TecnoLógicas

Artículo de Investigación/Research Article

Simulación del Péndulo Invertido Rotacional usando Easy Java Simulations y Matlab

Simulating a Rotational Inverted Pendulum Model by using Matlab and Easy Java Simulations

Oscar O. Rodríguez-Díaz[1]
Edwin L. Téllez-Valderrama[2]
Diego A. Gutiérrez-Ramírez[3]

Fecha de recepción: 14 de Julio de 2011
Fecha de aceptación: 07 de Abril de 2012

1 Grupo de Investigación DSP, Universidad Pedagógica y Tecnológica de Colombia, Sogamoso-Colombia, oscar.rodriguez@uptc.edu.co
2 Grupo de Investigación DSP, Universidad Pedagógica y Tecnológica de Colombia, Sogamoso-Colombia, edwin.tellez@uptc.edu.co
3 Grupo de Investigación DSP, Universidad Pedagógica y Tecnológica de Colombia, Sogamoso-Colombia, diegoandres.gutierrez@uptc.edu.co

Resumen

En este artículo se presenta el análisis, diseño y construcción de un simulador virtual, que representa gráficamente el comportamiento de un sistema no lineal como el del péndulo invertido rotacional. Esta interfaz se presenta por medio de un Applet de Java que permite a los usuarios hacer la variación de los parámetros del modelo, formando un puente entre los conceptos teóricos y los comportamientos reales del proceso. Del lado del servidor se utiliza Matlab/Simulink como motor de cálculo numérico dada su facilidad para construir modelos no lineales mediante diagramas de bloques. La interfaz de usuario ha sido diseñada mediante la herramienta de software gratuito *Easy Java Simulations*, que permite crear aplicaciones gráficas con alto grado de interactivi-dad como interfaces con objetos en 3D. Esta herramienta es de gran ayuda para la enseñanza del control automático.

Palabras clave

Péndulo; control automático; laboratorio virtual; sistemas no lineales.

Abstract

This paper presents the analysis, design and construction of a virtual simulator, in which the behavior of a non-lineal system as the rotational inverted pendulum is represented graphically in an interface. This uses an Applet of Java that allows users change parameters of the model. The use of this tool is a good alternative for bridging the gap between the theoretical concepts and the actual behaviors of a process. The server uses Matlab/Simulink as a calculation engine, taking advantage of its ease for constructing non-lineal models by using block diagrams. The user interface has been created by a free software tool called Easy Java Simulations that allows designing interactive graphical applications as 3D interfaces. Easy Java results an interesting tool for automatic control system education.

Keywords

Pendulum; automatic control; virtual laboratory; nonlinear system.

1. INTRODUCCIÓN

El péndulo invertido rotacional también conocido como péndulo de Furuta (Furuta *et al.*, 1992) es un sistema subactuado, con dos grados de libertad, muy estudiado en la teoría de control no lineal, desde entonces se han publicado artículos y tesis usando este sistema para describir leyes de control lineales y no lineales (Åkesson & Ästrom, 2001) pero muy pocos incluyen la dinámica completa. En este artículo se muestra la construcción de una interfaz gráfica para presentar de forma interactiva el prototipo desarrollado por Caipa *et al.* (2010) el cual emplea la dinámica del sistema no lineal presentada por Cazzolato & Prime (2008) en donde se incluye la dinámica dada por el motor DC y algunas otras conjeturas como la fricción. Buscando herramientas de apoyo para la enseñanza del control como las presentadas por Sánchez *et al.* (2005), Domínguez *et al.* (2005), Farias *et al.* (2006) y Dormido, (2008), donde mediante plataformas virtuales se simulan y presentan modelos matemáticos a sistemas reales con el fin de facilitar la enseñanza del control automático a estudiantes de ingeniería. Para la conexión a través de internet con Matlab se utiliza Jim (Java Internet Matlab) de Farias *et al.* (2006), con el fin de realizar una comunicación del modelo no lineal implementado en Simulink y la interfaz de usuario desarrollada con Easy Java Simulations (EJS) de Esquembre (2005). Esta comunicación permite constituir un laboratorio virtual híbrido dándole al usuario la capacidad de manipular la simulación sin necesidad de tener instalado Matlab/Simulink en su ordenador, ya que la aplicación desarrollada con EJS establece un enlace de red con un equipo remoto o servidor, soportado por Jim, el cual posee el software licenciado con Matlab/Simulink.

