Master de Ingeniería de Sistemas y de Control.

Laboratorios Virtuales y Remotos en Automática para un Curso Online en Abierto

Jacobo Sáenz Valiente

Director & Codirector:

Luis de la Torre Cubillo y Jesús Chacón Sombría

Junio, 2014

Master de Ingeniería de Sistemas y de Control.

Laboratorios Virtuales y Remotos en Automática para un Curso Online en Abierto

Jacobo Sáenz Valiente

Trabajo de Fin de Master

Director & Codirector:

Luis de la Torre Cubillo y Jesús Chacón Sombría

Autorización

Autorizamos a la Universidad Complutense y a la UNED a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la memoria de este Trabajo Fin de Máster, como el código, la documentación y/o el prototipo desarrollado.

Firmado:

Resumen del Proyecto.

La tecnología actual y el acceso generalizado a internet han permitido que tanto las enseñanzas teóricas como las prácticas puedan ayudarse de recursos compartidos, aplicaciones y todo tipo de datos en línea que conforman una base importante del proceso educativo. En el marco de las enseñanzas puramente experimentales el laboratorio ha sido siempre una parte fundamentalmente presencial, y por tanto el acceso a ellos ha tenido limitaciones. Para minimizar estas limitaciones la utilización de ordenadores interconectados ha dado la posibilidad de crear laboratorios simulados y laboratorios remotos.

Este trabajo de fin de master reúne una descripción cualitativa de la arquitectura y estructuras los laboratorios virtuales y remotos que han sido incluidos en un curso abierto y que están alojados en UNILabs:

1. El servo-motor de corriente continua.
2. El sistema heatflow.
3. Los motores eléctricos acoplados.

Palabras Clave.

Trabajo de fin de master. Laboratorios virtuales. Laboratorios remotos. Easy java simulations EJS. Labview.. UNILabs. Moodle. Sistema de Acceso a Recursos de LABoratorio SARLAB. Servo motor de corriente continua. Heatflow. Motores eléctricos acoplados.

Índice.

Índice de Figuras.

.

1. Introducción.

Los laboratorios remotos y virtuales son actualmente herramientas muy útiles en el proceso de aprendizaje de muchas personas. Numerosos usuarios hacen uso de estos medios para obtener acceso a equipos y recursos que de otra forma no podrían tener a su alcance. Estos laboratorios pueden utilizarse a modo de complemento educativo, pues permiten a los usuarios ampliar su experiencia práctica, o a modo de aprendizaje práctico a distancia. En ambos casos se pueden simular estos sistemas o se puede sustituir el acceso físico a los sistemas reales por un control sobre una planta remota, a través de aplicaciones interactivas.

Easy Java Simulations (EJS) [1] es una herramienta de autor en Java para crear estas simulaciones interactivas, ha sido creada por Francisco Esquembre y es parte del proyecto Open SourcePhysics [2]. Esta herramienta permite crear simulaciones de manera sencilla, pudiendo utilizar elementos de visualización con diferentes parámetros, en lugar de tener que definirlos previamente mediante código. De la misma forma el entorno de trabajo de EJS permite añadir elementos con funcionalidades y propiedades adicionales, los cuales forman una parte importante de los laboratorios que se presentan en este trabajo.

Un ejemplo aplicado de arquitectura para gestionar y trabajar con este tipo de aplicaciones de laboratorio es UNILabs, un portal web que consiste en una red distribuida de laboratorios virtuales y remotos de ingeniería y ciencias, los cuales son parte del programa de estudios superiores de varias universidades. Estos laboratorios web utilizan Moodle, un sistema de gestión de aprendizaje de código abierto y gratuito que provee a estudiantes y profesores, entre otras cosas, de una plataforma en la cual comunicarse, compartir links a recursos, ejercicios y teoría.

1.1. Descripción del Trabajo.

El objetivo de este trabajo ha sido la creación de un curso en abierto con acceso a laboratorios virtuales y remotos escritos en EJS y alojados en UNILabs, de forma que un usuario con una conexión a internet y un explorador con Java pueda realizar algunas de las experiencias prácticas correspondientes a un curso de automática.

1.2. Visión General.

En esta sección se realiza un análisis cualitativo de los componentes principales de los laboratorios del curso abierto, la infraestructura de la red de laboratorios de UNILabs, y la estructura de las aplicaciones Java que se han incluido en la sección de "Curso abierto con Laboratorios Accesibles".

1.3. Laboratorios Virtuales y Remotos

Una aplicación de laboratorio virtual consiste en la simulación del entorno de trabajo y de los resultados esperados bajo la acciones de los usuarios en un laboratorio real. Esto permite realizar múltiples experimentos cuyo objetivo sea el aprendizaje de conceptos teóricos a partir de una experiencia práctica o la preparación del usuario de cara a una

experimentación sobre una planta real. La naturaleza simulada de este tipo de experimentos permite también a los usuarios observar la respuesta esperada del sistema a acciones fuera de las restricciones comunes en un sistema real, por lo cual el usuario puede aprender patrones deseados a la hora de utilizar un laboratorio real concreto.

Los laboratorios remotos se basan en la observación de una planta real mediante cámaras, de varios posibles instrumentos y del propio sistema sobre el cual se llevan a cabo los experimentos. En este tipo de laboratorios, las acciones de los usuarios sobre la aplicación son transmitidas al sistema y la respuesta es recogida y enviada al usuario, como si el propio usuario manipulara los controles y observara los instrumentos de medida en el propio laboratorio. Estas interacciones están mediadas por una red de comunicaciones, que provee a los usuarios de un acceso remoto a los equipos que controlan los laboratorios reales.

1.3.1. Estructura General de los Laboratorios Virtuales y Remotos.

Como se ha comentado anteriormente los laboratorios, tanto remotos como virtuales, se pueden alojar en un portal Web, como en nuestro caso es UNILabs. El acceso a las aplicaciones desde este portal se realiza mediante dos tipos de arquitectura, correspondientes a los laboratorios virtuales y los remotos.

Los laboratorios virtuales son simulaciones de los sistemas reales y por tanto al no necesitan un intercambio de datos con un sistema real, pueden ejecutarse como aplicaciones web, en nuestro caso escritas en Easy Java. En la Figura 1 se muestra la arquitectura usada para el acceso a las simulaciones [3].

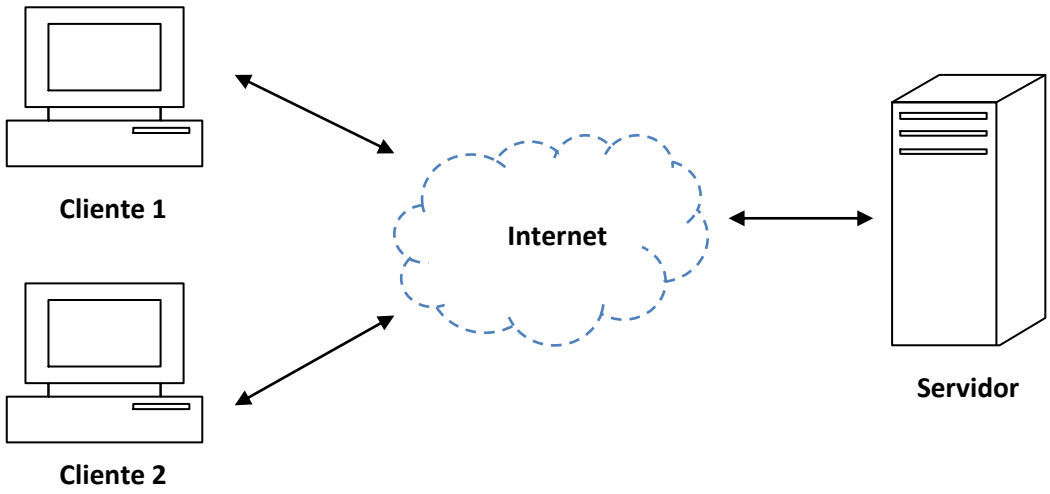

Figura 1. Arquitectura de acceso a laboratorios virtuales.

En el caso de los laboratorios remotos la arquitectura se vuelve algo más complicada aunque desde el punto de vista del usuario sigue constando, simplemente, de una aplicación en Java. En este tipo de laboratorio, la aplicación actúa a modo de interfaz, mientras que en el lado del servidor podemos diferenciar dos partes: el montaje experimental: la planta junto con los controladores, y el software encargado del control e interacción con el montaje experimental.

El montaje experimental consiste en el propio sistema y en los medios físicos necesarios para poder cumplir con los requerimientos de control. Dentro de estos medios físicos entran sensores, iluminación y actuadores de todo tipo. La segunda parte, el software, consiste en aquellos programas elegidos para soportar esta tarea, como podría ser Labview [4] o Matlab [5].

Al igual que sucede en el caso de los laboratorios presenciales, la operación de los laboratorios remotos debe realizarse individualmente, de forma que varios usuarios no modifiquen parámetros del experimento al mismo tiempo. Para realizar esta tarea se utiliza un sistema de registro el cual gestiona la entrada de los usuarios, previene el uso exclusivo de un solo usuario, facilita a las personas a cargo del experimento el control de acceso y en el caso de haber varios laboratorios remotos gestiona el tráfico entre los usuarios y las plantas que se están utilizando. El sistema de registro utilizado recibe el nombre SARLAB (Sistema de Acceso a Recursos de LABoratorio) [6], un sistema desarrollado en la Universidad de Huelva. SARLAB, entre otras cosas, mantiene a todos los ordenadores con el hardware/software necesario para controlar los laboratorios detrás de una red privada que gestiona las conexiones de los usuarios externos a estos ordenadores, además de proveer de un sistema de encendido y apagado de la alimentación del equipo que usen los laboratorios.

En la figura 2 se muestra una representación gráfica de esta arquitectura.

