COMP0212 MODELLING AND SIMULATION

Modelling and Control of the Furuta Pendulum

Muhammad Maaz and Hassan Shahzad

December 2024

CONTENTS

1. INTRODUCTION

1.1. Project Overview

The Furuta Pendulum is a renowned benchmark in control theory and robotics, representing a nonlinear, underactuated mechanical system. Unlike a traditional inverted pendulum that swings solely in a vertical plane, the Furuta Pendulum uses a rotary arm that operates in the horizontal plane while the pendulum swings vertically. This configuration introduces complex, coupled dynamics, making the system a platform for exploring advanced control strategies.

Our primary objectives in this project include:

- Developing a comprehensive mathematical model using Ordinary Differential Equations (ODEs) to accurately capture the Furuta Pendulum's dynamics. This involves deriving the equations of motion, identifying key parameters, and ensuring that the model reflects the physical behavior of the system under various conditions.

- Introducing realistic uncertainties by adding randomness and noise. By simulating sensor inaccuracies, actuator noise, and external disturbances, we can evaluate the controller's performance in scenarios that mimic real-world operational environments.

- Implementing and evaluating two control methodologies:

  1. **Swing-Up Control:** A strategy to transition the pendulum from a stable downward position to the unstable upright equilibrium. This involves designing a controller that can efficiently transfer energy to the pendulum, overcoming the gravitational potential barrier and positioning it for stabilization.
  2. **Linear Quadratic Regulator (LQR):** A control approach to stabilize the pendulum around the upright position with minimal oscillations and control effort. The LQR is designed by linearizing the system around the desired equilibrium point and determining the optimal feedback gains that minimize a defined cost function balancing state deviations and control effort.

- Assessing performance through statistical metrics such as settling time, overshoot, and control effort, alongside sensitivity analyses to gauge the impact of parameter variations and noise. This evaluation ensures that the controllers also maintain performance robustness in uncertainties and operational variances.

1.2. Industrial, Research, and Other Uses

The Furuta Pendulum's versatility makes it useful in various domains:

- **Industrial Robotics:** Used to design and test control algorithms for robotic arms, ensuring precision and stability in automated tasks. By simulating the dynamics of a robotic arm with a pendulum, engineers can develop controllers that maintain stability even under dynamic loads.

- **Aerospace Engineering:** Insights from controlling underactuated systems like the Furuta Pendulum assist in stabilizing spacecraft and satellites. Techniques developed can be applied to attitude control systems, ensuring that spacecraft maintain their orientation during missions.

- **Biomechanics:** Principles derived can be applied to prosthetics and wearable robotics, enhancing human-machine interaction. For example, balancing mechanisms in exoskeletons can benefit from the stability strategies employed in the Furuta Pendulum. This leads to the creation of more responsive and adaptive assistive devices that can improve mobility and quality of life for individuals with physical impairments.

- **Academic Research:** Serves as a foundational model for exploring advanced control theories and nonlinear dynamics. Researchers can use the Furuta Pendulum to validate new control algorithms before applying them to more complex systems. This includes studies on adaptive control, robust control, and intelligent systems that can autonomously adjust to changing environments and operational demands.

1.3. Similar Existing Products

Several educational kits and simulation tools emulate the Furuta Pendulum's dynamics:

- **Hardware Kits:** Available from various educational suppliers, these kits include actuated rotary arms and passive pendulums, complete with sensors and microcontrollers for real-time control experiments. These kits provide a hands-on approach to understanding the dynamics and control of underactuated systems. They often come with instructional manuals and example projects that facilitate experiential learning and experimentation.

- **Software Simulators:** Platforms like MATLAB/Simulink offer simulation frameworks for modeling and experimenting with control strategies. These simulators allow for rapid prototyping and testing of control algorithms without the need for physical hardware. They provide tools for visualizing system behavior, tuning controller parameters, and analyzing performance metrics in a controlled virtual environment.

These products vary in complexity but collectively contribute to a deeper understanding of underactuated system control.

1.4. Significance

The Furuta Pendulum encapsulates key challenges prevalent in modern control systems:

- **Nonlinearity:** The system's equations of motion are inherently nonlinear, requiring sophisticated modelling and control techniques. Nonlinear dynamics introduce complexities such as multiple equilibrium points and sensitivity to initial conditions, which must be addressed to achieve stable control. Understanding these nonlinear behaviors is important for developing controllers that can handle a wide range of operating conditions and disturbances.

- **Underactuation:** With fewer actuators than degrees of freedom, controlling such systems requires innovative strategies to manage energy and stability. Underactuated systems are common in robotics and aerospace, making the development of effective control methods highly relevant. Techniques such as energy-based control, feedback linearization, and optimal control are essential for managing the inherent challenges posed by underactuation.

- **Robustness:** Real-world applications demand controllers that can handle uncertainties and disturbances, making robustness a critical aspect of control design. Controllers must maintain performance despite variations in system parameters and external disturbances. This involves designing controllers that are not only effective under nominal conditions but also resilient to unexpected changes and perturbations.

1.5. Evolution and Similar Mechanisms

The Furuta Pendulum builds upon the traditional inverted pendulum by introducing a rotary base, adding a second degree of freedom and introducing coupling between the arm and pendulum motions. Similar mechanisms and evolved systems include:

- **Reaction Wheel Inverted Pendulum:** Used in spacecraft stabilization, where reaction wheels provide the necessary torque to control orientation. This system mimics the Furuta Pendulum's challenge of stabilizing an underactuated system through controlled torque application. Reaction wheels are important for fine-tuning the orientation of satellites and spacecraft without expending propellant, making them indispensable in space missions.

- **Flywheel-Based Stabilizers:** Employed in satellites and drones to maintain stability and orientation through flywheel dynamics. Flywheels store rotational energy, similar to how the Furuta Pendulum controller injects energy to achieve stability. They offer a means of passive stabilization, complementing active control strategies to enhance overall system resilience and performance.

2. THEORETICAL BACKGROUND

2.1. Schematic Diagram

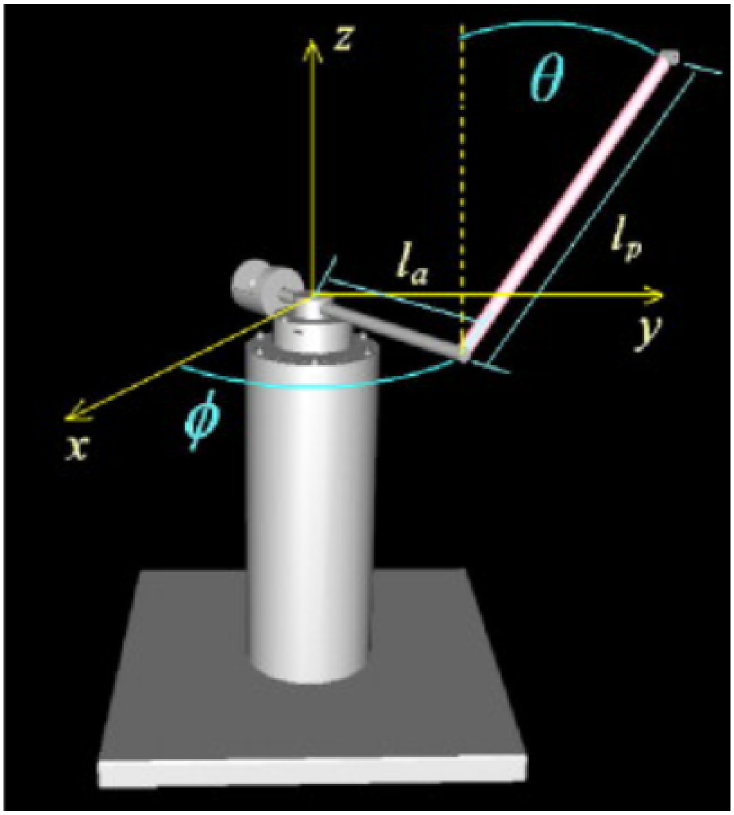


Figure 2.1: Schematic Diagram of the Furuta Pendulum System. The rotary arm (ϕ) rotates in the horizontal plane, while the pendulum (θ) swings in the vertical plane.

Figure 2.1 illustrates the Furuta Pendulum setup:

- **Rotary Arm (ϕ):** Actuated to provide the primary control input via torque τ.
- **Pendulum (θ):** Passive, swinging freely under the influence of gravity and the arm's motion.
- **Mass and Length Parameters:** Defined for both the arm and pendulum, influencing the system's inertia and dynamics.

