Diseño de un sistema péndulo invertido, sobre plataforma LEGO Mindstorms NXT, controlado mediante MATLAB

Design of an inverted pendulum system on LEGO MINDSTORMS NXT platform, controlled by MATLAB

Fernando Martínez Santa
Ingeniero en Control Electrónico e Instrumentación, Magister en Ingeniería Electrónica y de computadores. Docente de la Universidad Distrital Francisco José de Caldas, Bogotá, Colombia. Contacto: *fmartinezs@udistrital.edu.co*

Cindy Estefany Guerrero Cifuentes
Ingeniera Eléctrica. Ingeniera de diseño eléctrico de Schneider Electric S.A. Bogotá, Colombia. Contacto: *cindy510g@gmail.com*

José David Pérez Ramírez
Ingeniero Eléctrico. Inspector de calidad de mantenimiento a redes de energía, en JMSedinko. Bogotá, Colombia. Contacto: *jdperez2682@gmail.com*

Fecha de recepción: 24 Agosto de 2012 **Clasificación del artículo: Investigación**

Fecha de aceptación: 1 de Octubre de 2012 **Grupo de Investigación: Control Electrónico**

Palabras clave: *Algoritmos genéticos, espacio estados, identificación, localización de polos, péndulo invertido*

Key words: *genetic algorithms, identification, inverted pendulum, pole location, space states.*

RESUMEN

Este artículo muestra el diseño de un sistema péndulo invertido, sobre la plataforma LEGO MINDSTORMS NXT, así como el diseño y la implementación del controlador correspondiente. Como punto de partida se realiza la medición de parámetros físicos necesarios para el modelamiento del sistema en espacio de estados. Adicionalmente, el modelo es identificado por medio de algoritmos genéticos empleando Matlab, donde la adquisición de datos de los sensores y el servomotor se realizan con el toolbox RWTH - Mindstorms NXT. Luego se diseña un controlador usando el método de ubicación de polos, para ser posteriormente implementado en Simulink, entorno desde el cual se ejecuta Embedded Coder Robot NXT toolbox, encargado de la conversión, compilación y transferencia al bloque NXT del controlador. Como resultado se tiene el diseño de una planta física con un kit armable y el diseño e implementación de un controlador viable para dicha planta.

ABSTRACT

This paper shows the inverted pendulum system design using the LEGO MINDSTORMS NXT platform, as well as the design and implementation of the corresponding controller. As a starting point, the needed physical parameters measurement for modeling the system on state space is made. In addition, the model is identified through genetic al-

gorithms using Matlab, when the data acquisition from sensors and servo-motor is done through the RWTH - Mindstorms NXT Toolbox. The next step is the controller design, using the pole placement method, to be subsequently implemented in Simulink, from which the Embedded Coder Robot NXT is executed; this toolbox is responsible of the conversion, compilation and transfer of the controller to the NXT block. The result is a physical plant design using a buildable kit, and the design and implementation of a feasible controller for this plant.

the task of the rescue team. This paper presents the implementation of the Kalman filter to estimate position and to correct errors in the location of a robot.

* * *

1. INTRODUCCIÓN

El péndulo invertido es uno de los sistemas inestables más usado en la teoría del control. En la educación y en la industria tiene diversidad de aplicaciones y formas de estudio [1]. Las implementaciones difieren en la forma de construcción y el tipo de controlador planteado para su desempeño. Teniendo en cuenta que, según sea el diseño mecánico, puede ser vulnerable a agentes como fricción y banda muerta, lo cual lo hace un modelo no confiable [2]. Además sus aplicaciones no sólo se basan en la construcción de prototipos, sino también en la creación de simulaciones gráficas y sus correspondientes análisis [3]. La mayoría de trabajos realizados con anterioridad utilizan diversas piezas mecánicas tomadas de sistemas tales como impresoras o piezas creadas a la medida.

La principal característica de este trabajo es la utilización del kit de robótica "LEGO MINDSTORMS NXT®" como base para el diseño y construcción de la planta física, creando una nueva estructura para el sistema carro-péndulo. Además, hallar el modelo matemático del sistema por medio de algoritmos genéticos, ampliando de esta manera el tema de investigación para la identificación de sistemas dinámicos, mediante algoritmos de inteligencia artificial.

