Princess Sumaya University for Technology

King Abdullah II Faculty of Engineering

Electrical Engineering Department

Princess Sumaya جامعـــة
University الأميـرة سميّـة
for Technology للتكنولوجيا

Embedded Systems Lab 22448

Model-Based PID Ball Balancing System

Authors:

Qabas Ahmad	20210786	Computer Engineering
Selen Qarajeh	20210622	NIS Engineering

Supervisor:

Eng. Raghad Al-Harasis

June 1st, 2025

Abstract

This project explores the design and implementation of a model-based PID (Proportional-Integral-Derivative) ball balancing system, a classical example of a feedback control mechanism in embedded systems. The system utilizes an Arduino microcontroller to interface with a servo motor, ultrasonic distance sensor, and potentiometers for real-time control parameter adjustment. The primary goal is to maintain a solid ball at a predefined position on a tiltable track by continuously sensing its position and applying corrective action through PID control. The controller is modeled and tuned in MATLAB Simulink, leveraging external mode operation for dynamic parameter tuning and real-time feedback. The ultrasonic sensor provides the current ball position, which is compared to the desired set point to compute the control error. The PID controller processes this error and adjusts the servo motor angle to re-center the ball effectively. This report shows the mechanical setup, embedded software design, PID tuning process, and overall system performance. The results demonstrate successful implementation of a responsive, stable control loop, highlighting the strengths of model-based design in developing real-time embedded control systems.

TABLE OF CONTENTS

1 INTRODUCTION

The objective of this project is to develop a control system that keeps a ball at a specific position on a track using PID (Proportional-Integral-Derivative) control. Real-time feedback is obtained from an ultrasonic sensor, and control commands are sent to a servo motor to tilt the track accordingly. The PID controller is implemented and tuned in Simulink and deployed on an Arduino-based embedded platform.

1.1 OBJECTIVES

- Design a Ball Balancing Platform

Construct a mechanical setup consisting of a track, flexible joints, and a servo-actuated base that allows controlled tilting of the platform to influence the ball's position.

- Develop a PID Controller in Simulink

Design and simulate a PID controller model in MATLAB Simulink that calculates the control signal based on the error between the ball's actual and desired positions.

- Deploy Model-Based Software to Embedded Hardware

Convert the Simulink model into executable code and deploy it to an Arduino microcontroller, ensuring proper interfacing with input sensors and output actuators.

- Enable Real-Time Gain Tuning

Use three potentiometers to allow manual adjustment of Kp, Ki, and Kd gains in real time, facilitating dynamic tuning of the PID controller during system operation.

1.2 THEORY

The core principle behind the ball balancing system is feedback control, specifically implemented using a PID (Proportional-Integral-Derivative) controller. This type of controller is widely used in industrial and embedded systems because of its simplicity, and effectiveness in stabilizing systems.

A PID controller continuously calculates an error value as the difference between a set point (desired ball position) and a process value (actual ball position measured by the ultrasonic sensor). It then applies a corrective action to minimize this error by generating an output signal that controls an actuator—in this case, a servo motor that tilts the track.

The PID output is composed of three terms:

1. Proportional (P) Term:
   It produces an output that is directly proportional to the current error. It provides an immediate corrective response:

$$\mathrm{P} = K_p . e(t) \qquad (1)$$

   Where Kp is the proportional gain and e(t) is the error at time t. A higher Kp results in a stronger reaction to the current error but can lead to overshoot if too high.

2. Integral (I) Term:

The integral term accounts for the accumulation of past errors, helping to eliminate steady-state error:

$$\mathrm{I} = \mathrm{K_i} \int_0^\tau e(\tau) d\tau \qquad (2)$$

This term ensures that any small persistent error over time is corrected, but excessive integral gain Ki can cause instability.

3. Derivative (D) Term:
   The derivative term predicts future error based on its rate of change, damping the system and reducing overshoot:

$$\mathrm{D} = \mathrm{K_d} \frac{de(t)}{dt} \qquad (3)$$

It acts as a stabilizer, particularly useful when the process value changes rapidly.

