PID Ball Balancing System

Final ProjectReport

Fall 2023

Hyder Ochsner - Hyder.ochsner@tamu.edu
Eyad Nazir - Eanazir@tamu.edu

Table Of Contents

Summary and High-level Description

The Ball Balancing PID System project focuses on achieving stabilization and control of a ball on a flat surface through a feedback mechanism. The primary objective is to maintain the ball at a predefined location by continuously measuring its position and adjusting the orientation of the surface. The chosen control mechanism for this project is the Proportional-Integral-Derivative (PID) controller, a widely used feedback controller in industrial control systems. The values calculated by the PID controller are then translated to the angles of the servo motors. Thus allowing for movement of the balancing platform. The project is designed to offer valuable insights into feedback control systems and serves as a practical introduction to PID control, a foundational concept in control engineering.

Background

The concept of dynamic stabilization, exemplified in our PID Ball Balancing project, is a significant leap in control engineering. Drawing inspiration from advanced earthquake stabilization systems used in modern infrastructure, this project aims to dynamically control the position of a ball on a flat surface using a Proportional-Integral-Derivative (PID) controller. The interplay of hardware components, like servo motors, and sophisticated software algorithms, echoes the principles of seismic dampers in skyscrapers. These systems counteract disturbances, such as seismic waves, by adjusting their structure in real-time. Our project, while on a smaller scale, mirrors this intricate balance of mechanics and electronics, and serves as an educational tool in understanding the fundamental principles of vibration control and feedback systems in civil engineering.

Introduction

Purpose

The PID Ball Balancing System, initially developed as a collaborative engineering project, has gained recognition for its educational potential and will be used to provide students with a tool to visualize PID controls, addressing a key engineering problem. By integrating a Raspberry Pi with mechanical components to balance a ball on a tilting platform, the project offers a tangible demonstration of control systems in action. This hands-on approach aids in demystifying complex engineering concepts, particularly in control and feedback mechanisms. As an instructional tool, it not only enhances learning but also encourages innovative thinking in addressing real-world engineering challenges, making it a valuable addition to the educational landscape in engineering and technology.