El kit educativo utilizado es de referencia 9797, contiene 437 fichas, 5 sensores, 3 servomotores y el bloque "inteligente" NXT [4]. Adicional a esto, una caja de referencia 9695, la cual contiene 817 fichas de diferentes formas [5]. El bloque NXT cuenta con dos procesadores, el primero es un microprocesador ARM7 de 32 bits, memoria flash de 256 kB, 64 Bits de memoria RAM, y el segundo es un micro controlador AVR de 8 bits, con memoria flash de 4 kB, 512 Bits de memoria RAM. Además, el bloque cuenta con enlace bluetooth, un puerto USB de 12Mbits/s, cuatro puertos de entrada, para los sensores, y 3 puertos de salida para los servomotores [6]. Además, éste es compatible con lenguajes de programación tales como: JAVA, C/C++, LABVIEW, NXT-G, NXC, Matlab y Simulink. Estos últimos programas utilizados para el desarrollo de este trabajo dado su facilidad de uso y análisis de resultados.

La construcción de la planta se realiza con un total de 340 fichas, un sensor de ángulo, un servomotor y el bloque NXT. El sistema péndulo invertido sobre plataforma NXT, se compone de diferentes etapas. La principal es el ensamble donde se tienen en cuenta los parámetros físicos y mecánicos para el diseño. De igual manera se emplean técnicas de identificación, ya sea usando algoritmos estándar o algoritmos de inteligencia artificial, con el fin de obtener el modelo matemático de la planta. Luego se realiza la simulación del controlador, así como la transferencia de este al bloque NXT. Con el sistema en funcionamiento se ejecuta el análisis de resultados y de esta manera se concluye acerca del comportamiento de la planta.

2. METODOLOGÍA

2.1 Ensamble de la planta

En esta sección se hace referencia al ensamble sobre la plataforma LEGO, en el cual se tienen en cuenta los parámetros que constituyen un sistema péndulo invertido. Como primer paso se realizan pruebas para encontrar la ubicación adecuada del servomotor y las ruedas del carro, además de proporcionar equilibrio en el peso del mismo carro y en el péndulo. Como criterio principal de diseño, el sistema péndulo invertido debe permitir la adquisición de cuatro señales como son: posición y velocidad del carro, así como posición y velocidad del péndulo. Inicialmente se instala en el extremo superior del péndulo, un giroscopio HiTechnic NXT, empleado como sensor de velocidad angular, medición proporcionada en [º/s]. De manera práctica se puede usar este sensor para la obtención de posición del péndulo, sin embargo es una medición indirecta, calculada por medio de una integral discreta. Al desconocer la constante de adición para el resultado, este método genera un error el cual se vuelve acumulativo y hace que la medición no sea confiable. Por lo anterior se usa el sensor de ángulo HiTechnic NXT, este mide rotación de un eje de 0 - 359 grados (º) con 1 grado de precisión. Para el ensamble de la planta el sensor se ubica en la base del péndulo midiendo así la posición de este en una trayectoria de 0 – 180 grados (º). Se determina dejar sólo este sensor de ángulo y obtener su derivada, lo cual representa también la velocidad angular del péndulo. En la figura 1, se puede observar la comparación del valor medido por el giroscopio y el valor resultante de derivar la posición angular *(θ)*.

De acuerdo a la comparación con la señal obtenida del giroscopio, la medición indirecta de velocidad angular tiene un error promedio del 9,6% en su amplitud. La forma general de la gráfica de comportamiento es muy similar a la obtenida con el giroscopio; posteriormente se define usar únicamente esta medición indirecta como la velocidad angular del

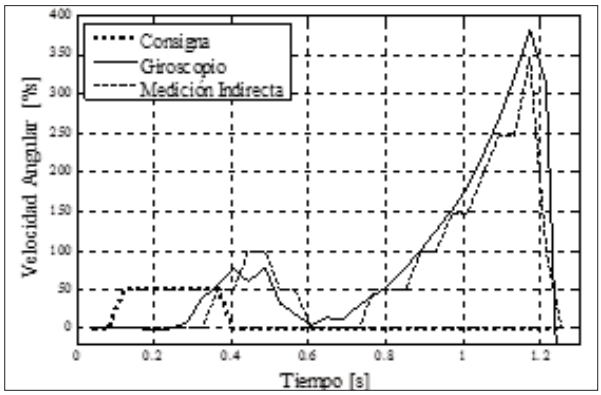

Figura 1. Medición de velocidad angular. Ante un impulso como valor de consigna, se realiza la comparación del valor medido por el giroscopio y el valor calculado mediante la derivada de la posición, medida por el sensor de ángulo.
Fuente: Elaboración propia.

sistema, dado que el cambio entre el uso de esta o la señal directa es mínima en cuanto a la identificación del sistema.

