PROJECT MILESTONE

By Edward Raphael & Harry Chiu

NAME OF PROJECT:

Lightning Modelling and Simulation in Python

Agenda

Objectives and Goals

Goal # 1

Simulate lightning formation using recursive fractal algorithms.

Goal # 2

Integrate physical principles, particularly electric field influence.

Goal # 3

Visualize natural lightning behavior with 2D plots and animations.

Goal # 4

Analyze behavior under environmental variations (e.g., pressure, humidity).

Description

01 What Causes Lightning?
Collisions between the ice particles and water droplets causes there to be a separation of positive and negative charges. This buildup of said charges creates an electric field between the clouds and the ground strong enough to generate lightning. The lightning is produced once the field exceeds a certain threshold where air becomes ionized and plasma channels form.

02 Fractal Segment Construction:
- Midpoint: $M = (A + B) / 2$
- Normal Vector: $n = [-(yB - yA), xB - xA] / ||...||$
- Offset Midpoint: $M' = M + offset * n$
- Recursive Subdivision builds complexity (generations)

Branching Mechanics:
- Probability: branch if random() < p_branch
- Rotation Matrix: rotates branch direction $R(\theta)$
- Segment Decay: L_new = L_old * decay_factor

03 Electric Field Approximation:

- $E(y) = 1 - y / MAX\_Y$
- y: height of midpoint, MAX_Y: total canvas height
- Simulates stronger field near ground ($y = 0$)

Growth Constraint:

- Segment only forms if $E(y) > E\_threshold$
- Branching only occurs in high-field zones

Purpose:

- Models lightning propagation toward ground
- Adds realism based on physics principles

Tested Variables

Test Variables: generations, max_offset, decay_factor, E_THRESHOLD, p_branch.

Constants: MAX_Y = 200, angle ranges, random seed (for testing).

Computed

segment midpoints, normal vectors, field strength.

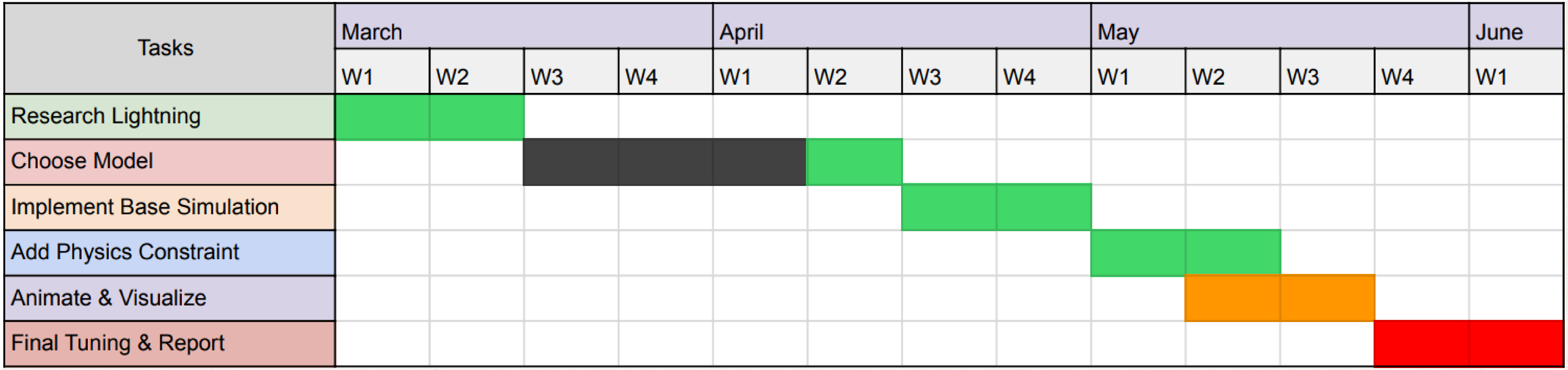
- Vary one test variable at a time and observe effects.

Timeline

Milestone	Description	Owner	Status
Research About Lightning	Conduct a comprehensive review of studies literature on lightning phenomena. Understanding the fundamental physics and behavior that cause the phenomena. This forms the foundation for our project by identifying variables and factors.	Harry	✅ Done
Choose Appropriate Model	Select the appropriate computational model that will be used for simulating lightning. This involves evaluating different algorithms based on their accuracy, efficiency, and compatability that suits out research.	Edward	✅ Done
Implement Basic Code (Simulation)	Develop and run an initial simulation based on the chosen model. This includes setting up the environment, ensuring basic functionalities are working and coding the simulation parameters.	Edward	✅ Done
Add Constraints and Variables	Introduce physical constraints and parameters to the simulation to enhance the simulation accuracy. This involves integrating known physics principles related to electricity, atmospheric conditions, and lightning behavior into the model.	Harry	✅ Done
Animate & Visualize	Create accurate visual representations of the simulation results. Includes generating simple animations that depict lightning strikes and visualizing the data in a way to make it comprehensible.	Edward	🟢 In Progress
Final Fixes & Report	Perform final adjustments to the simulation parameters to optimize accuracy and performance. Compile the results, analyses, and visuals into a report that summarizes the findings and implications of the research.	Harry	🔴 Not Done