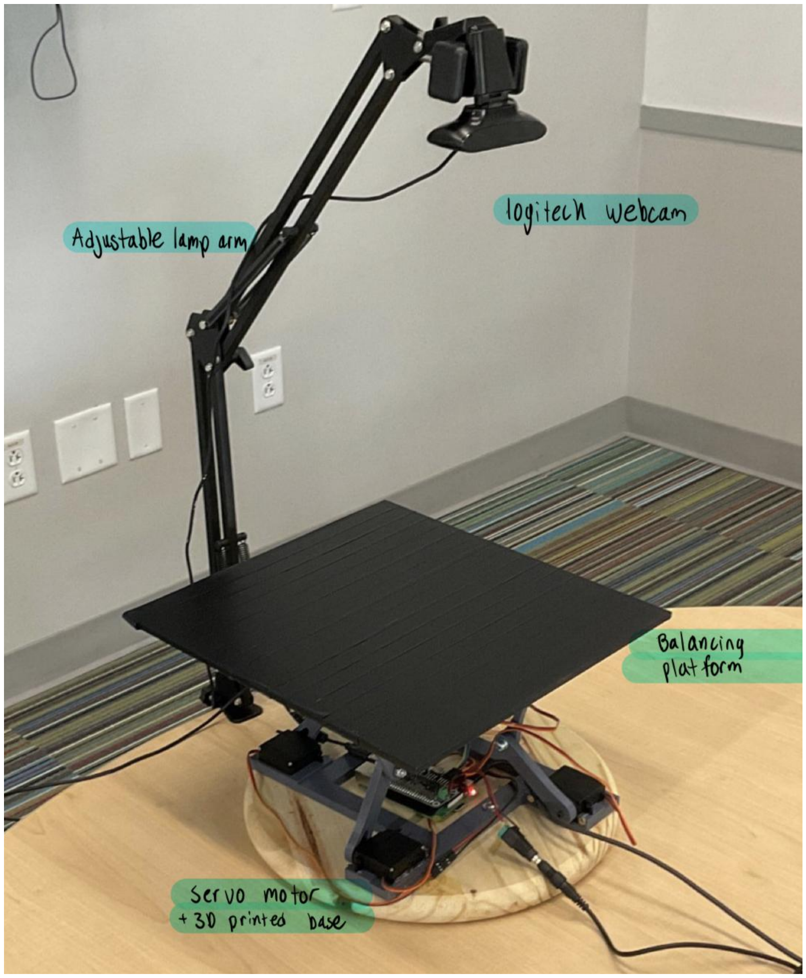
Environment

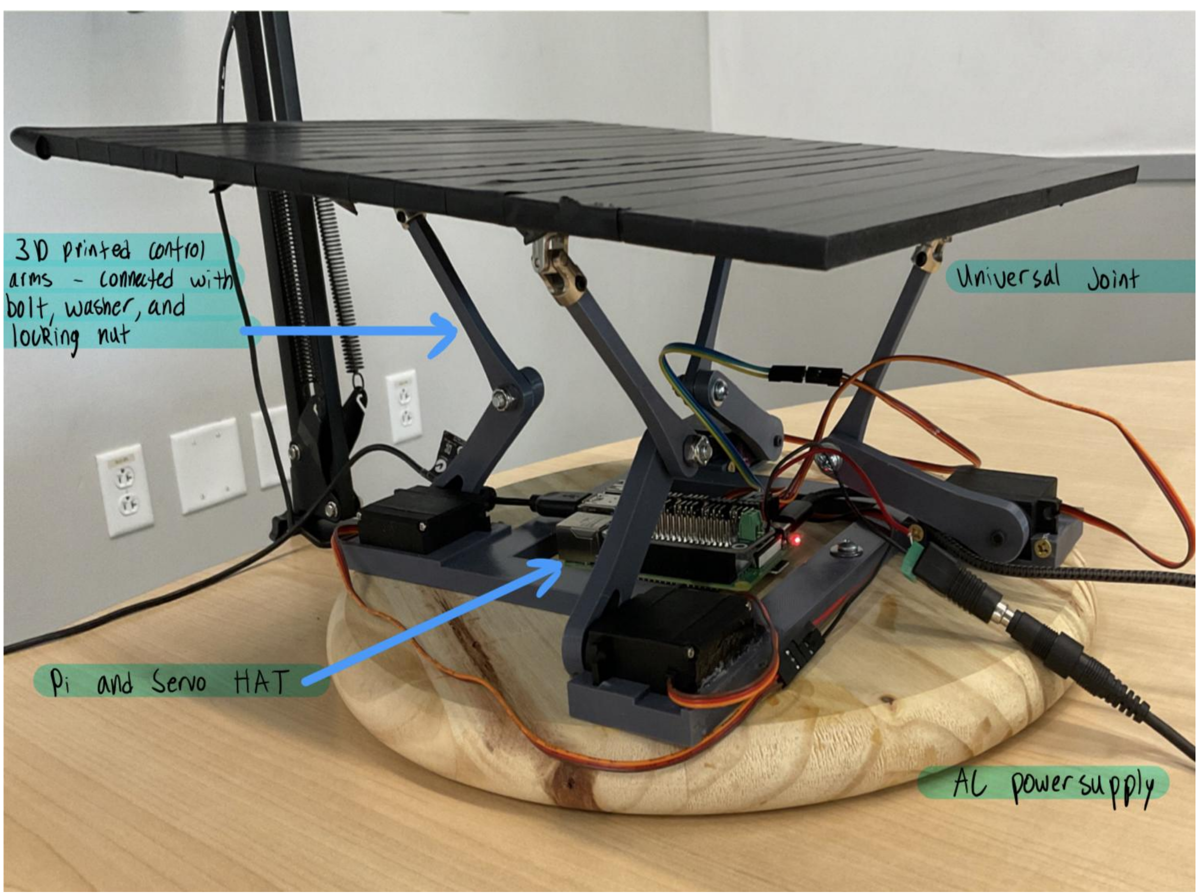
The PID Ball Balancing System integrates mechanical, electrical, and software components within a cohesive environment. Its mechanical aspects, primarily the platform and servo motors, interface directly with the physical world. The system's operation is confined within its structured design, encompassing a Raspberry Pi, sensors, and actuators, ensuring stability and precision in its function.