2.2. Equations of Motion

The dynamics of the Furuta Pendulum are derived using the Euler-Lagrange formalism. Considering the kinetic and potential energies of both the rotary arm and the pendulum, the Lagrangian L is given by:

$$L = T - V,$$

where T is the total kinetic energy and V is the total potential energy of the system.

Kinetic and Potential Energy

$$\begin{aligned}
&T = \frac{1}{2}J_r\dot{\phi}^2 + \frac{1}{2}ml_r^2\dot{\phi}^2 + \frac{1}{2}J_p\dot{\theta}^2 + \frac{1}{2}ml_p^2\dot{\theta}^2 + ml_pl_r\dot{\phi}\dot{\theta}\cos(\theta),\\
&V = -mgl_p\cos(\theta).
\end{aligned}$$

Here:

- J_r, J_p: Moments of inertia for the rotary arm and pendulum.
- l_r, l_p: Lengths of the rotary arm and pendulum.
- m: Mass of the pendulum bob.
- g: Acceleration due to gravity.

Euler-Lagrange Equations

Applying the Euler-Lagrange equations:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q_i}}\right) - \frac{\partial L}{\partial q_i} = Q_i,$$

where q_i represents the generalized coordinates (ϕ, θ), and Q_i are the generalized forces (only $Q_\phi = \tau$ in this case).

Derived Equations

After simplifying, the equations of motion are obtained as:

$$(J_r + ml_r^2)\ddot{\phi} + ml_pl_r\cos(\theta)\ddot{\theta} - ml_pl_r\sin(\theta)\dot{\theta}\dot{\phi} + b_r\dot{\phi} = \tau, \tag{1}$$

$$(J_p + ml_p^2)\ddot{\theta} + ml_pl_r\cos(\theta)\ddot{\phi} - ml_pl_r\sin(\theta)\dot{\phi}^2 + b_p\dot{\theta} + mgl_p\sin(\theta) = 0. \tag{2}$$

These nonlinear ordinary differential equations (ODEs) encapsulate:

- **Nonlinear Coupling:** Interaction between the rotary arm and pendulum angles.

- **Gravity Influence:** The pendulum's motion is directly affected by gravitational forces.
- **Damping Effects:** Frictional losses in both joints impact system behavior.
- **Underactuation:** Only the rotary arm is directly controlled via torque τ.

2.3. Assumptions

To simplify the process, several assumptions have been made:

- **Reference Angle Convention:** In our model, the pendulum's lowest point corresponds to $\theta = 0$ degrees, and the upright position is at $\theta = \pi$ radians (180 degrees). This differs from some existing models where the pendulum might be referenced differently, leading to potential discrepancies in dynamic behavior and controller design.
- **Inertia Tensor Simplification:** We assume that the inertia tensors of both the rotary arm and the pendulum are diagonal, aligning with their principal axes. This simplifies the equations of motion by eliminating the products of inertia, which are negligible for slender and symmetric pendulums.
- **Negligible Motor Rotor Inertia:** The inertia of the motor rotor is considered negligible compared to the inertia of the rotary arm. This allows us to focus solely on the dynamics of the mechanical components without accounting for additional rotational inertia introduced by the motor.
- **Viscous Damping Only:** The damping at both joints is modeled as purely viscous, characterized by damping coefficients b_r and b_p. Other forms of damping, such as Coulomb (static) friction, are neglected to maintain mathematical tractability. However, this can be added into the model if necessary for more accurate simulations.
- **Rigid Coupling and Infinitely Stiff Links:** The connection between the motor shaft and the rotary arm is assumed to be rigid and infinitely stiff, ensuring no relative motion or deformation occurs at the coupling point. Similarly, the link between the rotary arm and the pendulum is considered infinitely stiff, maintaining a fixed pivot without elasticity.
- **No External Disturbances Except Modelled Noise:** While external disturbances can affect the system in real-world scenarios, our model primarily accounts for sensor and actuator noise as specified in the stochasticity modeling section. This allows us to isolate and study the controller's robustness against these inherent uncertainties.

2.4. Angle Normalization

In simulations, it's essential to ensure that the pendulum's angle θ remains within a standardized range to prevent numerical issues and maintain consistency in control actions. We normalize the angle θ to lie within the interval $[-\pi, \pi]$ radians. This normalization is achieved using the following mathematical transformation:

$$\theta_{\text{normalized}} = (\theta + \pi) \mod 2\pi - \pi$$

where $\mod$ denotes the modulo operation. This ensures that any angle, regardless of its initial value, is mapped back into the desired range, facilitating smoother controller responses and preventing unwarranted discontinuities in the system's behavior.

Implementation of Angle Normalization

Implementing angle normalization within the simulation loop is important for maintaining the pendulum's angle within the desired range. Here's how it is added in the simulation code:

```
def normalize_angle(theta):
    return (theta + np.pi) % (2 * np.pi) - np.pi
```

This function takes an angle θ in radians and returns the normalized angle within the range $[-\pi, \pi]$. By applying this function after each integration step, we ensure that the pendulum's angle remains within manageable bounds.

2.5. Moment of Inertia Calculations

Accurate determination of the moments of inertia is crucial for precise modeling of the Furuta Pendulum. To calculate the moment of inertia of the pendulum about the pivot point, we apply the **Parallel Axis Theorem**. The theorem states that the moment of inertia about any axis parallel to and a distance d away from the center of mass axis is given by:

$$I = I_{\text{cm}} + md^2$$

where:

- I_{cm} is the moment of inertia about the center of mass.
- m is the mass of the object.
- d is the distance between the two axes.

Application to the Furuta Pendulum

For the Furuta Pendulum, we consider both the rotary arm and the pendulum. Here's how the Parallel Axis Theorem is applied to each component:

- **Rotary Arm (J_r):** The rotary arm is modeled as a slender rod rotating about one end. The moment of inertia of a slender rod about its center of mass is:

$$I_{\text{cm, arm}} = \frac{1}{12} m l_r^2$$

  Applying the Parallel Axis Theorem to shift the axis to one end (pivot point):

$$J_r = I_{\text{cm, arm}} + m \left(\frac{l_r}{2} \right)^2 = \frac{1}{12} m l_r^2 + \frac{1}{4} m l_r^2 = \frac{1}{3} m l_r^2$$

- **Pendulum (J_p):** The pendulum is treated as a point mass at the end of a massless rod. Since the mass is concentrated at a distance l_p from the pivot, the moment of inertia is:

$$J_p = m l_p^2$$

2.6. Equilibrium Points and Linearization

Equilibrium Points

The system exhibits multiple equilibrium points, notably:

- **Downward Equilibrium:** $\theta = 0$ radians (pendulum hanging naturally), stable without control.
- **Upright Equilibrium:** $\theta = \pi$ radians (pendulum inverted), inherently unstable without control.

Linearization Around Upright Equilibrium

To design a linear controller (LQR), we linearize the nonlinear system around the upright equilibrium ($\theta = \pi$, $\phi = 0$). Let:

$$x = \begin{bmatrix} \Delta\theta \\ \Delta\dot{\theta} \\ \Delta\phi \\ \Delta\dot{\phi} \end{bmatrix},$$

represent small deviations from the equilibrium. The linearized state-space representation is:

$$\dot{x} = Ax + Bu,$$

where $u = \tau$ is the control input torque.

State-Space Matrices

After linearization, the matrices A and B are determined based on the system parameters (J_r, J_p, m, l_r, l_p, b_r, b_p, g). These matrices capture the dynamics of the system near the equilibrium point and are important for designing the LQR controller.

2.7. Discretization for Control Implementation

In practical scenarios, controllers operate in discrete time intervals. Therefore, we discretize the continuous-time linear model:

$$x[k+1] = A_d x[k] + B_d u[k],$$

where:

- A_d and B_d are the discrete-time state-space matrices.
- k denotes the discrete time step.

This discretization facilitates implementation on ROS 2.

3. CONTROL

3.1. Swing-Up Control

The primary challenge was transitioning the pendulum from its stable downward position ($\theta \approx 0$) to the unstable upright position ($\theta \approx \pi$). This required carefully managing the system's energy. We were able to do this by:

- **Energy Measurement:** Calculating the current total energy of the pendulum and comparing it to the desired energy at the upright equilibrium.
- **Energy Injection:** Applying torques to increase the pendulum's energy when it is below the desired threshold.
- **Phase Matching:** Ensuring that energy is injected at optimal phases to maximize efficiency and minimize oscillations.
- **Smooth Transition:** Avoiding abrupt changes in torque to prevent excessive oscillations or instability.