Este trabajo es el resultado del proyecto de investigación apoyado por la dirección de investigaciones de la Universidad Pedagógica y Tecnológica de Colombia (UPTC) en el proyecto con código SGI 764 titulado "Desarrollo de una plataforma virtual y remota para la enseñanza e investigación en el área de control automático e instrumentación industrial", la plataforma virtual proporciona una herramienta pedagógica como apoyo en la enseñanza de control al grupo de procesamiento de señales DSP de la U.P.T.C, y se

encuentra organizado de la siguiente forma: en la sección 2 se describe la dinámica del péndulo y el análisis del modelo matemático incluyendo la dinámica del motor, también se construye el modelo linealizado mediante Matlab/Simulink y con base en este se realiza la conexión del modelo en Simulink con EJS. En la sección 3 se describe la comunicación de la interfaz de usuario con el servidor de internet para Matlab (JIM), mostrando los resultados de la interfaz del péndulo invertido rotacional, así como la presentación final al usuario.

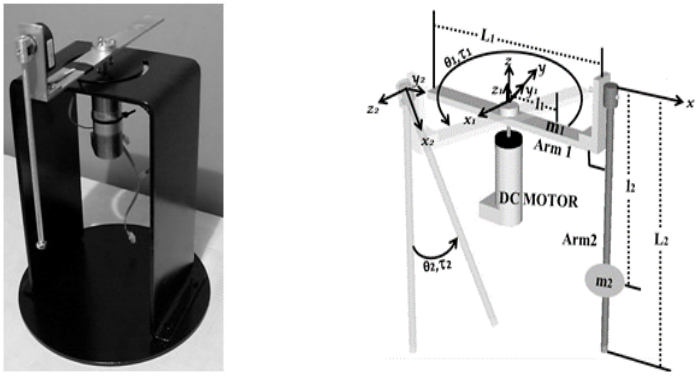
2. METODOLOGÍA

2.1 Modelado del Sistema de Péndulo Rotacional

Para la construcción del simulador virtual se inicia utilizando los parámetros del sistema de péndulo invertido rotacional desarrollado en la UPTC por Caipa *et al.* (2010), el cual se observa en la Fig. 1a. El prototipo se encuentra montado sobre un motor DC (Fig. 1b), el cual aplica un torque (τ) al brazo generando una rotación en el plano horizontal, acoplado al brazo se encuentra el péndulo quien puede rotar libremente en el plano vertical. El brazo y el péndulo tienen longitudes L_1 y L_2 y masas m_1 y m_2 a una distancia l_1 y l_2 respecto a su centro de masa, con momentos de inercia j_1 y j_2 respectivamente.

El momento total de inercia del brazo sobre la punta del eje es J_1 y la inercia total del péndulo sobre la punta de su eje es J_2. Cada unión rotacional tiene coeficientes de amortiguamiento viscoso b_1 y b_2, donde b_1 es el amortiguamiento suministrado por las escobillas del motor y b_2 es el amortiguamiento proveniente de la conexión entre el brazo y el péndulo (Caipa *et al.*, 2010).

Para definir las entradas, se usa el sistema coordenado dado por la regla de la mano derecha. La rotación angular del brazo (θ_1) es medida en el plano horizontal en dirección opuesta de las manecillas del reloj (positiva). La rotación angular del péndulo (θ_2) es medida en el plano vertical en dirección opuesta de las manecillas del reloj (positiva); cuando el péndulo está suspendido en equilibrio estable (abajo) la posición es $\theta_2 = 0$.



(a) (b)

Fig. 1. a) Prototipo físico real. Fuente: Caipa *et al.* 2010, b) Descripción física del péndulo invertido. Fuente: Autores.

El par que el motor aplica sobre el brazo, τ1, es positivo en el sentido contrario de las agujas del reloj (cuando es visto desde arriba). El par de perturbación, τ2, es experimentado por el péndulo y es positivo en el sentido contrario a las agujas del reloj (cuando es visto de frente).

A partir de la formulación de Newton – Euler se obtiene la dinámica del sistema deducida por Cazzolato & Prime (2008), (1) y (2) describen la dinámica no lineal del péndulo.

$$\ddot{\theta}_1\left(\hat{J}_0 + \hat{J}_2 sin^2(\theta_2)\right) + \ddot{\theta}_2 m_2 L_1 l_2 \cos(\theta_2) - m_2 L_1 l_2 \sin(\theta_2)\,\dot{\theta}_2^2 + \dot{\theta}_1\dot{\theta}_2\hat{J}_2 \sin(2\theta_2) + b_1\,\dot{\theta}_1 = \tau \quad (1)$$

$$\ddot{\theta}_1 m_2 L_1 l_2 \cos(\theta_2) + \ddot{\theta}_2\hat{J}_2 - \frac{1}{2}\dot{\theta}_1^2\hat{J}_2 \sin(2\theta_2) + g m_2 l_2 \sin(\theta_2) + b_2\dot{\theta}_2 = 0 \quad (2)$$

