Pendulo Invertido Control PID

D. Carlos Sánchez López, *Member IEEE,* M.A. Villarreal-González,

***Abstract*—Para este proyecto se realizo mediante el uso de arduino el control PID para un pendulo invertido usando 2 PIDs, uno para la posicion del carrito y otro para el balanceo de un pendulo invertido montado sobre un carro robotico de 4 ruedas.Utilizando como sensores de entrada el sensor ultrasonico hc-sr04 y para en angulo el acelerometro MPU6050.**

***Index Terms*—PID Discreto,Matlab,Arduino, Control,Carro Robotico,Ultrasonico,Sensor MPU6050**

I. Introduction

PARA la introduccion de este proyecto es nesecario conocer los principales temas de los cuales hablaremos durante este reporte, por lo cual explicaremos de manera mas detallada cada uno de sus componentes a continuacion.

- PID:Un controlador o regulador PID es un dispositivo que permite controlar un sistema en lazo cerrado para que alcance el estado de salida deseado. El controlador PID está compuesto de tres elementos que proporcionan una acción Proporcional, Integral y Derivativa. Estas tres acciones son las que dan nombre al controlador PID.
- Matlab: MATLAB es un sistema de cómputo numérico que ofrece un entorno de desarrollo integrado con un lenguaje de programación propio.
- Arduino:Arduino es una plataforma de desarrollo basada en una placa electrónica de hardware libre que incorpora un microcontrolador re-programable y una serie de pines hembra. Estos permiten establecer conexiones entre el microcontrolador y los diferentes sensores y actuadores de una manera muy sencilla (principalmente con cables dupont).

Con estos conceptos definidos de manera correcta daremos inicio a la implementacion del Controlador PID para un pendulo invertido.

II. Simulacion

Este reporte se dividira en 2 partes, por una parte sera la simulacion utilizando matlab y simulink para realizar el controlador PID tanto continuo como discreto y ver como este interactuara con el modelo fisico del cual hablaremos mas tarde.

Para comenzar nesecitamos conocer ciertas caracteristicas de nuestro modelo fisico para poder realizar una simulacion acertada de nuestro prototipo. Estas caracteristicas son las siguientes.

Masa del carrito (M)
Masa del contrapeso al final de nuestro pendulo (m)
Longitud de nuestra barra de pendulo (L)
Gravedad (g)

Tecnológico de Monterrey Campus Puebla

```
M=0.510;%kg
m=0.03; %kg
L=0.25;%m
g=9.81;%m/s^2
```

Fig. 1. Valores de caracteristicas del modelo fisico.

Una vez teniendo estos valores con sus respectivas unidades podremos proceder a obtener los valores continuos de nuestro PID.

A. PID continuo

Una vez teniendo las caracteristicas de nuestro sistema realizaremos el PID continuo utilizando Simulink de matlab. Gracias al siguiente esquematico podremos obtener los valores continuos de la constante Proporcional,Integral y Derivativa de nuestro PID haciendo uso de la funcion TUNE que nos ofrece el bloque PID de simulink.

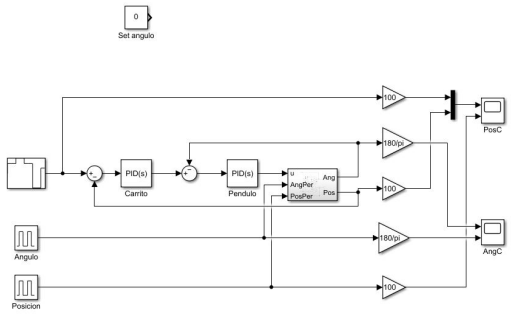

Fig. 2. Esquematico completo de Simulink para PID continuo

Como podemos observar tenemos algunos bloques los cuales fueron definidos por un usuario, por lo tanto revisaremos cada uno de estos bloques para poder realizar el proyecto.

El primer bloque que revisaremos sera el PID del carrito y el PID del pendulo,este bloque es un bloque que podemos obtener por medio de simulink que se llama PID 1dof, lo que nos permite realizar este bloque es sintonizar nuestro controlar PID con valores continuos usando la funcion de tune, lo cual nos regresara los valores correspondientes a cada una de nuestras constantes utilizadas en el PID, Proporcional,Integral y Derivativa asi como tambien el valor de nuestro Filtro que tambien utilizaremos mas adelante para pasar los valores continuos a discretos.