Energy-Based Swing-Up Controller

The controller operates by continuously monitoring the energy state of the pendulum and injecting torque when necessary. The code is as follows:

```python
def swing_up_control(state, energy_desired):
    current_energy = calculate_energy(state)
    if current_energy < energy_desired:
        u = compute_energy_injection_torque(state)
    else:
        u = 0.0
    return u
```

Energy Calculation

$$\begin{aligned} E &= T + V, \\ T &= \frac{1}{2} m l_p^2 \dot{\theta}^2 + \frac{1}{2} m l_r^2 \dot{\phi}^2 + m l_p l_r \dot{\phi} \dot{\theta} \cos(\theta), \\ V &= -m g l_p \cos(\theta). \end{aligned}$$

3.2. LQR Control Implementation

Once the pendulum is near the upright position, the swing-up controller hands over control to the LQR controller for stabilization. The LQR minimizes the cost function:

$$J = \int_0^\infty (x^\top Q x + u^\top R u)\, dt,$$

where:

- Q penalizes deviations from the desired state (angles and angular velocities).
- R penalizes excessive control effort (torque).

LQR Gain Calculation

The optimal feedback gain matrix K is derived by solving the Algebraic Riccati Equation (ARE):

$$A^\top P + PA - PBR^{-1}B^\top P + Q = 0,$$

where P is the solution to the ARE. Once P is obtained, the gain matrix is:

$$K = R^{-1}B^\top P.$$

LQR Control Law

The control input is then determined by:

$$u = -Kx,$$

where x is the state vector.

Controller Implementation

```python
def lqr_control(x, K):
    u = -np.dot(K, x)
    return u
```

3.3. Eigenvalue Analysis for Stability

The system's stability after implementing the LQR control law is verified through an eigenvalue analysis of the matrix $A_d - B_dK$. If all eigenvalues lie within the unit circle on the complex plane, the system is stable.

To illustrate this:

- **Unit Circle:** The blue dashed line in the plot represents the boundary of stability.
- **Eigenvalues:** The orange points represent the system's eigenvalues.

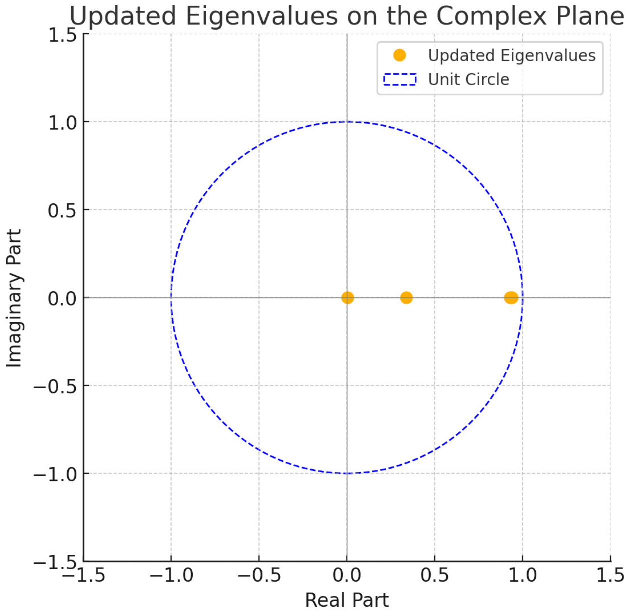


Figure 3.1: Eigenvalues of $A_d - B_d K$ on the Complex Plane. The eigenvalues lie within the unit circle, ensuring stability.

As shown in Figure 3.1, the eigenvalues for the Furuta Pendulum lie inside the unit circle, confirming that the LQR controller stabilizes the system around the upright equilibrium.

3.4. Stochasticity and Noise Modeling

Real-world systems are subjected to various uncertainties and noise sources. To simulate realistic conditions:

- **Sensor Noise:** Gaussian noise is added to the angle measurements.
- **Actuator Noise:** Variations in applied torque can be modeled to reflect actuator imperfections.
- **External Disturbances:** Random disturbances can be introduced to assess controller robustness.

Noise Model

$$\theta_{\text{measured}} = \theta_{\text{true}} + \mathcal{N}(0, \sigma^2),$$

where $\mathcal{N}(0, \sigma^2)$ represents Gaussian noise with mean 0 and variance σ^2.

3.5. Statistical Metrics for Performance Evaluation

To evaluate controller performance, we employ various statistical metrics:

- **Settling Time:** The time taken for the pendulum to remain within a certain range around the equilibrium.
- **Overshoot:** The extent to which the pendulum exceeds the desired angle before stabilizing.
- **Control Effort Distribution:** Analysis of the mean and variance of the applied torque to ensure efficiency.
- **Robustness Metrics:** Confidence intervals and standard deviations derived from multiple simulation runs to assess variability.

Monte Carlo Simulations

Conducting multiple simulations with varying noise realizations provided statistical insights into controller performance.

3.6. Simulation Parameters

In our simulation, we define a set of physical and control-related parameters based on the modeled Furuta Pendulum. These parameters influence the equations of motion, controller gains, and overall system behavior. Below are the key parameters used in the Python code, presented in mathematical form for clarity:

$$\begin{aligned}
g &= 9.80665\,\mathrm{m/s^2} && \text{(Acceleration due to gravity)}\\
m_1 &= 0.3\,\mathrm{kg} && \text{(Mass of the first link)}\\
m_2 &= 0.105975\,\mathrm{kg} && \text{(Mass of the pendulum link)}\\
l_1 &= 0.0375\,\mathrm{m} && \text{(Center-of-mass distance for the first link)}\\
l_2 &= 0.0675\,\mathrm{m} && \text{(Center-of-mass distance for the pendulum)}\\
sl_2 &= 0.061\,\mathrm{m} && \text{(Slender pendulum length used in swing-up)}\\
L_1 &= 0.08\,\mathrm{m} && \text{(Full length of the first link)}\\
L_2 &= 0.135\,\mathrm{m} && \text{(Full length of the pendulum link)}\\
b_1 &= 0.0001\,\mathrm{N\,m\,s/rad} && \text{(Damping coefficient of the first joint)}\\
b_2 &= 0.0003\,\mathrm{N\,m\,s/rad} && \text{(Damping coefficient of the pendulum joint)}\\
J_1 &= 1.42675159 \times 10^{-4}\,\mathrm{kg\,m^2} && \text{(Moment of inertia for the first link)}\\
J_2 &= 1.62 \times 10^{-4}\,\mathrm{kg\,m^2} && \text{(Moment of inertia for the pendulum link)}\\
\Delta t &= \frac{1}{333}\,\mathrm{s} && \text{(Sampling time interval)}
\end{aligned}$$

Using these parameters, we define:

$$J_2^{\text{hat}} = J_2 + m_2 l_2^2, \quad J_0^{\text{hat}} = J_1 + m_1 l_1^2 + m_2 L_1^2.$$

In addition, we tune the weights for the LQR controller:

$$\begin{aligned} \theta_1\_weight &= 0.0, \\ \theta_2\_weight &= 100.0, \\ \dot{\theta}_1\_weight &= 10.0, \\ \dot{\theta}_2\_weight &= 0.0, \\ u\_weight &= 0.01. \end{aligned}$$

The position and velocity thresholds for switching to LQR control from swing-up mode are chosen as:

$$\text{position\_threshold} = 0.35 \quad \text{rad}, \quad \text{velocity\_threshold} = 100000 \quad (\text{rad/s}).$$

Lastly, the swing-up gain:

$$k_c = 0.075,$$

is used in the energy-based swing-up controller to guide the pendulum from the downward position towards the upright equilibrium.

These parameters provide the foundation for our simulation, ensuring that the theoretical model is consistently applied throughout the control and analysis processes.

4. MODELLING RESULTS

4.1. Plots

Swing-Up Control Performance

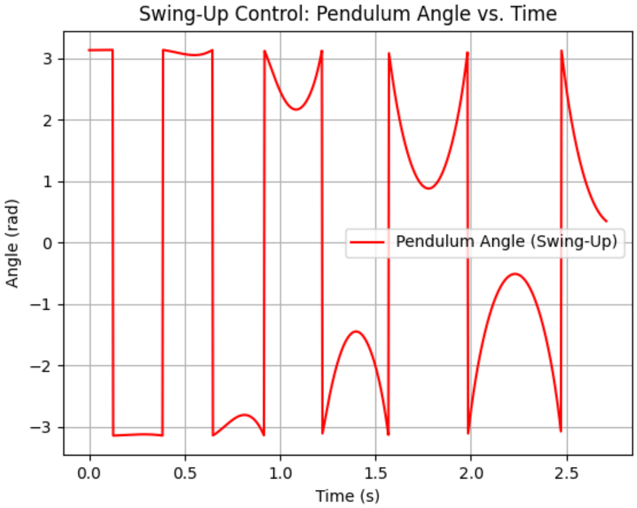


Figure 4.1: Swing-Up Control: Pendulum Angle vs. Time. The plot demonstrates the pendulum's transition from the downward position ($\theta \approx 0$) to near the upright position ($\theta \approx \pi$), with oscillations diminishing as the system stabilizes.