Ya que el sistema está montado sobre un motor DC, se utiliza la ley de Kirchhoff de voltaje para describir el subsistema eléctrico, esta se observa en (3), el torque producido por el motor está dado en (4).

$$V = L_m\dot{i} + R_m i + K_m\dot{\theta}_1 \quad (3)$$

$$\tau = K_m i \tag{4}$$

Dónde: "V" es el voltaje aplicado al motor, "Lm" es la inductancia eléctrica del motor, "Rm" es la resistencia eléctrica del motor, "Km" es la constante de torque, "τ" es el torque ejercido por el motor, y por último "i" es la corriente que fluye a través del motor.

Para la obtención del modelo linealizado se necesita acomodar la dinámica del sistema dado en (1) y (2) junto con (3) y (4) a la representación en espacio de estados, donde se incluye otra variable de estado: X_5=i que es la corriente del motor. El vector de estado corresponde al mostrado en (5).

$$[x_1 \;\; x_2 \;\; x_3 \;\; x_4 \;\; x_5]^T = \left[\theta_1 \;\; \theta_2 \;\; \dot{\theta}_1 \;\; \dot{\theta}_2 \;\; i\right]^T \tag{5}$$

Haciendo cero las ecuaciones de estado no lineales se obtienen los puntos de equilibrio del sistema representados en (6), en donde ß puede ser cualquier valor entre 0 y 2π para la posición del brazo, cuando n=1, se encuentra la posición de equilibrio inestable del péndulo (vertical-arriba), cuando n=0, se encuentra la posición de equilibrio estable (vertical-abajo).

$$x_{ieq} = [\beta \;\; n\pi \;\; 0 \;\; 0 \;\; 0]^T \tag{6}$$

Realizando una linealización en torno al punto de equilibrio inestable, se obtiene el modelo lineal en espacio de estados según (7) y por medio del Jacobiano, donde la entrada al sistema es el voltaje aplicado al motor.

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & A_{32} & A_{33} & A_{34} & B_{31}K_m \\ 0 & A_{42} & A_{43} & A_{44} & B_{41}K_m \\ 0 & 0 & -\dfrac{K_m}{L_m} & 0 & -\dfrac{R_m}{L_m} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \dfrac{1}{L_m} \end{bmatrix} V \tag{7}$$

Dónde:

$$A_{32} = \frac{a_1 a_2}{J_0 J_2 - {a_1}^2}\,,\; A_{33} = \frac{-J_2 b_1}{J_0 J_2 - {a_1}^2},\; A_{34} = \frac{-a_1 b_2}{J_0 J_2 - {a_1}^2},\quad B_{31} = \frac{J_2}{J_0 J_2 - {a_1}^2}$$

$$A_{42} = \frac{a_2 J_0}{J_0 J_2 - {a_1}^2},\ A_{43} = \frac{-a_1 b_1}{J_0 J_2 - {a_1}^2},\ A_{44} = \frac{-b_2 J_0}{J_0 J_2 - {a_1}^2},\quad B_{41} = \frac{a_1 a_2}{J_0 J_2 - {a_1}^2}$$

La salida del sistema está dado representado en (8), la cual representa las posiciones angulares (θ_1 y θ_2) medibles del sistema real del brazo y del péndulo respectivamente.

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} \tag{8}$$

2.2 Modelo en Simulink

El modelo no lineal se implementa mediante el uso de una S-Function de Matlab/Simulink, la cual contiene un archivo .m, donde se define el número de entradas, número de salidas, los estados y por último las ecuaciones no lineales en espacio de estados del sistema. El modelo no lineal implementado en Simulink mediante diagramas de bloques se muestra en la Fig. 2.

Los parámetros del Multiplexor en la entrada de la S-Function corresponden a los valores del sistema (ver Tabla 1) dados por Caipa et al (2010). Estos parámetros están visibles en la interfaz de usuario dada por la aplicación de Java, de manera que el usuario puede modificar estos valores del modelo y de esta forma ver los cambios en la respuesta del sistema, mediante las cinco variables obtenidas en el Demultiplexor (ver Fig. 2).

Evaluando la respuesta del sistema ante una condición inicial del péndulo en cercanías del punto de equilibrio inestable se obtienen las respuestas del sistema (ver Fig. 3). Como se puede observar las oscilaciones del brazo y el péndulo respectivamente disminuyen progresivamente hasta llegar al punto de equilibrio estable.

