Balancín de dos ruedas con controlador Pololu

Fernando Luis Rodríguez Gallegos
José Antonio Intriago Torres
Ing. Carlos Enrique Valdivieso Armendariz.
Facultad de Ingeniería en Electricidad y Computación (FIEC)
Escuela Superior Politécnica del Litoral (ESPOL)
Campus Gustavo Galindo, Km 30.5 vía Perimetral
Apartado 09-01-5863. Guayaquil-Ecuador
flrodrig@fiec.espol.edu.ec, antoniointriago@gmail.com, cvaldiv@fiec.espol.edu.ec

Resumen

El presente proyecto es un ejemplo clásico de la aplicación de los fundamentos del Control Automático a sistemas que utilizan microcontroladores. Combinación que da como resultado sistemas sumamente eficientes en consumo de energía, espacio utilizado y precisión; gracias al alto rendimiento de estos integrados. Un balancín de dos ruedas representa una planta de control particularmente complicada. Pues debido a su naturaleza sumamente inestable, errores relativamente poco significativos ocasionarían la caída del sistema. Para contrarrestar esto utilizamos precisamente microcontroladores, los cuales gracias a su alta velocidad de reacción y precisión nos permitieron censar continuamente el ángulo de inclinación del mismo. Aplicando a estas mediciones un lazo de control PID nos fue posible finalmente mantener al balancín de dos ruedas equilibrado.

Palabras claves: microcontroladores, sumamente inestable, ángulo de inclinación, control PID.

Abstract

This project is a classic example of the application of Automatic Control fundamentals to systems which require the use of microcontrollers. Combination that generates systems characterized by an efficient power consumption, space utilization and accuracy; due to the high performance of these components. A two-wheeled rocker represents a particularly complicated control plant. Its highly unstable nature causes that relatively insignificant errors could easily cause its crash. Precisely to counter this we used microcontrollers, their high speed response and precision allowed us to continuously measure its angle. By applying these readings to a PID control loop we were able to finally keep the two-wheeled rocker balanced.

Key words: microcontrollers, highly unstable, angle, PID control.

1. Introducción

El balancín de dos ruedas es un proyecto que se ha vuelto muy popular en el campo de la mecatrónica. La idea de un robot móvil tipo péndulo invertido ha surgido en años recientes y ha atraído la atención de los investigadores de sistemas de control de alrededor del mundo. Los robots o los medios de transporte de este tipo son mecánicamente inestables, por lo cual es necesario explorar las diversas posibilidades de implementación de sistemas de control para mantener el equilibrio.

Los principios por los cuales opera el balancín de dos ruedas son la base de los sistemas de locomoción de los robots bípedos. Esta no solo que es una aplicación sumamente interesante de este sistema, sino también una con un amplio campo de desarrollo a futuro.

Aunque el uso de robots bípedos se encuentre aun muy poco difundido, por motivos de costo, la evolución de estos autómatas se está dando a pasos agigantados. Debido por supuesto a la capacidad que poseen de transportarse libremente en el mismo entorno de los humanos.

2. Herramientas

2.1 Herramientas del Software

Estas herramientas cubren dos funciones; la compilación de nuestro código y la simulación del sistema. Para la compilación haremos uso del programa AVR Studio. Se utilizará Proteus para visualizar las conexiones del puente H que se implementará.

2.1.1 AVR Studio

Es el software que utilizaremos para programar al Orangutan SV-328. Para esto nos valdremos de uno de los dos compiladores que este software nos ofrece, el AVR GCC, el cual es un compilador de lenguaje C.

La capacidad de programar en C facilitará bastante la elaboración de nuestro código. Pues se trata de un leguaje de alto nivel, creado precisamente para simplificar el trabajo del programador. Ya que es un lenguaje muy bien estructurado que nos permite acceder a múltiples funciones asociadas a distintas librerías, mismas que son básicamente encapsulados de rutinas utilizadas frecuentemente. [1]

2.1.2 Proteus

Nos valdremos de este software para visualizar el comportamiento del hardware que utilizaremos, una vez se lo cargue con nuestro código. Este nos permite simular los componentes y sus conexiones.

Presenta una interfaz muy amistosa con el usuario, donde cada elemento posee una representación gráfica, en la cual se encuentran resaltadas sus entradas y salidas. [2]

2.2 Herramientas de Hardware

Los siguientes componentes constituyen la parte física de nuestro proyecto. Partimos de su controlador, el Orangutan SV-328, que se encarga de enviar una señal adecuada al motor, en función de las lecturas que recepta el sensor. Luego se describe al acelerómetro MX2125; finalmente se describen los motores y las llantas.

