Spherical Electromagnetic Pendulum — 1D Model

Oliver Chang

December 2025

1 Introduction

Calculations for the Minimum Proof of Concept (PoC) for the Spherical Electromagnetic Pendulum. Through these calculations, the goal is to isolate variables and validate the physics before scaling to the full spherical model.

1.1 Objectives

The primary objectives of this PoC are:

- To validate power generation in a 1-DOF system.
- To determine the constants and parameters of coil, magnet, and pendulum geometry.
- To build instrumentation to collect and get usable data as a framework for more experiments.

2 Methodology

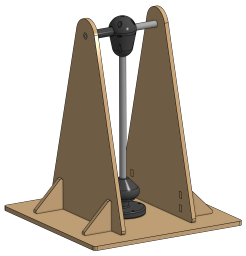

Figure 1: CAD Model of Setup

The prototype is constructed via a simple pendulum with a cylindrical magnet attached to the bottom. The magnet will swing over a round of copper coils positioned at the base from which we can measure power generation. CAD was completed through Onshape and allows the arm, magnet, and coils to adjust.

3 Theoretical Background

This section outlines the physics governing the pendulum's motion and the electromagnetic principles used to predict power generation through calculation.

3.1 Mechanical Dynamics

To generate power, the magnet must be in motion. As with previous generations of Wave Energy Converters, the efficiency of the converter depends on the pendulum's ability to resonate with ocean waves and reach high angular velocities.

Natural Frequency (f_n): The pendulum arm length (L) is tuned so that the natural frequency matches the expected frequency of ocean waves (≈ 1 Hz).

$$f_n = \frac{1}{2\pi}\sqrt{\frac{g}{L}} \tag{1}$$

Where g is the acceleration due to gravity ($9.81\, m/s^2$).

Maximum Angular Velocity (ω_{max}): Because voltage is proportional to velocity, we can use conservation of energy to solve for the maximum angular velocity occurring at the bottom of the swing:

$$\omega_{max} = \sqrt{\frac{2g(1-\cos\theta_0)}{L}} \tag{2}$$

Where θ_0 is the initial release angle.

3.2 Electromagnetic Principles

Now that the parameters of the pendulum are found, we can translate that into the generation of electricity, which is governed by the interaction between the permanent magnet's field and the copper coil.

3.2.1 Derivation of the Magnetic Field Model

The Python simulation calculates the magnetic field strength B_z using the **Equivalent Current Model**. Using the concepts from Physics 7B, we can treat the cylindrical permanent magnet as a solenoid with a surface current density K related to its remanence B_r by $K = B_r/\mu_0$.

We start with the Biot-Savart Law for a single circular current loop of radius R and current I. The field along the central axis at a distance z is:

$$B_{loop}(z) = \frac{\mu_0 I R^2}{2(R^2+z^2)^{3/2}} \tag{3}$$

To model a solid cylinder of length D, we treat it as a stack of infinite infinitesimal loops. We replace the current I with the differential current $dI = K dx$, where x is the distance from the measurement point to the slice. We integrate from the front face ($x = z$) to the back face ($x = z + D$):

$$B_{total} = \int_z^{z+D} \frac{\mu_0 (K dx) R^2}{2(R^2+x^2)^{3/2}} = \frac{B_r R^2}{2}\int_z^{z+D} \frac{dx}{(R^2+x^2)^{3/2}} \tag{4}$$

Using the standard integral $\int \frac{dx}{(a^2+x^2)^{3/2}} = \frac{x}{a^2\sqrt{a^2+x^2}}$, we evaluate the definite integral:

$$B_z(z) = \frac{B_r}{2}\left[\frac{x}{\sqrt{R^2+x^2}}\right]_z^{z+D} \tag{5}$$

Substituting the limits yields the final analytical equation used in the Python script:

$$B_z(z) = \frac{B_r}{2}\left(\frac{D+z}{\sqrt{R^2+(D+z)^2}} - \frac{z}{\sqrt{R^2+z^2}}\right) \tag{6}$$

3.2.2 Flux and Coupling

Magnetic Flux (Φ): As the reference papers detail, calculating the flux integral $\Phi = \iint \vec{B} \cdot d\vec{A}$ dynamically at every time-step is computationally expensive. Instead, it is more effective to borrow the curve-fitting method established in paper [1].

The method involves running a static simulation to generate "true" flux data points at various angles and then fitting them to the following function:

$$\Phi(\alpha) \approx \frac{a}{\sqrt{b\alpha^2 + c}} \tag{7}$$

Equation Rationale: This function mimics the geometric decay of a magnetic field. The magnetic field strength roughly follows an inverse-square law with distance ($1/r^2$). Since the distance between the pendulum arc and the flat coil increases with the square of the angle (α^2) for small oscillations, this term models the rapid drop-off of flux as the magnet swings away from the center.

