Péndulo invertido

1ero Lautaro Valentin Cabeza ¿ 2do Luca Sturla 3ero Natanael Robaina 4th Juan Pablo Rodriguez
lautarovalentincabeza@impatrq,com lucasturla@impatrq.com, natanaelrobaina@impatrq,com juanpablorodriguez@impatrq.com

5th Mateo Coria Reartes
mateocoriareartes@imptrq.com

***Abstract*—El proyecto consiste en equilibrar un péndulo en 90° mediante el movimiento de un brazo que está conectado a un motor.**

I. Introducción

Elegimos este proyecto porque consistía principalmente en un sistema de control de una variable, que en este caso era el equilibrio del péndulo. Nos llamó especialmente la atención el sistema de lazo cerrado implementado en el sistema. Durante el proceso de investigación, también descubrimos la aplicación del control PID, lo que nos permitió adentrarnos en este campo.

Originalmente, la idea planteada era levantar el péndulo desde abajo y equilibrarlo. Sin embargo, debido a la complejidad que esto presentaba, nos enfocamos en equilibrar el péndulo desde una posición inicial, que sería la referencia y la posición que queríamos que mantuviera a lo largo del tiempo.

Las principales complicaciones que enfrentamos fueron las imprecisiones en las mediciones del ADC de la Raspberry Pi Pico, ya que, incluso con el péndulo inmóvil, las lecturas fluctuaban significativamente. Durante el desarrollo del código, también nos encontramos con dificultades al implementarlo completamente desde cero y sin valores predefinidos para las constantes del PID. Tuvimos que realizar múltiples pruebas hasta aproximarnos a los valores ideales.

Otras complicaciones que enfrentamos fueron más estructurales. Esto se debió a que las roscas realizadas en las piezas impresas en 3D, con el tiempo, terminaron desgastándose. Como consecuencia, el movimiento del motor no se reflejaba en el brazo, ya que el acoplamiento se deslizaba y perdía efectividad.

II. Descripción funcional

A. Descripcion de circuitos

El circuito se compone principalmente de un potenciometro multivuelta, que funciona como elemento sensor de angulo, es multivueltas especificamente para medir la totalidad de la rotacion del pendulo.

Utilizamos el driver a4988 para poder darle instrucciones al motor, alimentado con 12V, y le seteamos una VREF de 0,48 para poder limitar la corriente por bobina a 0,6A y asi evitar que le llegue mas corriente que la debida al motor.

Usamos una fuente STEP-DOWN para poder alimentar el driver, ya que se uso de fuente un cargador de notebook.

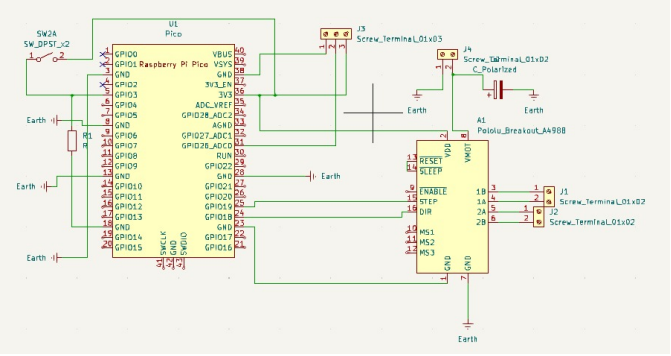

Fig. 1. ESQUEMATICO

Conectamos un pin entre V-BUS y MS1 del driver para poder sacar provecho del microstepping que ofrece el driver, que en ese caso era de 1/2 paso. Los circuitos se pueden ver en la Figura 1

B. DIAGRAMA EN BLOQUES

C. Diagrama de código

A lo largo del codigo, implementamos la formula del control PID con el fin de poder extraer una señal que nos indique el giro y la velocidad con la que se deberia mover el motor:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)\,d\tau + K_d \frac{d}{dt} e(t) \qquad (1)$$

Ademas se incluyo un boton para poder guardar una referencia, ya que en nuestro sensor, los 90° no siempre iban a estar en el mismo lugar. Y con el fin de que el motor no siga girando cuando el pendulo se cae,se implemento un rango de deteccion de caidas, en el que en el caso de estar dentro de ese rango, se detendra todo y se esperara a que se setee una nueva referencia.