En la Fig. 3 se muestra como el brazo puede variar con valores entre -1 y 1, correspondientes a -180° y 180°, de igual forma el péndulo oscila entre -3.1416 y 3.1416 (rad/s) que corresponden a –π y π (rad/s).

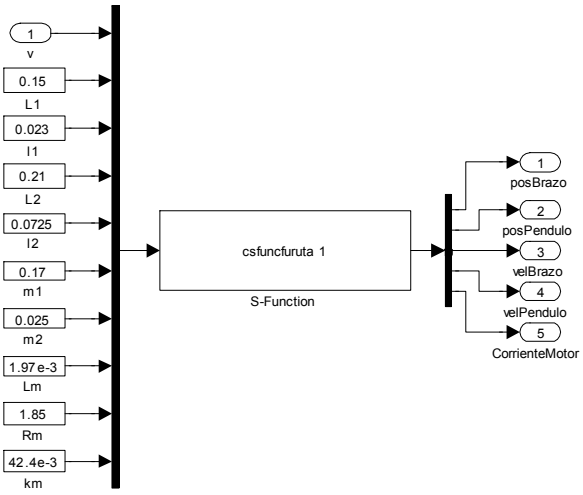


Fig. 2. Modelo no lineal en Simulink. Fuente: Autores

Tabla 1. Parámetros del sistema. Fuente: Caipa *et al.* 2010

Descripción	Notación	Valor	Unidades
Parámetros Motor			
Inductancia	L_m	1,97	mH
Resistencia	R_m	1,85	Ω
Constante torque	K_m	0,0424	Nm/A
Coeficiente Viscoso	b_1	3,7 E^{-6}	Nm/rad
Parámetros Brazo			
Longitud	L_1	0,15	m
Longitud al centro de masa(C.M.)	l_1	0,023	m
Masa	m_1	0,17	Kg
Momento de Inercia en C.M.	J_1	0,0016	Kgm^2
Coeficiente Viscoso(Estimado)	b_2	$1E^{-4}$	Nms/rad
Parámetros Péndulo Corto			
Longitud	L_2	0,21	m
Longitud al C.M.	l_2	0,0725	m
Masa	m_2	0,25	Kg
Momento de inercia en C.M.	J_2	0,0016	Kgm^2

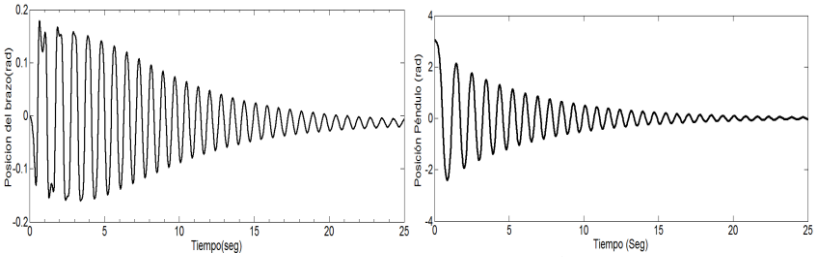


Fig. 3. Respuesta al modelo no lineal en Simulink. Fuente: Autores

2.3 Conexión del Modelo en Simulink con la Interfaz en EJS

El desarrollo de simulaciones interactivas en EJS se puede describir en tres pasos (ver Fig. 4), adoptando el paradigma de Modelo-Control-Vista de Dormido (2008). A partir de este paradigma se realiza la interfaz de usuario para el péndulo de Furuta en la interfaz de EJS.

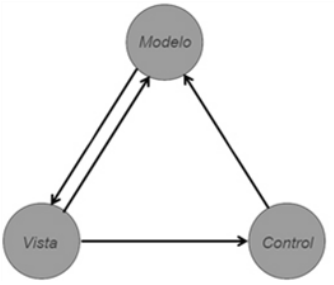
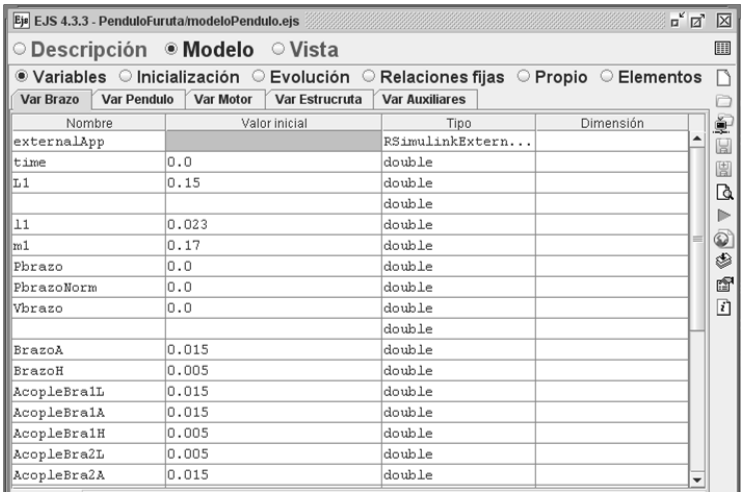


