Earthquake Stabilization Simulator

Team Members:

- **Hyder Ochsner - System Design Engineer:** (hyder.ochsner@tamu.edu)
  - Responsibilities:
    - Responsible for the overall system architecture, component selection, and ensuring that all components work together seamlessly
- **Eyad Nazir - Software & Control Engineer:** (eanazir@tamu.edu)
  - Responsibilities:
    - Responsible for writing the software that will run on the microcontroller, implementing the PID algorithm, and programming any additional features such as the bouncing patterns

Summary:

The Ball Balancing PID System project aims to stabilize and control the position of a ball on a flat surface using a feedback mechanism. By constantly measuring the ball's position and adjusting the surface's orientation, the system maintains the ball at a predefined location, even when disturbed. The control mechanism employed is a Proportional-Integral-Derivative (PID) controller, a popular type of feedback controller in industrial control systems. This project provides valuable insights into feedback control systems and can serve as a practical introduction to PID control, one of the foundational concepts in control engineering.

Technical Description:

1. Components:

- Platform: A flat surface on which the ball rolls. This can be controlled using actuators to tilt in various directions
- Ball: A spherical object that rolls on the platform
- Sensors: To detect the ball's position. Common choices include infrared sensors, ultrasonic sensors, or cameras
- Actuators: Mechanisms that can tilt the platform in response to control signals. Typically, servos or motors are used
- Microcontroller: A device that processes sensor data, implements the PID algorithm, and controls the actuators

2. Working:

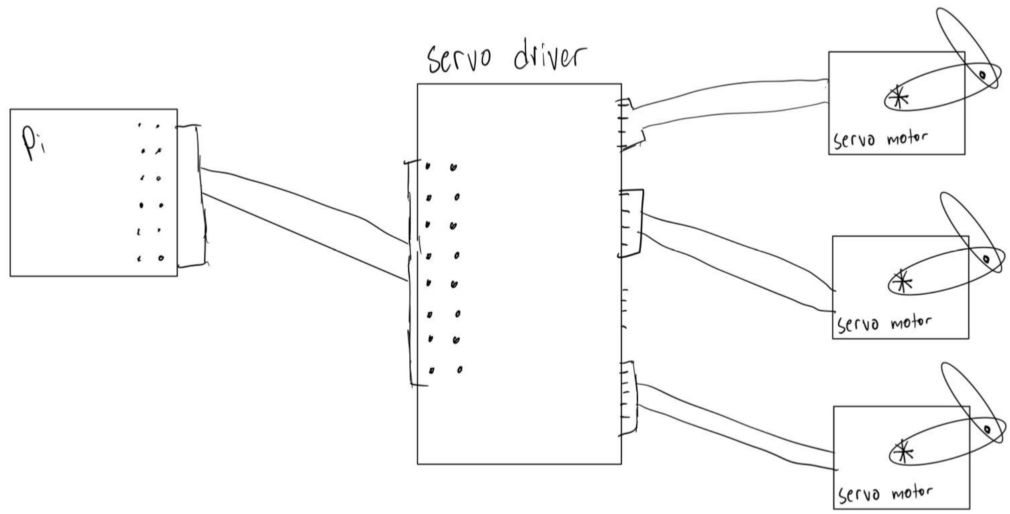
- The sensors constantly measure the ball's position on the platform and send this data to the microcontroller
- The microcontroller calculates the error, which is the difference between the ball's current position and the desired position
- Using the PID algorithm, the controller calculates the necessary adjustments to make to the platform's orientation

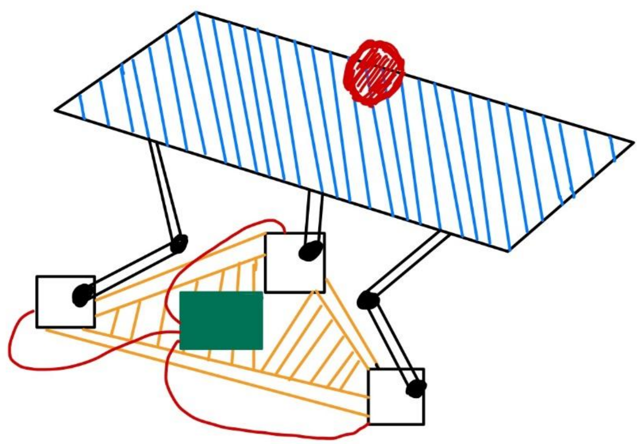
- The actuators adjust the platform's tilt based on the controller's signal, moving the ball towards the desired position

3. PID Controller:
   - Proportional (P): This component produces an output value proportional to the current error. It determines the reaction to the current error
   - Integral (I): Concerned with the accumulation of past errors. If the error has been present for an extended period, it will accumulate (integral of the error), and the controller will respond by changing the control output in relation to a sustained error
   - Derivative (D): This predicts the future error trajectory based on its rate of change. It provides a control output to counteract the rate of error change

4. Features:
   - To design and develop a system that can dynamically adjust its position to counteract disturbances, simulating the behavior of advanced earthquake stabilization systems used in modern infrastructure
   - System will include programmed bouncing patterns which when executed will bounce the ball in a predefined rhythm
   - Sync the bouncing action with music or sound effects. For example, the ball can bounce to the beat of a song, turning the system into an interactive musical instrument