No se implemento un promedio de mediciones del ADC ya que aunque mejoraria las mediciones, es mas dificultoso balancear un pendulo que en las que se toma en cuenta mediciones pasadas y no las mas presentes. Figura 3

III. Alcance logrado

Logramos unos segundos de estabilidad, que podían extenderse un poco más con la ayuda de topes. Sin embargo, debido a las problemáticas mencionadas anteriormente, no conseguimos que el sistema fuera constante ni que pudiera

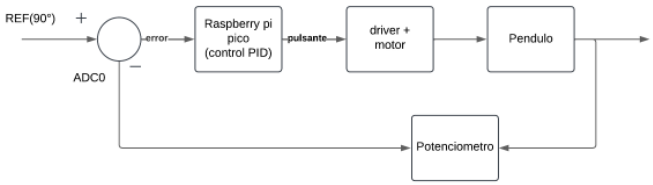

Fig. 2. Diagrama en bloques

- 1ero:Seteamos la referencia a 90°
- 2ndo:El error llega la Raspberry pi pico, que es el encargado de determinar la velocidad del motor dependiendo del error que puede haber provenido del elemento sensor. Esto lo hace con la señal de control proveniente de la etapa del control PID.
  Luego de multiples pruebas, los valores que usamos finalmente fueron:
  - K_p(Ganancia proporcional)=25
  - K_i(Ganancia integral)=0.3
  - K_d(Ganancia derivativa)=0.1
- 3ero: Esta señal de control es enviada al driver a4988, que es el encargado de ,segun la señal de control, darle una velocidad especifica variando el ancho de pulsos que es enviado al motor.Ademas de controlar la velocidad del motor, le da la direccion de giro. El motor usado fue un NEMA 17, que es un motor paso a paso.
- 4to: Todo esto se vera reflejado en el pendulo que es el objeto que queremos lograr un control en su equilibrio.
- 5to: Particularmente el sensor usado es un potenciometro multivueltas.

corregir los errores introducidos al mover el péndulo manualmente. Quizás nos acercamos a los valores óptimos del PID para el péndulo, pero definitivamente no eran exactos.

El control PID logró estabilizar el péndulo durante breves períodos, pero el sistema no fue capaz de mantener la estabilidad durante tiempos prolongados. La imprecisión en las lecturas del sensor y el desgaste de las piezas impresas fueron factores que contribuyeron a la inestabilidad.

IV. Conclusiones

En definitiva, a lo largo del trayecto adquirimos una gran variedad de conocimientos, pero, al reflexionar retrospectivamente, identificamos aspectos que podríamos haber mejorado. Para empezar, podríamos haber incorporado más elementos matemáticos para guiarnos mejor al estructurar el sistema. Además, habría sido útil incluir una referencia externa con un regulador para la Raspberry Pi Pico, lo que nos habría permitido obtener mediciones más precisas del ángulo.

Otro error que cometimos fue en la elección del péndulo. Hubiera sido más conveniente utilizar un diseño en el que el peso se concentrara en la punta, mientras que el resto fuera liviano y uniforme, como una varilla con un peso en un extremo. Esto habría facilitado el control y mejorado el desempeño del sistema en general.