Fig. 4. Interacción entre Modelo-Control-Vista. Fuente: Autores

El Modelo describe el modelo bajo estudio, se definen y describen las variables y relaciones, así como las inicializaciones y demás características del sistema a simular. El control vincula las variables modelo con la interfaz gráfica, define las acciones que el usuario puede realizar sobre la simulación. En la sección Vista se define el árbol de elementos que tomarán parte de la simulación a partir de los elementos de la interfaz de dibujo 2D y 3D incluida en EJS. Existe una gran cantidad de elementos para utilizar en las animaciones.

2.4 Modelo EJS

Como se observa en la Fig. 5, en este panel se definen las variables y los diferentes comportamientos del sistema, también las inicializaciones y demás características del prototipo a simular.



Nombre	Valor inicial	Tipo	Dimensión
externalApp		RSimulinkExtern...	
time	0.0	double	
L1	0.15	double	
		double	
l1	0.023	double	
m1	0.17	double	
Pbrazo	0.0	double	
PbrazoNorm	0.0	double	
Vbrazo	0.0	double	
		double	
BrazoA	0.015	double	
BrazoH	0.005	double	
AcopleBra1L	0.015	double	
AcopleBra1A	0.015	double	
AcopleBra1H	0.005	double	
AcopleBra2L	0.005	double	
AcopleBra2A	0.015	double	

Fig. 5. Ventana de variables del entrono gráfico en la interfaz de EJS. Fuente: Autores

2.4.1 Variables

El panel de modelo contiene cinco ventanas de variables, las del brazo, péndulo, motor y estructura tienen como función definir tanto las variables destinadas a la conexión con el modelo de Simulink como las que se usan para la construcción de los objetos 3D que conforman el sistema. Las variables auxiliares son usadas para dar una proporción entre la masa, la longitud del brazo y el péndulo en la medida que el usuario varíe sus parámetros.

Se usa una variable llamada *externalApp* que corresponde al nombre de la variable usada en el subpanel de inicialización para especificar la posición del archivo de Simulink de extensión .mdl, el valor inicial se deja en blanco y en el tipo se especifica que es un archivo *RSimulinkExternalApp*, este tipo de variable indica que la conexión con Simulink es de forma remota.

2.4.2 Inicialización

Corresponde a los métodos de Java usados para la conexión con Matlab/Simulink existen tres tipos variables usadas para el enlace: *de entrada*, *de salida y de parámetros*. Primero se especifica el tipo de enlace usado para la comunicación entre la simulación local existente y el servidor remoto, que para este caso se elige asíncrona, seguida de la dirección IP del servidor, el puerto para la comunicación con JIM y luego el nombre del modelo de Simulink.

Las *variables de entrada* son las que modifican los valores de las variables en el modelo de Simulink. Las *variables de salida* leen los valores obtenidos de Simulink, es decir que, cualquier cambio que se haga a sus variables asociadas en EJS no afectará su valor en el modelo de Simulink. Los *parámetros de Simulink* también pueden obtener o modificar desde EJS, pero sólo se garantiza el nuevo valor en Simulink si el modelo no ha comenzado simular. Es decir, se debe modificar el valor de los parámetros antes de ejecutar el modelo por primera vez, o inmediatamente después de un reinicio de la simulación.

2.4.3 Evolución

Para el control de la simulación de Simulink, se utiliza el método de la case RSimulinkExternalApp:(1), este método indica el número de pasos de integración en Matlab usados para refrescar los valores leídos de las variables de salida de Simulink que estén enlazadas con las variables de EJS. Se usa un paso de integración para obtener cada valor de las variables de forma ininterrumpida.

2.4.4 Relaciones fijas

Ya que cada variable no depende únicamente del tiempo, es decir que la variación de una variable puede verse reflejada en otra, a este lazo se le denomina "Relaciones Fijas", para este caso se definen los métodos de JAVA que permiten rotar las posiciones de los objetos en dos ejes, de acuerdo con los valores leídos del modelo del péndulo y el brazo en Simulink. Estos métodos son:

```
Matrix3DTransformation A = Ma-trix3DTransformation.rotationEJE(B);
_view.C.setTransformation(A);
```

Dónde:

A: Es una variable de tipo double, declarada en este espacio de trabajo, en la cual almacena un vector de posición que está cambiando en torno al EJE elegido (ver Fig. 6), en la medida que B cambia. B: Corresponde a la variable que para este caso se lee de Simulink y corresponde a las posiciones del brazo y del péndulo. C: Es el nombre objeto 3D o grupo de objetos 3D, que se desea rotar en torno al EJE dado.