The combined PID output is:

$$\text{Output} = K_p.e(t) + \mathrm{K_i} \int_0^\tau e(\tau) d\tau + \mathrm{K_d} \frac{de(t)}{dt} \qquad (4)$$

In the context of this project, the PID controller is used to control the angle of the track by adjusting the servo motor's position. The ball's position is measured by an ultrasonic sensor placed at one end of the track. Any deviation from the target position is considered an error and is corrected by tilting the track appropriately:

- Set Point (SP): The desired position of the ball on the track (fixed by the user).
- Process Value (PV): The current position of the ball, measured by the ultrasonic sensor.
- Control Output: The angle command to the servo motor, scaled from the PID output
- Error: The value used by the PID controller to determine how to manipulate the output to bring the process value to the set point.

$$\text{Error} = \text{Set point} - \text{Process Value} \qquad (5)$$

- Gain: The Multiplication Factor, it controls how much effect the PID controller has on the output, and how the controller will react to different changes in the process value.

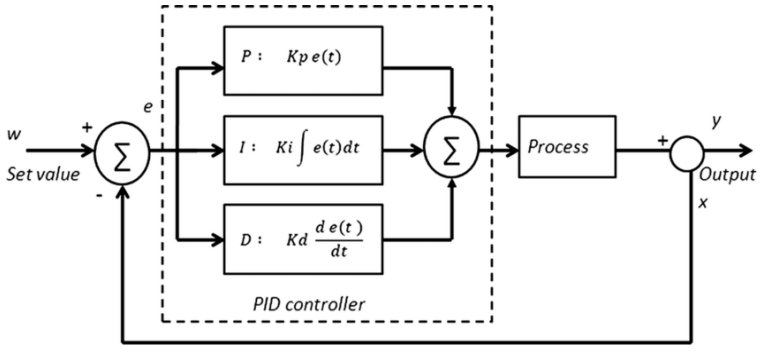


Figure 1 PID Controller Block Diagram

2 PROCEDURE AND METHODS

2.1 HARDWARE SETUP

The mechanical structure was designed with movable joints allowing the track to tilt when actuated by the servo motor. The ultrasonic sensor was placed at one end of the track to detect the distance of the ball in real-time. Potentiometers were installed to adjust PID gains dynamically.

2.1.1 Wooden Base and Foam Board

Using a wooden base as the plate for a PID ball balancing system is important because wood provides a stable and lightweight surface that minimizes external disturbances.

The two platforms connected at 90 degrees allow smooth ball movement along both axes, enabling precise PID control. This setup also ensures the ultrasonic sensor has a clear, unobstructed view of the track for accurate position measurements.

2.1.2 Servo Motor

The servo motors control the tilt of each platform, allowing the system to adjust the ball's position accurately. Their precise and responsive motion is essential for effective PID control and real-time balancing.

Figure 2 Servo Motor Component

2.1.3 Ultrasonic Sensor

The ultrasonic sensor measures the distance to the ball, providing real-time position data. This feedback is crucial for the PID controller to adjust the servo motors and keep the ball balanced.

Figure 3 Ultrasonic Sensor

2.1.4 Potentiometer

The potentiometer provides analog input to manually adjust the system's setpoint or control parameters, allowing real-time tuning and testing of the PID response.

Figure 4 Potentiometer

2.1.5 Arduino Uno

All components are connected to the Arduino Uno, which acts as the central controller. It reads sensor data, processes it through the PID algorithm, and sends commands to the servos.

Figure 5 Arduino Uno

2.1.6 Breadboard

The breadboard is used to make temporary connections between the Arduino, sensors, and other components, simplifying circuit setup and allowing easy modifications during testing.

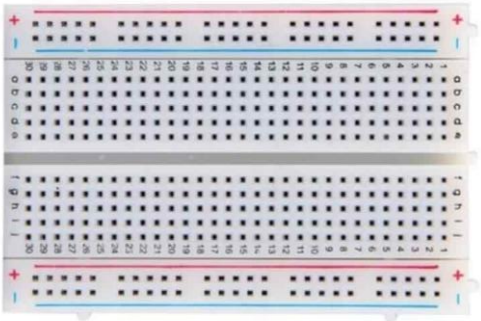


Figure 6 Breadboard

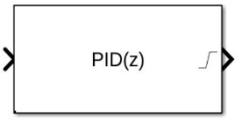
2.2 SOFTWARE SETUP

The Simulink model receives real-time distance data, calculates the error, and applies PID control to determine the required servo motor angle. The PID block operates in external mode to enable real-time gain tuning.