2.2.1 Orangutan SV-328

Es un controlador para robots muy compacto, que usaremos como procesador central de nuestro sistema. Este generará respuestas adecuadas en función de los datos que receptará de los sensores.

Cabe recalcar que esta tarjeta tiene incorporado el microcontrolador ATmega328P, un TB6612FNG, capaz de controlar 2 motores DC y 8 entradas analógicas. En la Figura 1 se destacan los componentes que integran esta tarjeta. [3]

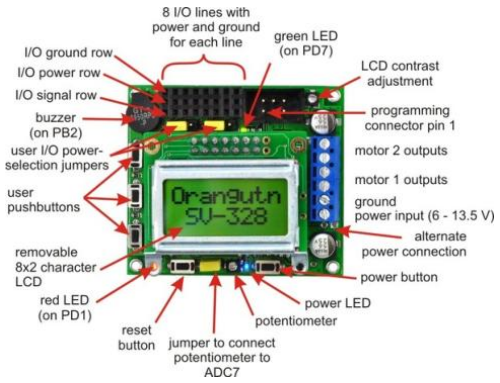

Figura 1. Orangutan SV-328

2.2.2 Acelerómetro MX2125 [4]

A pesar de que este sensor es un acelerómetro, es capaz de generar lecturas de posición angular (característica que se usará en este trabajo). Esto es posible gracias a su particular diseño. En el cual se dispone de una cámara que contiene gas y un elemento calorífico, de tal manera que el aire caliente tenderá a subir al contrario del más frio. Gracias a este principio los sensores de temperatura que este componente contiene permiten estimar la inclinación a la que se encuentra.

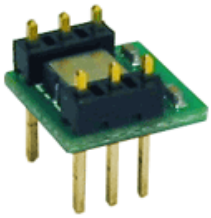

Figura 2. Acelerómetro MX2125

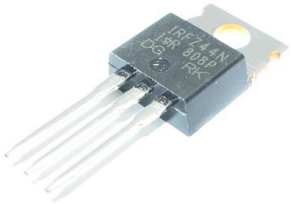

Figura 4. MOSFET IRFZ44N

2.2.3 Motor 19:1 de 37Dx52L mm con encoder 64 CPR

Poderoso motor de 12 voltios con un encoder de cuadratura integrado. Gracias al cual nos será posible determinar la velocidad de nuestro sistema. Posee una prescicion de 1216 conteos por revolución y es capaz de trabajara a 6v mas esta supuesto a trabajar con 12v. En la siguiente vista lateral del motor (Figura 14), se observan los cables que corresponden a la alimentación (rojo y negro) y los del encoder. [3]

Figura 3. Motor con encoder

2.2.4 Llantas 90x10mm

En este trabajo se utilizarán llantas plásticas con neumáticos de silicón, cuyas dimensiones son de de 90x10mm.

2.2.5 MOSFET IRFZ44N

Se utilizan cuatro de estos MOSFETS para armar el puente H que intermediará entre el Orangutan SV-328 y los motores. Puesto que los motores pueden llegar a demandar más corriente que la que los drivers toleraran. Soportan corrientes de hasta 49 A, lo cual excede con creces el tope de 5 A que cada motor puede llegar a requerir.

3. Diseño de la solución

A continuación se presenta el procedimiento que se empleó para encarar este problema. Básicamente nos valemos del acelerómetro MX2125, sensor que envía lecturas respecto a una variable específica a un controlador. Este aplica a las mismas un lazo PID, en función del cual se hará llegar a los motores pulsos (PWM) con el fin de recuperar la posición de equilibrio. El controlador PID a utilizarse es de la forma:

v= $\text{kp} * \text{error} + \text{ki}(\text{error} + \text{error}_{\text{anterior}}) + \text{kd}(\text{error} - \text{error}_{\text{anterior}})$

v= velocidad del motor

kp= constante proporcional

ki= constante integral
Kd= constante diferencial

El error (0° - ángulo actual) será calculado a partir de los pulsos emitidos por el MX2125. Dichos pulsos representan en su duración el ángulo experimentado por este sensor. Cuando este se encuentra en la posición de equilibrio genera pulsos altos de 5ms; por lo cual este valor será una importante referencia. La siguiente fórmula es recomendada por el fabricante para estimar el ángulo de inclinación (en radianes) del MX2125 y por supuesto es implementada en nuestro código: [5]

aceleración= (((duración del pulso en alto / 10.0) - 500) * 8) / 100

10 representa el periodo de la señal (10 ms constantes)

