Actas de las XXXIX Jornadas de Automática, Badajoz, 5-7 de Septiembre de 2018
https://doi.org/10.17979/spudc.9788497497565.0114

ENTORNO DE EXPERIMENTACIÓN PARA LABORATORIOS EN LÍNEA: EL CASO DEL PÉNDULO DE FURUTA

Daniel Galán, Luis de la Torre, Dictino Chaos, Ernesto Aranda, José Sánchez, Sebastián Dormido
Dpto. de Informática y Automática, UNED, Madrid, 28040, España.

Resumen

En este trabajo se propone una nueva forma de experimentar con laboratorios interactivos en línea, tanto virtuales como remotos. Se ha diseñado un entorno que utiliza un lenguaje de programación gráfico que permite comunicarse con laboratorios interactivos y que incluye herramientas para definir gráficos personalizables. Los profesores y diseñadores de laboratorio únicamente deben centrarse en el modelado del sistema y no en las tareas de experimentación que se pueden llevar a cabo con ella. Este entorno ofrece ventajas tanto a los profesores como a los alumnos: los primeros tienen un abanico más amplio de posibilidades a la hora de considerar los trabajos que se pueden proponer y los segundos adquieren un mayor conocimiento del sistema a estudiar al enfrentarse con un enfoque del problema que se basa más en la investigación y la práctica. Para demostrar la utilidad y posibilidades de este entorno, se detallan los resultados obtenidos con el laboratorio remoto del Péndulo de Furuta.

Palabras clave: Laboratorios remotos, laboratorios virtuales, experimentación en línea, lenguaje de experimentos, Blockly

1. Introducción

Parte del aprendizaje en disciplinas como Ciencias, Tecnología, Ingeniería o Matemáticas (STEM) ha de ser práctica para afianzar y demostrar los conocimientos teóricos. En la educación tradicional, esta experimentación se viene realizando en laboratorios o en investigaciones de campo. Sin embargo, en la enseñanza en línea y semipresencial, los dispositivos electrónicos (ordenadores, tabletas y móviles) constituyen una herramienta fundamental y necesaria para llevar a cabo experimentos. De esta necesidad nacieron las simulaciones, los laboratorios virtuales y los laboratorios remotos.

En el marco de la educación en línea o semipresencial, son varios los trabajos en que se detallan las ventajas de este tipo de laboratorios respecto a los tradicionales [2, 5, 6, 8, 10].

La gran mayoría de laboratorios virtuales y remotos se basan en la experimentación interactiva [9]. Este tipo de experimentación consiste en la interacción de los estudiantes con la interfaz gráfica del laboratorio. Ésta permite la modicicación de ciertos parámetros del sistema implementado en el laboratorio y la visualización de la evolución del sistema en tiempo real. Sin embargo, a pesar de los probados beneficios de este modo de trabajar, la experimentación interactiva tiene algunos inconvenientes importantes. Quizás, el más relevante es que los estudiantes a veces tratan de resolver problemas por prueba y error [7]. En lugar de pensar previamente cómo obtener resultado deseado del experimento/sistema y desarrollar una estrategia inteligente para alcanzarlo. Los estudiantes simplemente manipulan el laboratorio para ver qué sucede, esperando que alguna combinación de botones y parámetros de entrada pueda resultar en una respuesta cercana a la que están buscando. Por lo tanto, trabajar con estos laboratorios no requiere necesariamente del conocimiento de los conceptos subyacentes. En contraposición con esta manera de proceder, los experimentos automatizados no tienen esta carencia. Los estudiantes deben pensar de antemano cómo enfrentarse al problema y cómo trabajar con el laboratorio para obtener los resultados buscados.

Este trabajo presenta un entorno de experimentación gráfica que combina la interactividad de los laboratorios y la automatización en los experimentos. Por un lado, la experimentación interactiva permite a los estudiantes modificar parámetros para ver inmediatamente reflejados los cambios de salida y la evolución del comportamiento del sistema en la interfaz gráfica. Por otro lado, cuando los estudiantes deben realizar alguna acción programada, el proceso de experimentación requiere de ellos más implicación que simplemente pulsar botones, introducir valores en los campos de entrada y/o ver cómo los datos se trazan automáticamente en gráficos predefinidos. Creemos que, en con este entorno, los estudiantes no sólo están más motivados para trabajar, sino que también : 1) se les pide que demuestren más conocimiento acerca de

cómo resolver el problema y 2) desarrollan mejor sus habilidades científicas/técnico.