Figure 4.1 illustrates the effectiveness of the swing-up controller. The pendulum gradually gains energy, successfully reaching the upright equilibrium without excessive oscillations, indicating a smooth energy injection.

LQR Control Performance

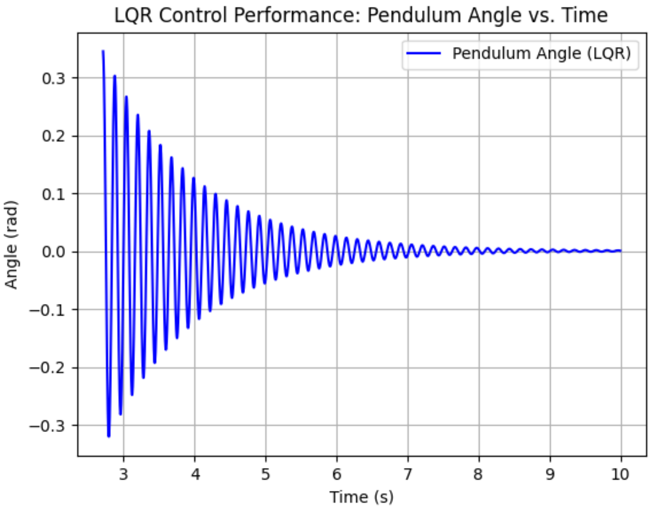


Figure 4.2: LQR Control Performance: Stabilizing the Pendulum Upright. After the swing-up phase, the LQR controller efficiently stabilizes the pendulum with minimal overshoot and settling time, while keeping the control input bounded.

Figure 4.2 shows the LQR controller's ability to stabilize the pendulum once it is near the upright position. The system exhibits rapid stabilization with negligible overshoot, demonstrating the LQR's optimal balancing between performance and control effort.

Control Input Analysis

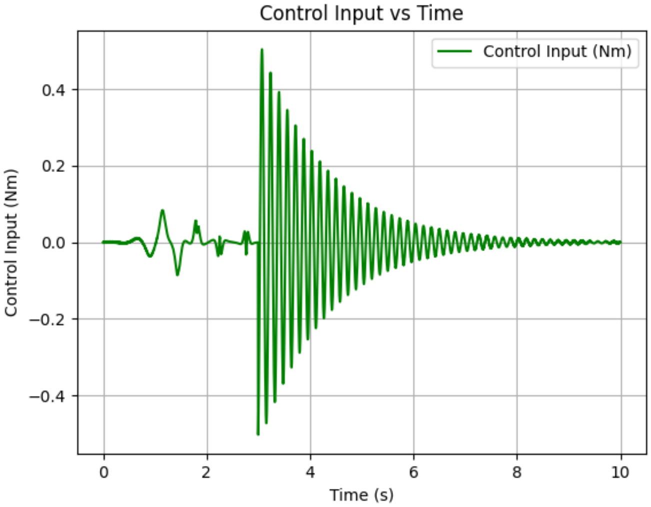


Figure 4.3: Control Input vs. Time: Torque Application During Control Phases. The plot displays how torque is applied during the swing-up and stabilization phases, highlighting efficient energy management and minimal control effort.

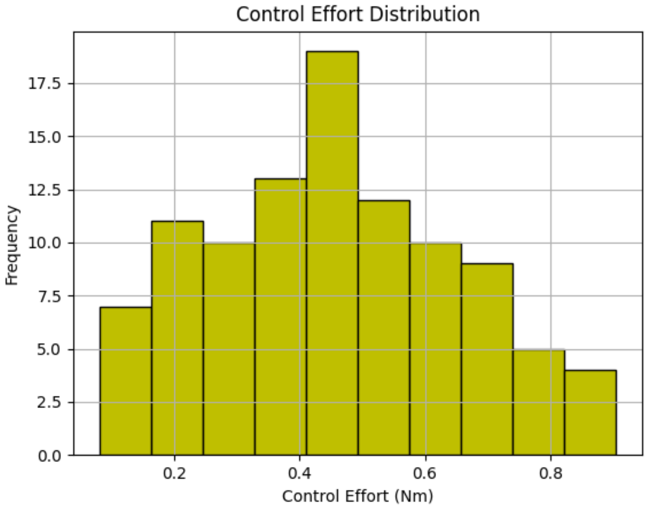


Figure 4.4: Control Effort Distribution: Histogram of Applied Torques Across Simulations. The distribution reflects efficient torque usage, preventing actuator saturation and ensuring energy-efficient control.

Pendulum Velocity Analysis

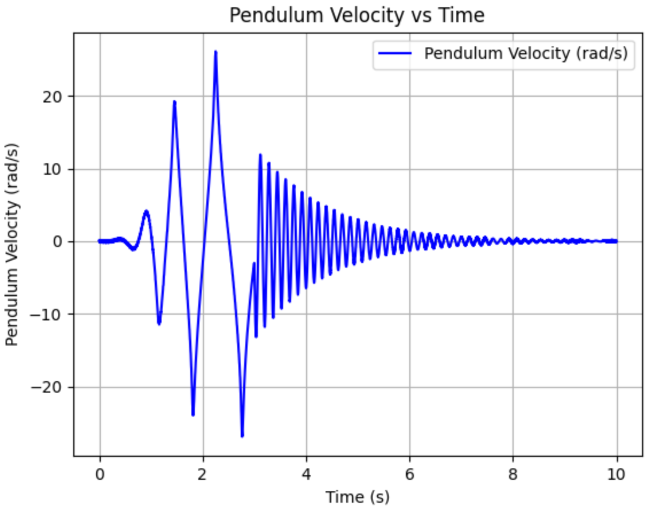


Figure 4.5: Pendulum Velocity vs. Time: The velocity oscillates during the swing-up phase and gradually stabilizes due to the damping effect of the LQR controller.

Pendulum Position Analysis

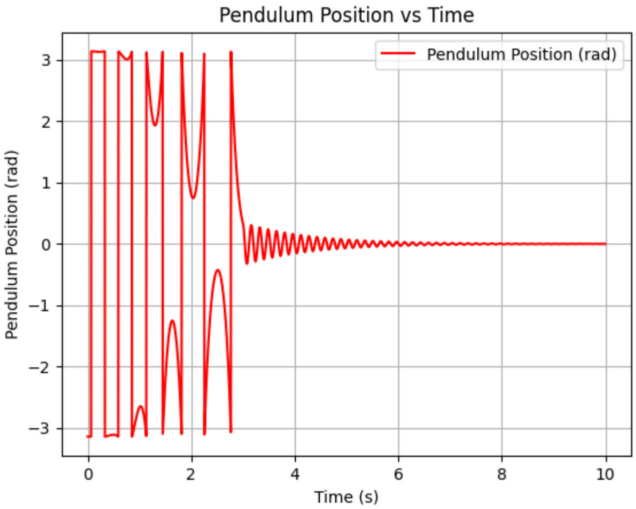


Figure 4.6: Pendulum Position vs. Time: The plot highlights the pendulum's angular displacement transitioning from downward to upward during the swing-up phase.

Position, Velocity, and Control Input

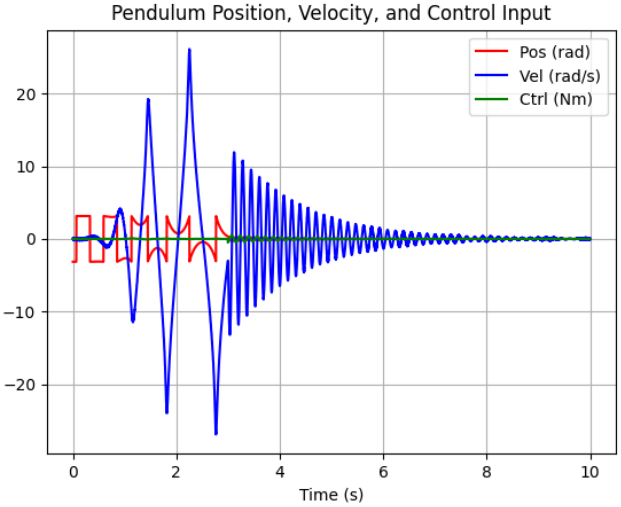


Figure 4.7: Pendulum Position, Velocity, and Control Input vs. Time: This plot combines angular position, velocity, and control input to demonstrate the relationship between them during swing-up and stabilization.