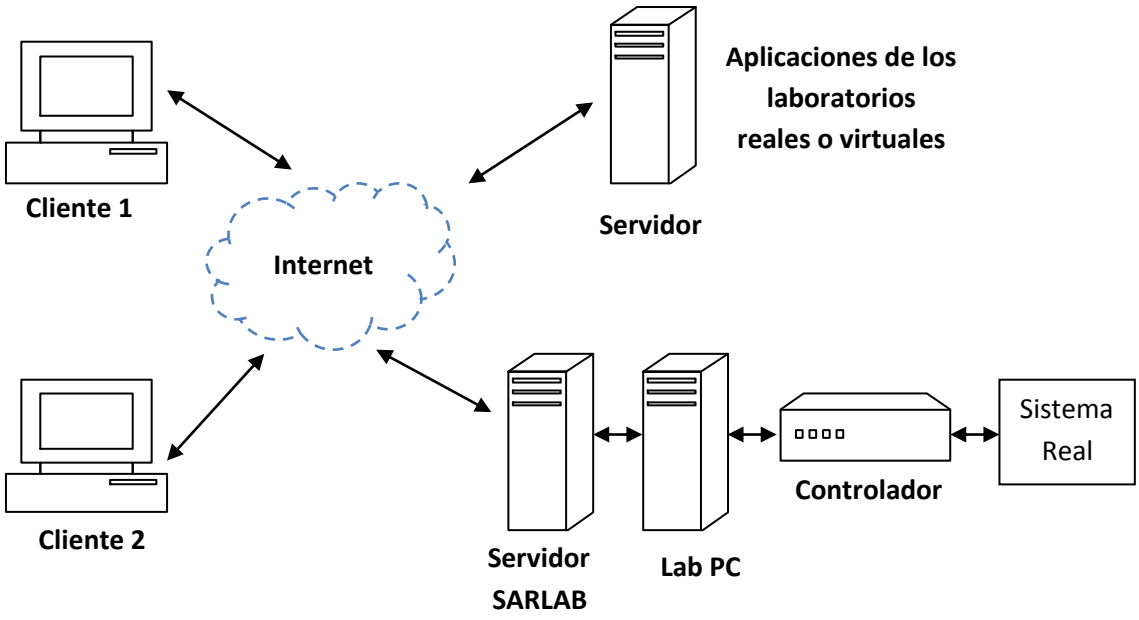

Figura 2. Arquitectura de acceso a laboratorios remotos y/o virtuales.

1.3.2. Comunicación Labview-EJS: JIL (Java Internet Labview)

Para simplificar la comunicación entre las aplicaciones basadas en EJS y los instrumentos virtuales de Labview en los equipos remotos se utiliza una capa software intermedia, la cual está basada en un servidor Java - Internet - Labview. Esta capa consta

de una aplicación llamada JIL-server [7] y una librería Java [8]. JIL-server se ejecuta en el lado del laboratorio remoto donde está Labview instalado mientras que en la aplicación en EJS se añade un elemento para vincular las variables del laboratorio con las variables de la aplicación EJS.

1.4. Estado del Arte.

En el campo de la experimentación sobre laboratorios remotos y virtuales podemos encontrar otras plataformas así como otras soluciones a nivel tecnológico.

1.4.1. PhET (Physics Education Technology Project).

Una de las otras plataformas destinadas a simulaciones es PhET [9], un proyecto de la universidad de Colorado que consiste en una colección de simulaciones de fenómenos relacionados con fenómenos físicos de las ciencias. Estas aplicaciones están escritas en Java o en Flash, por tanto pueden ejecutarse en un navegador web de uso común que tenga estos instalados. Esas aplicaciones permiten al usuario interaccionar mediantes deslizadores, botones, haciendo clic y arrastrando elementos gráficos de la propia simulación, y provee también de instrumentos de medida simulados, como reglas o voltímetros que permiten al usuario realizar medidas durante la simulación.

El portal web de PhET está orientado principalmente a profesores y alumnos, como material complementario de enseñanza en el aula, sin que esto limite los posibles usos por parte de otros usuarios con interés en campos de las ciencias como Física, Química, Matemáticas, Biología o ciencias de la Tierra. Se orienta también a usuarios con mayores conocimientos en campos específicos dentro de las simulaciones de investigaciones avanzadas, aquí encontramos seis simulaciones correspondientes a temas más especializados como por ejemplo un Modelo de Partículas Auto-Impulsadas.

1.4.2. RCL (Remotely Controlled Laboratories).

Una de las plataformas alternativas destinadas a laboratorios interactivos remotos es RCL [10], un proyecto de la Universidad Tecnológica Kaiserslauternque se encarga de la instalación de experimentos reales junto con sistemas de observación como webcams y con la posibilidad de interactuar e intercambiar datos con el usuario.

Estos laboratorios están orientados principalmente a estudiantes universitarios de ciencias, pudiendo encontrar entre sus sistemas remotos experimentos de Física, Electrónica y Robótica a los que se puede acceder siempre y cuando no estén en uso. Estos experimentos permiten una visualización de laboratorio a través de una webcam y una serie de botones dentro de la propia página web que permiten interaccionar con partes del sistema.

1.4.3. VISIR (Virtual Instrument System In Reality)

VISIR [11] es un proyecto de la Blekinge Institute of Technology, de Suecia, basado en la educación a distancia y la experimentación remota en electrónica. El laboratorio de electrónica de VISIR permite la emulación de un laboratorio y su instrumental en el ordenador del usuario, de forma local. Un estudiante matriculado puede cablear y configurar su circuito, y una vez completado puede enviar estos datos a los equipos del laboratorio real, el cual devuelve los resultados al usuario.

Los laboratorios constan de un equipamiento como PICs, FPGA y microprocesadores como parte del aprendizaje de los alumnos y que además son capaces de implementar y probar los circuitos que hayan sido configurados por el usuario.

El proyecto VISIR colabora con universidades como la UNED y con sociedades como la IEEE Education Society para crear una red de laboratorios de todas las universidades colaboradoras.

1.4.4. iSES(Internet School Experimental System).

El sistema iSES [12] es una de las posibles soluciones alternativas a novel tecnológico, pues consiste en una herramienta de adquisición en tiempo real, tanto local como remota, procesamiento de datos y control de experimentos y procesos.

Los componentes principales del sistema se pueden dividir en dos grupos, hardware y software. El hardware de iSE consiste en una serie de sensores, una placa de control y una tarjeta de adquisición de datos para el ordenador. Los datos adquiridos por los sensores pueden ser procesados matemáticamente y ser representados en paneles o gráficas configuradas por los usuarios. Este sistema también funcionará a modo de servidor para los casos en que quiera utilizarse en modo remoto.

Este sistema se ha utilizado tanto para estudiantes de secundaria como para estudiantes universitarios de Física, además de contener una práctica de automática de control del nivel de agua

1.4.5. GRAASP.

GRAASP [13] es una plataforma web que ofrece a sus usuarios la posibilidad de crear y gestionar espacios compartidos, en los cuales, entre otras cosas pueden añadir recursos, actividades y aplicaciones. Esta plataforma ofrece la posibilidad de crear un espacio de aprendizaje (Inquiry Learning Space: ILS), el cual puede contener laboratorios, bibliografía, aplicaciones y otros recursos destinados a los alumnos.

Una de las utilidades principales de GRAASP es la posibilidad de utilizar los recursos existentes en la propia plataforma, de forma que podemos buscar laboratorios, entre los cuales encontraremos VISIR, PhET, RCL y muchos más.

2. Estructura General de las Aplicaciones EJS

2.1. Configuración estándar de las aplicaciones

Las aplicaciones destinadas al curso en abierto, tanto para laboratorios remotos como para laboratorios virtuales se han diseñado siguiendo una estructura común en cuanto a visualización nos referimos. Esta estructura se muestra en la figura 3, donde podemos diferenciar al menos 3 zonas: Menú, ventana de representación y ventana de evolución. Pueden verse ejemplos concretos de implantación de esta estructura para cada uno de los laboratorios virtuales y remotos presentes en el curso en abierto en la sección 3.

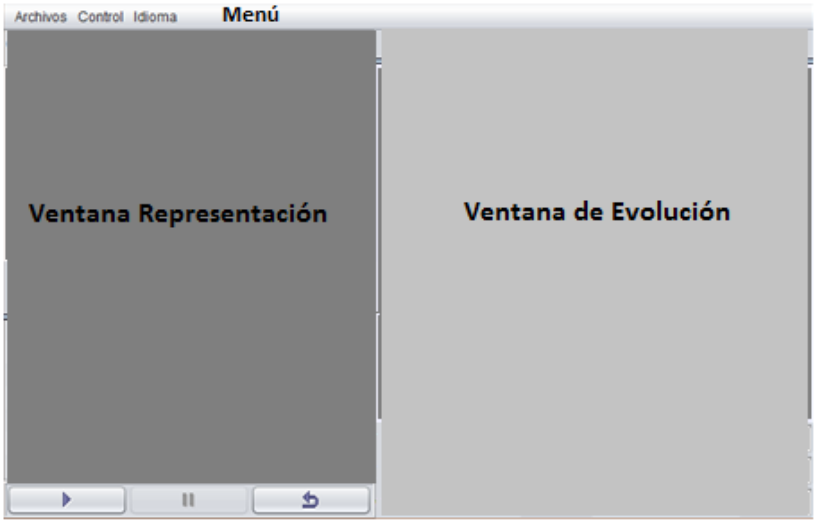

Figura 3. Ventana general de las aplicaciones virtuales y remotas.

2.1.1. Menú.

En la parte superior izquierda de la ventana hay una serie de opciones, cada una de las cuales abre un menú desplegable al pulsarlas.

- Files.

Guardar Gráficas(.gif). Permite guardar una imagen de la evolución de las variables controladas en formato .gif., es decir, guarda una imagen de las gráficas que se encuentran en la ventana de representación y del valor de los parámetros que se están registrando, cuya información se encuentra debajo de las gráficas en la ventana de evolución. Una vez guardada esta información se puede visualizar en el mismo o exportarla a un directorio.

Guardar Medidas(.m). Comienza a grabar en un fichero .m los parámetros de los controladores y los valores de la evolución temporal de las variables controladas y manipuladas. De esta forma, exportando a MATLAB esta información se pueden representar las gráficas que se deseen relativas a los experimentos que se están realizando.

- Control.

En el mismo menú donde está la opción de Archivos pero a la derecha de esta se encuentra la opción Control, cuyo menú desplegable permite realizar dos posibles acciones sobre el sistema. Dichas acciones son las siguientes:

MANUAL. Permite que el sistema trabaje en modo manual, es decir, el usuario debe ajustar los parámetros de acción sobre la planta, para obtener las consignas deseadas del sistema.

AUTOMÁTICO. Permite que el sistema pase a estar controlado por controladores PID o por el código del controlador, según este seleccionada o no la opción "Usar el Código" de la pestaña "Controlador" o no.

- Idioma.

En el mismo menú de archivos donde están las opciones de Archivos y Control (parte superior de la ventana de representación) pero a la derecha de esta última opción se encuentra la opción Idioma, cuyo menú desplegable permite cambiar el idioma de la simulación. Por defecto se encuentra en Inglés (seleccionando English) pero es posible pasarla a Español (seleccionando Spanish). Se recomienda pausar la simulación antes del cambio del idioma, para prevenir la perdida de información.

2.1.2. Ventana de Representación.

En la ventana de representación podemos encontrar una central donde se visualiza el sistema que se está estudiando, en el caso de un laboratorio virtual tendremos una imagen simulada, mientras que en el caso remoto tendremos una imagen real. En este último caso, puede encontrarse además, un selector: "augmented" o "aumentada" el cual nos ofrece la posibilidad de ver el sistema real junto con ayudas visuales para mejorar la experiencia durante el desarrollo de la práctica.