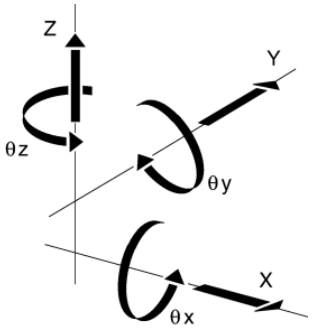


Fig. 6. Ejes de movimiento Rotacional. Fuente: Autores

2.4.5 Propio

Contiene métodos de Java, escritos con el fin de mantener una proporción en los objetos 3D si se varia algún parámetro del modelo, por ejemplo cuando el usuario aumenta la longitud del péndulo, el objeto 3D que lo representa aumenta su largo y mantiene el ancho, lo que en un sólido real representa un aumento de masa proporcional a su longitud, por lo tanto si el objeto virtual aumenta su longitud pero no su masa, no sería comparable con un sólido real.

3. RESULTADOS

El panel *Vista* de EJS está dedicado a la construcción de interfaz gráfica de la simulación, el entorno de simulación para el sistema requiere un diseño simple y a su vez completo y debe darle al

usuario la posibilidad de navegar en él, de forma intuitiva. Para cumplir con esta particularidad se expone la distribución en bloques de la interfaz de simulación (ver Fig. 7), caracterizada por su interactividad con la simulación, visualización dinámica y accesibilidad a variación de parámetros. En este panel de EJS se tiene también la opción de utilizar JAVA3D para darle un efecto más suavizado a los objetos en 3D.

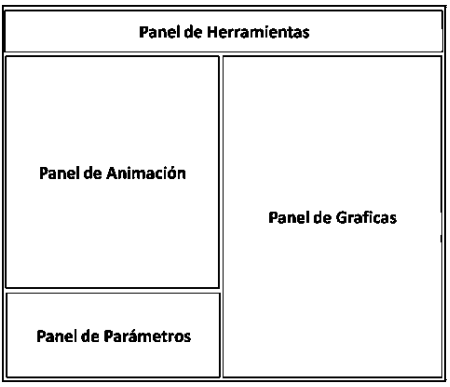


Fig. 7. Distribución de Paneles en la interfaz de simulación. Fuente: Autores

El Panel de Herramientas está compuesto por los botones que permiten iniciar, pausar y reiniciar la simulación del proceso. El Panel de Animación muestra una representación gráfica del prototipo, se construye a partir de elementos en 3D, con el propósito de mostrar el comportamiento de la planta. El Panel de Gráficas muestra gráficamente la respuesta de las variables principales del sistema (Posiciones y velocidades), estos registros se muestran de forma continua en la medida que la simulación transcurre, los cambios en las señales del sistema se pueden dar por acciones del usuario, como variaciones de parámetros con un deslizador o un valor ingresado por teclado en el panel de parámetros.

El Panel de Parámetros contiene los parámetros del sistema que el usuario puede modificar con la ayuda de deslizaderos y/o campos alfanuméricos, se divide en tres categorías a las cuales se puede acceder a través de pestañas, los parámetros del péndulo, brazo y motor, los cuales son los valores de los parámetros propios del modelo matemático estudiado.

Después del diseño de la distribución de los elementos que conforman la interfaz de usuario, se construye la aplicación en EJS como se observa en la Fig. 8, la animación de este sistema se construye con elementos 3D formando una estructura similar a la planta física real.