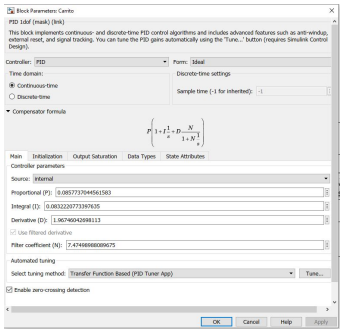

Fig. 3. Bloque PID para carrito

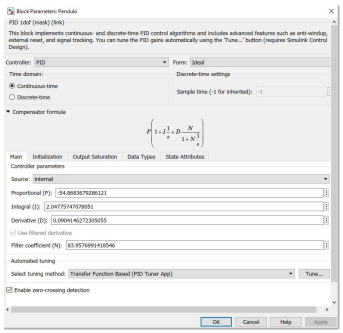

Fig. 4. Bloque PID para pendulo

Como se puede observar en las imagenes yo ya tengo valores para cada una de las variables debido a que ya se ha sintonizado este PID , pero la primera vez que entremos tendremos que realizar el siguiente procedimiento, lo primero a realizar es el PID del pendulo, para esto simularemos que el angulo del pendulo se encuentra en 0 gracias a nuestra variable SetAng que se observa en la figura [2]. realizando la conexion como se muestra en la figura 4, despues de esto procederemos a realizar el tuning de nuestro PID.

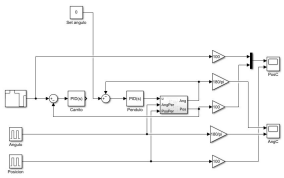

Fig. 5. Conexion en Simulink para realizar sintonizacion de PID continuo pendulo

Una vez realizada la conexion procederemos a dar click en el boton tune de nuestro bloque PID y nos debera aparecer la siguiente pantalla.

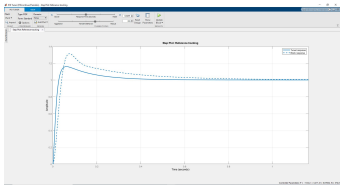

Fig. 6. Bloque tune dentro de PID

Dentro de esta pantalla procederemos a realizar el tuning de nuestro sistema a como sea nesecario dependiendo de nuestra aplicacion. Una vez realizada este tuning daremos ok y regreseramos a realizar el tuning de nuestro segundo PID para el carrito de la misma manera, solo que esta vez desconectaremos nuestro SetAng y volveremos a realizar la conexion como se muestra en la figura 2.

Ahora que tenemos nuestros valores de las constantes para nuestro PID continuo procederemos a revisar el subsistema que se ha realizado para el PID continuo.

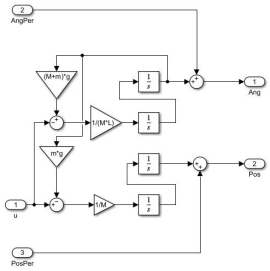

Fig. 7. Bloque subsistema PID continuo

Una vez tengamos estos valores podremos proceder a correr el programa en simulink y haciendo uso del scope podremos verificar nuestros calculos de manera simulada, como podemos observar tenemos 2 scopes uno para la posicion de nuestro carrito y otro para el angulo en el cual se encuentra nuestro pendulo. Con la simulacion bien realizada deberiamos poder obtener valores similares a los siguientes.

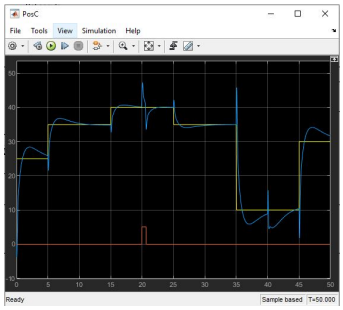

Fig. 8. Simulacion posicion

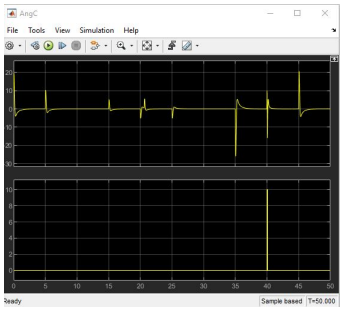

Fig. 9. Simulacion Angulo

Con esto podremos proceder a realizar la simulacion para nuestro PID discreto.

B. PID discreto

Una vez tengamos nuestros valores continuos obtenidos mediante simulink realizaremos la sintonizacion para valores discretos de nuestro PID, para este proyecto se utilizo la forma ideal, cabe recalcar que dependiendo el valor que seleccionemos en nuestro bloque PID continuo podremos elegir entre Ideal o Paralelo para realizar nuestro PID con diferencias en las ecuaciones a utilizar para este proyecto.