En la parte más baja de la zona de visualización y fuera del grupo de pestañas, se encuentran una serie de botones que permiten al usuario controlar las principales operaciones relativas a la evolución del sistema. En concreto puede realizar las acciones siguientes:

Play: Tan solo en simulación, sirve para iniciar la simulación.

Pause: Tan solo en simulación, sirve para establecer una pausa en la simulación.

Reset: Tan solo en simulación), sirve para resetear la simulación en curso e iniciar de nuevo la simulación desde 0.

Connect: Tan solo en modo remoto, permite la conexión con la planta en modo remoto.

En la parte intermedia, debajo de la representación gráfica del sistema se encuentran cinco pestañas para seleccionar entre los posibles funcionamientos de la planta (ver figura 1). Dichas pestañas son generalmente Controles, PID y Controlador.

- Pestaña CONTROL

Se proporciona un conjunto de sliders, o barras de desplazamiento, así como una serie de botones que permiten definir diversas situaciones en la dinámica del proceso, por ejemplo, realizar un cambio en la consigna o introducir los valores de las entradas del sistema de forma directa.

- Pestaña PID

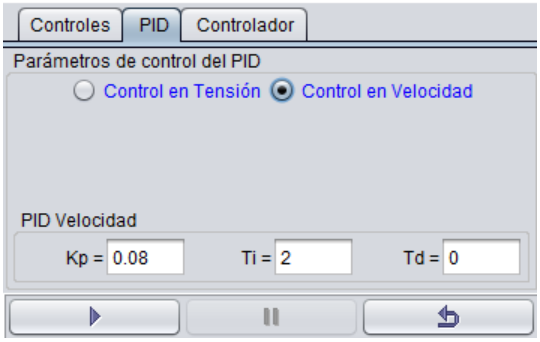

Figura 4. Pestaña PID de la Ventana de Representación

En la pestaña PID se proporcionan tres o seis campos numéricos que permiten variar los parámetros de los controladores asociados a cada tipo de control (ver figura 4). Se puede por tanto variar el valor de la ganancia proporcional (Kp), el tiempo integral (Ti) y el tiempo derivativo (Td), de cada uno de los dos posibles controladores a colocar en el sistema, los cuales serán visibles al activar los diferentes modos de control del sistema. Inicialmente tendrán asignados unos valores que permiten operar con el sistema de una forma razonable.

- Pestaña Controlador

En la pestaña Controlador se abre un panel con un cuadro de texto, dos botones y dos selectores (véase figura 5). Este panel permite escribir, editar, cargar y guardar y utilizar un código en Java o JavaScript para usarlo a modo de bloque controlador del sistema.

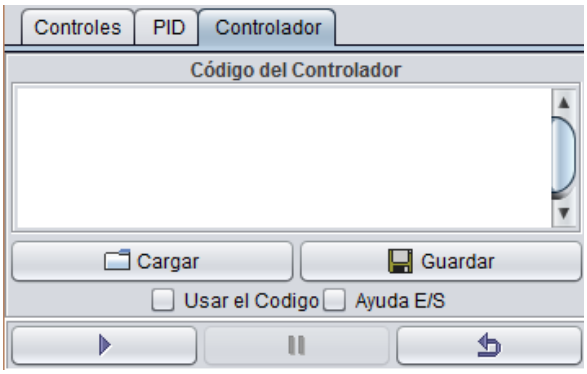

Figura 5. Pestaña Controlador de la Ventana de Representación

A continuación vamos a describir cada uno de estos elementos de arriba hacia abajo:

Código del Controlador. Muestra un cuadro de texto en el cual puede escribirse el código. Las variables que se utilizaran como entradas y salidas de este controlador se describen seleccionando Ayuda E/S. Las variables de entrada podrán leerse pero no modificarse, cualquier variable interna al código no afectará al resto de la simulación, tan solo se mantendrán los cambios que se realicen sobre el vector de salida. Este vector de salida tiene reservadas las posiciones u[0] y u[1] para las señales de control de los motores 1 y 2 respectivamente. El resto de elementos del vector, desde el u[2] hasta el u[9], podrán utilizarse a modo de variables auxiliares del usuario, como por ejemplo para acumular el error de la salida a lo largo de varios pasos de simulación.

Cargar. Muestra una ventana con un explorador de archivos que permitirá cargar algún código de controlador en .txt.

Guardar. Muestra una ventana con un explorador de archivos que permitirá guardar el código contenido en el cuadro "Código del Controlador" en formato .txt.

Usar el Código. Una vez seleccionado utiliza el código escrito en el cuadro "Código del Controlador". Si hubiera algún error nos devolvería una ventana de error y se pausaría la simulación. En caso de editarse el código este selector deberá ser pulsado de nuevo, para poder actualizar el código en ejecución.

Ayuda E/S. Muestra una ventana de ayuda en la cual se especifican las variables de entrada y salida que son utilizables en el código, junto con los comentarios correspondientes a cada una de ellas.

2.1.3. Ventana de Evolución.

A la derecha de la ventana de representación aparece lo que se conoce como ventana de evolución del sistema en la que se muestra como su propio nombre indica la evolución de las principales variables del proceso (ver figura 6).

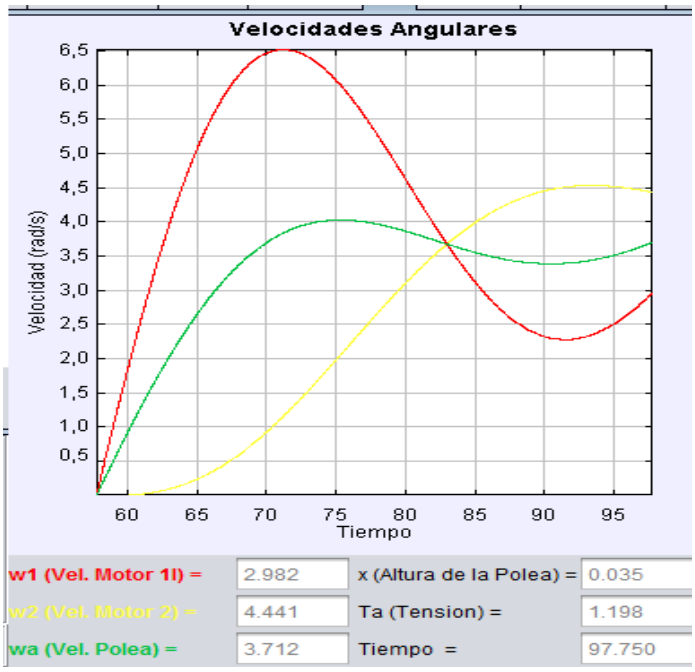

Figura 6. Ejemplo de una ventana de Evolución.

2.1.4. Gráficas y datos de la ventana de evolución

En la parte inferior de la ventana de evolución se muestran un conjunto de campos numéricos que indican el valor de las distintas variables de interés del proceso. En concreto están disponibles los siguientes campos, descritos de arriba hacia abajo y de izquierda a derecha:

En la parte superior de la ventana de evolución se observa un panel de pestañas cada una de las cuales contiene una gráfica con la evolución de las variables del proceso.

Una de las pestañas que se repite es la pestaña titulada Auxiliar, se muestran los datos y la gráfica donde se representan las variables auxiliares utilizables el código del controlador en modo automático. Esta pestaña tan solo se activará cuando se esté utilizando el código para obtener las señales de control.

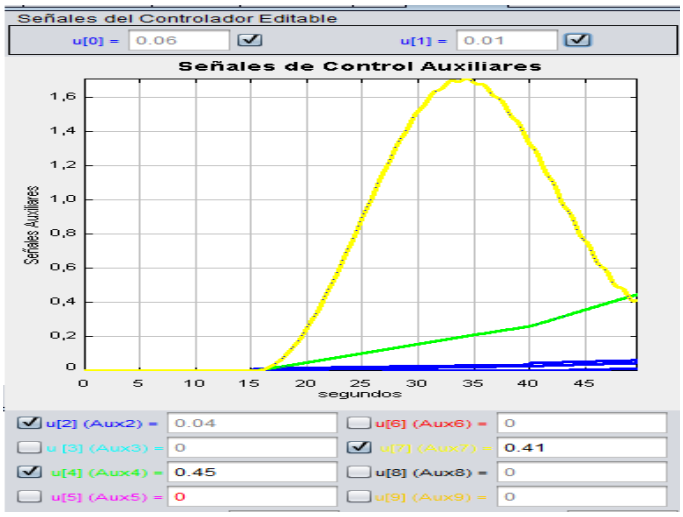

Figura 7. Pestaña Auxiliar de la ventana de evolución.

En la parte superior de esta pestaña se muestran los dos campos numéricos del vector de salida del controlador correspondientes a las señales de control de los motores, u[0] y u[1]. Junto a cada una de ellas podemos ver un selector que nos permite incluir o excluir de la representación gráfica esas señales.

En la parte media de la pestaña se encuentra la gráfica donde se pueden representar todos los elementos del vector de salidas.

En la parte inferior vemos un conjunto de campos numéricos del vector de salida del controlador correspondientes a las señales de auxiliares, u[2] hasta u[9]. Junto a cada una de ellas podemos ver un selector que nos permite incluir o excluir de la representación gráfica esas señales.

2.2. Elemento Evaluador de Código de Controlador.

La estructura general de las aplicaciones de laboratorios virtuales y remotos ha sido presentada en el apartado anterior y en esta sección se desarrolla con más detalle la funcionalidad de los elementos evaluadores de código, tanto en la propia aplicación, EJS, como en el software del ordenador encargado del control de la planta, Labview. Ambos elementos, juntos o por separado, ofrecen la posibilidad de utilizar código creado por el usuario y evaluarlo dentro del lazo de control del sistema virtual y remoto.

2.2.1. Evaluador en EJS.

Se ha hablado en la introducción de los elementos de EJS, los cuales añaden funciones y propiedades a las aplicaciones desarrolladas, para la evaluación de código en Java o JavaScript se ha utilizado el elementoCodeControllerEvaluator, el cual ha sido desarrollado por la doctora María Guinaldo Losada del departamento de Informática y Automática de la UNED.

En la figura 8 puede verse un ejemplo de la interfaz de enlace de variables de entrada y el vector de salida del bloque evaluador.