Coupling Coefficient (K): This coefficient represents the system's sensitivity ($d\Phi/d\alpha$) [1]. By analytically differentiating the fitted flux equation, we obtain a computationally efficient formula for voltage prediction:

$$K(\alpha) = \frac{d\Phi}{d\alpha} = \frac{-ab\alpha}{(b\alpha^2 + c)^{3/2}} \tag{8}$$

3.3 Electrical Output

Using Faraday's Law of Induction, we calculate the instantaneous voltage and power delivered to the load. Both values are functions of time (which translates to the pendulum's angle).

Instantaneous Voltage ($V(\alpha)$): The electromotive force is the product of the coupling coefficient and the angular velocity:

$$V(\alpha) = -N \cdot K(\alpha) \cdot \omega(\alpha) \tag{9}$$

Instantaneous Power ($P(\alpha)$): The power dissipated by the load varies throughout the swing cycle:

$$P(\alpha) = \frac{V(\alpha)^2}{R_{coil} + R_{load}} \tag{10}$$

The *average power* is found by integrating this pulse over one full period, while the *peak power* occurs at $\alpha = 0$ (bottom of the swing).

Power Output (P): The instantaneous power dissipated by the load is determined by the voltage and the total resistance [1]:

$$P = \frac{V^2}{R_{coil} + R_{load}} \tag{11}$$

4 Python Simulation

In order to have a numerical calculator using all of our input parameters, this Python script calculates the theoretical magnetic flux and predicts the output voltage based on the equations defined above.

Currently, there are assumptions for constants (such as magnet grade and exact wire gauge) that will be updated as testing data helps us determine the specific variables.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit

# ========================================
# 1. ASSUMPTIONS & CONSTANTS
# ========================================
Br = 1.32       # Remanence (Tesla) [N42]
R_mag = 0.015   # Magnet radius (m)
D_mag = 0.010   # Magnet thickness (m)
N_turns = 500   # Coil turns
```

```
r_coil = 0.018  # Coil avg radius (m)

# Resistance (Approx for 30 AWG)
rho_cu = 1.68e-8
area_30awg = 0.05e-6
len_wire = N_turns * 2 * np.pi * r_coil
R_coil = (rho_cu * len_wire) / area_30awg
print(f"Coil Resistance: {R_coil:.2f} Ohms")

# Dynamics
L = 0.10        # Pendulum length (m)
gap = 0.002     # Min air gap (m)
theta_0 = 15    # Release angle (degrees)
g = 9.81

# ==========================================
# 2. PHYSICS MODELS
# ==========================================
def calc_Bz(z):
    # Biot-Savart Law for Cylinder on Axis
    term1 = (D_mag + z) / np.sqrt(R_mag**2 + (D_mag + z)**2)
    term2 = z / np.sqrt(R_mag**2 + z**2)
    return (Br / 2) * (term1 - term2)

def get_flux_data(angles_deg):
    # Calculate Flux vs Angle
    rads = np.radians(angles_deg)
    # Effective distance z increases as magnet swings away
    z_eff = np.sqrt(gap**2 + (L * np.sin(rads))**2)
    B_vals = calc_Bz(z_eff)
    return B_vals * (np.pi * r_coil**2)

# Fitted Model from Literature
def flux_model(alpha, a, b, c):
    return a / np.sqrt(b * alpha**2 + c)

# ==========================================
# 3. SIMULATION ROUTINE
# ==========================================
# A. Generate Data & Fit Curve
angles = np.linspace(0, theta_0, 100)
flux_data = get_flux_data(angles)
popt, _ = curve_fit(flux_model, angles, flux_data, p0=[1, 1, 1])
a, b, c = popt

# B. Calculate Velocity vs Angle (Conservation of Energy)
# omega = sqrt( 2g/L * (cos(alpha) - cos(theta_release)) )
rads = np.radians(angles)
rad_0 = np.radians(theta_0)
omega = np.sqrt((2*g/L) * (np.cos(rads) - np.cos(rad_0)))

# C. Calculate Coupling Coefficient K (dPhi/dAlpha)
# Derivative of a/sqrt(b*x^2 + c)
d_flux_deg = -0.5 * a * (b * angles**2 + c)**(-1.5) * (2 * b * angles)
K_vals = d_flux_deg * (180/np.pi) # Convert to V/(rad/s)

# D. Calculate Voltage & Power Pulse
voltage = -N_turns * K_vals * omega
# Matched load: R_load = R_coil
power = (voltage**2) / (2 * R_coil)

print(f"Peak Voltage: {np.max(np.abs(voltage)):.2f} V")
print(f"Peak Power:   {np.max(power)*1000:.2f} mW")