Por otro lado, el servomotor Lego es ubicado en el centro del chasis del carro, permitiéndole a este igual fuerza para el desplazamiento hacia adelante y hacia atrás en las dos ruedas laterales y, de esta manera, las ruedas de los extremos continúan con el movimiento producido por el motor. Este servomotor tiene internamente un encoder, sensor de

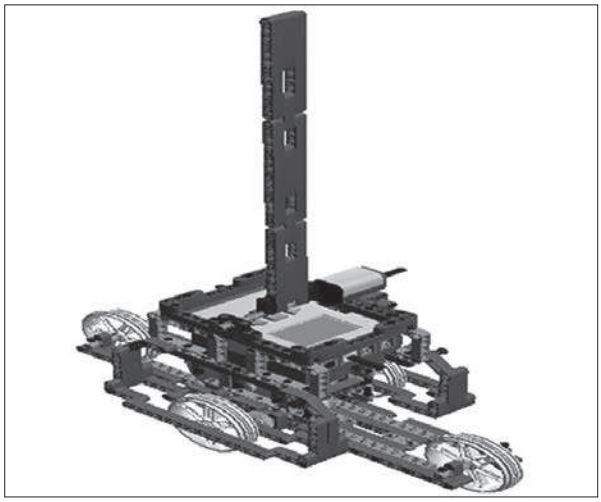

Figura 2. Modelo ensamblado. Sistema péndulo invertido, sobre la plataforma LEGO MINDSTORMS NXT, digitalizado en LEGO digital designer

posición, con resolución de un grado y de 170 rpm para la velocidad de rotación del eje [7]. Aprovechando la estructura del servo motor se pueden adquirir las dos mediciones faltantes, sin embargo, una medición se hace indirecta. La posición del carro *(χ)*, es proporcionada directamente del servomotor, en cambio la velocidad del carro *(χ)*, se obtiene derivando su posición. Al conseguir el diseño final de la planta, esta permite que los sensores envíen información de la ubicación del carro y el péndulo al bloque NXT, y éste consecuentemente por comunicación USB envía los datos a Matlab. En la figura 2, se encuentra el modelo final, ensamblado para el sistema péndulo invertido.

2.2 Modelado del sistema

Un sistema péndulo invertido se considera como un sistema inestable y se hace necesario identificar cada uno de los factores que influyen en su comportamiento y de qué manera lo hacen. Cada una de estas variables, influye en el modelo de la planta, según sea su planteamiento. En la figura 3. Se observan las variables físicas que intervienen en el sistema péndulo invertido de este artículo. En el modelamiento matemático se desarrolla el diagrama de cuerpo libre e implementación de las leyes de Newton para obtener el modelo como función de transferencia y en espacio de estados [8].

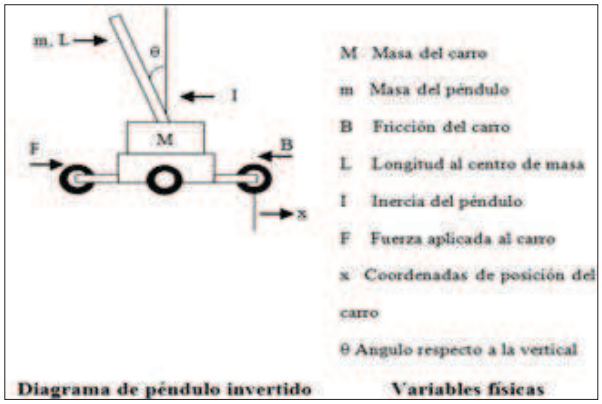

Figura 3. Diagrama del péndulo invertido. Contiene cada uno de los parámetros del sistema; a partir de este se realiza el análisis de fuerzas para obtener el sistema de ecuaciones de la dinámica de la planta y realizar el modelamiento en espacio de estados.
Fuente: Elaboración propia

La interacción con el sistema permite identificar el número de entradas, salidas y estados. Estas variables deben ser expresadas de forma vectorial, donde un sistema de orden n, es separado en n ecuaciones de estado, por lo que siempre va a ser un sistema de orden uno [9]. El modelo en espacio de estados de este artículo, cuenta con cuatro salidas medibles, que son: ángulo del péndulo (θ), velocidad angular ($\dot{\theta}$), posición del carro (χ) y velocidad del carro ($\dot{\chi}$). De manera general un sistema por espacio de estados puede representarse por medio de la ecuación de estado en el dominio del tiempo, como se observa en la ecuación (1).