2. El entorno de experimentación

Proponemos un entorno de experimentación (EE) que permite:

1. Usar los laboratorios existentes sin requerir ninguna adaptación adicional.
2. Escribir experimentos para esos laboratorios en un lenguaje de programación gráfica.
3. Ejecutar los experimentos con un intérprete (de modo que los experimentos se ejecuten en tiempo real).
4. Recopilar y visualizar datos de los experimentos.
5. Guardar, cargar y compartir los experimentos entre alumnos y profesores.

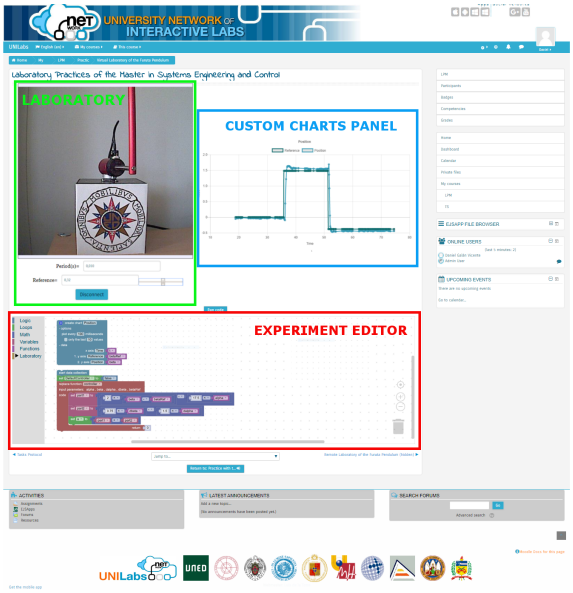


Figura 1: Arquitectura general del entorno de experimentación

El EE propuesto en este trabajo consta de tres elementos, claramente visibles en la Figura 1: (1) el Editor de experimentos (marcado con un rectángulo rojo en la parte inferior), (2) el Panel de gráficas personalizadas (cuadro azul en la parte superior derecha) y (3) el Laboratorio en línea (recuadrado en verde en la parte superior izquierda).

2.0.1. Editor de experimentos

El Editor de experimentos es el componente principal del entorno. En el se realiza la programación de los experimentos automatizados. Está formado por un área de edición y un menú, situado a la izquierda. Éste contiene categorías en las que los alumnos pueden encontrar los distintos bloques con los que realizar la programación. Estos bloques se arrastran al área de edición y se unen formando un programa, como si de un rompecabezas se tratara. Los programas creados aquí deben interpretarse automáticamente. De esta manera, los estudiantes no necesitan recompilar los experimentos una y otra vez con cada pequeño cambio que hagan en su código. En su lugar, pueden simplemente ejecutarlo, ver si los resultados son los que esperaban y continuar creando el experimento.

2.0.2. Panel de gráficas personalizadas

El entorno propuesto en este trabajo permite la creación y personalización de gráficos. Los estudiantes primero deciden los datos que quieren visualizar y luego, pueden definir el título, el período de muestreo y el número máximo de puntos que quieren trazar. Una vez definidos, los gráficos se crean automáticamente y la información se muestra en tiempo real. También es posible crear diferentes gráficos para el mismo experimento o confrontar datos de diferentes ejecuciones de un mismo experimento en el que se ha variado algún parámetro.

2.0.3. Laboratorios en línea

Los laboratorios están preparados por defecto para ser utilizados en el EE sin necesidad de adaptación o programación adicional. El tipo de laboratorio, ya sea virtual o remoto, es totalmente transparente para el EE y las funciones de éste son adaptadas automáticamente en función de su naturaleza.

3. Aplicación

Esta sección detalla cómo se aborda la implementación de los tres componentes (el Editor de experimentos, el Panel de gráficas personalizadas y los Laboratorios en línea) del entorno de experimentación gráfica.

3.1. Blockly para el editor de experimentos

Blockly es un proyecto de código abierto desarrollado por Google. Consiste en una librería JavaScript que permite la programación gráfica de código con bloques. El proyecto comenzó en 2011 y actualmente se utiliza en una amplia lista de aplicaciones web. Este proyecto tiene como objetivo

enseñar conocimientos básicos de programación a nuevos usuarios uniendo piezas de código como un rompecabezas. Los scripts creados con Blockly pueden generar código JavaScript, Python, Dart y PHP. Sin embargo, se puede personalizar con nuevos bloques y con nuevas funciones para codificar otros lenguajes [12]. En el EE presentado, el lenguaje generado es JavaScript porque encaja perfectamente con la implementación de los laboratorios. Se han definido nuevos bloques en Blockly para acceder a las variables y funciones del laboratorio en línea, permitiendo un control total del laboratorio a través de la programación.