# ==========================================
# 4. PLOTTING
# ==========================================
fig, ax1 = plt.subplots(figsize=(8, 5))
```

```
80
81 color = 'tab:red'
82 ax1.set_xlabel('Swing Angle (Degrees)')
83 ax1.set_ylabel('Power Output (mW)', color=color)
84 ax1.plot(angles, power*1000, color=color, linewidth=2)
85 ax1.tick_params(axis='y', labelcolor=color)
86 ax1.grid(True, alpha=0.3)
87
88 ax2 = ax1.twinx()  # Instantiate a second axes
89 color = 'tab:blue'
90 ax2.set_ylabel('Voltage (V)', color=color)
91 ax2.plot(angles, np.abs(voltage), color=color, linestyle='--')
92 ax2.tick_params(axis='y', labelcolor=color)
93
94 plt.title(f"Power & Voltage vs. Angle (Release: {theta_0}  )")
95 fig.tight_layout()
96 plt.show()
```

Listing 1: Python Simulation Script

5 Testing and Validation

The Python simulation relies on several theoretical assumptions. In order to improve the accuracy of the Python calculations and scale to the full 3D model, we ran experiments to calibrate the specific constants.

5.1 Experiment 1: Resistance Calibration

Objective: To correct the electrical impedance model.

- **Procedure:** Use a multimeter to measure the DC resistance of the physical copper coil.
- **Code Update:** Compare the measured value to the script's output `Calculated Coil Resistance`. Change the `r_coil_avg` variable to match the tested value.

5.2 Experiment 2: Magnetic Field Strength Verification

Objective: To validate the Magnet properties (B_r) and Air Gap (g).

- **Procedure:** Set the pendulum to the release angle ($\theta_0 = 15°$) and release it. Measure the peak open-circuit voltage (V_{peak}) using an oscilloscope across the coil terminals.
- **Analysis Logic:** We compare the measured voltage (V_{meas}) to the script's predicted voltage (V_{pred}). The error is likely due to the Air Gap (g) or Magnet Strength (B_r). Especially because this is a magnet purchased on Amazon without specific standardization, it will likely differ from existing N35 and N42 standards.

5.3 Experiment 3: Load Optimization

Objective: To confirm maximum power transfer.

- **Procedure:** Connect a potentiometer in series with the coil. Swing the pendulum and measure power output at various resistance settings.
- **Code Update:** Plot the experimental Power vs. Resistance curve. The peak of this curve should occur when $R_{load} \approx R_{coil}$. This validates the internal resistance calculations and confirms the efficiency of the generator.

6 Limitations and Model Assumptions

6.1 Uniform Flux Distribution Error

The current simulation calculates the magnetic field strength B_z at the exact center of the coil and assumes this maximum value applies uniformly across the entire coil area (A_{coil}).

$$\Phi_{sim} = B_{center} \times \pi r^2 \tag{12}$$

In reality, the magnetic field of a cylindrical magnet is non-uniform, with intensity dropping significantly towards the edges of the 18mm coil radius. This simplification leads to an overestimation of the total magnetic flux Φ, and since power scales with Φ^2, this introduces the largest source of error.

6.2 Omission of Lenz's Law (Magnetic Braking)

The dynamic model calculates angular velocity ω using the Conservation of Energy for a free-falling pendulum ($\omega = \sqrt{2gh/L}$). It neglects the counter-electromotive force (Back EMF) generated by the induced current.

$$\tau_{brake} = \frac{V_{induced}^2}{R_{coil} \cdot \omega} \tag{13}$$

As the generator produces power, this magnetic braking torque will physically slow the pendulum at the bottom of the swing, reducing the actual ω_{max} and, consequently, the voltage output.

7 Next Steps Before Scaling to the Full 3D Model

The following tasks are identified for the transition to the full prototype:

- **Universal Joint for Pitch & Roll:** Mechanics for 2-DOF motion.
- **Spherical Magnet Array + Optimization:** Optimizing the magnet layout for 3D coverage.
- **Power Electronics Integration:** Adding circuitry for the angle sensor and battery charging.
- **3D Dynamics Simulation:** Predicting behavior in chaotic sea states.
- **Frequency Tuning:** Adding counterweights to match resonance frequency.
- **Coulomb Friction & Damping:** Accounting for mechanical losses.
- **Coil Geometry Optimization:** Tuning ID, OD, and thickness.
- **Multi-Coil Flux Interaction:** Simulating overlaps between adjacent coils.

References

[1] H. Lou, T. Wang, and S. Zhu, "Design, modeling and experiments of a novel biaxial-pendulum vibration energy harvester," *Energy*, vol. 254, p. 124 431, 2022.