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t) \qquad (1)$$

Donde

x(t)=Vector de estados

u(t)= Kx, preserva la linealidad y da origen al sistema en lazo cerrado.

De igual manera en la ecuación (2), *y (t)* es la salida del sistema.

$$y(t) = C(x) + du(t) \qquad (2)$$

De las ecuaciones (1) y (2) se puede decir que, no importa el sistema que se modele, ni la complejidad que este pueda tener, puesto que por estar en espacio de estados el sistema va a ser de orden uno. A continuación se muestra el modelo calculado en espacio de estados.

Matriz de estados, *A*

$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -0.2056 & 0.0415 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -0.4771 & 22.836 & 0 \end{bmatrix}$$

Vector de entrada, *B*

$$\begin{bmatrix} 0 \\ 1.1661 \\ 0 \\ 2.7058 \end{bmatrix}$$

Matriz de resultados, *C*

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Vector *d*

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Donde describe la influencia directa sobre la salida [10], se elige igual a cero para que no tenga transición sobre el sistema.

El modelo calculado como función de transferencia, es un sistema de orden 3, como se muestra en la ecuación (3).

$$H(s) = \frac{2,706s}{s^3 + 0,2056s^2 - 22,84s - 4,676} \quad (3)$$

2.3 Identificación del modelo

Para la identificación del modelo, se toma como base el modelo calculado en espacio de estados. Es necesario obtener el comportamiento real del sistema péndulo invertido ante una perturbación y capturarlo en el computador. En este proceso se utiliza el RWTH - Mindstorms NXT, toolbox que cuenta con protocolos de comunicación bluetooth y USB con los robots LEGO MINDSTORMS NXT, más exactamente con el bloque NXT, permitiendo ser controlados y programados desde Matlab. En este artículo la adquisición de datos se realiza mediante conexión USB. Luego de realizar la configuración de la comunicación con Matlab, es enviada una señal de pulso de duración determinada al bloque NXT (ver figura 1), teniendo en cuenta la alimentación del motor en un rango de 1 a 100% de potencia. La potencia seleccionada es 50% como valor neutro para ser multiplicado por la señal de consigna y de esta manera generar movimiento en el carro, y consecuentemente un cambio de posición del sensor de ángulo. Para la toma de muestras el péndulo es ubicado geométricamente a 90 grados (º), pero para el sistema esta posición es un ángulo cero.

Ahora, para la generación de los datos se declara la apertura del sensor de ángulo, la potencia de entrada para el motor y la función para la generación de la señal de consigna. Luego se realiza un ciclo donde se calcula la cantidad de datos según el intervalo de tiempo asignado. En el momento de arranque del carro, el péndulo cae generando una variación en la posición del mismo péndulo, capturada dentro del ciclo por el sensor de ángulo. En la adquisición de datos se calcula el tiempo de muestreo del sistema, valor importante para la identificación e implementación del controlador. El tiempo calculado para la adquisición es de 0,0252 [s]. Valor propio del toolbox y la comunicación USB, que a su vez depende del procesador del PC en el que se esté realizando la toma de datos.

La identificación del modelo se realiza por medio de algoritmos genéticos, método adaptativo basado en el proceso evolutivo de los organismos vivos, para la solución de problemas de la vida real [11]. La herramienta de Matlab utilizada para la identificación del sistema se denomina *"gatool"*. Los algoritmos genéticos optimizan una función objetivo o función de fitness, la cual representa el problema que se quiere minimizar o solucionar. En este caso la función de fitness tiene en cuenta el grado de similitud entre las señales reales (adquiridas por medio del servomotor y los sensores) y cada una de las iteraciones generadas por el algoritmo, es

decir, que dicha función simula el comportamiento de un sistema dinámico con parámetros aleatorios entregados por el algoritmo genético, y lo compara con el comportamiento real del sistema, donde entre más similitud sea encontrada, menor será el valor de salida de la función de fitness, tanto que al minimizar esta función, se escogerá un sistema lo suficientemente cercano en comportamiento al sistema real y así hacer un proceso de identificación.

En el modelo calculado se presentaron 6 variables, estas se encuentran en las matrices y, del modelo en espacio de estados, y corresponden a la posición de los valores no nulos del sistema modelado, esto con el fin de conservar la estructura básica del modelo por espacio de estados original.