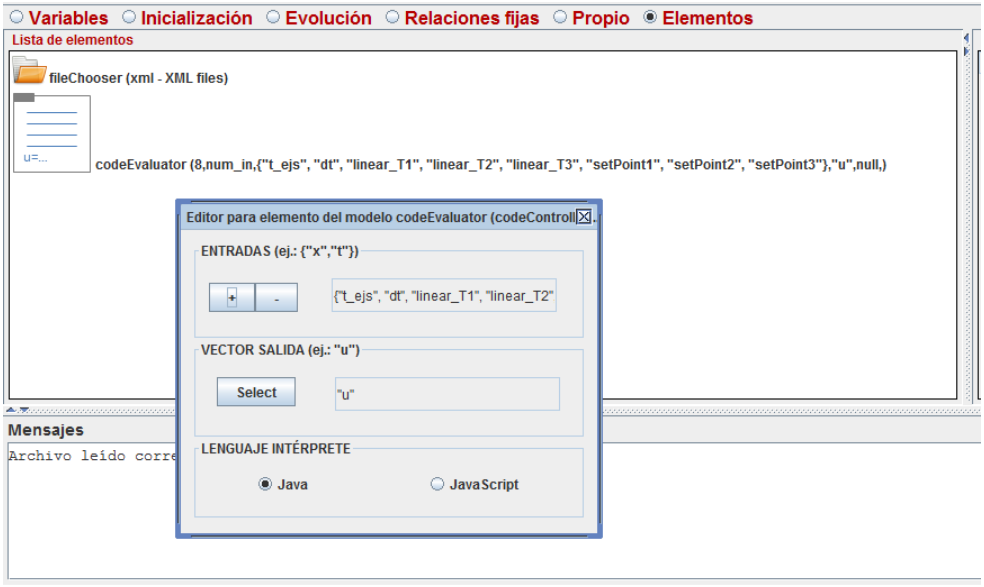

Figura 8. Ejemplo de visualización de la interfaz para enlazar variables del evaluador de código.

Este bloque permite utilizar los nombres de las variables de entrada y salida en el código introducido de forma que las salidas u[i] puedan ser obtenidas como la salida de una función del objeto codeControllerEvaluator:

codeControllerEvaluator.evaluateCode(code,codeInterpreter)

Este elemento ofrece además la posibilidad de mostrar una ayuda acerca de las variables definidas en EJS para que el usuario tenga un conocimiento más detallado de las variables de entrada que puede utilizar. En caso de error al usar las variables o de errores de sintaxis el elemento muestra una ventana de error para informar al usuario.

2.2.2. Evaluador en Labview.

El software Labview forma una parte importante en el desarrollo de los laboratorios del curso en abierto, pues como se ha comentado permite el control de las plantas a través de una programación mediante bloques. El bloque encargado de la evaluación de código ha sido desarrollado por el doctor Dictino Chaos García del departamento de Informática y Automática de la UNED.

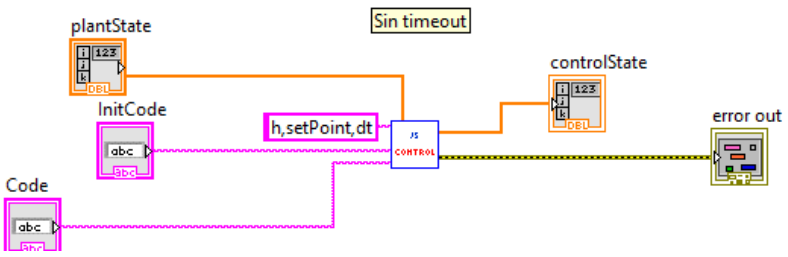

Figura 9. Ejemplo de uso del evaluador de código en Labview.

En la figura 9 puede verse un ejemplo de uso del bloque evaluador de código en Labview. *InitCode* y *Code* son variables de entrada que se envían desde EJS cuyo valor inicial es:

```
u[0] = 0;
u[1] = 0;
u[2] = 0;
```

El cual garantiza que no se introduzcan señales en la planta si no se ha enviado un código alternativo. Se introducen también en el bloque los nombres {h,setPoint,dt} y con ello el orden en el vector de variables de entrada *plantState*, estas variables son aquellas pueden utilizarse como entradas en *Code.* Por último el bloque devuelve el vector *controlState*, el cual contiene las variables de control u[i], o en caso de errores de sintaxis devuelve un mensaje en *error out.*

3. Laboratorios del Curso en Abierto.

Inicialmente se han incluido laboratorios virtuales y remotos en la web de UNILabs, en las sección de Curso Abierto con Laboratorios Accesibles, en los apartados siguientes se describen estos sistemas con mayor detalle.

La colección de laboratorios virtuales abarca tres sistemas: el servo-motor, el heatflow y los motores eléctricos acoplados. Cada uno de estos laboratorios virtuales tiene dos aplicaciones distintas, "Ajuste de un PID" o "Programación de un Controlador".

Los laboratorios remotos son correspondientes a dos de los sistemas anteriores, el servo-motor y el heatflow, ya que el laboratorio remoto de los motores eléctricos acoplados está aún en desarrollo. Cada uno de estas aplicaciones de laboratorios remotos tiene al igual que en el caso virtual dos modos distintos, "Ajuste de un PID" o "Programación de un Controlador".

Las aplicaciones de los laboratorios de los sistemas del servo-motor y del heatflow ya habían sido desarrolladas y se han modificado para: 1) añadir la opción de que el usuario pueda programar un controlador propio, 2) funcionen a través del sistema SARLAB, descrito en la sección 1, en el caso de los laboratorios remotos y 3) asegurar y/o mejorar su correcto funcionamiento con las últimas versiones de EJS. Usando estas aplicaciones se han puesto en funcionamiento en modo remoto 3 plantas correspondientes al servo-motor y una planta del heatflow.

Los tres laboratorios virtuales han sido compartidas y subidas a la colección de simulaciones de Open Source Physics (OSP) de la librería digital comPADRE [14], la cual es una red de colecciones de recursos destinados a profesores y alumnos de Física y Astronomía.

3.1. Servo Motor.

El sistema del servo motor, figura 10, consiste en un motor de corriente continua, con un tacómetro para medir la velocidad y un potenciómetro para medir la posición. Además utiliza un disco de acero como carga y cuenta con un freno magnético variable que aplica una fricción viscosa.

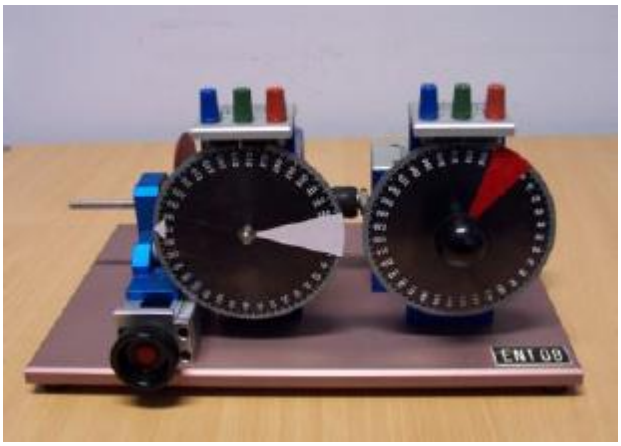

Figura 10. Sistema de Servo-Motor

La entrada a este sistema es el voltaje aplicado sobre el motor y las salidas son voltajes que son traducirlos a posiciones y velocidades en radianes y radianes por segundo. En ambos laboratorios, virtual y remoto, se ha utilizado el modelo desarrollado en la Escuela Politécnica Federal de Laussane, en Suiza.[15]

El modelo matemático para simular la dinámica de este sistema se ha obtenido del desarrollo matemático que se muestra en el *Anexo A1*.

El curso de UNILabs se configura en general de la siguiente manera:

1. Una primera aplicación virtual permite al usuario ajustar los parámetros del PID de forma que la salida del sistema, posición o velocidad, cumpla determinadas condiciones.
2. En una segunda aplicación, el usuario puede ajustar los parámetros de PID y también escribir el código de su propio controlador, de forma que pueda hacer comparaciones con el control PID. Como ejemplo, al acceder a este laboratorio virtual se ofrece a los alumnos un código para un controlador PID, con unos parámetros tal que salida no sea inestable. (Puede verse el código en el *Anexo B*)

3.1.1. Virtual.

El laboratorio virtual del servomotor simula la dinámica del sistema utilizando su modelo matemático. Este modelo permite variar las entradas de voltaje en modo manual y las referencias, tanto de velocidad como de posición, en modo automático.

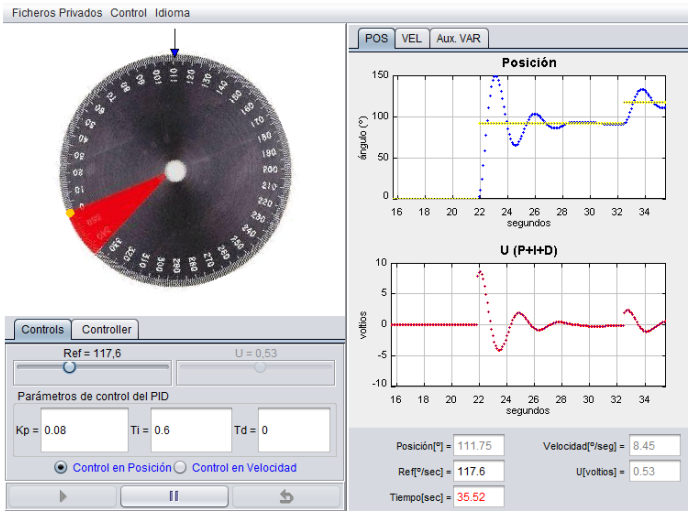

Figura 11. Aplicación del Servo Motor virtual.

Esta aplicación virtual permite a los usuarios visualizar al mismo tiempo la señal de control y la posición/velocidad según en que pestaña de la ventana de evolución nos encontremos. Adicionalmente permite perturbar al sistema a través del objeto desplazable amarillo que se ve en la ventana de representación de la figura 11. Al hacer clic sobre este objeto la simulación se pausa hasta que el usuario suelte el objeto, y por tanto el disco, en su nueva posición, tras lo cual la simulación seguirá ejecutándose.

3.1.2. Remoto.

La aplicación del laboratorio remoto del servomotor conecta al usuario con el ordenador que controla el sistema intercambiando entre ellos las variables que son necesarias para el control y la representación. Esta aplicación permite variar las entradas de voltaje en modo manual y las referencias, tanto de velocidad como de posición, en modo automático.