The Ultrasonic Sensor block emits sound waves and measures the time it takes for the echo to return, enabling distance calculation in simulations of robotic or automotive systems.

The Continuous Servo Write block sends a control signal to a continuous rotation servo motor, allowing it to rotate in either direction at varying speeds based on the input value.

The Median Filter block removes noise from signals by replacing each data point with the median value of neighboring samples.



The PID Controller implements an algorithm to regulate output by minimizing the error between a desired setpoint and the actual output.

A limiter circuit using switches and a threshold of 60 is designed to restrict a signal's maximum value. This setup involves a Switch block that compares the input signal to the threshold value.

If the signal exceeds 60, the switch outputs a fixed value otherwise, it passes the input unchanged. This prevents the signal from going beyond the specified limit, protecting downstream components and maintaining system stability.

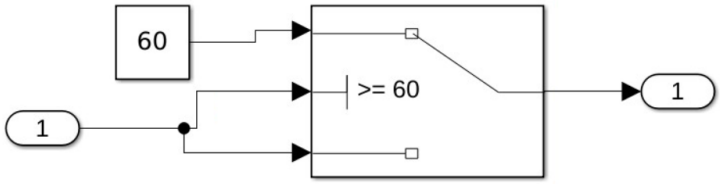
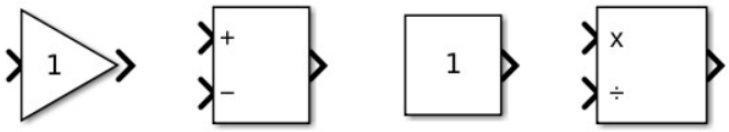


Figure 7 Limiter Circuit

Additionally, the Subtract block finds the error, the Constant sets the target value, gain blocks adjust the PID response, and the Divide block helps with scaling. Together, they process signals to keep the system balanced and responsive.



2.3 3D Model Design

Creating a 3D model of the design helps visualize the entire PID system, including the ball platform, servo mounts, and sensor placement. It ensures proper fit, balance, and wiring layout before physical assembly, reducing trial-and-error and improving build accuracy.

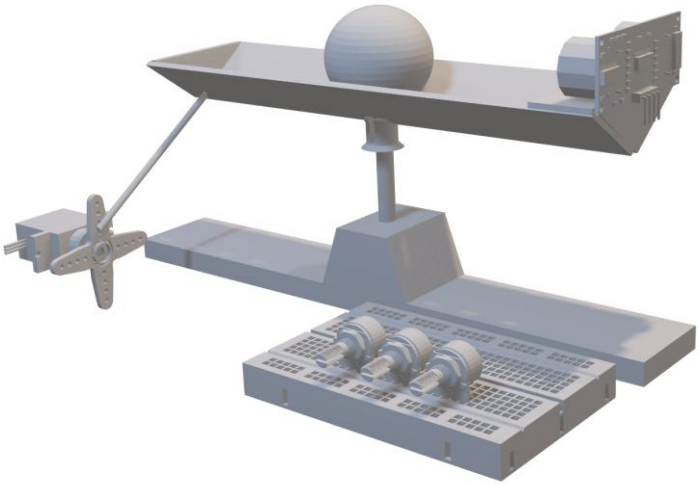


Figure 8 3D Model

3 Results and Discussions

3.1 Hardware Connection

All hardware components were securely mounted on a wooden base, with proper wiring via a breadboard, and powered through the Arduino. The ultrasonic sensor, servo motor, and potentiometers were carefully positioned for accurate input and control.