Matriz de estados, *AA*

$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \mathbf{v}1 & \mathbf{v}2 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \mathbf{v}3 & \mathbf{v}4 & 0 \end{bmatrix}$$

Vector de entrada, *BB*

$$\begin{bmatrix} 0 \\ \boldsymbol{v}5 \\ 0 \\ \boldsymbol{v}6 \end{bmatrix}$$

La función de fitness se define como el promedio de los errores medios cuadráticos entre las cuatro salidas de los sistemas reales y los obtenidos por el algoritmo, como se muestra en la ecuación (4).

$$f = \frac{(e1 + e2 + e3 + e4)}{4} \qquad (4)$$

Los valores y configuración de parámetros para la estimación del modelo se encuentran en la tabla 1

.

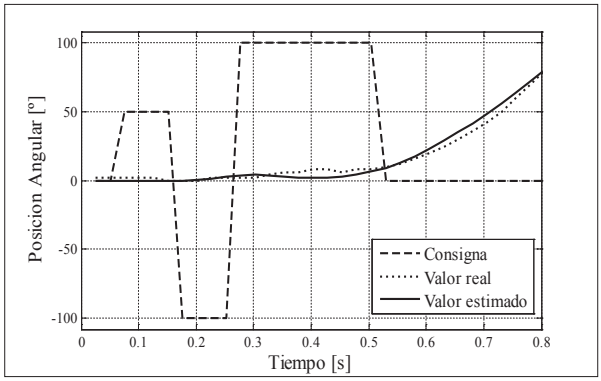

Figura 4. Señal de salida del sensor de ángulo. Se busca que el valor estimado, tenga el mismo comportamiento al valor real del sensor. Esto se realiza por medio de la herramienta de algoritmos genéticos de Matlab, con la cual se logró una respuesta de la función de fitness de 6,05 de error, para este caso.
Fuente: Elaboración propia.

Tabla 1. Parámetros para estimación del modelo por AG.

Población	
Tipo de población	Doble vector
Tamaño de población	20
Rango inicial	[0;1]
Mutación	
Función de mutación	Gaussiana
Escala	2
Criterios de parada	
Generaciones de estancamiento	100

Fuente: Elaboración propia.

Luego de 120 iteraciones el sistema obtenido tiene un valor de fitness de 6.05, valor, que para este caso representa el error entre la señal medida y la señal estimada por medio de algoritmos genéticos, como se observa en la figura 4. Luego del resultado obtenido en la identificación del sistema en espacio estados, este se transforma a función de transferencia, la cual representa el modelo final y se puede observar en la ecuación (5).

$$H(s) = \frac{-7,675s + 179,3}{s^3 + 15,01s^2 - 13,58s - 105,9} \qquad (5)$$

2.4 Controlador

En esta sección se presenta el diseño del controlador, para la sintonización de éste, se usa *sisotool* de Matlab, en el cual se observa la ubicación de los polos del sistema, además de su respuesta. Esta herramienta permite la adición de polos y ceros, tanto como sean necesarios. Para la estabilización del sistema, consecuentemente es calculado el controlador. En la sintonización, el primero de los mejores resultados es un controlador con dos ceros. Sin embargo, a pesar de ser un sistema con un tiempo de estabilización de 2 [s] y una oscilación de 0,25 [ciclos/s], al momento de discretizar el controlador, la señal de control se vuelve inestable. El segundo controlador sintonizado, presenta un polo y dos ceros, con un tiempo de estabilización de 1 [s] y una oscilación de 0,4 [ciclos/s]; se considera un controlador rápido, pero de igual manera al discretizarlo se afecta la efectividad. Por último el controlador seleccionado, cuenta con dos polos y dos ceros, mostrado en la ecuación (6).

$$H(s)=\frac{21,25s^2+204,6s+498}{s^2+36,43s-333,5} \qquad (6)$$

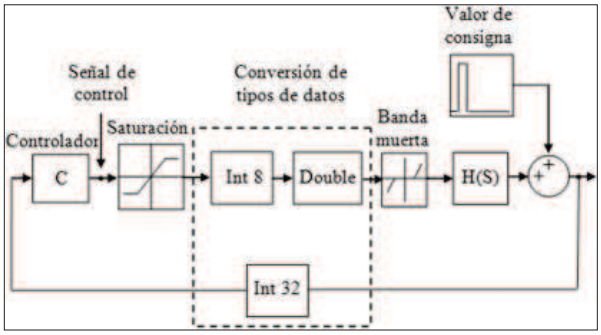

Figura 5. Diagrama de bloques del controlador. Se realiza en Simulink para la simulación del controlador sintonizado.
Fuente: Elaboración propia

El controlador cuenta con un tiempo de estabilización de 2 [s] y una oscilación de 0.5 [ciclos/s], además de un sobre pico de 29,2%. Este último se considera apropiado para el sistema péndulo invertido. En el proceso de diseño del controlador es importante realizar la simulación previa del sistema en el diagrama de bloques de la figura 5, teniendo en cuenta cada una de las configuraciones en la interfaz de diseño y la recepción de datos del sensor y el servomotor.