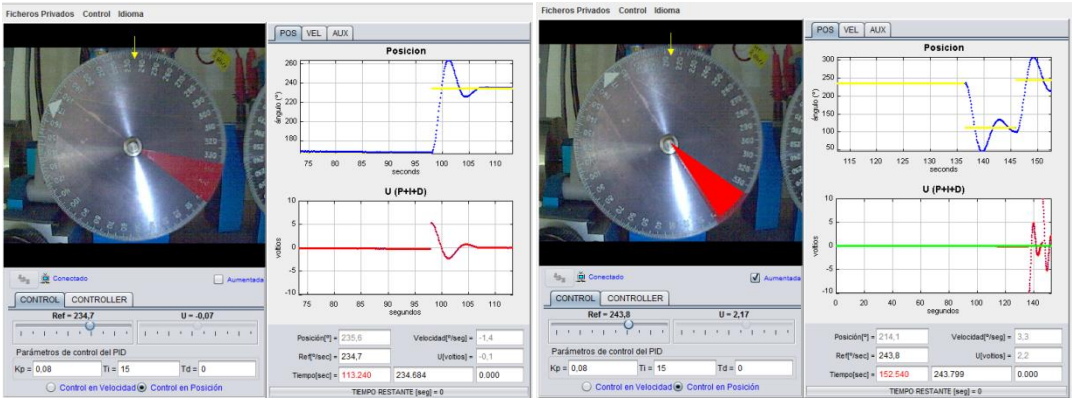

Figura 12.Izq. Servo-Motor funcionando en remoto. Dcha. Servo-Motor funcionando en remoto con realidad aumentada.

En ambos casos de control automático el cálculo de las señales de control se lleva a cabo en la propia planta, si se utiliza un controlador editado por el usuario, la aplicación intercambia el código que se ejecutará en el ordenador del laboratorio utilizando el bloque de evaluación de código de la sección 2.4.2.

En la figura 12 puede verse el sistema real en la ventana de representación visto a través de la webcam y la ventana de evolución, que es igual que en caso virtual. En la imagen izquierda, sin realidad aumentada, apreciamos que una flecha amarilla nos sirve de punto de medida para la posición angular del motor. Por otra parte, el sistema puede utilizar la realidad aumentada mostrando con mayor intensidad el triángulo rojo que indica el paso por cero, imagen de la derecha.

3.2. HeatFlow.

Este sistema heatflow, fabricado por la empresa Quanser Consulting de Canadá [16], utiliza un ventilador para establecer un flujo de aire a través una malla calefactora, el cual se desplaza a lo largo del recinto interno variando su temperatura por interacción las paredes del contenedor. Esta variación de temperatura es medida en distintas posiciones a lo largo del recinto interno a través de transductores de platino. En la figura 13 podemos ver una fotografía de la planta donde se indican los componentes fundamentales de la planta.

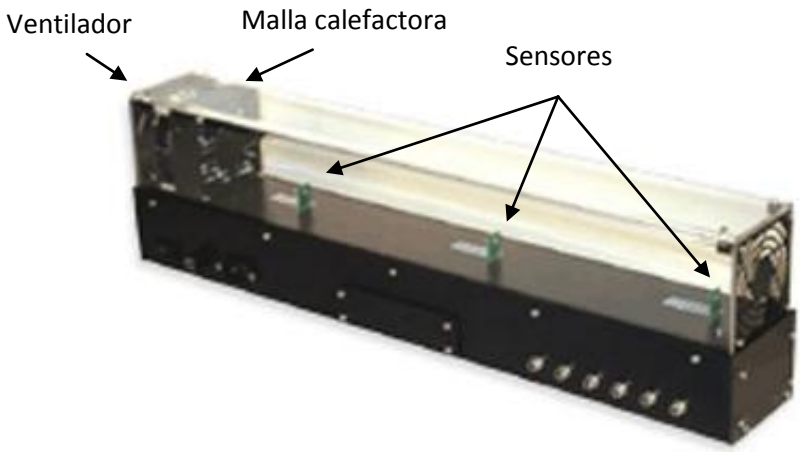

Figura 13. Sistema HeatFlow de Quanser.

Las entradas y las salidas del sistema son señales analógicas, las primeras controlan el calefactor y el ventilador, y las salidas son las señales devueltas por los transductores.

El modelo matemático y los parámetros para simular la dinámica de este sistema se han obtenido del desarrollo matemático que se muestra en el *Anexo A2*.

Al igual que en el caso del servo-motor este curso de UNILabs se configura de la siguiente manera:

1. Una primera aplicación virtual permite al usuario ajustar los parámetros del PID de forma que la salida del sistema, cualquiera de las tres temperaturas, cumpla determinadas condiciones.
2. En una segunda aplicación, el usuario puede ajustar los parámetros de PID y también escribir el código de su propio controlador, de forma que pueda hacer comparaciones con el control PID. Como ejemplo, al acceder a este laboratorio virtual se ofrece a los alumnos un código para un controlador PID,

con unos parámetros tal que salida no sea inestable. (Puede verse el código en el *Anexo C*)

3.2.1. Virtual.

El laboratorio virtual del heatflow simula la dinámica del sistema utilizando su modelo matemático. Este modelo permite variar las entradas de voltaje para el calentador y el ventilador en modo manual, representando las salidas correspondientes a los tres sensores que se encuentran a lo largo de la estructura del heatflow. Las referencias de temperatura se pueden cambiar en modo automático, donde la entrada de voltaje del ventilador se utiliza como perturbación.

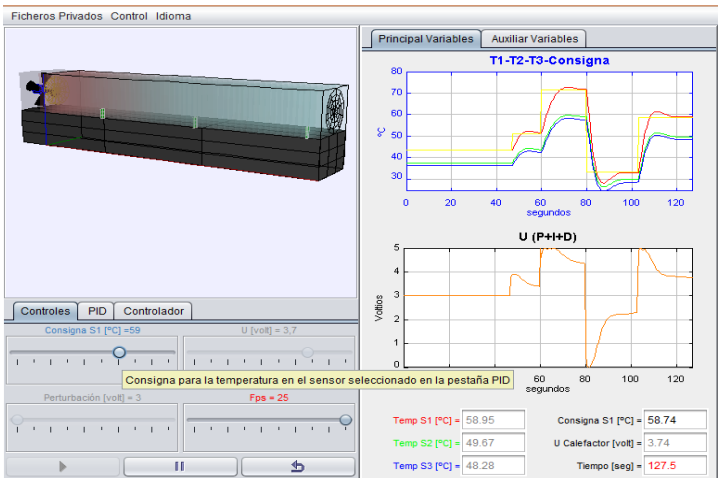

Figura 14. Aplicación del HeatFlow, virtual.

En la figura 14 tenemos un ejemplo de la aplicación virtual funcionando, en la ventana de representación podemos observar una figura en 3D que representa el heatflow, en su interior podemos apreciar un campo de colores que representa la temperatura en el recinto interno. En la ventana de evolución observamos la temperaturas T_1, T_2 y T_3, en los colores rojo, verde y azul, junto con la señal de control del sistema en la gráfica inferior.

3.2.2. Remoto.

La aplicación del laboratorio remoto del heatflow conecta al usuario con el ordenador que controla el sistema intercambiando entre ellos las variables que son necesarias para el control y la representación. Esta aplicación permite variar las entradas de voltaje en modo manual y las referencias, tanto de velocidad como de posición, en modo automático.

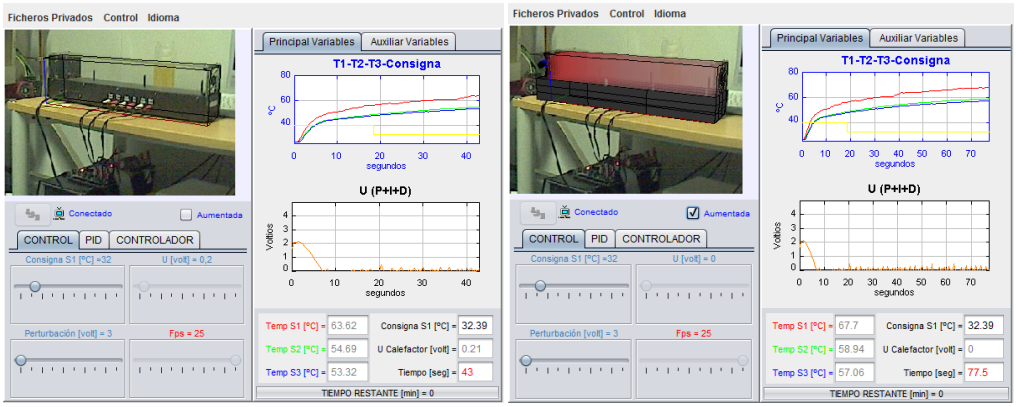

Figura 15. Izq. HeatFlow funcionando en remoto. Dcha. HeatFlow funcionando en remoto con realidad aumentada.

En ambos casos de control automático el cálculo de las señales de control se lleva a cabo en la propia planta, si se utiliza un controlador editado por el usuario, la aplicación intercambia el código que se ejecutará en el ordenador del laboratorio utilizando el bloque de evaluación de código que se comentó en 2.4.2.

En la figura 15 puede verse el sistema real en la ventana de representación visto a través de la webcam y la ventana de evolución, que es igual que en caso virtual. En la imagen izquierda, sin realidad aumentada, no es posible apreciar las variaciones de temperatura internas, pero puede obtenerse este tipo de información adicional del sistema utilizando la realidad aumentada que muestra una figura similar a la del caso del laboratoro virtual superpuesta con la planta real del heatflow.

3.3. Motores Acoplados.

Se ha utilizado el Sistema de Motores Acoplados CE8 de TQ Education and Training Ltd [17], el cual representa el problema clásico de los motores eléctricos acoplados, para obtener el modelo teórico y los parámetros básicos. Este sistema corresponde a esquema representado en la figura 16.

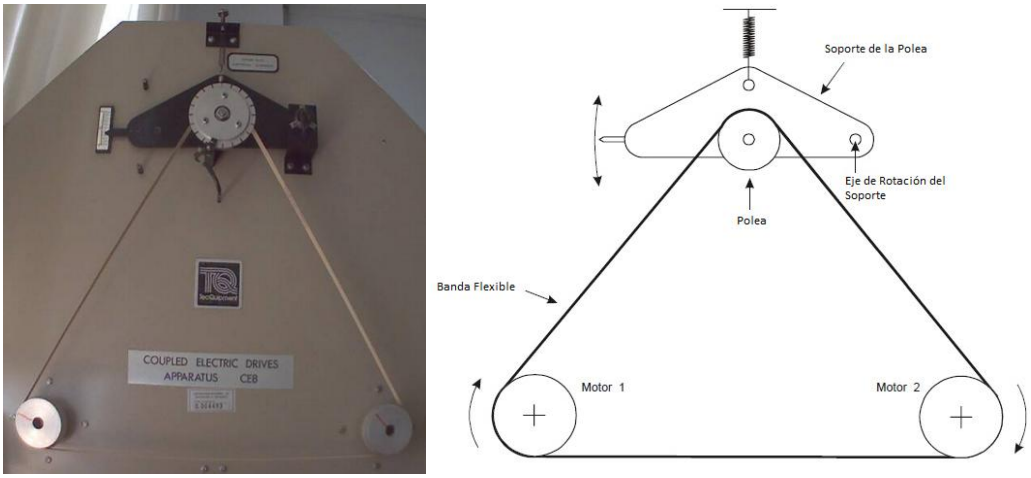

Figura 16. Sistema de los Motores Eléctricos Acoplados.

