Inverted Pendulum Physics: Mathematical Foundations and Implementation

Introduction

The inverted pendulum is a classic control problem in dynamics and control theory that simulates the behavior of a pendulum balanced in an unstable equilibrium position. This document outlines the physics, mathematical model, numerical integration techniques, and key parameters used in our implementation, with special attention to tuning parameters for optimal gameplay.

1. Mathematical Model

1.1 Governing Equation of Motion

The dynamics of our inverted pendulum are described by the following second-order nonlinear differential equation:

$$\ddot{\theta} = k_a \sin(\theta) - k_s \theta - k_b \dot{\theta} + k_j u(t)$$

Where:

- θ is the angular displacement from the vertical (positive is clockwise)
- $\dot{\theta}$ is the angular velocity
- $\ddot{\theta}$ is the angular acceleration
- $u(t)$ is the control input (external force)

The system coefficients are defined as:

$$k_a = \frac{m L g}{mL^2 + I_z}$$

$$k_s = \frac{k_{sp}}{mL^2 + I_z}$$

$$k_b = \frac{b}{mL^2 + I_z}$$

$$k_j = \frac{K_{joy}}{mL^2 + I_z}$$

Where:

- m is the mass at the end of the pendulum (kg)
- L is the length of the pendulum (m)
- g is gravitational acceleration (9.81 m/s^2)

- I_z is the moment of inertia about the pivot point (kg·m^2)
- k_{sp} is the torsional spring constant (N·m/rad)
- b is the damping coefficient (N·m·s/rad)
- K_{joy} is the force scaling factor for control inputs

1.2 Physical Interpretation

The equation consists of four key terms:

1. **Gravitational term** ($k_a \sin(\theta)$): This term causes the pendulum to fall away from the unstable equilibrium. It's proportional to the sine of the angle.

2. **Spring term** ($-k_s \theta$): This term models a torsional spring that provides a restoring force proportional to the displacement angle.

3. **Damping term** ($-k_b \dot{\theta}$): This term represents viscous damping, slowing the pendulum's motion proportionally to its angular velocity.

4. **Control input** ($k_j u(t)$): This term represents the external torque applied to the system (player input).

2. Numerical Integration

2.1 State-Space Formulation

To numerically integrate the system, we convert the second-order differential equation into two first-order differential equations:

$$\dot{\theta} = \omega$$ $$\dot{\omega} = k_a \sin(\theta) - k_s \theta - k_b \omega + k_j u(t)$$

Where ω is the angular velocity.

2.2 Runge-Kutta 4th Order Method (RK4)

We use the RK4 method for numerical integration due to its excellent balance of accuracy and computational efficiency. For a system of ODEs $\dot{y} = f(t, y)$, the RK4 algorithm advances the solution from t_n to t_{n+1} using:

$$k_1 = h \cdot f(t_n, y_n)$$ $$k_2 = h \cdot f(t_n + \frac{h}{2}, y_n + \frac{k_1}{2})$$ $$k_3 = h \cdot f(t_n + \frac{h}{2}, y_n + \frac{k_2}{2})$$ $$k_4 = h \cdot f(t_n + h, y_n + k_3)$$ $$y_{n+1} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

Where h is the time step.

Our implementation also supports alternative numerical methods:

- Simple Euler method (first-order)
- Improved Euler method (second-order)

The default is RK4 as it provides superior stability and accuracy for this system.

3. Key Parameters and Their Effects

3.1 Parameter Sensitivity Analysis

The inverted pendulum's behavior is highly sensitive to several key parameters:

Parameter	**Symbol**	**Effect when increased**
Mass	m	Increases inertia, slows oscillations, makes pendulum harder to control but more stable once positioned
Length	L	Increases moment of inertia, slows oscillations, makes system more difficult to stabilize
Spring constant	k_{sp}	Increases restoring force, helps system return to center, can cause oscillations if too high
Damping coefficient	b	Reduces oscillations, makes system more controllable but less responsive
Force scaling	k_j	Increases player control authority, affects how responsive the pendulum is to inputs

3.2 Parameter Tuning for Gameplay

Based on our analysis, we recommend the following parameter adjustments to address the current issue of excessive force application:

1. **Reduce the force scaling factor (k_j)**: The current implementation appears to apply too much force with each button press. Reducing k_j will make control inputs more gradual.

2. **Increase damping (b)**: A moderate increase in damping will make the system more controllable and prevent wild oscillations.

3. **Consider non-linear force application**: Implementing a force curve rather than linear scaling could provide more nuanced control.

Suggested parameter ranges for gameplay:

- k_j: 0.05 to 0.2 (reduced from current values)
- b: 0.1 to 0.5 (increased slightly)
- k_{sp}: 5.0 to 15.0 (adjust based on desired "centering" force)

4. Visualization Components

4.1 Pendulum Rendering

The visual representation maps the mathematical state to a graphical display using:

$$x_{bob} = x_{pivot} + L \cdot \sin(\theta)$$ $$y_{bob} = y_{pivot} - L \cdot \cos(\theta)$$

Where (x_{pivot}, y_{pivot}) is the fixed pivot point and (x_{bob}, y_{bob}) is the position of the pendulum bob.

4.2 Phase Space Visualization

The phase space plot (θ vs. θ) provides valuable insight into the system dynamics:

- Stable points appear as attractors
- Unstable equilibria appear as repellers
- Limit cycles indicate oscillatory behavior
- Chaotic behavior displays as irregular trajectories

This visualization is particularly useful for understanding the pendulum's behavior under different parameter regimes.

5. Implementation Notes

5.1 Boundary Conditions

Our implementation includes a fall boundary condition to handle when the pendulum falls beyond a recoverable angle:

```
if abs(position) > fallBoundary {
    position = position > 0 ? fallBoundary : -fallBoundary
    velocity = 0
}
```

This prevents the pendulum from continuously rotating and resets it to a fallen state.

5.2 Time Step Considerations

The simulation uses a fixed time step of 1/60 seconds (60Hz) matching typical display refresh rates. The stability of the numerical integration depends on this time step - smaller steps provide more accurate results but require more computation.

6. Recommendations for Further Development

1. **Adaptive difficulty**: Gradually adjust parameters based on player performance to maintain an optimal challenge level.

2. **Perturbation modeling**: Add random disturbances to simulate environmental factors for a more realistic and challenging experience.

3. **Impulse response analysis**: Study the system's response to different input patterns to fine-tune the control dynamics.

4. **Multiple control schemes**: Implement alternative control methods (continuous force vs. impulse) to assess which provides the most engaging gameplay.

Conclusion

The inverted pendulum model provides a rich platform for exploring dynamics, control theory, and numerical methods while offering an engaging gameplay challenge. By carefully tuning the physical parameters, particularly the force scaling factor and damping coefficient, we can achieve the desired balance between challenge and playability.

For questions or further discussion on parameter optimization, please contact me directly.