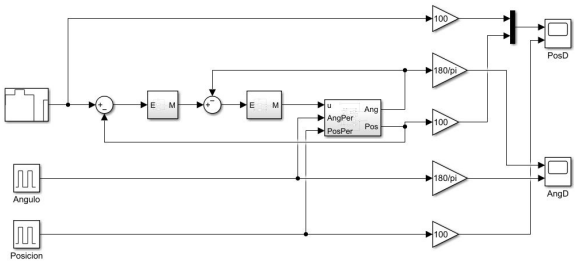

Fig. 10. Esquematico completo de Simulink para PID discreto

Se utilizara el siguiente esquematico para nuestro PID discreto en simulink, donde posteriormente hablaremos de cada subsistema con mas detalle.

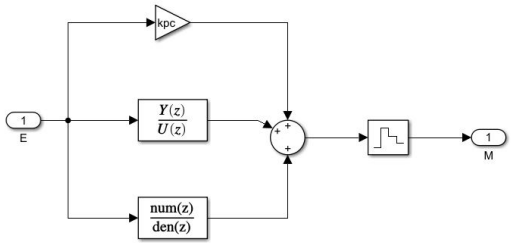

Fig. 11. Bloque de subsistema para cada uno de nuestros PID discretos

Para realizar estos subsistemas de cada uno de los PIDs se utilizaron las siguientes ecuaciones.

Proportional =kpc
Integral =kic*Ts/2*(z+1)/(z-1)
Derivative =2*Nc*kdc*(z-1)/((2+Nc*Ts)*z+(Nc*Ts-2))

Fig. 12. Ecuaciones a utilizar en cada uno de nuestros subsistemas para PID discreto

Una vez se haya configurado de manera correcta cada subsistema procederemos a correr el programa para verificar los resultados en nuestro scope, si todo ha salido bien podremos observar una salida similar a nuestras graficas de posicion y angulo continuo pero esta vez de manera discreta como observamos a continuacion.

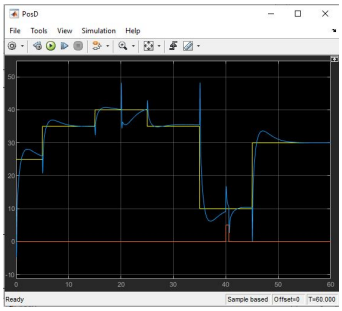

Fig. 13. Simulacion posicion Discreto

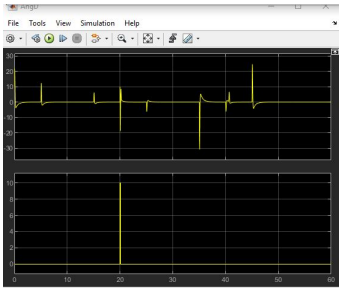

Fig. 14. Simulacion Angulo Discreto

Y con esto dariamos fin a la parte de simulacion realizada en Matlab en conjunto con simulink por lo cual ahora se procedera a explicar el funcionamiento de nuestro codigo arduino asi como el aspecto de nuestro pendulo invertido de manera Fisica.