En la figura 16 se observan dos motores y un polea unidos entre sí por bandas flexibles. Estas bandas circulan alrededor de los motores y de la polea circular y por tanto están sometidas a una tensión que depende de la diferencia de velocidades entre los dos motores. Esta tensión se transmite en parte a la polea la cual está montada en un soporte el cual a su vez es soportado por un resorte. Con esta configuración puede obtenerse la tensión de forma directa a través de la posición o del ángulo del soporte de la polea.

En los sistemas de este tipo, las entradas del sistema de los motores eléctricos acoplados serán los voltajes de funcionamiento de los motores, mientras que las salidas serán la tensión y velocidad angular de la polea. Esta elección de entradas y salidas hace que se clasifique como un sistema de múltiples entradas y múltiples salidas (MIMO por sus siglas en inglés).

El modelo matemático y los parámetros para simular la dinámica de este sistema se han obtenido del desarrollo matemático que se muestra en el *Anexo A3*.

3.3.1. Virtual.

En el laboratorio virtual del sistema de los motores eléctricos acoplados se han introducido algunos elementos adicionales que no se han incluido en el apartado de "estructura general de las aplicaciones EJS", los cuales se explican a continuación. En la figura 17 puede verse la aplicación virtual en ejecución.

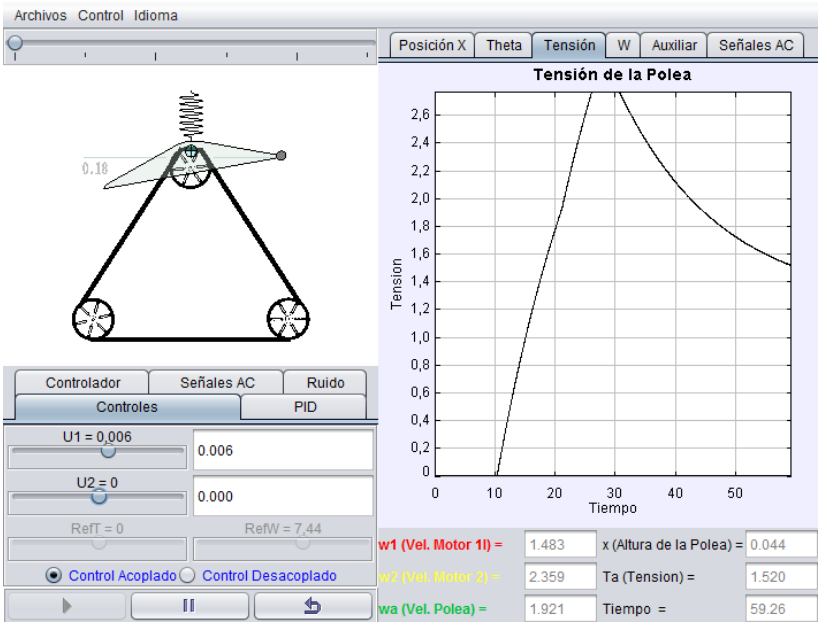

Figura 17. Motores Eléctricos Acoplados Virtual.

Puesto que en este laboratorio virtual encontramos muchas variaciones, se describirán una a una en los siguientes apartados.

3.3.1.1. Ventana de Representación

- **Pestaña Controles:** En la figura 17, en la pestaña seleccionada de la ventana de representación, pueden verse dos selectores: Control Acoplado y Control

Desacoplado. Estos permiten seleccionar si la simulación utilizará el sistema en modo múltiples entradas, múltiples salidas, o por el contrario se utilizará como dos sistemas de una sola entrada y una sola salida, en cuyo caso la pestaña PID presentará 6 campos para poder variar los parámetros de los dos controladores.

- **Pestaña Señales AC:** Esta pestaña ha sido incluida para poder introducir señales de amplitud y frecuencia variable en los motores, lo cual permite hacer un estudio de la respuesta en frecuencia del sistema.

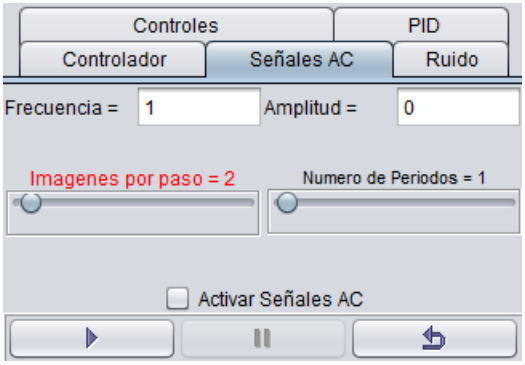

Figura 18. Pestaña de Señales AC con sus valores por defecto.

Los parámetros variables en de esta pestaña conforman la señal alterna: frecuencia y amplitud, establecen el número de imágenes mostradas gráficamente por cada paso de simulación y el número de periodos sobre el cual se hace la integración para la representación de ganancia y fase. Tras un proceso de toma de datos de tensión y frecuencia puede obtenerse un diagrama como el que se muestra en la figura 19, usando un software como Matlab. En esta figura puede verse el pico de resonancia debido a los dos polos complejos conjugados de la función de transferencia (Apéndice A3).

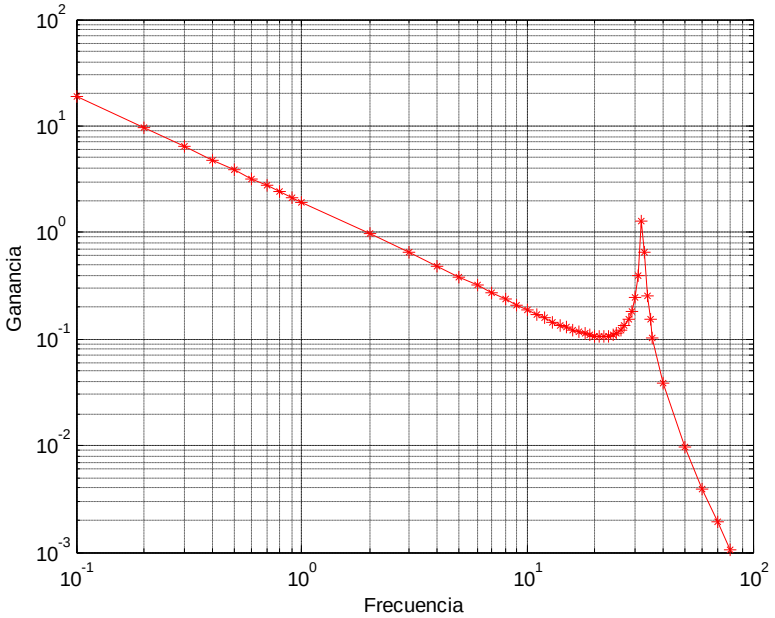

Figura 19. Diagrama de variación de ganancia con la frecuencia.

Actualmente estas señales pueden introducirse únicamente para el control manual y para el control en tensión. Pues, debido a la estructura del guión práctico no es necesario que los usuarios introduzcan estas señales para el control en velocidad.

- **Pestaña Ruido:** Esta pestaña ofrece la capacidad de introducir señales pseudo-aleatorias con una amplitud máxima y una frecuencia mínima en los motores, lo cual permite hacer un estudio de la respuesta del sistema frente al ruido.

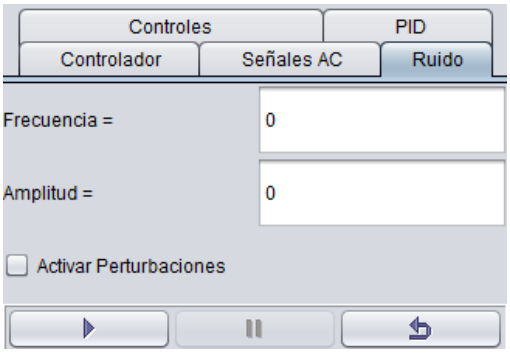

Figura 20. Pestaña de ruido de la Ventana de Representación.

En la figura 20 se muestra la pestaña Ruido, donde pueden verse los dos parámetros variables y el selector para activar la función.

3.3.1.2. Ventana de Evolución.

En la ventana de evolución tan solo ha sido añadido una ventana adicional para el caso en el que se utilicen señales alternas en las entradas de los motores.

- **Pestaña Señales AC:** En esta pestaña podemos observar una estimación de la ganancia y la fase que resulta de introducir señales alternas en el sistema.

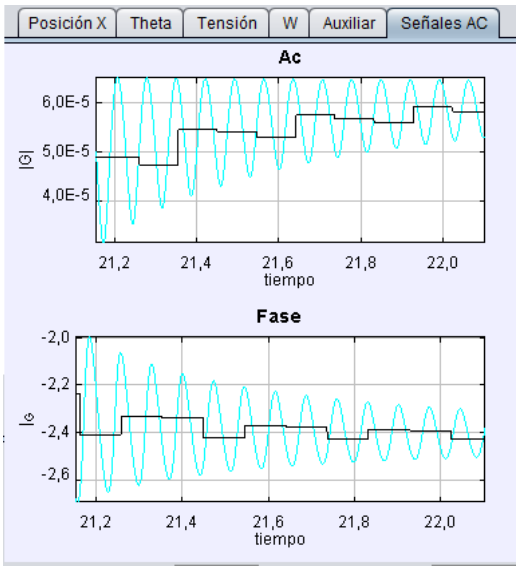

Figura 21. Ganancia y fase en la pestaña Señales AC. Ventana de Evolución.

En la figura 21 pueden verse distintas señales representadas, en azul vemos una estimación de los valores de la ganancia y la fase mientras que en negro se observan una media en bloques de doscientos datos. Si se aumenta lo suficiente el número de periodos de integración obtendremos una señal oscilante dominada por una señal decreciente, la cual tiende asintóticamente con el tiempo a un valor determinado, el cual puede anotar el usuario como ganancia y/o fase del sistema a esa frecuencia.

4. Conclusiones.

El objetivo principal de este trabajo está cumplido, el primer curso en Moodle abierto y con acceso a laboratorios virtuales y remotos se ha puesto en marcha, siendo además uno de los primeros cursos de este tipo en un portal web de automática. Hemos podido observar en el registro de los laboratorios que han sido visitados por numerosos usuarios, lo cual implica que este tipo de recursos despiertan interés en la población.

Como ventaja adicional las versiones anteriores de los laboratorios han sido actualizadas de forma que:

- Tiene la nueva opción de que el usuario pueda programar un controlador propio.
- Los laboratorios remotos funcionen a través del sistema SARLAB, añadiendo seguridad al manejo de las plantas y un sistema de registro y gestión de usuarios mejor.
- Estén actualizadas asegurando y/o mejorando el funcionamiento con las últimas versiones de EJS.