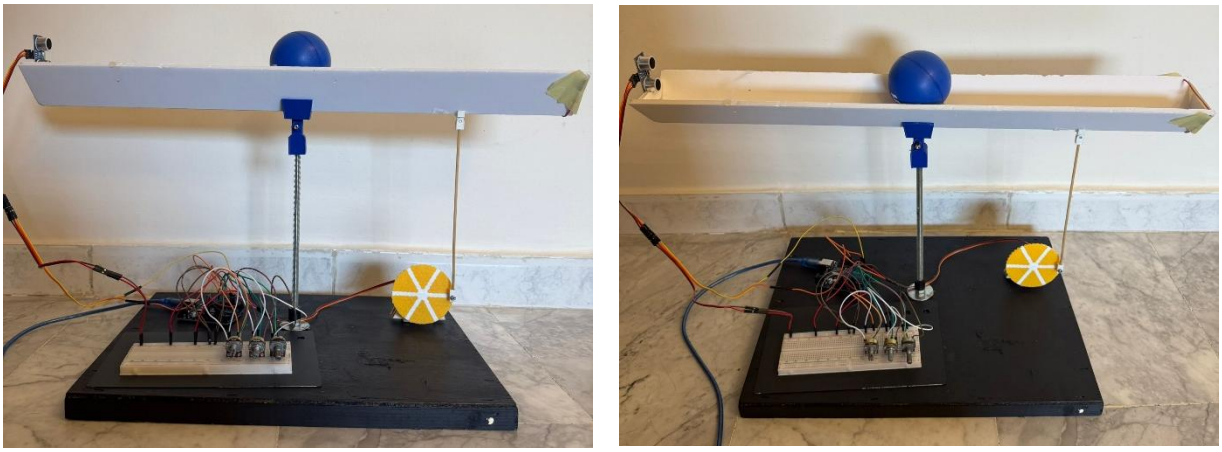


Figure 9 Hardware Setup

The system was tested iteratively, and the PID values were tuned manually using potentiometers to achieve stable and responsive behavior.

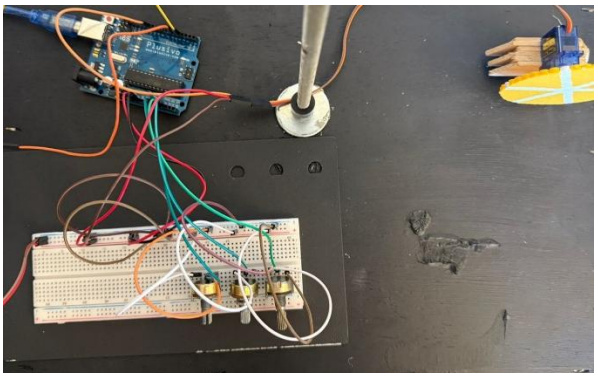


Figure 10 Physical Wiring

3.2 SIMULINK SIMULATION

The following figure represents the control system used for the ball-balancing platform. An ultrasonic sensor reads the distance of the ball, and the signal is first scaled, filtered using a median filter, and limited to valid ranges. The processed value is then compared to a reference (25 cm) to compute the error.

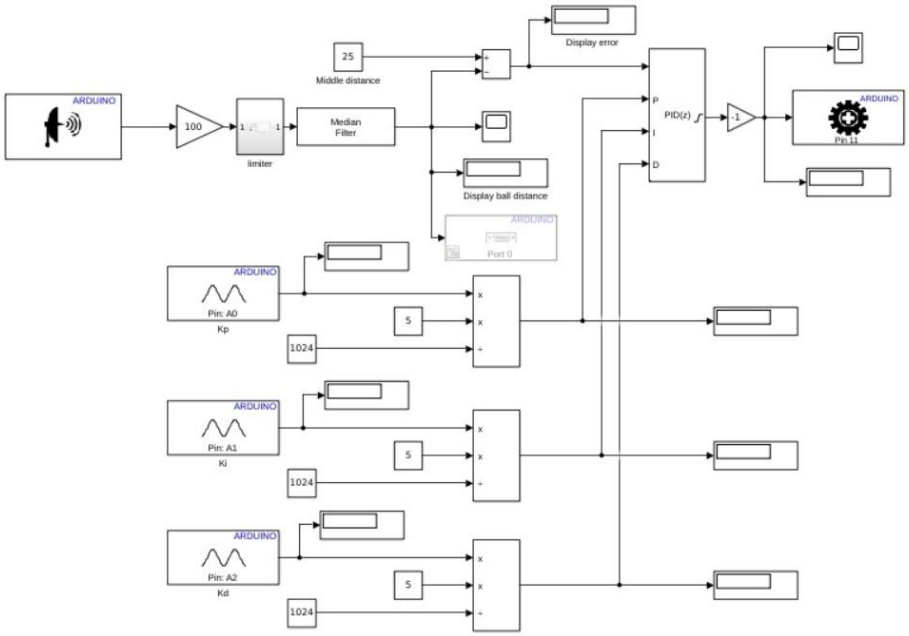


Figure 11 Simulink Ball Balancing

This error feeds into a PID controller, where the proportional, integral, and derivative gains are dynamically set using potentiometers connected to the Arduino's analog pins.