3.2. Chart.js para las gráficas personalizables

Chart.js es una librería JavaScript de código abierto que permite la creación de gráficos HTML5. Es la librería elegida por Moodle para crear sus gráficos, por lo que se adapta perfectamente al entorno. Tiene varias características notables:

- Incluye ocho tipos diferentes de gráficos que pueden ser mezclados, personalizados o incluso animados.
- Como está construido sobre HTML5 y tiene un gran rendimiento en los navegadores modernos, lo que es muy importante en los gráficos en tiempo real como los de experimentación.
- Los gráficos tienen un diseño sensible al dispositivo utilizado por lo que el entorno de experimentación se puede visualizar tanto en ordenadores como teléfonos móviles o tabletas.

3.3. EjsS para los laboratorios en línea

Los avances tecnológicos han supuesto que el software que rodea a los laboratorios virtuales y remotos se haya vuelto cada vez más complejo. La creación de una interfaz gráfica para utilizar el laboratorio no es trivial, especialmente para aquellos que no tienen amplios conocimientos de programación. Afortunadamente, EjsS ofrece interfaces gráficas de alto nivel para crear simulaciones por ordenador. Los usuarios con pocos conocimientos de programación informática pueden concentrar sus esfuerzos en definir el modelo matemático del laboratorio y el aspecto que debe tener la interfaz gráfica de usuario. Cualquier otro aspecto de la implementación se genera automáticamente. EjsS forma parte del OSP, que ofrece gratuitamente colecciones de recursos en línea a través de la biblioteca digital ComPADRE. Entre estos recursos, los usuarios pueden encontrar más de 500 aplicaciones creadas con EjsS: `http://www.opensourcephysics.org`.

3.4. UNILabs como plataforma para el uso del EE

UNILabs (University Network of Interactive Labs) es una web 2.0 de laboratorios interactivos, un Sistema de Gestión del Aprendizaje (*Learning Management System*), donde cualquier universidad puede incluir libremente sus laboratorios y ofrecerlos a sus estudiantes [11]. Creado en 2013, hoy ofrece más de 20 laboratorios remotos y virtuales distribuidos en 31 cursos de diversas especialidades (control automático, matemáticas, óptica, automatización, sistemas lineales, física moderna y clásica, entre otras). UNILabs está basado en Moodle, que ofrece excelentes funcionalidades para gestionar los recursos de experimentación por parte de los profesores, e incluye los plugins de EJSApp Moodle para hacer uso de los laboratorios EjsS.

La página web de UNILabs (`http://unilabs.dia.uned.es`) incluye más información sobre los cursos y universidades que colaboran.

4. El laboratorio online del péndulo de Furuta

Se ha desarrollado un laboratorio virtual que consiste en una simulación de las ecuaciones de péndulo (1) y (2) (donde β es el ángulo descrito por el brazo rotatorio, y α es el ángulo descrito por el péndulo), usando EjsS. La simulación incorpora una representación 3D y algunas entradas para interactuar (ver Figura 2). La interfaz de usuario es lo más sencilla posible y sólo permite las acciones más básicas: cambiar la posición del péndulo, la referencia y detener, pausar o reanudar la simulación.

$$\begin{aligned}\ddot{\beta} =& \frac{h_2 h_3 \cos^2\alpha \sin\alpha \dot{\beta}^2}{H} \\ & \frac{h_3 \cos\alpha \left(-B_1\dot{\alpha} + h_5 \sin\alpha\right)}{H} \\ & \frac{-h_4\dot{\beta}\left(B_0 + 2h_2\dot{\alpha}\cos\alpha\sin\alpha\right)}{H} \\ & \frac{-h_4\left(-\tau + h_3\dot{\alpha}^2\sin\alpha\right)}{H}\end{aligned} \tag{1}$$

$$\begin{aligned}\ddot{\alpha} =& \frac{\left(h_2\sin^2\alpha + h_1\right)h_2\cos\alpha\sin\alpha\dot{\beta}^2}{H} \\ & \frac{\left(h_2\sin^2\alpha + h_1\right)\left(-B_1\dot{\alpha} + h_5\sin\alpha\right)}{H} \\ & \frac{-h_3\dot{\theta}_0\cos\alpha\left(B_0 + 2h_2\dot{\theta}_1\cos\alpha\sin\alpha\right)}{H} \\ & \frac{-h_3\cos\alpha\left(-\tau + h_3\dot{\theta}_1^2\sin\alpha\right)}{H}\end{aligned} \tag{2}$$