Dado que las aplicaciones y simulaciones de laboratorios son un recurso importante en la actualidad las correspondientes a los laboratorios virtuales ya están disponibles en la biblioteca de recursos en física y astronomía OSP-Compadre, y por lo tanto forman parte de una librería de laboratorios virtuales más grande. Además han sido subidas a Moodle.net [18], que es una web donde pueden publicarse y compartirse contenidos, cursos descargables o participativos con otros usuarios del mundo.

Los trabajos futuros en relación a este trabajo consistirían en la puesta en funcionamiento la planta real de los motores acoplados, completando así la parte remota de este laboratorio. Además de la mantenimiento y mejora de las aplicaciones añadiendo nuevas funcionalidades y elementos a las aplicaciones para, por ejemplo, poder usar un conjunto de controladores distintos.

A. A1: Desarrollo Teórico del Servo-Motor.

El sistema del servo motor de corriente continua se basa en el modelo de un motor de con dos bobinas separadas, la armadura o el inducido y el campo o la excitación. Un esquema de un motor de este tipo de puede ver en la figura siguiente:

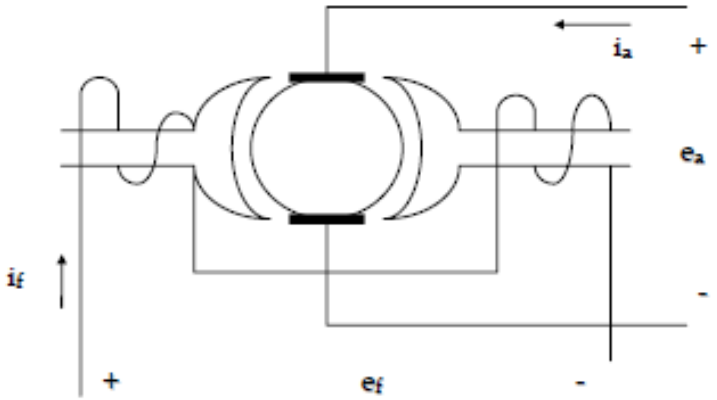

Figura 22 Esquema de un motor con inducido y excitado..

En este esquema podemos ver que tenemos la posibilidad de utilizar como entradas los potenciales en el inducido $e_a(t)$ o los potenciales del excitado $e_f(t)$. Haciendo las suposiciones siguientes, de que el flujo de campo magnético es proporcional a la corriente en el excitado y que el par desarrollado es proporcional al flujo magnético y a la corriente del inducido tenemos las siguientes relaciones:

$$\phi(t) = K_f i_f$$

$$T_m(t) = K_m \phi(t) i_a(t)$$

Si mantenemos el potencial y la corriente del excitado constantes podemos ver que le flujo magnético también lo sería y por tanto podemos concluir que el torque será proporcional a la corriente del inducido.

$$T_m(t) = K_m K_f i_f i_a(t) = K_1 i_a(t)$$

Si el flujo se mantiene constante la única contribución a la fuerza contraelectromotriz proviene de la variación del ángulo del propio eje del motor:

$$e_b(t) = K_b \frac{d\theta(t)}{t} = K_b \omega$$

Suponiendo que tanto el inducido como el excitado pueden modelarse como una inductancia en serie con una resistencia podemos obtener también la ecuación correspondiente del circuito del inducido:

$$e_a(t) = L_a \frac{di_a(t)}{dt} + R_a i_a(t) + e_b(t)$$

Por otra parte tenemos una ecuación correspondiente al torque del motor, si tenemos en cuenta todas las componentes, torque total del motor, torque debido a la carga y fricción, obtenemos la expresión del par útil desarrollado:

$$J_m \frac{d^2\theta(t)}{dt^2} = T_m(t) - B_m \frac{d\theta(t)}{dt} - T_L$$

Si sustituimos todos las expresiones anteriores obtenemos las dos ecuaciones que se muestran a continuación, en función de la corriente del inducido y el ángulo de giro:

$$e_a(t) - R_a i_a(t) = L_a \frac{di_a(t)}{dt} + K_b \frac{d\theta(t)}{t}$$

$$J_m \frac{d^2\theta(t)}{dt^2} + B_m \frac{d\theta(t)}{dt} = K_1 i_a(t) - T_L$$

Con estas ecuaciones podemos simular el comportamiento del sistema con una buena elección de las constantes del sistema, si además suponemos que la carga del sistema es nula podemos escribir las transformadas de Laplace y la función de transferencia con todos los términos considerados.

$$E_a(s) - R_a I_a(s) = s L_a I_a(s) + sK_b\theta(s)$$

$$s^2 J_m\theta(s) + s B_m\theta(s) = K_1 I_a(s)$$

$$\frac{\theta(s)}{E_a(s)} = \frac{K_1}{s\,(L_a J_m s^2 + (L_a B_m + R_a J_m)s + R_a B_m + K_1 K_b)}$$

Si consideramos que la inductancia de la armadura es muy pequeña podemos despreciarla y obtenemos la siguiente función de transferencia siguiente:

$$\frac{\theta(s)}{E_a(s)} = \frac{K_1}{s\,(\,(L_a B_m + R_a J_m)s + R_a B_m + K_1 K_b)}$$

A. A2: Desarrollo Teórico del Servo-Motor.

En este caso no podemos obtener un modelo simple de ecuaciones diferenciales que describa el modelo del heatflow, que se muestra en la figura, pues depende de la termodinámica lo cual complica las ecuaciones. Por ello se ha utilizado el artículo "creación de laboratorios virtuales y remotos usando Easy Java Simulations y Labview. El sistema Heatflow como un caso de estudio".

El modelo genérico puede expresarse de la forma que se muestra en la siguiente expresión, la temperatura del sensor n como función de los voltajes aplicados al calefactor y al ventilador, la temperatura ambiente y la distancia a la que esta colado el sensor.

$$T_n = F(V_h, V_b, T_a, x_n)$$

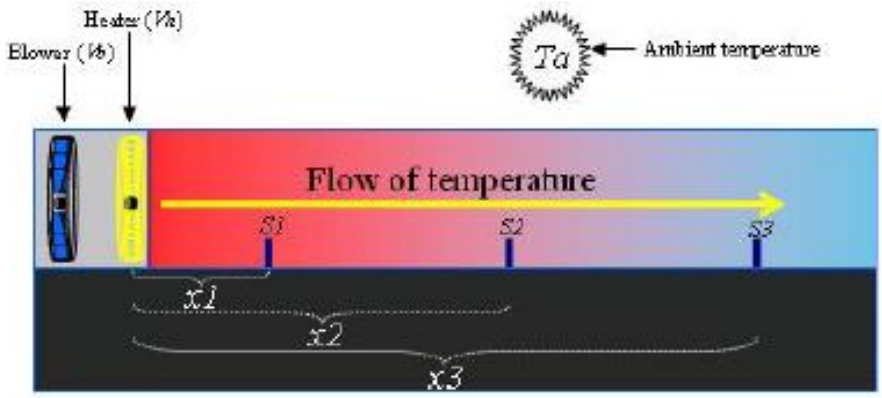

Figura 23. Sistema y variables del sistema heatflow

De este modo podemos ver que la función de transferencia del sistema puede identificarse como:

$$G(s) = \frac{T_n(s)}{V_h(s)} = \frac{K_p(1+\tau_3 s)e^{-\tau_d s}}{(1+\tau_1 s)(1+\tau_2 s)}$$

Si consideramos tan solo el modelo de segundo orden simplificado obtenemos la siguiente función de transferencia:

$$G(s) = \frac{T_n(s)}{V_h(s)} = \frac{K_p(1+\tau_3 s)}{(1+\tau_1 s)(1+\tau_2 s)}$$

A. A3: Desarrollo Teórico de los Motores Acoplados.

La planta de los motores eléctricos acoplados es un sistema de tipo multivariable diseñado para realizar el control de la tensión y la velocidad de las bandas de unión entre los motores y la polea..Utilizando la teoría lagrangiana debemos obtener primero las expresiones de la energía cinética y de la energía potencial, los cuales se muestran a continuación.

La forma cilíndrica de los motores y de la polea implican que debemos tratarlos como sólidos rígidos, y por tanto la expresión total de la energía cinética y el término de disipación se muestran en la expresión 1:

$$T = \frac{1}{2}m\dot{x}^2 + \frac{1}{2}I_1\dot{\theta}_1^2 + \frac{1}{2}I_2\dot{\theta}_2^2 + \frac{1}{2}I_p\dot{\theta}_p^2; \quad R = \left[\frac{1}{2}b_1\dot{\theta}_1^2 + \frac{1}{2}b_2\dot{\theta}_2^2 + \frac{1}{2}b_{pa}\dot{\theta}_p^2 + \frac{1}{2}b_{pt}\dot{x}_p^2\right] \quad [1]$$

La naturaleza elástica de las bandas y por tanto del acoplamiento permite realizar una aproximación a un resorte, quedando las expresiones de la energía potencial pueden verse en la expresión 2, donde k_0 es la contante del muelle del soporte de la polea.

$$V = \frac{1}{2}k[r(\theta_1 - \theta_2)]^2 + \frac{1}{2}k\left[r\left(\theta_1 - \theta_p\right) - x\cos\alpha\right]^2 + \quad [2]$$

$$\frac{1}{2}k\left[r\left(\theta_p - \theta_2\right) - x\cos\alpha\right]^2 + \frac{1}{2}k_0x^2$$

Si hacemos la suposición de que tanto el momento de inercia como la disipación debido a la rotación de la polea son despreciables (I_p = b_{pa} = 0) y si tenemos en cuenta que el ángulo de giro de la polea puede expresarse como $\theta_p = \frac{1}{2}(\theta_1 + \theta_2)$, obtenemos las expresiones algo más simplificadas 3, 4 y 5.

$$. T = \frac{1}{2}m\dot{x}^2 + \frac{1}{2}I_1\dot{\theta}_1^2 + \frac{1}{2}I_2\dot{\theta}_2^2 ; \quad [3]$$

$$R = \left[\frac{1}{2}b_1\dot{\theta}_1^2 + \frac{1}{2}b_2\dot{\theta}_2^2 + \frac{1}{2}b_{pt}\dot{x}_p^2\right] \quad [4]$$

$$V = \frac{1}{2}k[r(\theta_1 - \theta_2)]^2 + \frac{1}{2}k[r(\theta_1 - \theta_2) - 2x\cos\alpha]^2 + \frac{1}{2}k_0x^2 \quad [5]$$

Podemos obtener las ecuaciones dinámicas del sistema a partir de las ecuaciones de Lagrange,

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}_i} - \frac{\partial L}{\partial \theta_i} + \frac{\partial R}{\partial \dot{\theta}_i} = u_i$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{x}} - \frac{\partial L}{\partial x} + \frac{\partial R}{\partial \dot{x}} = 0$$

$$I_1\ddot{\theta}_1 + b_1\dot{\theta}_1 + k\cdot r^2\cdot\left[\frac{3}{2}\theta_1 - \frac{3}{2}\theta_2 - \frac{x}{r}\cos\alpha\right] = u_1 \quad [6]$$

$$I_2\ddot{\theta}_2 + b_2\dot{\theta}_2 + k\cdot r^2\cdot\left[-\frac{3}{2}\theta_1 + \frac{3}{2}\theta_2 + \frac{x}{r}\cos\alpha\right] = u_2 \quad [7]$$

$$m\ddot{x} + b_{pt}\dot{x} + k \cdot r \cdot \left[-\theta_1 \cos\alpha + \theta_1 \cos\alpha + \frac{x\, k_0}{r\, k} + \frac{2x}{r}\cos^2\alpha\right] = 0 \qquad [8]$$

En esta última expresión puede verse que el acoplamiento desaparece cuando $\alpha = \pi/2$, ya que obtenemos la expresión del oscilador armónico amortiguado, y es máximo con $\alpha = 0$.