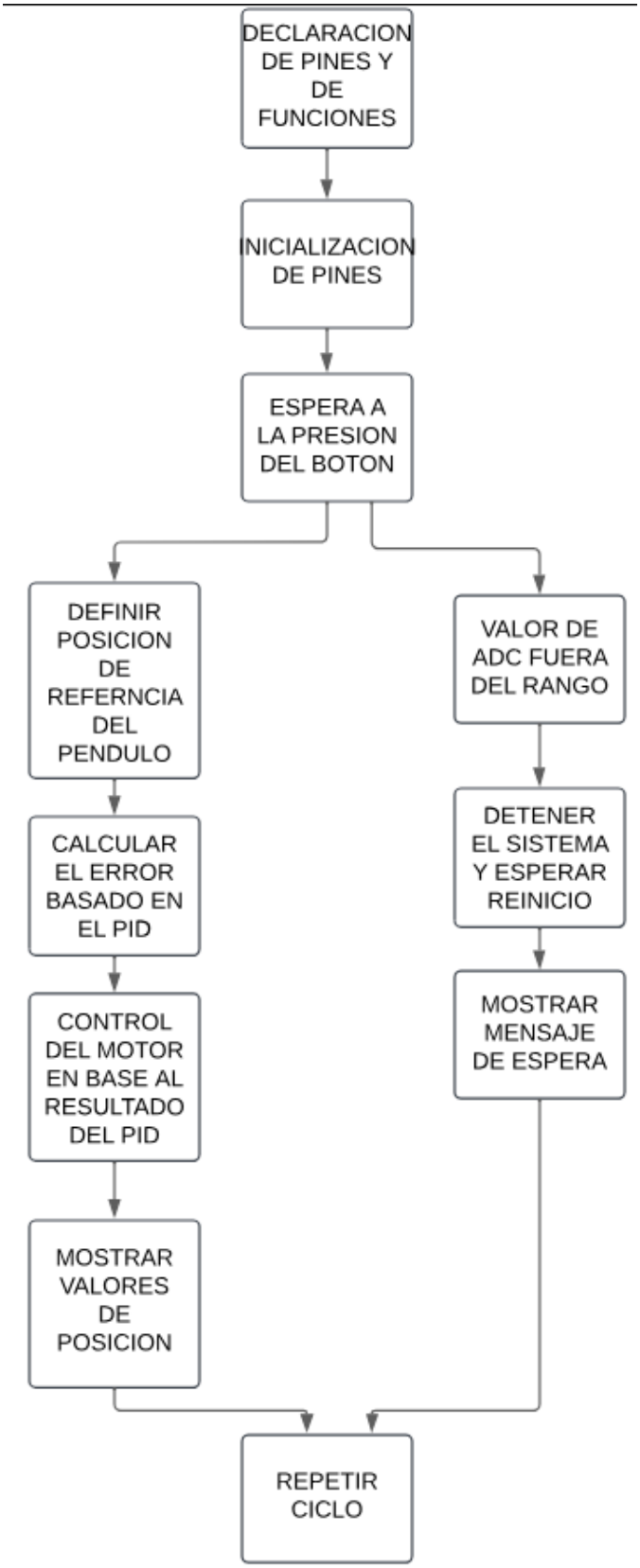

Fig. 3. DIAGRAMA DE CODIGO

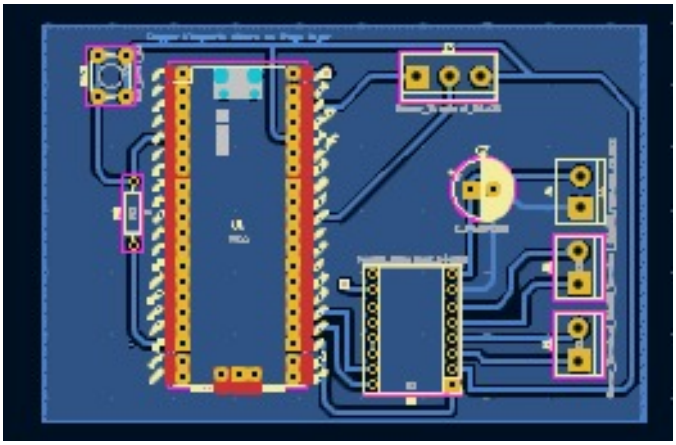

Fig. 4. PCB

Finalmente, a pesar de las dificultades, el proyecto nos permitió aprender acerca de la importancia de los sistemas de control, la implementación de algoritmos PID y el manejo de sensores, lo que representa una valiosa experiencia en ingeniería.

V. Anexos

A. PCB

Figura 4

B. Repositorio del proyecto

Link:https://github.com/JuanPablo2905/PROYECTO-PENDULO/tree/main

C. Lista de materiales

Para la estructura fuero utilizados los siguientes materiales:

- 2x placas de acrílico (20x10 cm)
- 1x lámina de metal (10x10 cm)
- 8x escuadras metálicas
- 1x placa de cobre (10x10 cm)
- 2x planchas de madera
- 2x impresiones en 3D
- 32x tornillos.
- 32x tuercas.
- Cables

D. Lista de componentes

- 1x Capactitor de 100uF x 50v.
- 1x Motor paso a paso NEMA 17 17hs2408.
- 3x Tiras de pines hembra.
- 1x Raspberry Pi pico.
- 1x Driver a4988.
- 1x Boton.
- 1x Resistencia de 10kohms.
- 3x Bornera de 2 pines
- 1x Bornera de 3 pines