La tarjeta controladora, Orangutan SV-328, dispone del microcontrolador ATmega328P. El cual receptará la lectura del sensor y generará una respuesta acorde. Esta señal generada (PWM) controlará el estado de los MOSFETS IRFZ44N (mismos que integran el puente H) por medio del driver TB6612FNG. Este también viene incorporado al Orangutan SV-328. De esta

manera se controlará la velocidad y sentido de los motores para recuperar la posición de equilibrio.

Debido a que los motores pueden llegar a demandar hasta 5A y el Orangutan SV-328 provee solo hasta 3A, nos valdremos de un arreglo de MOSFETS (puente H) para precautelar la integridad de la tarjeta controladora. De tal forma que los pulsos (PWM) emitidos desde el Orangutan SV-328, controlarán el estado del puente y este a su vez será el que alimente de manera directa a los motores.

Finalmente, nuestro sistema será energizado con una fuente de 12V, a excepción del acelerómetro MX2125. Pues este requiere una alimentación de 3.3V a 5V y consume menos de 4mA. Por lo cual utilizaremos uno de los pines Vcc del Orangutan SV-328, pues estos generan 5V y son capaces de proveer hasta 100mA.

4. Simulaciones y pruebas

4.1 Simulación del puente H

Con el fin de facilitar la visualización del puente H empleado se presenta a continuación la representación del mismo en Proteus:

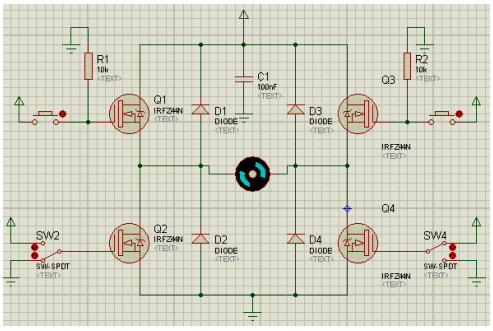

Figura 5. Puente H en “avance”

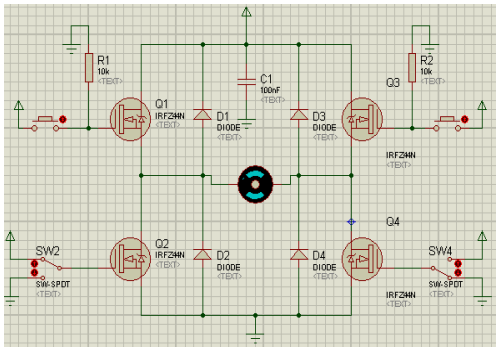

Figura 6. Puente H en “retroceso”

Nótese que los MOSFETS de la parte “baja” son alimentados por switches, mientras que los de la parte “alta” por botoneras. Lo cual es muy semejante a la implementación real de nuestro sistema, pues la parte baja recibe una señal constante (alto o baja) de los pines PD0 y PD1 del Orangutan SV-328. La parte baja en cambio es excitada por pulsos (PWM) generados en la tarjeta. Esto se hizo por supuesto con el fin de evitar problemas de desincronización.

4.2 Implementación

Presentamos a continuación el balancín de dos ruedas armado y operativo. Mismo que está por supuesto integrado por todos los materiales anteriormente descritos. En la parte superior se encuentran acoplados la tarjeta controladora, el sensor y el puente H.