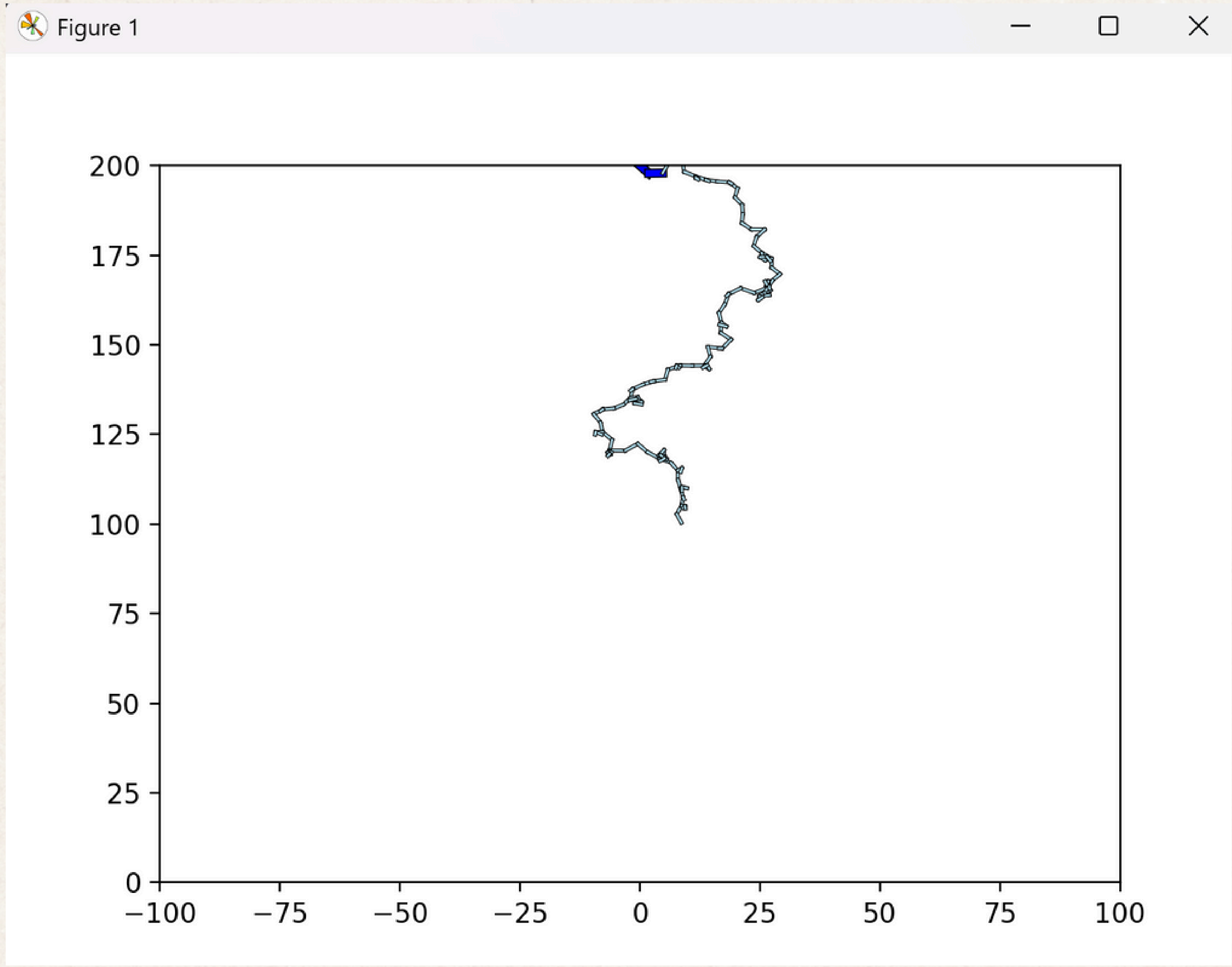
Gantt Chart

Display of our work timeline



- Completed
- In Progress
- Not complete
- Hiatus

Visualizations



Program

```python
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation

# Constants
E_THRESHOLD = 0.3  # Threshold for electric field strength
MAX_Y = 200        # Maximum height for the simulation

def get_electric_field(point):

    y = point[1]
    return y / MAX_Y  # Normalized field strength (0 at top, 1 at bottom)

def create_segments(segments, generations, max_offset, decay_factor):
    if generations == 0:
        return segments

    new_segments = []

    for segment in segments:
        start, end = segment
        mid = 0.5 * (start + end)

        # Check if the electric field at the midpoint is sufficient
        E_field = get_electric_field(mid)
        if E_field < E_THRESHOLD:
            continue

        # Calculate the normal vector for offsetting
        normal = np.array([-(end[1] - start[1]), end[0] - start[0]], dtype=float)
        normal /= np.linalg.norm(normal)
        offset = np.random.uniform(-max_offset, max_offset)
        mid += normal * offset

        new_segments.append((start, mid))
        new_segments.append((mid, end))

        if np.random.rand() < 0.3 and E_field > E_THRESHOLD + 0.1:
            direction = end - start
            direction /= np.linalg.norm(direction)
            angle = np.random.uniform(-np.pi / 4, np.pi / 4)
            rot_matrix = np.array([
                [np.cos(angle), -np.sin(angle)],
                [np.sin(angle),  np.cos(angle)]
            ])
            direction = rot_matrix @ direction
            branch_end = mid + direction * decay_factor
            new_segments.append((mid, branch_end))

    return create_segments(new_segments, generations - 1, max_offset / 2, decay_factor)
```

Thank you

References

https://siliconwit.com/modelling-and-simulation-in-python/lightning-strike

Riousset, J. A., Pasko, V. P., Krehbiel, P. R., Thomas, R. J., & Rison, W. (2007). Three-dimensional fractal modeling of intracloud lightning discharge in a New Mexico thunderstorm and comparison with lightning mapping observations. Journal of Geophysical Research Atmospheres, 112(D15). https://doi.org/10.1029/2006jd007621