Constraints

The system has a few limiting factors:

1. The Ball needs not to be thrown too fast as the platform will not have the chance to react due to the 60fps nature of the camera
2. The Ball needs not to be thrown at an excessive speed leading it to fall of the platform very quickly
3. The Ball needs to have a desired weight to ensure it does not roll too fast due to the low friction of the platform


logitech webcam
Adjustable lamp arm
Balancing platform
Servo motor
+ 3D printed base


3D printed control arms - connected with bolt, washer, and locking nut
Universal Joint
Pi and Servo HAT
AC power supply

System Design

Hardware/Mechanical Components

The system's functionality relies on three main components: servo motors for platform control, the balancing platform, and a camera for ball position tracking. To ensure connectivity between these components, various subcomponents are integrated into the system.

1. Servo Motors and Mounting

   The servo motors are mounted on a solid surface to ensure stability during operation. A solid wood plank with a 3D printed bracket is utilized for this purpose. The bracket includes evenly spaced mounts for the four servo motors, each fastened securely to the mount with two screw holes. This configuration eliminates unwanted position shifts of our motors during execution. .

2. Platform Design and Connection to Motors

   The platform consists of a lower arm connected to an upper arm at a movable joint, both 3D printed and fastened together using a bolt, washer, and locking nut. The upper arm connects to the platform through a universal joint, allowing the platform to move while maintaining a stable connection with the arm. This design ensures flexibility and stability in the system.

3. Camera Positioning

   The camera is positioned above the platform to capture the ball's movement fully and is stabilized to reduce variability. An adjustable lamp arm with a clamp is used for this purpose, providing a steady and adjustable position on the table.

In essence, the hardware and mechanical components are intricately designed and assembled to create a robust system capable of implementing the PID control mechanism for ball balancing. Each component plays a crucial role in ensuring the system's stability and effectiveness in maintaining the ball at the desired location on the flat surface.

Software

The software component of the PID Ball Balancing System is a blend of image processing for ball tracking and intricate PID control logic. This combination exemplifies the project's technical sophistication and educational value.

Image Processing and Ball Tracking: The system uses OpenCV for real-time image analysis. It identifies the ball's position on the platform by analyzing video feed from a camera interfaced with the Raspberry Pi. Our code implements functions to recognize circular shapes and filter out irrelevant contours, ensuring accurate ball tracking. This is crucial for determining the ball's deviation from the desired position. In addition we dynamically search for the center of the platform using contours. We then compare the two centers to find the error.

PID Controller: The core of our system is the PID controller algorithm. The Proportional, Integral, and Derivative components are meticulously tuned to achieve optimal system response. We engaged in an iterative process to determine the PID constants, adjusting them based on the system's performance in various test scenarios.

- **Proportional (P) Term:** It calculates an output proportional to the current error (the difference between the desired and actual ball position). We experimented with different values to find a balance that minimizes overshoot yet responds swiftly to disturbances.
- **Integral (I) Term:** This term accumulates the past errors to eliminate residual steady-state error. Finding the right integral constant was crucial to prevent integral windup and ensure smooth corrections over time.
- **Derivative (D) Term:** It predicts future error trends based on their rate of change, providing damping and improving system stability. Our adjustments in the derivative constant aimed to counteract the ball's motion effectively without inducing unnecessary oscillations.

Our code's sophistication lies in integrating these elements into a cohesive system. We programmed the Raspberry Pi to process the image data, calculate the error in the ball's position, and generate control signals for the servo motors. This interactive loop of sensing, computing, and actuating embodies the intricacies of modern control systems.

Design Process

Stage one: Motor Bracket Design

Upon gathering all the required components for our project, we soon recognized the necessity of custom CAD (Computer-Aided Design) parts for optimal functionality. Our immediate focus was on designing a Motor Bracket, a critical component tasked with securing the servo motors firmly to the base. Given the servos' rapid and continuous movement, precision in the bracket's design was paramount to ensure stable operation without any interference.

Our initial prototype of the Motor Bracket revealed a slight miscalculation: the back-side was designed 2 millimeters too short, resulting in an improper fit for the motors. To rectify this, we quickly adapted our design and extended the back-side by 2 millimeters, ensuring a snug and accurate fit. This adjustment was crucial; it highlighted the importance of precision in mechanical design, especially in a system where even minor discrepancies could significantly impact performance. The redesigned bracket was then 3D printed, marking a successful step in the iterative process of engineering and design refinement. We would then have to come back to add it to the base assembly.