III. Prototipo Fisico

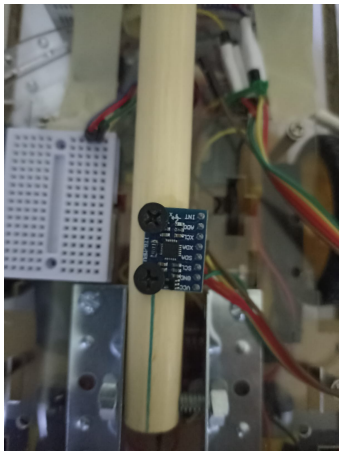

Fig. 15. Acelerometro MPU6050

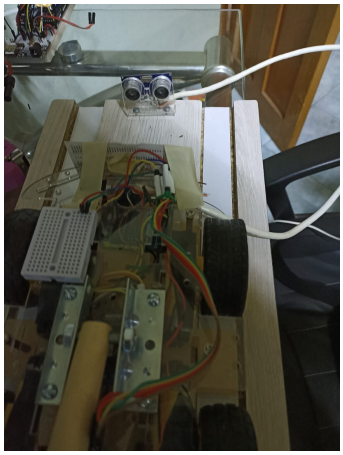

Fig. 16. Ultrasonico Hc-Sr04

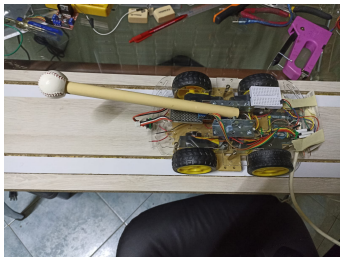

Fig. 17. Carrito con pendulo Invertido

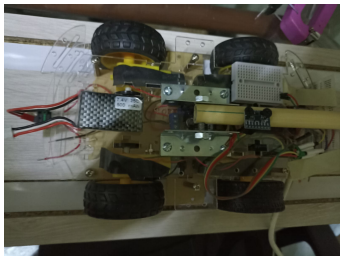

Fig. 18. Carrito con pendulo invertido

Durante la realizacion de este proyecto se tuvieron que realizar ciertos cambios debido a la complejidad de este proyecto, por lo tanto se trato de reducir los problemas lo mas posible , es por esta razon que el sensor ultrasonico fue conectado de manera alambrica en lugar de mandar los datos por Radiofrecuencia entre otros problemas encontrados, pero se encontro que este diseño funcionaba de manera adecuada por lo cual se mantuvo como prototipo final.

A. *Diagrama de Conexiones*

Debido a este siendo un prototipo fisico se implemento un diagrama de conexiones para cada uno de los sensores asi como el uso de un puente H L298N, el cual de igual manera se explicara su funcionamiento.

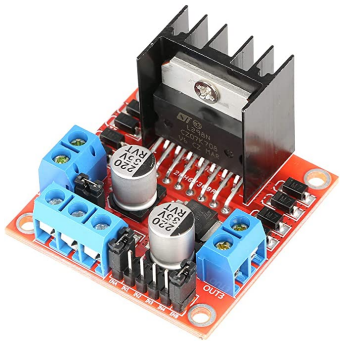

Fig. 19. Puente H L298N

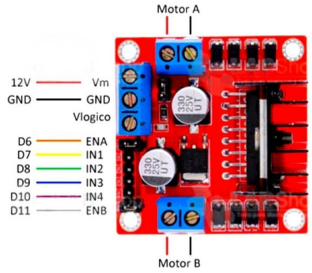

Fig. 20. Diagrama de Pines Puente H L298N

El driver puente H L298N es el modulo más utilizado para manejar motores DC de hasta 2 amperios. El chip L298N internamente posee dos puentes H completos que permiten controlar 2 motores DC o 4 en nuestro caso conectando 2 motores a la misma salida de motor A o B.

El módulo permite controlar el sentido y velocidad de giro de motores mediante señales TTL que se pueden obtener de microcontroladores y tarjetas de desarrollo como Arduino, Raspberry Pi o Launchpads de Texas Instruments. El control del sentido de giro se realiza mediante dos pines para cada motor, la velocidad de giro se puede regular haciendo uso de modulación por ancho de pulso (PWM por sus siglas en inglés).

Gracias a este modulo se pudo mover los 4 motores de nuestro carrito sin problema alguno teniendo como fuente para los motores una bateria LIPO de 7.4v a 500mAh como se muestra en la Imagen 18.

Ahora pasaremos al Diagrama electrico completo de nuestro sistema.

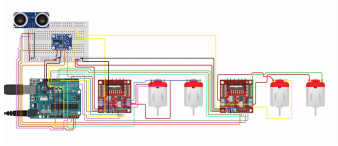

Fig. 21. Diagrama electrico de nuestro sistema

B. *Arduino*

Para revisar el codigo completo favor de seguir la siguiente liga de github donde estara el repositorio completo para este proyecto.

https://github.com/ManuelAmadeoVillarrealGonzalez
/ProyectosUniversidad/tree/main/ControlPIDPenduloInvertido

En este repositorio se encontraran los codigos y simulaciones realizadas en Matlab y Arduino documentados lo mejor posible.

IV. Conclusion

Como conclusion este proyecto fue muy retador debido a su alta complejidad, pero a pesar de esto fue muy interesante de poder realizar ya que reafirmo temas como PIDs tanto discretos como continuos y gracias a esto se pudo realizar el proyecto de manera adecuada. Si bien el pendulo no funciono como esperado se llegaron a resultados muy positivos.

Gracias a este proyecto me fui de mucha utilidad ya que para su servidor la materia de control en general ha sido muy dificil para mi, pero este proyecto me enseño que con dedicacion y empeño cualquier proyecto es posible incluso los mas dificiles.