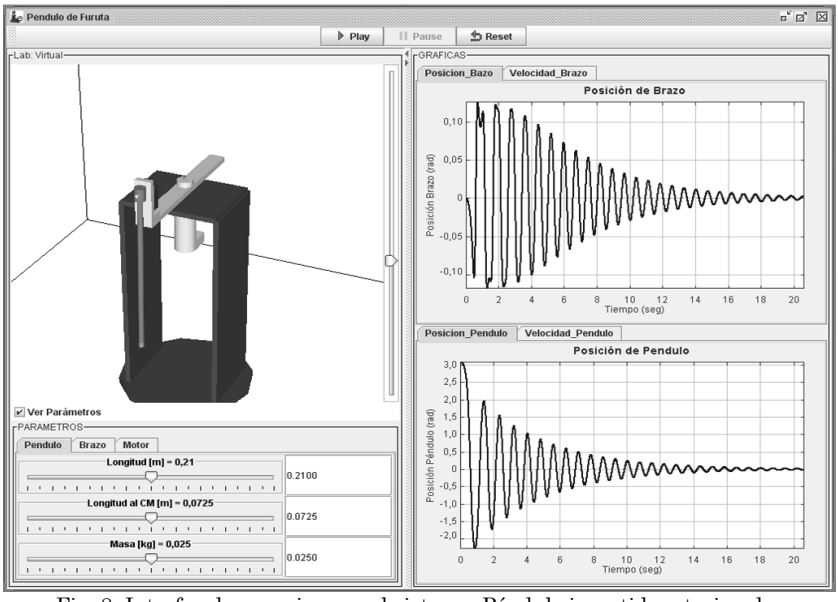


Fig. 8. Interfaz de usuario para el sistema: Péndulo invertido rotacional.
Fuente: Autores

En la Fig. 8 se puede apreciar la interfaz final a la cual pueden ingresar hasta diez usuarios simultáneamente debido a la capacidad del servidor de internet para Matlab llamado Jim. Con esta interfaz del péndulo rotacional es posible validar los modelos según la variación de los parámetros, formando una herramienta pedagógica útil para el estudio de sistemas dinámicos en asignaturas de control automático y control no lineal en el área de ingeniería, de igual forma el grupo de investigación está trabajando en la implementación de la interfaz de usuario para el péndulo invertido sobre carro deslizante.

3.1 Conexión de la Interfaz de Usuario con el Servidor JIM

La conexión entre la aplicación desarrollada con EJS y Matlab/Simulink a través de internet es posible gracias a JIM *(Java Internet Matlab)* el cual es un servidor para Matlab diseñado por DIA-UNED "laboratorio virtual". Este paquete de Software permite conectar un modelo de Simulink con una aplicación de EJS de forma remota. Se aclara que JIM es una aplicación escrita en JAVA que se conecta con una versión de Matlab local y no con Matlab Web-Server. Se utiliza la librería de JMatlink de Müller & Waller (1999) para la conexión entre Matlab y JIM.

La comunicación entre *JIM* y *EJS* es del tipo cliente/servidor y se resuelve mediante la utilización de sockets TCP/IP (ver Fig. 9), el lado del servidor es soportado por *JIM* mientras que *EJS* o la aplicación desarrollada con *EJS* corresponde al lado del cliente y pueden comunicarse mediante enlaces síncrono (Jara *et al.*, 2007) y asíncrono (Farias *et al.*, 2006). La ejecución de una simulación del modelo de Simulink remotamente comprende la realización de todas las etapas o fases de la tabla (ver Tabla 2).

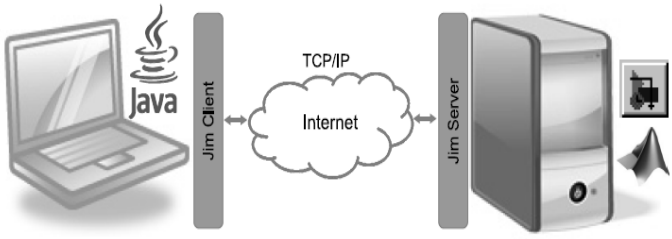


Fig. 9. Protocolo de comunicación remoto JIM. Fuente Farias *et al.*, 2006

Se observa que el enlace síncrono es bidireccional, ya que en cada paso de integración se realizan las seis faces de descritas, por lo que la comunicación fluye en ambos sentidos, por otro lado el enlace asíncrono será unidireccional en la mayoría de los pasos de evolución, presentando una bidireccionalidad solo cuando el usuario interactué con la simulación.

Tabla 2. Pasos realizados para la ejecución de un modelo Simulink remoto. Fuente: Autores

Fase	Acción
1	La simulación envía a JIM los valores de las variables de entrada del modelo.
2	JIM actualiza los valores de las variables de entrada y solicita a Matlab la ejecución de un paso de integración.
3	Matlab ejecuta un paso de integración del modelo Simulink.
4	JIM obtiene del espacio de trabajo de Matlab los valores actualizados de las variables de salida del modelo Simulink.
5	JIM envía los valores actualizados a la simulación.
6	Finalmente, se actualiza la vista de la simulación.

En la Fig. 10 se presentan las fases de la Tabla 2 cuando se ejecuta una simulación remotamente, se resalta que estas faces no dependen del tipo de enlace seleccionado.