The resulting control output adjusts a servo motor via Arduino pin 11 to maintain the ball at the desired central position. Display blocks are used throughout to monitor the system’s behavior in real time.

The setup for serial communication with an Arduino through COM6 at a baud rate of 9600. Data is read from the COM6 port and then passed to display blocks, allowing real-time monitoring of the incoming values.

Figure 12 Simulink Receiver

To verify the functionality of the system, various figures of the PID control response were plotted, namely the immediate distance obtained from the ultrasonic sensor readings.
By placing the scope at the ultrasonic sensor's output after filtering, the plot showcases the motion of the ball on the track.

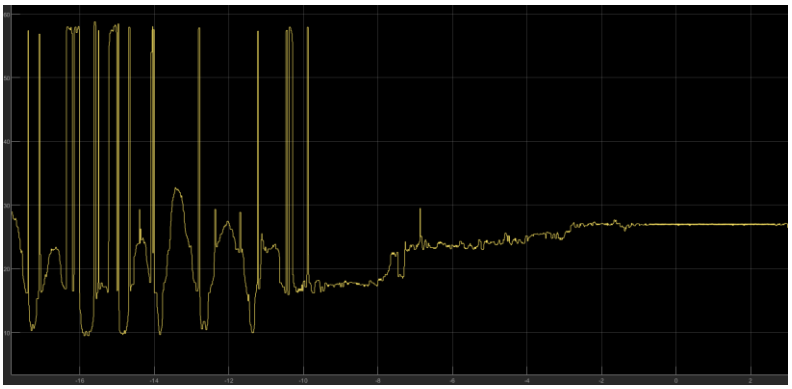


Figure 13 Distance Graph

The oscillations in the distance graph are due to a stopping edge placed near the ultrasonic sensor, which causes the ball to bounce slightly upon contact, preventing it from hitting the sensor directly.

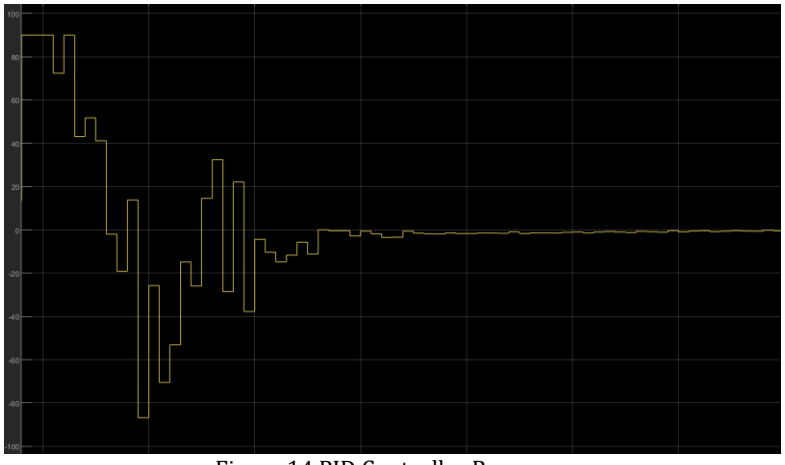


Figure 14 PID Controller Response

The PID controller generates an output ranging from -90 to 90 degrees, which is used to set the position of the servo motor and adjust the platform accordingly.

4 CONCLUSIONS

The project demonstrated a functional model-based PID control system for ball balancing using an Arduino, Simulink, an ultrasonic sensor, and a servo motor. By continuously measuring the ball's position and adjusting the track angle, the system maintained its stability and responded effectively to disturbances. Manual tuning with potentiometers allowed real-time optimization of Kp, Ki, and Kd gains, resulting in fast response and minimal error. Overall, the project met its objectives and provided practical experience in implementing real-time embedded control systems.

5 REFERENCES

[1] "Wikipedia PID Controllers," 2025. [Online]. Available: https://en.wikipedia.org/wiki/Proportional%E2%80%93integral%E2%80%93derivative_controller.

[2] "PID Balance + Ball" 2025. [Online]. Available: https://www.youtube.com/watch?v=JFTJ2SS4xyA&ab_channel=Electronoobs.

[3] "PID Explained," 2025. [Online]. Available: https://pidexplained.com/pid-controller-explained/