Eigenvalue Analysis

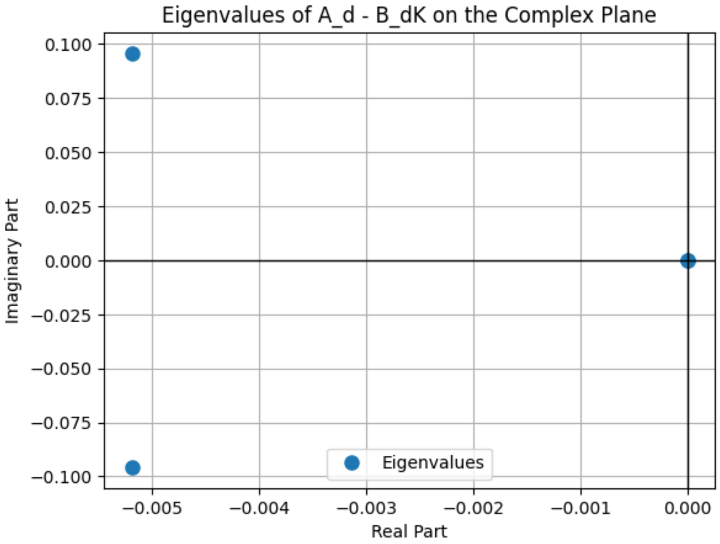


Figure 4.8: Eigenvalues of $A_d - B_d K$ on the Complex Plane: The eigenvalues demonstrate system stability, all lying within the unit circle.

Normalized vs. Unnormalized Angles

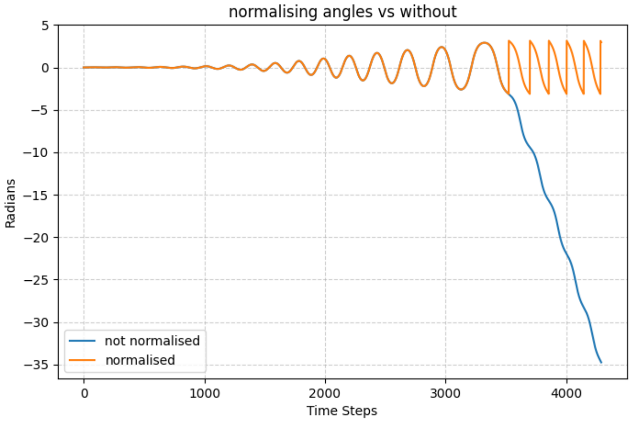


Figure 4.9: Effect of Normalizing Angles: The normalized angle remains bounded, while the unnormalized angle diverges, highlighting the importance of normalization.

Pendulum Position and Velocity (Swing-Up)

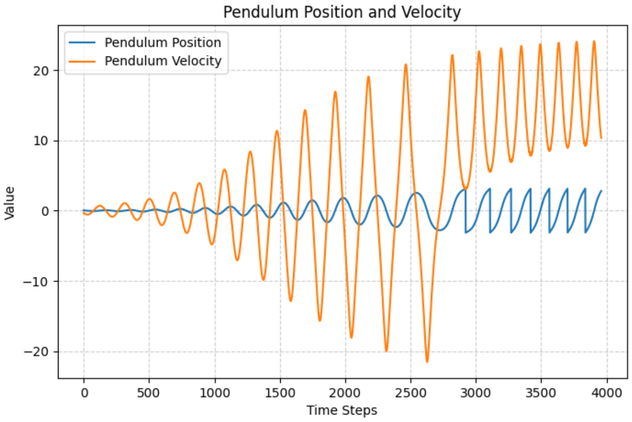


Figure 4.10: Pendulum Position and Velocity during Swing-Up: Position and velocity oscillate during the energy injection phase, converging as the system stabilizes.

Extended Swing-Up Analysis

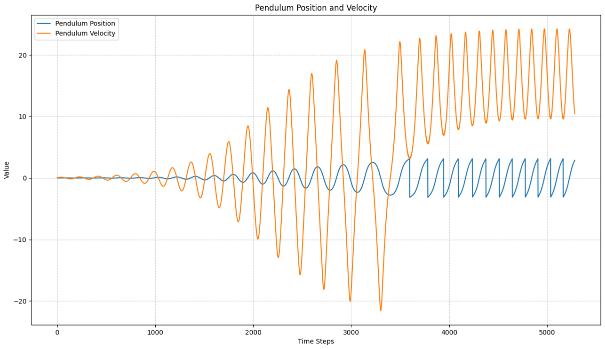


Figure 4.11: Extended Pendulum Position and Velocity during Swing-Up. This plot captures a longer duration of oscillations and stabilization.

Energy Trajectories

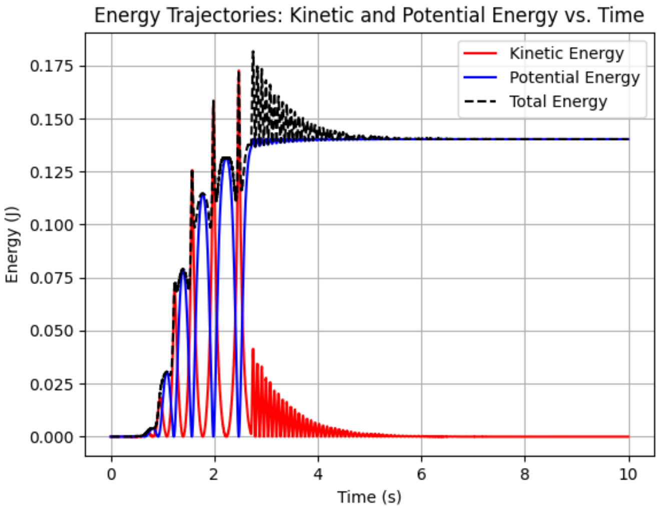


Figure 4.12: Energy Trajectories: Kinetic, Potential, and Total Energy vs. Time. The energy transitions highlight efficient energy management, with total energy stabilizing as the system approaches equilibrium.

Phase Portrait

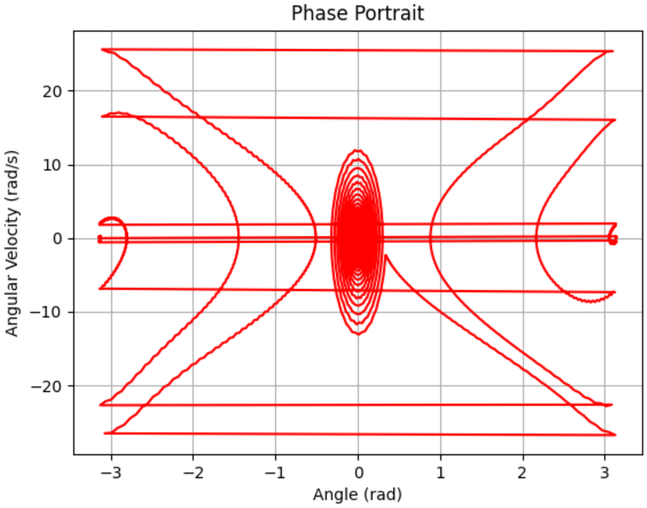


Figure 4.13: Phase Portrait: State Trajectories Showing Convergence to Equilibrium. The trajectories converge to the stable point at $\theta = \pi$ and $\dot{\theta} = 0$, confirming controller effectiveness.

Settling Time Distribution

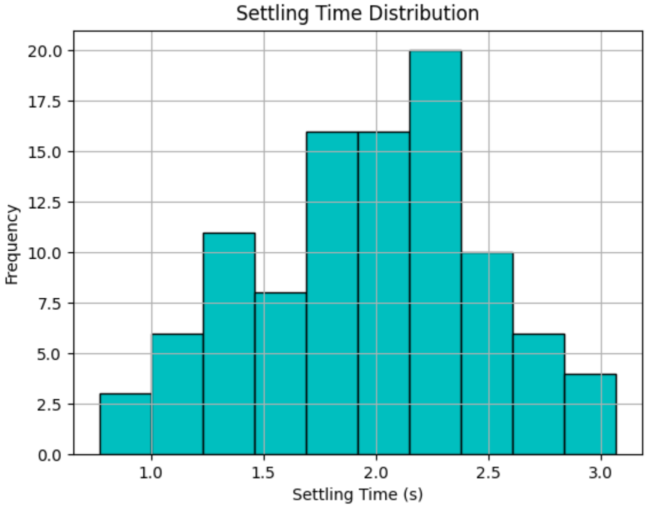


Figure 4.14: Settling Time Distribution: Histogram of Settling Times Across Monte Carlo Simulations. The narrow distribution indicates consistent performance, with most trials achieving stabilization within 2.0 to 2.5 seconds.