El controlador se discretiza, ya que ECRobot, trabaja en tiempo discreto. El método de discretización implementado en Matlab es el método *tustin* o trapezoidal, el cual aproxima numéricamente las integrales, es decir, hace la sumatoria de áreas bajo la curva sumando trapecios [12].

Los datos adquiridos del sensor de ángulo son enteros de 32 bits y el servomotor recibe valores enteros de 8 bits. Por esta razón se deben considerar en la simulación esta clase de conversiones: la de tipos de dato. En el diagrama de bloques en Simulink se adiciona una saturación antes del bloque de conversión Int 8 para enviar la señal al servomotor. Este último punto es crítico debido a la señal de control. Cualquier pérdida de datos puede ocasionar fallas al controlar el sistema.

Otro aspecto importante a tener en cuenta es la banda muerta del sistema, este fenómeno se presenta por la inercia que debe romper el motor (actuador), hasta alcanzar el movimiento efectivo del carro. Dicha inercia se debe a la masa del carro y a la relación de trasformación existente por los engranajes. Esta banda muerta se define entre -13% y 13% del valor de la potencia del motor, la cual no es tenida en cuenta en el modelamiento ni en la identificación del problema.

Al determinar el controlador adecuado para la planta, basado en la respuesta del sistema, se procede a realizar el diagrama de bloques del controlador en el entorno de ECRobot, donde el diagrama realizado en Simulink, es convertido a un archivo de extensión *.rxe por medio del compilador GNU ARM GCC, archivo compatible con el bloque NXT. Para esto es necesario instalar una versión modificada del firmware original previamente en el bloque para poder ejecutar el sistema operativo NXTOsek, necesario para la realización del diagrama de bloques de Simulink.

3. RESULTADOS

Como principal resultado se obtiene un sistema péndulo invertido tal como se mostró en la figura 1, con un peso de 820 gramos, 48 centímetros de altura, 36 centímetros de largo, 16 centímetros de ancho y un péndulo de 34 centímetros.

El comportamiento del sistema mejora cuando el tiempo de muestreo es menor, se realizó la identificación en dos equipos, el primero consta de un procesador Intel Core Duo E8400, velocidad del procesador de 3 G.Hz y memoria RAM de 3 GB, con el cual se obtiene un tiempo de muestreo de 0,035 [s]. El segundo tiene un procesador de Intel Core i5-2400 y una RAM de 3.1 G.Hz y memoria RAM de 3,45 GB, con el que se obtiene un tiempo de muestreo de 0,0252 [s], permitiendo adquirir más muestras del comportamiento y dando al sistema y al controlador mejores resultados.

Inicialmente al obtener el modelo calculado en espacio de estados se diseña un controlador por realimentación de estados o control LQR, donde se halla el vector, el cual contiene las cuatro constantes de realimentación, para cada una de las salidas del sistema, como se puede ver en la ecuaciones (7). En la simulación el sistema se estabiliza. Este comportamiento se puede observar en la figura 6. Sin embargo, al transferirse al bloque NXT, la planta

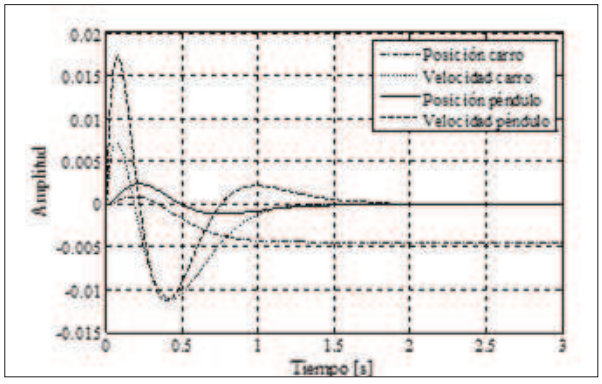

Figura 6. Respuesta del sistema, ante un controlador por realimentación de estados en tiempo continúo.
Fuente: Elaboración propia.