El laboratorio remoto tiene dos componentes principales: 1) el lado del servidor, desarrollado en LabVIEW, que acepta conexiones usando un componente llamado JIL [3] y 2) la aplicación del laboratorio en EjsS, cuyas variables están vinculadas a LabVIEW. La interfaz del laboratorio remoto es muy similar a la virtual, pero muestra imágenes a tiempo real de una cámara web del péndulo en lugar de la vista simulada en 3D (ver Figura 2).

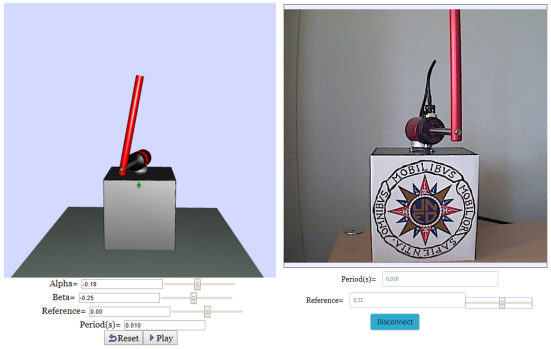


Figura 2: Laboratorio virtual (izquierda) y remoto (derecha)

La interfaz de usuario del laboratorio virtual y remoto se ha simplificado al máximo, pero sin perder usabilidad. Esta es la diferencia crucial con los diseños anteriores que incluyen muchas herramientas y gráficas enfocadas en resolver tareas o experimentos muy específicos.

A modo de comparación, la Figura 3 muestra una interfaz de usuario antigua para este laboratorio. Esta interfaz necesitaba anticiparse a las necesidades de los usuarios. Por lo tanto, mostró muchos elementos para la visualización de variables y varios menús con opciones avanzadas relacionadas.

Una característica importante de este laboratorio es que permite cambiar el controlador del sistema. Esto se ha hecho también en laboratorios anteriores, pero utilizando un lenguaje específico para el dominio de aplicación, que necesita un servidor especial para su ejecución, [4] detalla una implementación previa exitosa de esta idea. En este trabajo, el controlador se define con código en JavaScript, por lo tanto, el mismo código puede interactuar sin modificación con el laboratorio virtual y con el sistema real o remoto. En este último caso, el código JavaScript se ejecuta en un *Sandbox* en LabVIEW.

Además, el laboratorio desarrollado en EjsS no tiene un editor ni opciones para cargar y guardar archivos. Todo (desde el controlador hasta la visualización) puede construirse usando el lenguaje de bloques del entorno, como se ilustra en la siguiente sección. Por lo tanto, la interfaz de usuario es general, simple y puede ser utilizada para muchos propósitos y experimentos diferentes sin ninguna modificación.

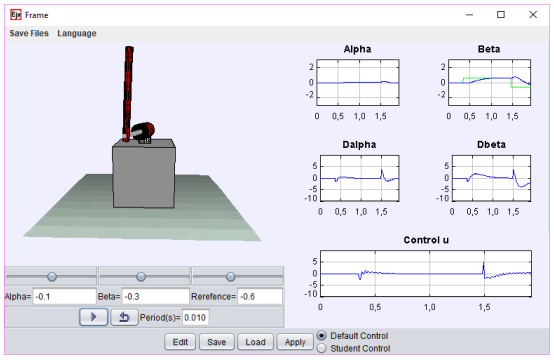


Figura 3: Antiguo laboratorio con una compleja interfaz gráfica de usuario

5. Experimentos planteados

Esta sección presenta algunos experimentos que los estudiantes realizan durante sus estudios en el Máster de Ingeniería de Sistemas y Control de la UNED utilizando el laboratorio remoto de péndulos Furuta. Gracias al EE, los estudiantes no sólo pueden llevar a cabo estos experimentos técnicamente complicados de una manera sencilla e intuitiva, sino que, como entorno flexible, se les anima a investigar y probar sus propios controladores. De esta manera, los profesores pueden utilizar el entorno como una herramienta para aplicar técnicas de aprendizaje basado en investigación.