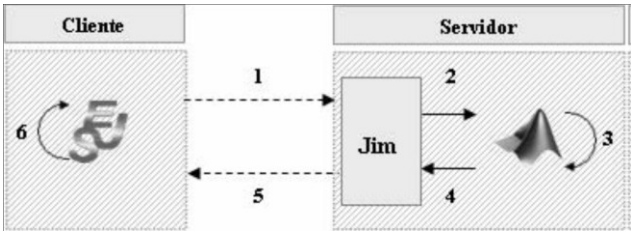


Fig. 10. Fases en la ejecución de una simulación remota. Fuente: Farias *et al.*, 2006

4. CONCLUSIONES

Las nuevas tecnologías de la información y la comunicación permiten crear herramientas de apoyo a la educación en control como los son los simuladores virtuales, en el caso particular de péndulo rotacional, contribuye a la comprensión del comportamiento del sistema a través de una interacción directa con el modelo, mostrando en gráficas y animaciones en 3D de forma instantánea la respuesta a una variación en los parámetros.

El uso de un servidor de Matlab diseñado con Java como lo es *JIM*, le permite extender sus capacidades a través del Internet, sin necesidad de usar otras herramientas que requieren una licen-

cia como es el caso de Matlab Web Server® de Math-Works Inc. Además *JIM* también es multiusuario, lo que permite que varios usuarios interactúen de manera simultánea con el modelo de Simulink alojado en el ordenador del servidor.

La construcción del aplicativo de Java (applet), tiene dos posibilidades para la construcción de elementos 3D; mediante una versión simple de objetos 3D o mediante JAVA3D, el uso del applet con objetos simples de 3D en una página html muestra distorsiones en la imagen al mover el cursor sobre los objetos, mientras que con el uso de JAVA3D no se presenta ningún tipo de distorsiones en la imagen y aporta una visualización con objetos más suavizados.

El uso del servidor para Matlab (JIM), se limita a un máximo de diez usuarios simultáneamente, ya que demanda una capacidad de procesamiento que debe ser soportada por el ordenador (servidor) el cual recibe, procesa y devuelve las peticiones del ordenador del cliente. Esto puede generar retardos en la comunicación afectando la visualización de los resultados.

Para un número de usuarios mayor al soportado por JIM, se puede trabajar con servidores de respaldo, por medio de la misma página html, y/o desde el aplicativo (applet), direccionando a una nueva IP y puerto de comunicación.

5. REFERENCIAS

Åkesson, J., Åström, K. (2001); Safe manual control of the furuta pendulum, in Proceedings 2001 IEEE International Conference on Control Applications (CCA'01), 890-895.

Caipa J., Rodriguez-Diaz, OO; Rodriguez, JL (2010); Modeling, Design And Implementation Control Of Swing Up And Swinging Of An Inverted Rotary Pendulum, Revista Colombiana de Tecnologías de Avanzada, 1(15), 134-141.

Cazzolato, B., Prime, Z. (2008); The dynamics of the furuta pendulum, Technical report, The University of Adelaide.

Domínguez, M. Reguera, P. Fuertes, J.J. (2005); Laboratorio Remoto para la enseñanza de la automática en la universidad de león (España),

Revista iberoamericana de la automática e informática industrial, 2(2), 36-45.

Dormido S. (2008); Desarrollo de laboratorios virtuales y remotos con Easy Java Simulations (Ejs), Departamento de Informática y Automática de la UNED.

Esquembre F., (2005); Creación de Simulaciones Interactivas en Java, Pearson Educación S.A., Madrid.

Farias G., Esquembre F., Sánchez J., S. Dormido, (2006); Laboratorios Virtuales Remotos Usando Easy Java Simulations Y Simulink, XXVII Jornadas de Automática.

Furuta, K., Yamakita, M., Kobayashi, S. (1992); Swing-up control of inverted pendulum using pseudo-state feedback, Journal of Systems and Control Engineering, 206(6), 263-269.

Jara C.A., Candelas F.A., Torres F., Esquembre F., Dormido S. (2007); Comunicación síncrona de simulaciones interactivas desarrolladas con Easy Java Simulations, V Jornadas de Enseñanza a través de Internet Web de la Ingeniería de Sistemas y Automática.

Müller S. & Waller, H. (1999); Efficient Integration of real-time hardware and web based services into MATLAB, Technical report, Ruhr-Universitat Bochum, 11th European Simulation Symposium. Erlangen, Alemania.

Sánchez J., Dormido S., Esquembre F. (2005); The learning of control concepts using interactive tools, Computer Applications in Engineering Education, 13(1), 84-98.