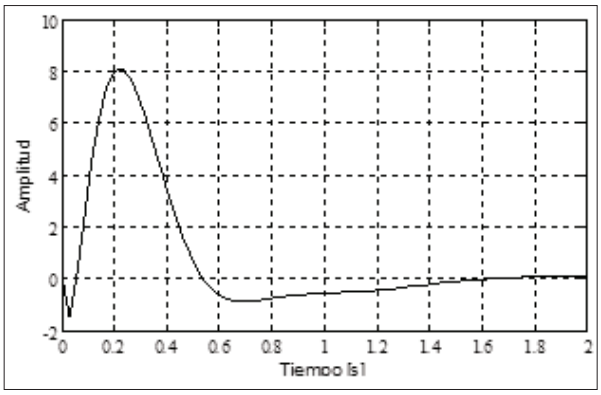

Figura 7. Respuesta de la posición angular, controlada por un sistema de orden dos, hallado por medio de ubicación de polos, ante un impulso en tiempo continúo.
Fuente: Elaboración propia

se hace inestable y no hay respuesta al controlador. Esto puede ocurrir por diferentes aspectos que influyen en el sistema como errores en la medición de los parámetros físicos del sistema, poca exactitud del modelo utilizado, el método de discretización, la medición indirecta de señal de velocidad y la banda muerta. Por estas razones el controlador obtenido puede no ser el apropiado, dado que el sistema con el que fue calculado puede diferir significativamente del real.

$$K = [-44,7214 \quad -28,2393 \quad 94,6142 \quad 19,6361] \qquad (7)$$

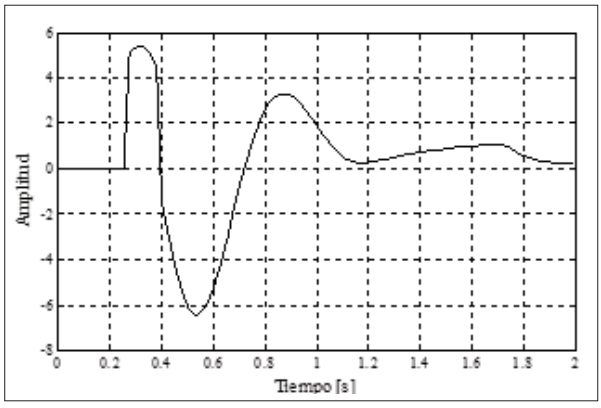

Figura 8. Respuesta de la posición angular ante un impulso, luego de la discretización del controlador y simulado en Simulink.
Fuente: Elaboración propia.

El controlador diseñado para el manejo de la posición angular, sintonizado por ubicación de polos, cuenta con la respuesta en tiempo continuo ante un impulso, éste comportamiento se presenta en la figura 7. Este mismo controlador es discretizado, para analizar su comportamiento antes de transferirlo al bloque NXT. La respuesta del sistema ante el controlador se muestra en la figura 8., donde se aplica un impulso en el entorno discretizado, además de estar simulados todos los componentes del sistema, como los tipos de datos y la saturación.

Al discretizar el controlador sintonizado se cambia de manera significativa su forma. En la simulación sus señales de control se hacen grandes, de valores entre 200 y 300. Si tenemos en cuenta que el servomotor es un actuador de 8 bits y que para su funcionamiento se antepone una saturación de -128 a 127, los valores superiores a éste serán aproximados a los límites del saturador. Entonces, los valores de las señales de control quedan recortados, por lo tanto el controlador no actuaria de manera adecuada sobre el sistema.

4. CONCLUSIONES

El principal error a la hora de implementar el controlador sobre el bloque NXT se encuentra en la saturación que se debe anteponer al actuadorya que este es de tipo Int 8y los valores obtenidos son aproximadamente el doble. Razón por la cual el controlador no logra emitir señales de control al actuador, obteniendo un sistema con una señal contante de 127, lo que hace imposible para el modelo adquirido cambiar por medio del controlador dichos valores máximos.

El código generado por GNU GCC se hace extenso al pasar del entorno Simulink al lenguaje de programación del bloque. Esto también hace que tome mayor tiempo para su ejecución, debido básicamente a la doble compilación generada, de Simulink a C y de este a lenguaje máquina. Por lo tanto la implementación del controlador directamente en lenguaje C podría mejorar el tiempo de ejecución y así el comportamiento del sistema.

Si el sistema fuera establecería posible mejorar la identificación y por tanto los resultados de la función de fitness o en su defecto la forma de adquisición de los datos para un sistema más factible.