High-level design:





Expected Materials:

- **RaspberryPi (Already Owned)**
- **Camera Extension Cable ($7.42)**
    - https://www.amazon.com/Adafruit-Flex-Cable-Raspberry-Camera/dp/B00XW2NCKS/ref=sr_1_1?crid=1AB8E0FOFQYMO&keywords=Adafruit+Camera+cable+extension&qid=1697483624&sprefix=adafruit+camera+cable+extension%2Caps%2C2121&sr=8-1
- **RaspberryPi Camera ($24.71)**
    - https://www.amazon.com/Raspberry-Pi-Camera-Module-Megapixel/dp/B01ER2SKFS/ref=sr_1_1?crid=20OBBNFZQUO1D&keywords=Pi%2BCamera%2BV2&qid=1697483565&sprefix=pi%2Bcamera%2Bv2%2Caps%2C292&sr=8-1&th=1
- **Servo Driver HAT for Raspberry Pi ($18.25)**

    - https://www.amazon.com/waveshare-Raspberry-16-Channel-Interface-Pinheader/dp/B07XKGHJ8B/ref=sr_1_1?crid=CZ2EL95D3LYG&keywords=waveshare%2Bservo%2Bhat&qid=1697483033&sprefix=waveshare%2Bserv%2Caps%2C354&sr=8-1&th=1
- **Servo Motors (x4 for $14.92)**
    - https://www.amazon.com/diymore-MG90S-Digital-Servos-Helicopter/dp/B07NV7SD3F/ref=sr_1_5?crid=31BRYNLS6DJ11&keywords=diymore%2Bservos&qid=1697482946&sprefix=DIYMORE%2Bservos%2Caps%2C134&sr=8-5&th=1
- **3D Printed Control Arms (~ $10)**
- **3D Printed Flat Plane (~ $8)**

Technical Requirements and challenges:

Mechanical:

- Design a stable base for the servo motors so that they are in a fixed position to minimize error
- Ensure the control arms are securely fastened to the flat platform
- Have the RaspberryPi camera in a fixed position so that there is no variation in repeated tests
- Utilized contrasting colors between the ball and platform so the positioning is easily detectable
- 3d printing the platform and the movable arms, which proposes the problem of working with CAD software and keeping costs down.
- Camera - Have a viable camera that is able to properly recognize the ball movement and transmit data onto PI

Software:

- Utilize OpenCV to track the location of the ball and determine the difference in the current position compared to the center of the plane
- With the calculated errors in the position, pass the necessary output necessary for each servo using the AdaFruit Servo library
- Design a PID algorithm which is capable of performing the outlined features with minimal error

Hardware:

- Wiring the mechanical moving arms to the breadboard might include either soldering or using the mentioned wire casings in class that connect multiple wires
- Bolts will be needed to connect the mechanical arm sections together to allow for flexible movement.

Workplan:

Week 0: Initial Preparation & Team Coordination:

● Objective: Establish communication, understand roles, and set preliminary goals.

Tasks:

1. Meet and understand the roles of System Design Engineer (SDE) and Software & Control Engineer (SCE).
2. Discuss the project's vision, goals, and preliminary design ideas.

3. Set up a shared workspace (online or physical) for collaboration and documentation.

Week 1: Planning, Research & Acquisition (Phases 1 & 2):
● Objective: Set specific objectives and secure all necessary components.
Tasks:
1. Define the project's detailed objectives and desired outcomes.
2. SDE to research and order components: sensors, actuators, microcontroller, etc.
3. SCE to set up the software development environment and begin preliminary research on PID algorithms.

Week 2: Assembly & Initial Software Development:
● Objective: Begin hardware assembly and foundational software development.
Tasks:
1. SDE to assemble the basic platform and install actuators.
2. SCE to develop software for sensor data collection and basic actuator control.
3. Test the platform's response to simple software commands.

Week 3: PID Implementation & Calibration:
● Objective: Implement the PID algorithm and begin calibrations.
Tasks:
1. SCE to finalize and test the PID algorithm.
2. SDE to assist in calibrating the system by adjusting the platform and observing the ball's behavior.
3. Jointly conduct initial system tests and document results.

Week 4: Advanced Features & Debugging:
● Objective: Implement advanced features like bouncing patterns and identify any system issues.
Tasks:
1. SCE to program bouncing patterns and synchronization with music.
2. SDE to adjust and optimize the platform's hardware to accommodate these new features.
3. Identify, document, and resolve any emerging issues.

Week 5: Integration & System Testing:
● Objective: Integrate all system components and conduct comprehensive testing.
Tasks:
1. SCE to finalize the software, ensuring all features work cohesively.
2. SDE to finalize the hardware, ensuring it responds well to software controls.
3. Conduct thorough system tests, recording performance and identifying areas of improvement.

Week 6: Final Adjustments & Documentation:
● Objective: Make final adjustments and document the entire project.

Tasks:

1. Make any final tweaks to the software or hardware based on testing feedback.
2. Document the project, including schematics, code comments, user manual, and potential future enhancements.
3. Conduct a project review meeting to discuss successes, challenges, and learnings.