5.1. Mantener el péndulo hacia arriba mientras el brazo rotatorio sigue los cambios de la referencia

En muchos laboratorios remotos de Ingeniería de Control, un enfoque típico es implementar el controlador en el lado del servidor. Es decir, en lugar de tener el controlador implementado en el lado del cliente, enviando periódicamente la señal de control al servidor, el código del controlador se envía una vez al servidor, y se ejecuta allí.

El entorno de experimentación gráfico discrimina, sin necesidad de intervención del usuario, entre el código a ejecutar localmente y el código a ejecutar remotamente. La Figura 4 muestra un experimento con código local y remoto para controlar el péndulo y visualizar las salidas.

El sistema tiene un controlador predeterminado que coloca el péndulo hacia arriba y un controlador diseñado por los estudiantes que puede ser reemplazado. Por lo tanto, el primer paso es deshabilitar el controlador por defecto y el segundo reemplazarlo por un controlador de retroalimentación lineal (mostrado en detalle en la Figura 5a).

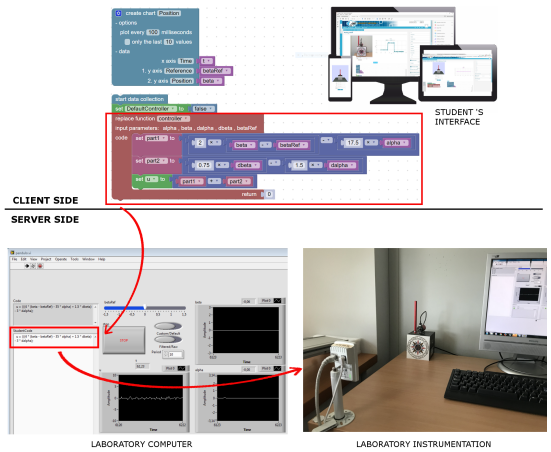
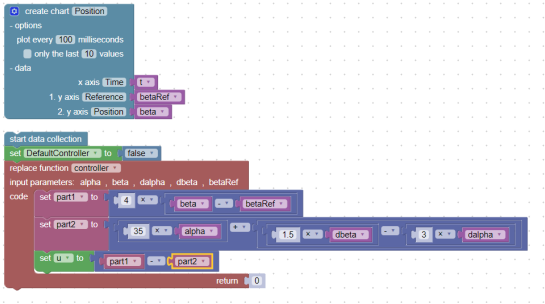


Figura 4: Cliente y servidor del entorno de experimentación cuando se ejecuta el experimento del controlador de posición

El código que sustituye a la función del controlador es el código que se ejecutará de forma remota (resaltado en rojo), mientras que el resto se ejecutará de forma local.



(a) Código ejecutado en el lado del cliente

```
StudentCode
u = (((4 * (beta - betaRef) - 35 * alpha) + 1.5 * dbeta)
- 3 * dalpha);
```

(b) Código recibido en el lado del servidor

Figura 5: Vista detallada del código del experimento y del controlador recibido en el servidor

La Figura 4 presenta una captura de pantalla del ordenador servidor. En ella se muestra el programa LabVIEW que controla las entradas y salidas del sistema de péndulo Furuta. El código remoto que se desarrolla en el entorno y se envía automáticamente al servidor se resalta en rojo. Estos dos elementos también se ilustran más claramente en la Figura 5: en la parte superior, el controlador diseñado con Blockly, en la parte inferior (Figura 5b), el código que llega al servidor una vez que se ha ejecutado el experimento. Una vez recibido el código del controlador, el servidor lo ejecuta en cada paso de ejecución y envía el nuevo valor de acción de control al péndulo.

El experimento diseñado con Blockly puede verse en la Figura 5a. El bloque "Create Chart", situado en la parte superior, se utiliza para definir el gráfico. Dibujará las trazas de la referencia del ángulo descrito por el brazo rotatorio ("betaRef") y su medida real ("beta") en función del tiempo ("t"). Cada 100 milisegundos, se realizará una medición. Después de esta definición, hay otro conjunto de bloques en el experimento: el bloque "'start data collection" inicia la adquisición de datos para la visualización; el bloque "'set" modifica una bandera para indicar al servidor que se utilizará un controlador definido por el usuario; y finalmente, el bloque "'replace function"'se utiliza para reemplazar la función del controlador. El contenido de este bloque será enviado al servidor y utilizado como controlador en el laboratorio remoto. Para ello, los bloques se traducirán a código JavaScript y se enviarán al servidor cuando se ejecute el experimento.