Se comprueba que la utilización de un algoritmo genético como método de identificación puede identificar sistemas dinámicos inestables o con señales de entrada y de salida desequilibradas.

Se puede mejorar significativamente el tiempo de respuesta del sistema modificando la planta diseñada básicamente de dos formas: cambiando la estructura de tipo carro a tipo riel, donde la masa de la parte móvil sea únicamente la del péndulo, o haciendo que la acción de control se realice directamente sobre el eje de giro del péndulo. Este último implica cambiar un poco el modelo (segway).

Se puede proponer un compensador para la banda muerta del sistema para hacer más efectivo el comportamiento. Este compensador podría ser utilizado antes o después del cálculo del controlador.

El cálculo de controladores basados en sistemas dinámicos identificados es más efectivo que el realizado con sistemas modelados, porque trabajan directamente sobre datos reales del sistema.

5. TRABAJO FUTURO

Dada la complejidad del sistema péndulo invertido es posible, en trabajos futuros, utilizar control adaptativo y control óptimo, así como diversas técnicas bio-inspiradas por medio de inteligencia computacional como: Algoritmos genéticos, Redes Neuronales, entre otros.

Para trabajos futuros se propone crear una metodología completa para la identificación de sistemas dinámicos utilizando algoritmos genéticos y, a partir de estos, analizar los resultados obtenidos,

abriendo así, la posibilidad de plantear otros proyectos enfocados a la investigación en este tema.

Es posible seguir utilizando el embedded coder (directamente, sin el uso de ECRobot) de Matlab para generar código C/C++ para otros dispositivos tales como: Sistemas embebidos, DSPs y microcontroladores. De esta forma no se restringiría la investigación al uso del kit NXT.

6. FINANCIAMIENTO

Este artículo surge del proyecto de investigación titulado "Diseño y control de un sistema péndulo invertido, sobre plataforma Lego MINDSTORMS NXT", realizado con el fin de contribuir a la formación del laboratorio de control del proyecto curricular "Tecnología en Electricidad e Ingeniería Eléctrica". El aval y financiamiento de este proyecto fue otorgado por la Universidad Distrital Francisco José de Caldas, Facultad Tecnológica.

REFERENCIAS

[1]. S. Mallo, V. Mazzone, *"Construcción y diseño de Controladores de un péndulo invertido rotante,"* Universidad Nacional de Quilmes, Argentina, 2003.

[2]. F. Castaños, R. Carrera *"Levantamiento y control de un péndulo invertido con un esquema de control reconfigurable,"* Universidad Nacional Autónoma de México, México, 2004.

[3]. J. Beltran, *"Simulación de un péndulo invertido,"* Universidad politécnica de Valencia, España, 2010.

[4]. LEGO Education *"LEGO MINDSTORMS NXT ®,"* [En línea]. Disponible: http://education.lego.com/en-us/lego-education-product-database/mindstorms/9797-lego-mindstorms-education-base-set/

[5]. LEGO Education *"LEGO MINDSTORMS NXT ®,"* [En línea]. Disponible: http://education.lego.com/en-us/lego-education-product-database/mindstorms/9695-lego-mindstorms-education-resource-set/

[6]. M. Silva, *"Construcción y programación de un grupo de robots móviles sobre la base del producto LEGO MINDSTORMS NXT,"* Escuela Politécnica Nacional, Ecuador, 2011.

[7]. L. Pinto, G. Bermúdez, *"Determinación de los Parámetros para el Servomotor NXT LEGO Mindstorms®* con Técnicas de Identificación de Sistema," Universidad Distrital Francisco José de Caldas, Colombia, 2008.

[8]. Control tutorials for Matlab, *"Modeling an Inverted Pendulum by University of Michigan,"* [En línea]. Disponible: http://www.engin.umich.edu/group/ctm/examples/pend/invpen.html

[9]. Notas grupos, *"Espacio estado,"* [En linea]. Disponible: http://gemini.udistrital.edu.co/comunidad/profesores/drairan/frecuencia_2008_II.htm

[10]. D. Rairán, *Análisis de sistemas dinámicos y control PID*. Primera edición, Colombia: Universidad Distrital Francisco José de Caldas, 2007.

[11]. N. Gil, *"Algoritmos genéticos,"* Universidad Nacional de Colombia, Colombia, 2006.

[12]. C. Herrera, R. Romero, *"Controlador digital para una planta de levitación magnética con un grado de libertad,"* Pontificia Universidad Javeriana, Colombia, 2010.