Stage two: Motor Arm and Free Arm Design

Motor Arm is the arm connected to the motor with a very small bolt and then connected to that arm is the Free Arm in which it is held loosely to the Motor arm allowing the flexing and movement of the platform easily. Our initial task was to CAD design the Motor Arm for a precise fit with the motor's hole. However, budget constraints precluded us from producing a high-fidelity print with integrated threads. This limitation led to an issue where the motor shaft would rotate, but the arm remained stationary due to the absence of threads. To overcome this, we revised the design to include a cutout for the Horn, which would be attached directly to the motor. The Motor Arm was then designed to mount onto this Horn, ensuring synchronized movement.

Once this prototype was successfully printed, we proceeded to design and print the Free Arm. This part required only a single iteration, featuring holes at each end for attachment to the Motor Arm and a universal joint, respectively.

We initially used a standard nut to connect the two arms with a bolt and washer. However, we encountered a problem where the rotation of the arm caused the nut to loosen and unscrew. To resolve this, we switched to a locking nut, which provided the necessary stability and prevented the disassembly of the bolt during operation. This adjustment was a critical step in ensuring the reliability and effectiveness of our design.

Stage three: Base Assembly

Following the successful verification of our prototyped parts for fit and compatibility, we shifted our focus to the final CAD assembly, integrating the Motor Brackets with the 6-inch by 6-inch square base. The design called for the placement of one bracket on each side of the base, arranged in a clockwise direction for balanced support and operational efficiency. Each bracket, measuring 2-inches in width, contributed to the total dimensions, extending the length and width of the base to 8-inches at its widest points. This careful consideration in the design ensured a robust and stable structure, essential for the smooth functioning of our system.

Stage four: Prototype Construction

Upon acquiring the essential components, including a wooden base, a base for holding the motors, four Motor Arms, four Free Arms, and some universal joints, we commenced the assembly of our prototype. However, during the construction process, we identified the need for a coupler to link the base with the universal joints. This coupler was designed with one circular end to attach to the joint and another end to be adhered to the platform, which we determined should measure 12 inches by 12 inches for optimal balance and space.
The assembly of the prototype involved meticulously fitting all the arms together using bolts and screws. To enhance the structure's stability, we decided to secure the 3D-printed base to the wooden base using two screws. This step was crucial in ensuring that the base remained firmly anchored, providing a solid foundation for the system's operation and preventing any unwanted movement or instability during testing.

Stage five: Circuitry

The circuitry of our system includes: four servo motors and the servo driver HAT. Upon connecting our components and simple testing, we discovered the motors were not acting as expected as the RaspberryPi’s power source did not produce enough current to power the motors and the Pi. As a result, we connected a power source directly to the servo HAT. After experimenting with multiple 9 V batteries, we concluded the need for AC power supply. We obtained a 9 V, 1 A power supply and the motors performed as expected.

Stage six: Software

The software of our system contains two main components: the PID controller and image processing. We began by researching the necessary libraries and determined the ADAFRUIT Servo library and OpenCV would suffice. We first developed the ball tracking software aimed at detecting the center of our balancing platform and the center of the ping pong ball. We faced numerous challenges due to variations in lighting. To overcome these, we added adjustable options within our program like the ability to select the color of the ball being used and a slider to adjust the brightness threshold.