En la parte superior derecha de la Figura 1, se puede ver el gráfico obtenido tras la ejecución del experimento. Los cambios en la referencia de ángulo, realizados durante la ejecución del experimento automático utilizando la interfaz gráfica del laboratorio, son visibles observando los pasos del gráfico. Primero se cambió a 1.48 radianes, y luego a -0.38 radianes.

Gracias a este experimento, se puede ver cómo los parámetros establecidos para el controlador son apropiados, ya que, además de hacer que el péndulo permanezca en equilibrio, el sistema es capaz de seguir con eficiencia los puntos de referencia marcados. La cámara IP, junto con los gráficos definidos en el entorno de experimentación, permite a los usuarios corroborar este hecho. Además, es fácil probar otros parámetros y ver si son tan efectivos como los utilizados en el experimento, o hacen que el sistema sea inestable y el péndulo caiga.

5.2. Creación del controlador de posición con la función de balanceo hacia arriba

Siguiendo el mismo proceso que en el experimento anterior, es posible diseñar un controlador más complejo. Este controlador es capaz no sólo de controlar la posición de la base, sino también de girarla hacia arriba si el péndulo cae. De esta manera, si el controlador detecta que el péndulo está lejos de la región de equilibrio esperada, ejecutará el código que lo empujará hacia arriba. De lo contrario, ejecutará el código visto en el experimento anterior.

Para la implementación de este controlador, basa-

do en el diseño de Åström [1], en lugar de utilizar los bloques tradicionales, se ha utilizado un bloque de evaluación de código JavaScript. Al ser un controlador que requiere más código para su implementación, el uso de bloques puede dificultar el proceso de creación, por lo que el entorno de experimentación ofrece la posibilidad adicional de escribir código a usuarios avanzados.

Como el sistema eleva el péndulo automáticamente, es suficiente, en primer lugar, enviar un controlador en el que la señal de control es siempre 0 ($u = 0$). Esto dejará caer el péndulo y permitirá que se verifique el controlador requerido.

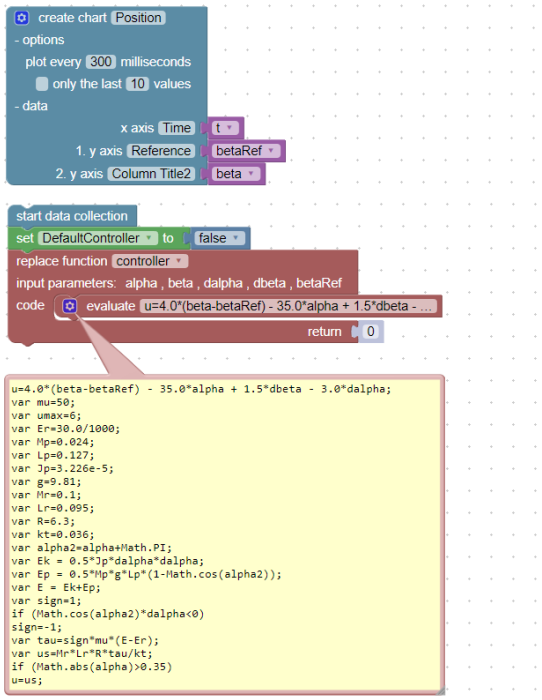


Figura 6: Experimento de balanceo hacia arriba y control de posición

La Figura 6 ilustra el experimento creado. Tiene la misma estructura que el experimento anterior, pero el controlador ha sido definido en JavaScript. Para esto, en lugar de usar los bloques para establecer variables y ecuaciones, se usa un bloque "evaluar". En este bloque, los usuarios pueden escribir código JavaScript (ver la parte inferior del experimento definido). Para comprobar su funcionamiento, basta con observar a través de la cámara IP cómo se levanta el péndulo cuando se realiza el experimento y el controlador comienza a funcionar. Una vez levantado, el comportamiento es el mismo que en el experimento anterior.

