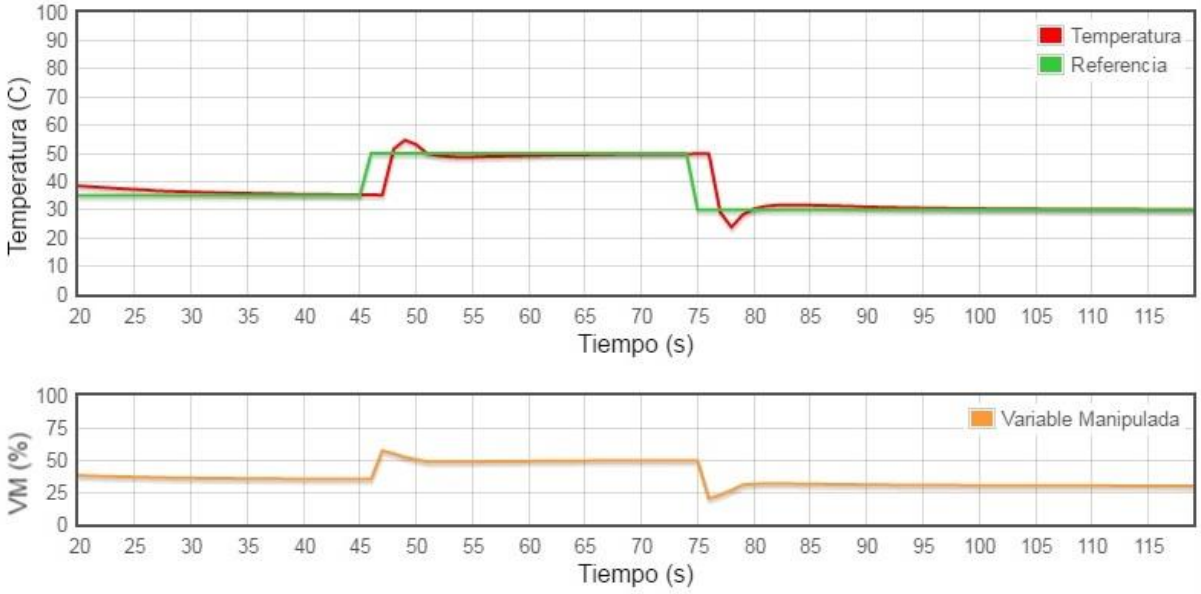
Anexo Figura 38: Escalón sin filtro



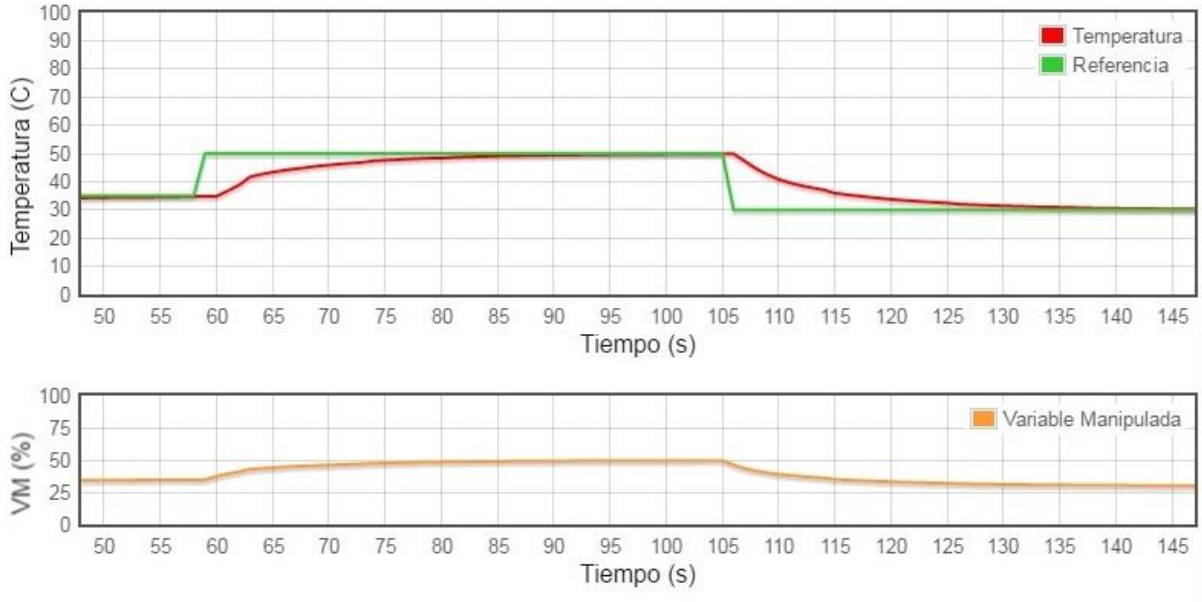
Anexo Figura 39: Filtro 0.5



Anexo Figura 40: Filtro 0.8



Anexo Figura 41: Ponderación PI-D



Anexo Figura 42: Ponderación I-PD