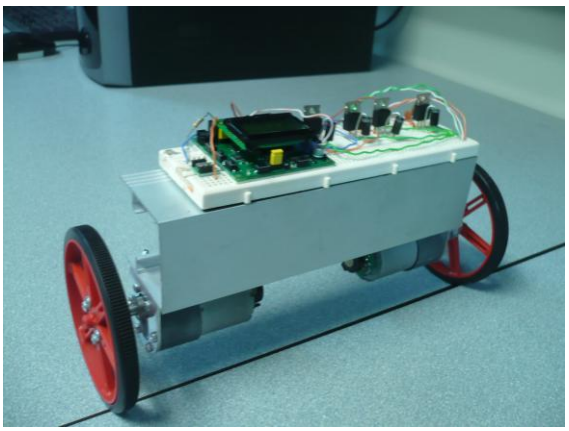

Figura 7. Balancín de dos ruedas con controlador Pololu

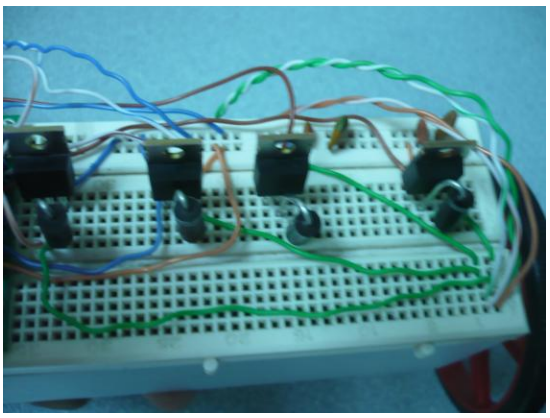

Figura 8. Puente H

Figura 9. Mensaje inicial

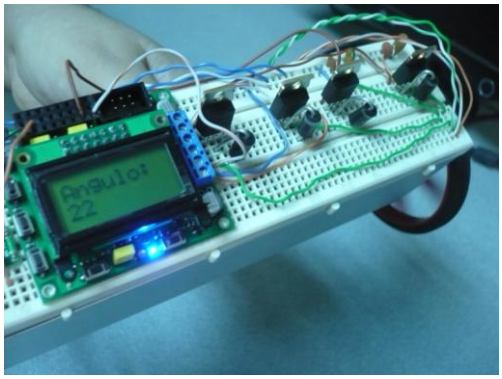

Figura 10. Medición del ángulo

5. Conclusiones

1. El modelo que trabajamos representa una planta de control relativamente complicada. Puesto que la mayoría de sistemas no presentan una sensibilidad tan alta como el nuestro. Es decir son sistemas capaces de mantenerse operativos aun con errores de consideración. Mientras que en el presente caso, literalmente todo el sistema se desplomaría.
2. El uso de librerías de Pololu facilitó bastante nuestro trabajo al momento de programar, pues estas incluyen funciones que miden la duración de pulsos y controlan la velocidad de los motores. No obstante para su correcta implementación se requiere analizar minuciosamente la estructura de las mismas.
3. La tarjeta Orangutan SV-328 fue un gran apoyo para la culminación de este proyecto. Pues como se menciona anteriormente no solo que nos permitió trabajar con comandos que se ajustaban a las necesidades específicas de este proyecto, sino que también incluye un microcontrolador y un driver para los motores. Los cuales son elementos de vital importancia para nuestro trabajo.

6. Recomendaciones

1. Para un mejor rendimiento del sistema de control se deberían considerar más pulsos pasados. En este caso solo se consideró el error inmediato anterior.
2. Los motores utilizados pueden llegar a demandar hasta 5A mientras que el Orangutan SV-328 provee hasta 3A. Por lo cual, bajo ninguna circunstancia estos dos elemento deben interconectarse directamente.
3. Al momento de determinar experimentalmente las constantes kp y ki. Es recomendable empezar por la proporcional manteniendo ki en 0. Una vez se encuentre una constante que mantenga el sistema con un cierto equilibrio, se debe proceder a variar el valor de ki para refinar la estabilidad del sistema.

7. Agradecimientos

A Dios por habernos dado la vida y la voluntad para esforzarnos tanto, a nuestras familias y amigos que supieron estar en el momento adecuado dando palabras de aliento.

8. Referencias

[1] Cornell University, Características del AVR Studio
http://courses.cit.cornell.edu/ee476/AtmelStuff/doc1019.pdf
Fecha de Consulta: *22/04/2011*

[2] Labcenter Electronics, Información referente al programa Proteus
http://www.labcenter.com/products/vsm_overview.cfm
Fecha de Consulta: *24/04/2011*

[3] Pololu Corporation, Datos generales referentes al hardware del proyecto
http://www.pololu.com
Fecha de Consulta: *24/04/2011*

[4] Parallax Inc., Características del acelerómetro MX2125
http://www.parallax.com/dl/docs/prod/compshop/SICMemsicTut.pdf
Fecha de Consulta: *26/04/2011*

[5] Ingeniería de Microsistemas Programados S.L., Análisis de la salida del MX2125
http://www.msebilbao.com/notas/downloads/Acelerometro%20de%202%20ejes%2028017.pdf
Fecha de Consulta: *11/05/2011*