Las ecuaciones 6, 7 y 8 son aquellas que han sido utilizadas para simular el sistema de los motores acoplados, y para ello se han utilizado los siguientes parámetros :

$$I_1 = I_2 = I = 8\ 10^{-4}\ kg\ m^2$$

$$m = 0.35\ kg$$

$$r = 0.03\ m$$

$$k = 50\ N/m$$

$$k_0 = 200N/m$$

$$b_1 = b_2 = b = 9\ 10^{-2}\ Nm/s$$

$$b_{pt} = 0.5\ N/s$$

Para obtener las expresiones de las funciones de transferencia así como el valor de los polos y ceros se ha utilizado Matlab y un valor de $\alpha = \pi/6$, cuyos resultados se muestran en las expresiones siguientes.

$$G_\omega = \frac{0.5}{I \cdot s + b}$$

$$G_x = \frac{1.7 \cdot k \cdot r}{2m \cdot Is^4 + \left(2mb + 2Ib_{pt}\right)s^3 + \left(6mkr^2 + 2Ik_0 + 2bb_{pt} + 3Ik\right)s^2 + \left(2bk_0 + 3bk + 6k\, r^2 b_{pt}\right)s + 6k^2r^2 + 6kr^2k_0}$$

Si en esta última expresión substituimos los parámetros del sistema y simplificamos obtenemos:

$$G_x = \frac{4600}{(28s^4 + 3190s^3 + 31200s^2 + 0.248\ 10^7 s + 0.338\ 10^7)} \rightarrow$$

$$\rightarrow G_x = \frac{4600}{(s^2 + 1.6s - 790)(s^2 + 111s + 1)}$$

Donde puede verse una función de transferencia para la posición x con dos polos reales y dos polos complejos conjugados.

A. B: Ejemplos de Código de Controladores PID: Servo-Motor.

- Aplicación del laboratorio Virtual del servo-motor: Control en Posición

En el siguiente cuadro de texto podemos ver el código de un controlador PID discreto, donde las salidas de control se encuentran en el vector uC[i]: uC[0], para el control en posición y uC[1], para el control en velocidad. El vector contiene en total 10 elementos para que el usuario pueda reutilizar variables en el propio código, por ejemplo uC[2] guarda la acción integral de este PID.

```
Kp = 0.08;
Ti = 0.6;
Td = 0;
N = 3;
e = (Refth - th);
ew = (Refw - w);

if (Ti == 0){
        uC[2] = 0;
}else{
        uC[2] = uC[2] + Kp*dt/Ti * e;
}
if (Td == 0){
        uC[3] = 0;
}else{
        uC[3] = Td/(Td+N*dt)*uC[3] - Kp*Td*N/(Td+N*dt)*(th - uC[4]);
}
if (uC[2] < -10)        uC[2] = -10;
if (uC[3] < -10)        uC[3] = -10;
if (uC[2] > 10)         uC[2] = 10;
if (uC[3] > 10)         uC[3] = 10;

uC[4] = th;
uC[0]= Kp*e + uC[2] + uC[3];
uC[1] =0;
```

- Aplicación del laboratorio remoto del servo-motor: Control en Posición

En el siguiente cuadro se observa el código de un controlador PID discreto, donde las salidas de control se encuentran en el vector, uC[i]: uC[0] para el control en posición y uC[1], para el control en velocidad. El vector contiene en total 10 elementos para que el usuario pueda reutilizar variables en el propio código, por ejemplo uC[2] guarda la acción integral de este PID.

En este código se añaden estructuras if-else adicionales para evitar que la señal de control se haga mayor que los límites permitidos por el sistema remoto.

```
Kp = 0.2;
Ti = 5;
Td = 0;
N = 3;
e = (Refth - th);
ew = (Refw - w);

if (Ti == 0){
        u[2] = 0;
}else{
        u[2] = u[2] + Kp*dt/Ti * e;
}
if (Td == 0){
        u[3] = 0;
}else{
        u[3] = Td/(Td+N*dt)*u[3] - Kp*Td*N/(Td+N*dt)*(th - u[4]);
}
if (u[2] < -10)         u[2] = -10;
if (u[3] < -10)         u[3] = -10;
if (u[2] > 10)          u[2] = 10;
if (u[3] > 10)          u[3] = 10;

u[4] = th;
u[0]= Kp*e + u[2] + u[3];
u[1] =0;

if (u[0] < -10)         u[0] = -10;
if (u[1] < -10)         u[1] = -10;
if (u[0] > 10)          u[0] = 10;
if (u[1] > 10)          u[1] = 10;
```

A. C: Ejemplos de Código de Controladores PID: HeatFlow.

- Aplicación del laboratorio Virtual del HeatFlow: Control de temperatura en el sensor 1.

En el siguiente cuadro de texto podemos ver el código de un controlador PID discreto, donde las salidas de control se encuentran en el vector uC[i]: uC[0], para el control de la temperatura en el sensor 1 y uC[1], para el control de la temperatura en el sensor 2, uC[2], para el control de la temperatura en el sensor 3. El vector contiene en total 10 elementos para que el usuario pueda reutilizar variables en el propio código, por ejemplo uC[2] guarda la acción integral de este PID.

```
Kp = 0.1;
Ti =2;
Td = 0;
N = 3;

e = (setPoint1 - linear_T1);
if (Ti == 0) {
        u[3] = 0;
} else {
        u[3] = u[3] + Kp*dt/Ti * e;
}
if (Td == 0) {
        u[4] = 0;
} else {
        u[4] = Td/(Td+N*dt)*u[4] - Kp*Td*N/(Td+N*dt)*(linear_T1 - u[5]);
}
if (u[3] < 0)    u[3] = 0;
if (u[4] < 0)    u[4] = 0;
if (u[2] > 5)    u[2] = 5;
if (u[3] > 5)    u[3] = 5;

u[0]= Kp*e + u[3] + u[4];
u[1] = 0;
u[2] = 0;
u[5] = linear_T1;
```

- Aplicación del laboratorio Remoto del HeatFlow: Control de temperatura en el sensor 1.

```
Kp = 0.1;
Ti =2;
Td = 0;
N = 3;
e = (setPoint1 - temp1);

if (Ti == 0) {
        u[3] = 0;
} else {
        u[3] = u[3] + Kp*dt/Ti * e;
}
if (Td == 0) {
        u[4] = 0;
} else {
        u[4] = Td/(Td+N*dt)*u[4] - Kp*Td*N/(Td+N*dt)*(temp1 - u[5]);
}

if (u[3] < 0)    u[3] = 0;
if (u[4] < 0)    u[4] = 0;
if (u[2] > 5)    u[2] = 5;
if (u[3] > 5)    u[3] = 5;

u[0]= Kp*e + u[3] + u[4];
u[1] = 0;
u[2] = 0;
u[5] = temp1;
```

Abreviaturas.

EJS	**E**asy **J**ava **S**imulations
JIL	**J**ava **I**nternet **L**abview
MOODLE	**M**odule **O**bject-**O**riented **D**ynamic **L**earning **E**nvironment
SARLAB	**S**istema de **A**cceso a **R**ecursos de **L**aboratorio
UNED	**U**niversidad **N**acional de **E**ducación a **D**istancia

Bibliografía.

[1]. F. Esquembre. Easy java simulations: A software tool to create scientific simulations in java. *Computer Physics Communications*, 156(6):199–204,January 2004.

[2]. Wolfgang Christian, Francisco Esquembre, and Lyle Barbato. Open source physics. Science, 334(6059):1077–1078, 2011.

[3]. De la Torre Cubillo, Luis. *New generation virtual and remote laboratories: Integration into web environments 2.0 with learning management system*s. (Tesis doctoral) Departamento de Informática y Automática E.T.S.I. Informática, UNED.

[4]. National Instruments, Labview: http://www.ni.com/labview/esa/

[5]. Mathworks, Matlab: http://www.mathworks.es/products/matlab/

[6]. Andrés Mejías, Marco A. Márquez, José Manuel Andújar, and María Reyes Sánchez. A complete solution for developing remote labs. In Advances in Control Education 2013, 2013.

[7]. Héctor Vargas, José Sánchez-Moreno, Sebastián Dormido, Christophe Salzmann, Denis Gillet, and Francisco Esquembre. Web-enabled remote scientific environments. Computing in Science and Engineering, 11:36–46, May 2009. ISSN 1521-9615.

[8]. G. Farias, R. De Keyser, S. Dormido, and F. Esquembre. Developing networked control labs: A matlab and easy java simulations approach. IEEE Transactions on Industrial Electronics, 57(10):3266–3275, 2010. ISSN 0278-0046

[9]. PhET : https://phet.colorado.edu/es/

[10]. RCL : http://rcl-munich.informatik.unibw-muenchen.de/

[11]. VISIR : http://ohm.ieec.uned.es/portal/?page_id=76

[12]. iSES : http://www.ises.info/index.php/en/ises

[13]. GRAASP laboratorios : http://graasp.epfl.ch/#item=space_2665

[14]. OSP-comPADRE : http://www.compadre.org/osp/index.cfm

i. Servo-Motor: http://www.compadre.org/osp/items/detail.cfm?ID=13208
ii. Heatflow: http://www.compadre.org/osp/items/detail.cfm?ID=13231
iii. Coupled Drives: http://www.compadre.org/osp/items/detail.cfm?ID=13238

[15]. École Polytechnique Fédérale de Lausanne: www.epfl.ch/index.en.html

[16]. Quanser Consulting inc.: http://www.quanser.com/

[17]. Education and Training Ltd : http://www.tq.com/

[18]. Moodle.net