6. Conclusiones

La mayoría de los laboratorios actuales, tanto virtuales como remotos, son aplicaciones cerradas que ofrecen tareas de experimentación basadas exclusivamente en la interactividad, lo que limita el trabajo de los estudiantes con los laboratorios. Este artículo presenta un entorno de experimentación capaz de superar estas limitaciones. Para ello, el entorno desarrollado añade nuevas capacidades a los laboratorios interactivos, como por ejemplo: 1) la posibilidad de diseñar experimentos automatizados con un lenguaje visual y fácil de usar, resolviendo las limitaciones de la experimentación interactiva; 2) la capacidad de añadir gráficas diseñadas por los estudiantes a la interfaz visual del entorno, dando nuevas posibilidades para visualizar los resultados; y 3) la capacidad de redefinir funciones y modificar variables que el diseñador podría no haber considerado interesantes en ese momento, abriendo los laboratorios a nuevas tareas de experimentación que antes no eran posibles. Para mostrar la utilidad del entorno de experimentación, se presentan dos experimentos diferentes para un laboratorio remoto del péndulo de Furuta.

Agradecimientos

Este trabajo ha sido financiado por el Ministerio de Economía y Competitividad en el marco del proyecto CICYT DPI2014-55932-C2-2-R.

English summary

ONLINE LABORATORIES EXPERIMENTATION ENVIRONMENT: FURUTA'S PENDULUM CASE

Abstract

This paper proposes a new way of experimenting with interactive online laboratories, both virtual and remote. The designed environment uses a graphical programming language that allows communication with interactive laboratories and includes tools to define customizable graphics. Teachers and designers should only focus on the plant modeling and not on the experimentation tasks that can be carried out with it. This environment offers advantages to both teachers and students: the former have a wider range of possibilities when considering the work that can be proposed and the latter acquire a greater knowledge of the plant to be studied when faced with an approach to the problem that is more based on research and practice. To

demonstrate the usefulness and possibilities of this environment, the results obtained with the Furuta Pendulum remote laboratory were detailed.

Keywords: Remote laboratories, virtual laboratories, online experimentation, experimentation language, Blockly.

Referencias

[1] Åström, K. J., and Furuta, K. Swinging up a pendulum by energy control. *Automatica 36*, 2 (2000), 287–295.

[2] Brinson, J. R. Learning outcome achievement in non-traditional (virtual and remote) versus traditional (hands-on) laboratories: A review of the empirical research. *Computers and Education 87* (2015), 218–237.

[3] Chacon, J., Vargas, H., Farias, G., Sanchez, J., and Dormido, S. Ejs, jil server, and labview: An architecture for rapid development of remote labs. *IEEE Transactions on Learning Technologies 8*, 4 (2015), 393–401.

[4] Chaos, D., Chacón, J., Lopez-Orozco, J. A., and Dormido, S. Virtual and remote robotic laboratory using ejs, matlab and labview. *Sensors 13*, 2 (2013), 2595–2612.

[5] de la Torre, L., Sanchez, J. P., and Dormido, S. What remote labs can do for you. *Physics Today 69* (2016), 48–53.

[6] Frerich, S., Kruse, D., Petermann, M., and Kilzer, A. Virtual labs and remote labs: Practical experience for everyone. In *Engineering Education 4.0* (2016), Springer, pp. 229–234.

[7] Guzman, J. L., Costa-Castello, R., Dormido, S., and Berenguel, M. An interactivity-based methodology to support control education: How to teach and learn using simple interactive tools [lecture notes]. *IEEE Control Systems 36*, 1 (2016), 63–76.

[8] Heradio, R., de la Torre, L., and Dormido, S. Virtual and remote labs in control education: A survey. *Annual Reviews in Control 42*, Supplement C (2016), 1 – 10.

[9] Heradio, R., de la Torre, L., Galan, D., Cabrerizo, F. J., Herrera-Viedma, E., and Dormido, S. Virtual and remote labs in education: A bibliometric analysis. *Computers & Education 98* (2016), 14–38.

[10] Potkonjak, V., Gardner, M., Callaghan, V., Mattila, P., Guetl, C., Petrović, V. M., and Jovanović, K. Virtual laboratories for education in science, technology, and engineering: A review. *Computers & Education 95* (2016), 309–327.

[11] Sáenz, J., Chacón, J., De La Torre, L., Visioli, A., and Dormido, S. Open and low-cost virtual and remote labs on control engineering. *IEEE Access 3* (2015), 805–814.

[12] Trower, J., and Gray, J. Sigcse. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (2015), ACM, pp. 677–677.