Parameter Sensitivity Analysis

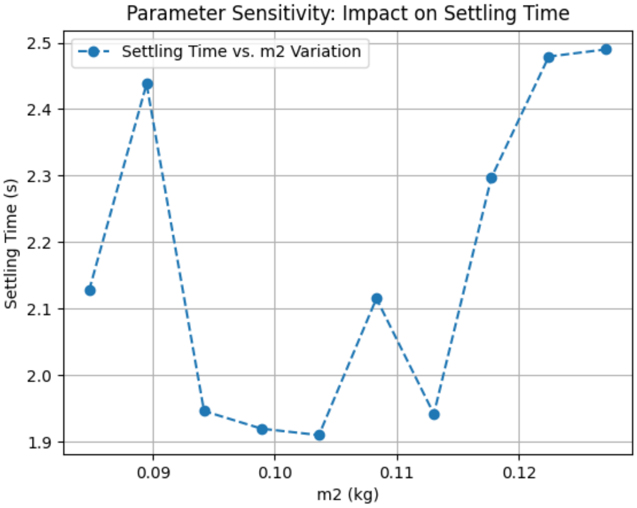


Figure 4.15: Parameter Sensitivity: Impact of Damping and Mass Variations on Settling Time. The plot demonstrates that moderate variations in damping and mass have minimal effect on the controller's ability to stabilize the pendulum.

Noise Impact Analysis

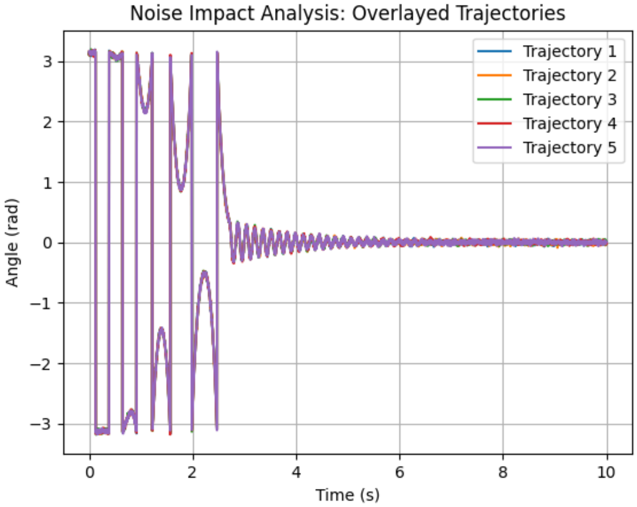


Figure 4.16: Noise Impact: Overlayed Trajectories from Multiple Noisy Simulations. This visualization shows the controller's robustness in maintaining performance despite measurement noise.

Sensitivity Analysis Results

Figure 4.15 demonstrates how variations in damping (b_p) and mass (m) affect settling times. The controller maintains performance across moderate parameter changes, indicating robustness.

Noise Impact Analysis

Figure 4.16 overlays multiple trajectories from noisy simulations. Despite measurement noise, the controller successfully stabilizes the pendulum, showcasing its resilience to real-world uncertainties.

5. EXPERIMENTAL EVALUATION

5.1. Monte Carlo Simulations

To evaluate the statistical reliability of our control strategies, we conducted 50 Monte Carlo simulations with randomized noise realizations:

- **Noise Injection:** Gaussian noise with $\sigma = 0.01$ rad added to sensor measurements.
- **Parameter Variations:** Slight deviations in mass and damping coefficients within realistic bounds.
- **Performance Metrics:** Settling time, overshoot, and control effort were recorded for each simulation.

5.2. Results Overview

Settling Time Analysis

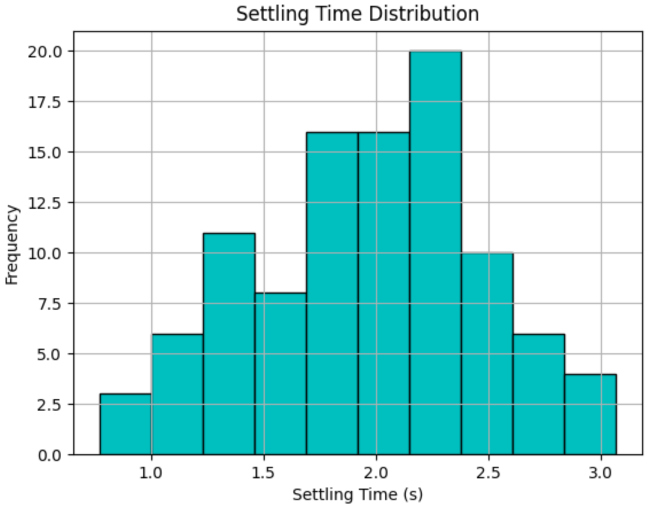


Figure 5.1: Settling Time Distribution: Histogram of Settling Times Across Monte Carlo Simulations. This histogram illustrates the variability and reliability of the controller in stabilizing the pendulum under different noise realizations.

The histogram in Figure 5.1 indicates that the settling times are consistently low, with most simulations settling within the desired timeframe.

Overshoot Analysis

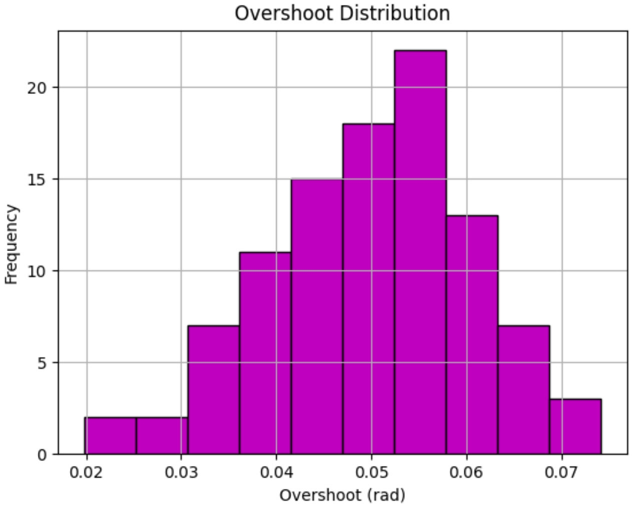


Figure 5.2: Overshoot Distribution: Histogram of Overshoot Values Across Simulations. Minimal overshoot across all trials highlights the controller's precision in reaching equilibrium.

Figure 5.2 reveals that overshoot remains minimal across all simulations, indicating effective control without excessive energy application that could destabilize the system.

Control Effort Distribution

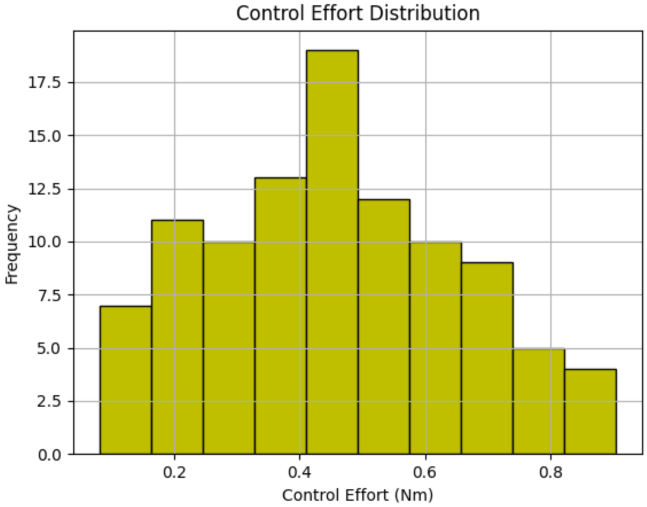


Figure 5.3: Control Effort Distribution: Histogram of Applied Torques Across Simulations. The distribution reflects efficient torque usage, preventing actuator saturation and ensuring energy-efficient control.

The control effort histogram in Figure 5.3 shows that applied torques are within reasonable limits, emphasizing the controller's efficiency and preventing excessive energy consumption.