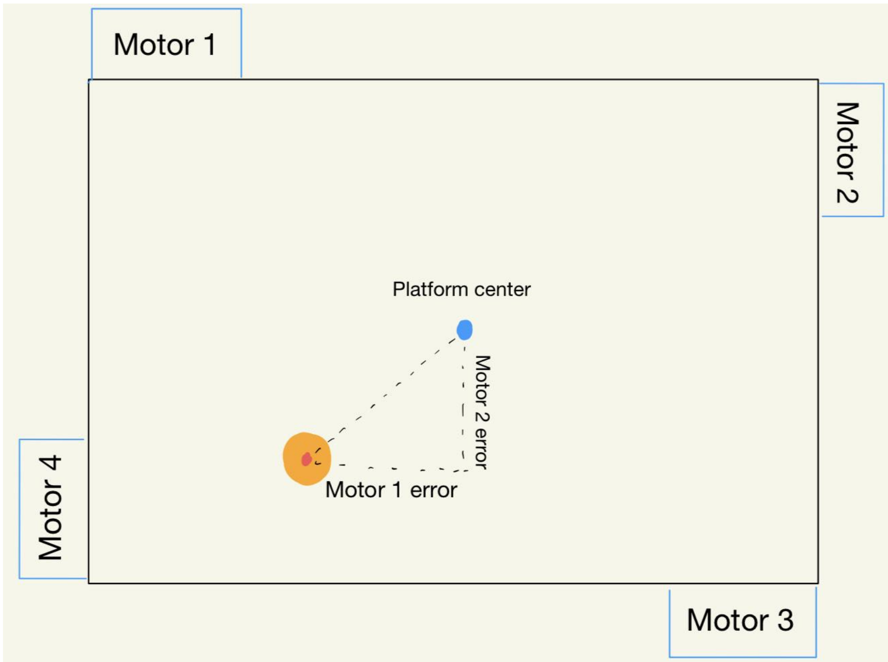
The PID controller required numerous iterations as the system relies heavily on trial and error.

Bill of Materials

Item	Source	Price
(4x) Servo motors	Amazon	$18.39
Servo driver HAT	Amazon	$18.20
(4x) 3D printed Motor Arm	FEDC Design Center	N/A
(4x) 3D printed Free Arm	FEDC Design Center	N/A
3D printed Base	FEDC Design Center	N/A
(4x) 3D printed couplers	FEDC Design Center	N/A
(4x) Universal joints	Amazon	$10.99
Raspberry Pi 3	Provided	N/A
(4x) Washer, locking nut, bolt	Home Depot	~ $6.00
Wooden base	Home Depot	~ $12.00
Sq. foot PVC platform	Amazon	$17.47
Adjustable lamp arm	Amazon	$19.47
Logitech webcam	Amazon	$59.99
9 V, 1 A power supply	Provided	N/A

Appendix (PID Model/Github Link)

- Github link: https://github.com/Eanazir/PID_system



- Error Calculation:
  - **Motor 1 (Top edge)** $Error[1] = Ball_y - Platform_y$: calculates the vertical distance between the ball's current and desired positions. A positive error indicates the ball is below the desired position, requiring the top edge to lift.
  - **Motor 2 (Right edge)** $Error[2] = Ball_x - Platform_x$: measures the horizontal distance on the right side. A positive error suggests the ball is left of the desired position, necessitating the right edge to lift.
  - **Motor 3 (Bottom edge):** $Error[3] = Platform_y - Ball_y$: is similar to Motor 0 but for the bottom edge. A positive error means the ball is above the desired position, signaling the bottom edge to lift.
  - **Motor 4 (Left edge):** $Error[4] = Platform_x - Ball_x$: assesses the horizontal distance on the left side. A positive error indicates the ball is to the right of the desired position, requiring the left edge to lift.

- $PID = K_p \times Error + K_i \times \int Error\, dt + K_d \times \frac{d(Error)}{dt}$
  - **Proportional (P) Term (*Kp*):** Determines the reaction to the current error. If the ball is far from the desired position, a large correction is made.
  - **Integral (I) Term (*Ki*):** Addresses the accumulation of past errors (integral of the error), helping to eliminate residual steady-state error.
  - **Derivative (D) Term (*Kd*):** Predicts future error trajectory based on its rate of change, providing a dampening effect to avoid overshooting.
  - The PID value would be negative when the motor needed to tilt the platform up and positive when the motor needed to tilt the platform down.

Note: *The tuning process is often iterative, and it involves finding the right balance between the three terms to achieve stability, fast response, and minimal overshoot or oscillation. This process relies heavily on trial and error.*

- $New\ Angle = Initial\ Angle + \frac{PID}{Scaling\ Factor}$
  - The New Angle here is calculated based on its initial angle plus the PID number divided by a scaling factor that was obtained by trial and error. This was done for each motor.
  - For our hardware as the angle increased the motor arm went down decreasing the platform level.