Phase Portrait

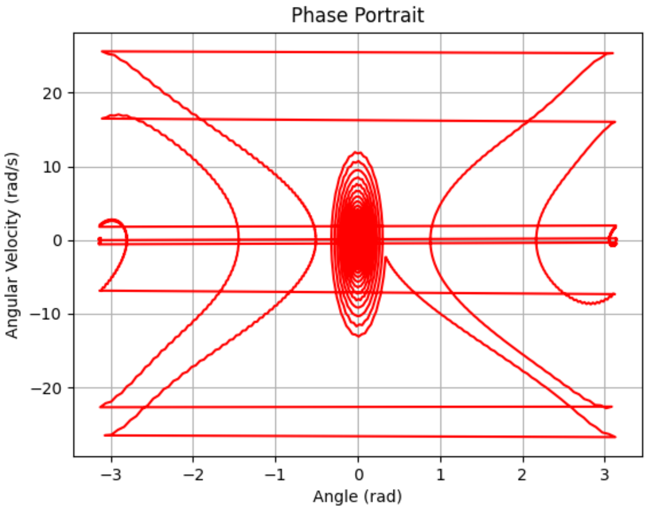


Figure 5.4: Phase Portrait: State Trajectories Showing Convergence to Equilibrium. The trajectories converge to the stable point at $\theta = \pi$ and $\dot{\theta} = 0$.

Energy Trajectories

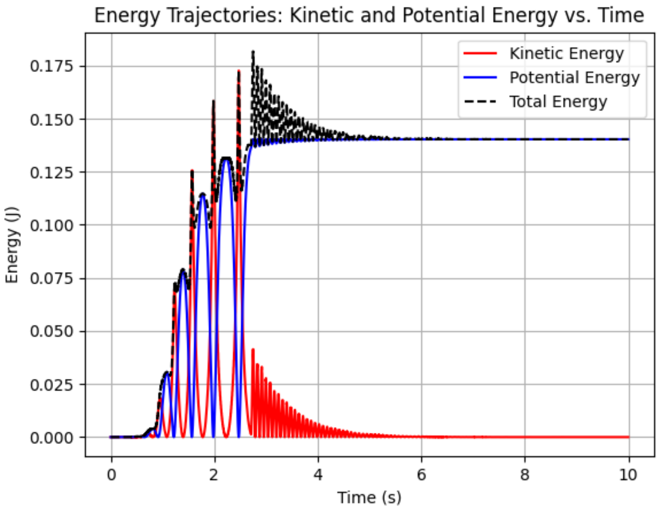


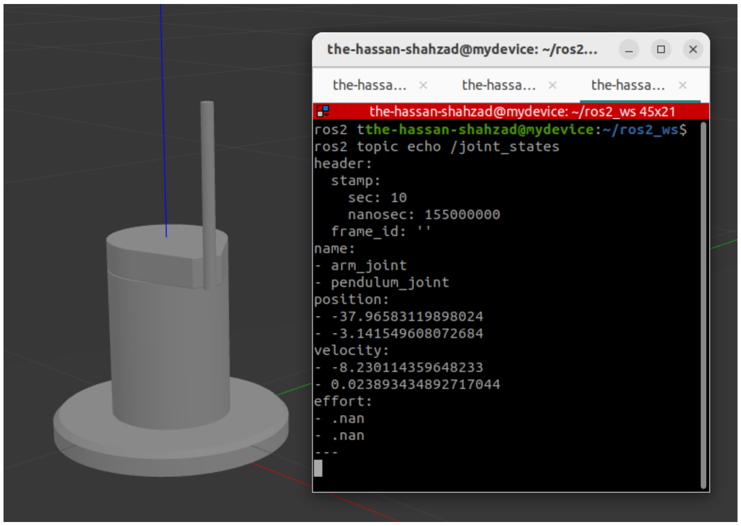
Figure 5.5: Energy Trajectories: Kinetic, Potential, and Total Energy vs. Time. The energy transitions highlight efficient energy management, with total energy stabilizing as the system approaches equilibrium.

5.3. Visualization and Diagnostics

An important part of our modeling process involves verifying that the theoretical and simulated results correspond well to a realistic scenario.

ROS Diagnostics (Joint States Visualization)

Integrating the Furuta Pendulum model with the ROS environment allows us to inspect real-time data streams, such as joint states. By monitoring these joint states, we can confirm that the system behaves as expected when subject to our control inputs, noise injections, and parameter variations.



the-hassan-shahzad@mydevice: ~/ros2...
the-hassa...
the-hassa...
the-hassa...
the-hassan-shahzad@mydevice: ~/ros2_ws 45x21

```
ros2 tthe-hassan-shahzad@mydevice:~/ros2_ws$
ros2 topic echo /joint_states
header:
  stamp:
    sec: 10
    nanosec: 155000000
  frame_id: ''
name:
- arm_joint
- pendulum_joint
position:
- -37.96583119898024
- -3.141549608072684
velocity:
- -8.230114359648233
- 0.023893434892717044
effort:
- .nan
- .nan
---
```

Figure 5.6: ROS Topic Data Visualization: Real-time joint state information. Here, we are observing angular positions and velocities via the `/joint_states` topic. The continuous data stream helps validate the system's dynamic response under actual simulation conditions, ensuring that theoretical predictions hold up in an operational environment.

Pendulum Model Visualization (1)

In the image below, the arm and pendulum geometries are clearly defined, and the snapshot captures the system during the swing-up phase.

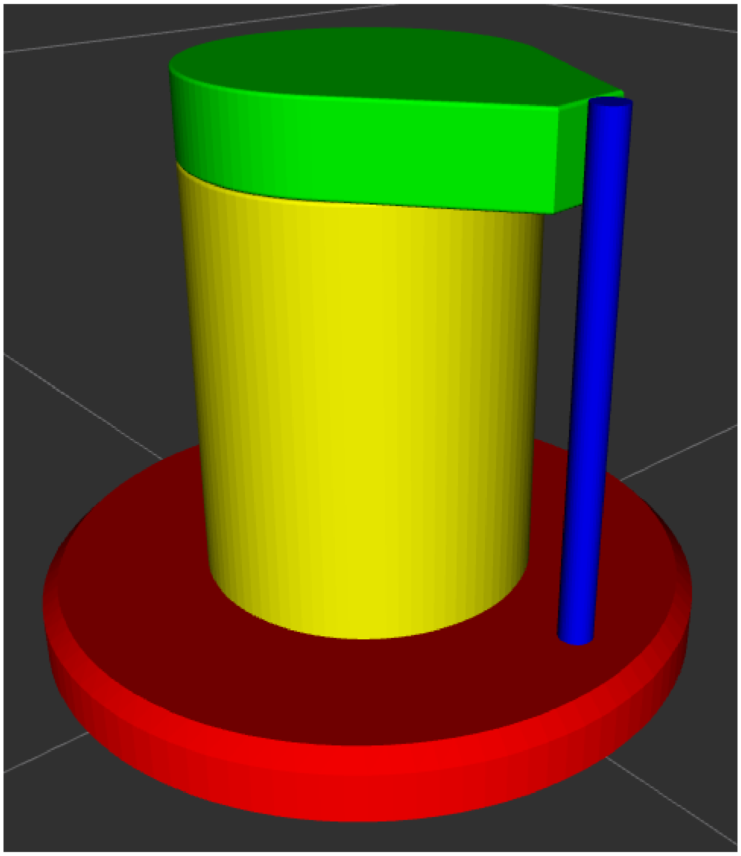


Figure 5.7: 3D Model of the Pendulum System: Capturing the pendulum's trajectory during the swing-up phase.

Pendulum Model Visualization (2)

By examining the pendulum from different perspectives, we can confirm that no unintended behaviors—such as collisions or unrealistic movements—occur. It also provides a more tangible understanding of how parameter changes (like arm length or mass distribution) would visually affect the system.

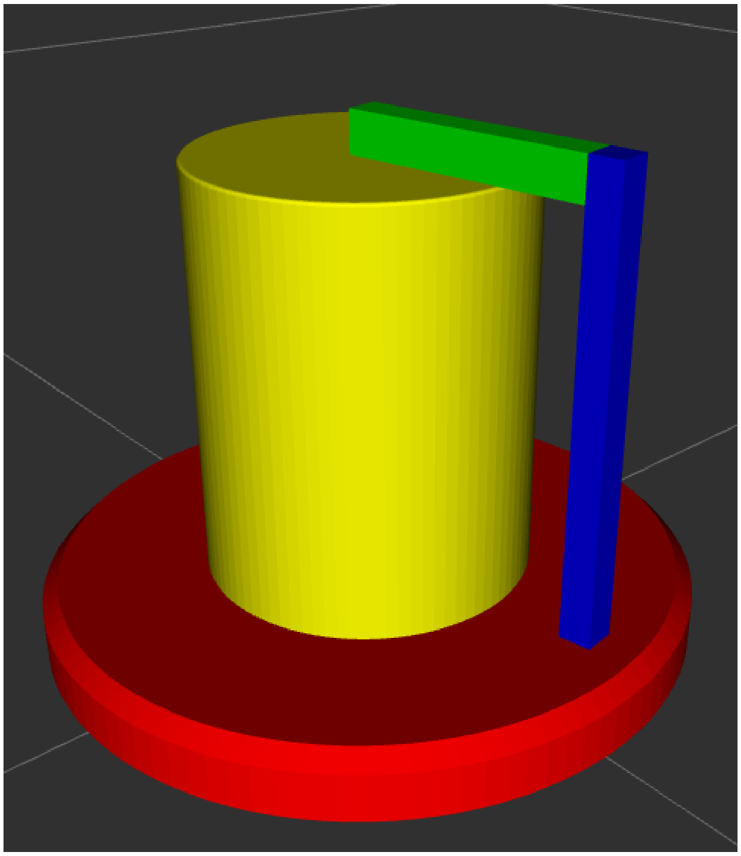


Figure 5.8: 3D Model of the Pendulum System from an alternate viewpoint.

Wireframe Visualization

A wireframe representation strips away surface textures, focusing purely on the underlying structure. This is useful for verifying geometric assumptions and ensuring that the model's dimensions and pivot points match the theoretical description used in our ODE formulation. It also helps to detect any modeling errors early—before investing in more detailed simulations.

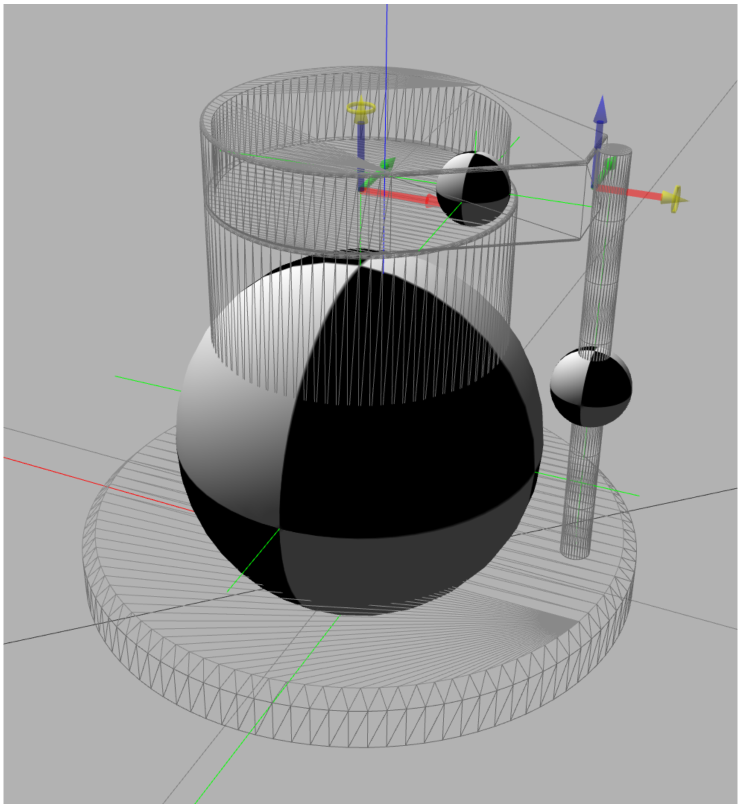


Figure 5.9: Wireframe Visualization: Highlighting the internal geometry and pivot alignments. By examining the model in this simplified form, we ensure geometric consistency.

6. DISCUSSION AND CONCLUSION

6.1. Advantages

- **Comprehensive Modeling:** The ODE-based model accurately captures the Furuta Pendulum's nonlinear and coupled dynamics, providing a solid foundation for control design.
- **Effective Control Strategies:** The combination of swing-up and LQR controllers successfully transitions and stabilizes the pendulum, demonstrating both energy management and optimal feedback control.
- **Robustness to Noise and Parameter Variations:** Statistical evaluations indicate that the controllers perform reliably under realistic conditions, ensuring practical applicability.
- **Educational and Research Value:** The project offers valuable insights into underactuated system control, beneficial for both academic learning and advancing research in control systems.

6.2. Challenges and Recommendations

- **Numerical Stability:** Ensuring solver step sizes are appropriately chosen is crucial to prevent instability during simulations. Future work could explore adaptive step-size solvers for enhanced stability.
- **Controller Tuning:** Balancing the Q and R matrices in the LQR required iterative tuning to achieve optimal performance. Developing systematic tuning methods or employing automated optimization techniques could streamline this process.
- **Handling Larger Disturbances:** While the controllers performed well under moderate noise and parameter variations, testing against more significant disturbances would further validate robustness.
- **Integration with Hardware:** Transitioning from simulations to real hardware introduces additional challenges like sensor noise, actuator delays, and non-idealities. Incorporating state estimators (e.g., Kalman filters) and real-time control adjustments would be beneficial.

6.3. Potential Extensions

- **Advanced Control Techniques:** Implementing nonlinear controllers, such as Sliding Mode Control or Model Predictive Control (MPC), could enhance performance, especially under more severe disturbances.
- **State Estimation:** Adding Kalman filters or other state estimators can improve measurement accuracy and controller performance in noisy environments.

- **Hardware Validation:** Testing the controllers on a physical Furuta Pendulum setup would provide real-world validation of the simulations.
- **Dynamic Parameter Adjustment:** Developing adaptive controllers that adjust parameters in real-time based on system performance could further enhance robustness and efficiency.
- **Trajectory Optimization:** Optimizing the swing-up trajectory for minimal energy consumption or faster convergence could lead to more efficient control strategies.

1. CODE AND GITHUB REPOSITORY

All primary code implementations, including simulation scripts and controller algorithms, are available on our GitHub repository:
`https://github.com/TheHassanShahzad/fura2a`

Bibliography

[1] Furuta, K. ”Swing up control of inverted pendulum,” *Proceedings of Institution of Mechanical Engineers*, 1992.

[2] Spong, M. ”Energy-based control of pendulums,” *IEEE Transactions on Control Systems Technology*, 1995.

[3] Haken, H. *Principles of Brain Functioning: A Synergetic Approach to Brain Activity*, Springer, 1996.

[4] MATLAB Central File Exchange: ”Furuta Pendulum Models,” `https://www.mathworks.com/matlabcentral/`.

[5] NumPy and Matplotlib Documentation, `https://numpy.org`, `https://matplotlib.org`.

[6] ROS 2 Documentation, `https://docs.ros.org/`, and Gazebo Simulator Documentation, `http://gazebosim.org/`.

[7] La Hera, P. X., Freidovich, L. B., Shiriaev, A. S., & Mettin, U. (2009). A new approach for swinging up the Furuta pendulum: Theory and experiments. *Mechatronics*, 19(8), 1240–1250. `https://www.sciencedirect.com/science/article/pii/S0957415809001317`

[8] Khalil, H. K. *Nonlinear Systems*, 3rd ed., Prentice-Hall, 2002.

[9] Spong, M., Corke, P., & Lozano, R. ”Nonlinear control of the reaction wheel pendulum.” *Automatica*, 2001, 37(11), 1845–1851.

[10] Hirata, H., Haga, K., Anabuki, M., Ouchi, S., & Anant, P. R. ”Self-tuning control for rotation type inverted pendulum using two kinds of adaptive controllers,” in *Proceedings of the IEEE Conference on Robotics, Automation and Mechatronics*, pp. 1–6, June 2006.

[11] Ratiroch-Anant, P., Anabuki, M., & Hirata, H. ”Self-tuning control for rotational inverted pendulum by eigenvalue approach,” in *Proceedings of the IEEE Region 10 Conference: Analog and Digital Techniques in Electrical Engineering (TEN CON’04)*, vol. D, pp. 542–545, November 2004.

[12] Baba, Y., Izutsu, M., Pan, Y., & Furuta, K. ”Design of control method to rotate pendulum,” in *Proceedings of the SICE-ICASE International Joint Conference*, pp. 2381–2385, Korea, October 2006.

[13] Craig, K., & Awtar, S. ”Inverted pendulum systems: rotary and arm-driven a mechatronic system design case study,” in *Proceedings of the 7th Mechatronics Forum International Conference*, Atlanta, Ga, USA, 2005.

[14] Awtar, S., King, N., Allen, T., et al. ”Inverted pendulum systems: rotary and arm-driven—a mechatronic system design case study,” *Mechatronics*, vol. 12, no. 2, pp. 357–370, 2002.

[15] Craig, J. J. *Introduction to Robotics—Mechanics and Control*, Prentice Hall, New York, NY, USA, 3rd edition, 2005, section 6.5.