Final Report: Simulation of Spacecraft Aerodynamics Using Particle Dynamics

Abstract:

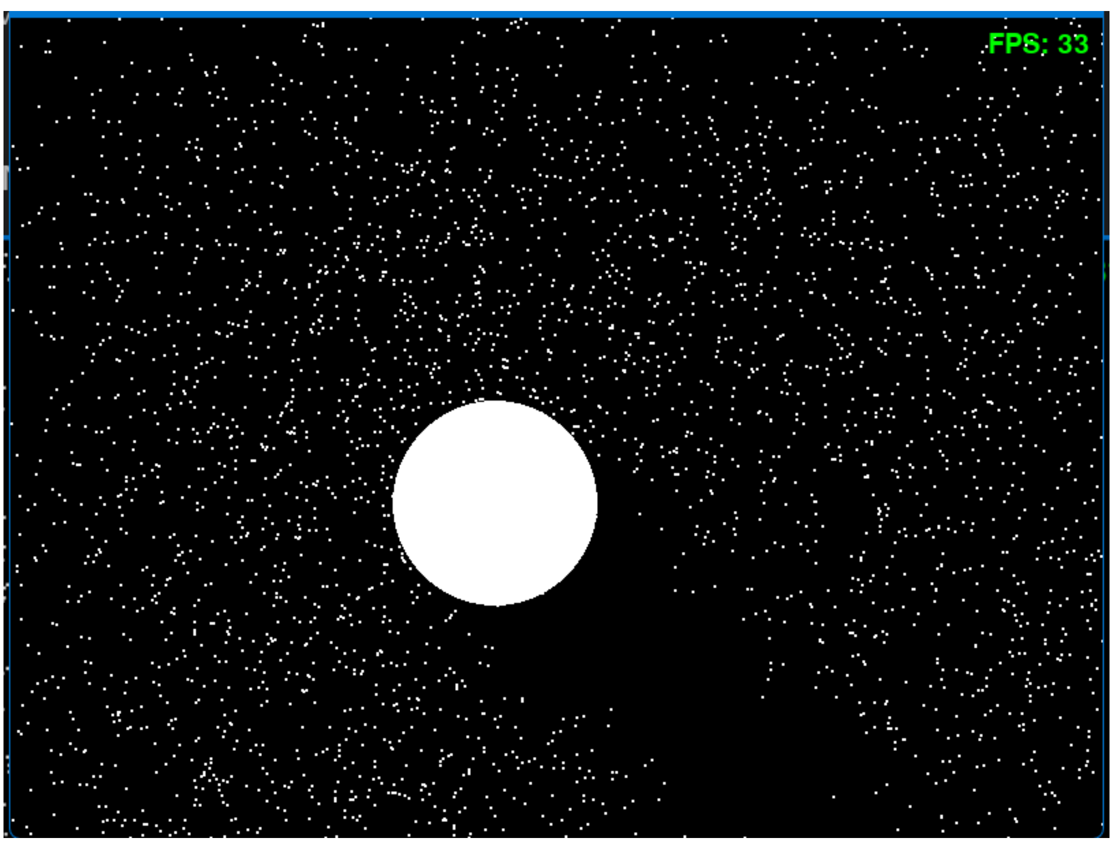
This report presents a detailed account of a collaborative programming project that simulates the aerodynamic interactions of air particles with a spacecraft. The simulation models the behavior of particles as they encounter a spacecraft, represented by a square obstacle, and the resultant aerodynamic forces.

Introduction:

The simulation was developed using Python and the Pygame library, creating a visual and interactive representation of air particles colliding with a spacecraft. The SquareObstacle class, representing the spacecraft, is central to the simulation, providing the framework for collision detection and response.

Methodology: SquareObstacle Class and Collision Detection

We started with the circle shape first because it's easier.

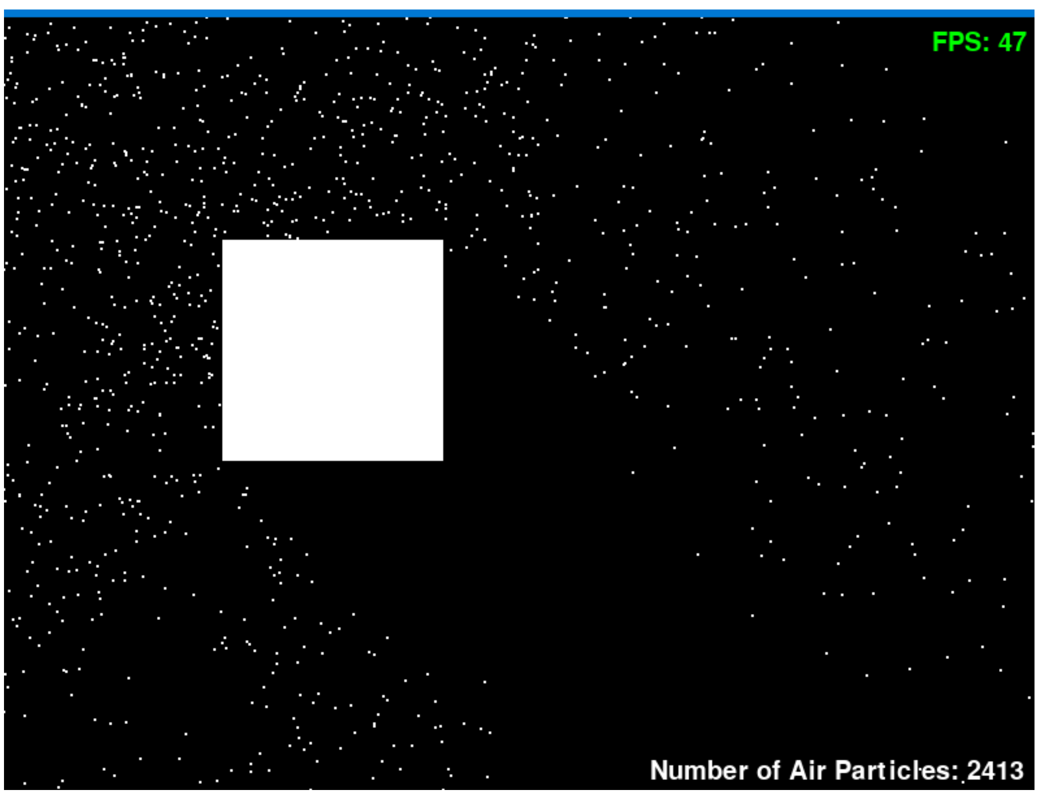


then we went for "Square" object with the difficulty of edges

The **SquareObstacle** class serves as a fundamental building block in our simulation. Here's how it works:

1. **Instantiation and Parameters**:
   - The **SquareObstacle** class is instantiated with essential parameters: **size**, **mass**, **initial_position**, and **velocity**.
   - These parameters define the characteristics of the spacecraft (or obstacle) within the simulation.
   - For instance, **size** determines the dimensions of the square obstacle, while **mass** represents its mass.

2. **Collision Detection**:
   - The **check_collision** method plays a crucial role in our simulation.
   - It determines whether a collision has occurred between an **air particle** (representing the surrounding air) and the spacecraft.
   - If the air particle enters the bounds of the square obstacle, a collision is detected.



3. **Impulse Calculation**:

    - Upon collision, we calculate the resulting **impulse** based on the **conservation of momentum**.
    - The impulse represents the change in momentum due to the collision.
    - The formula for impulse is **Impulse (J) = Change in Momentum (Δp)**.

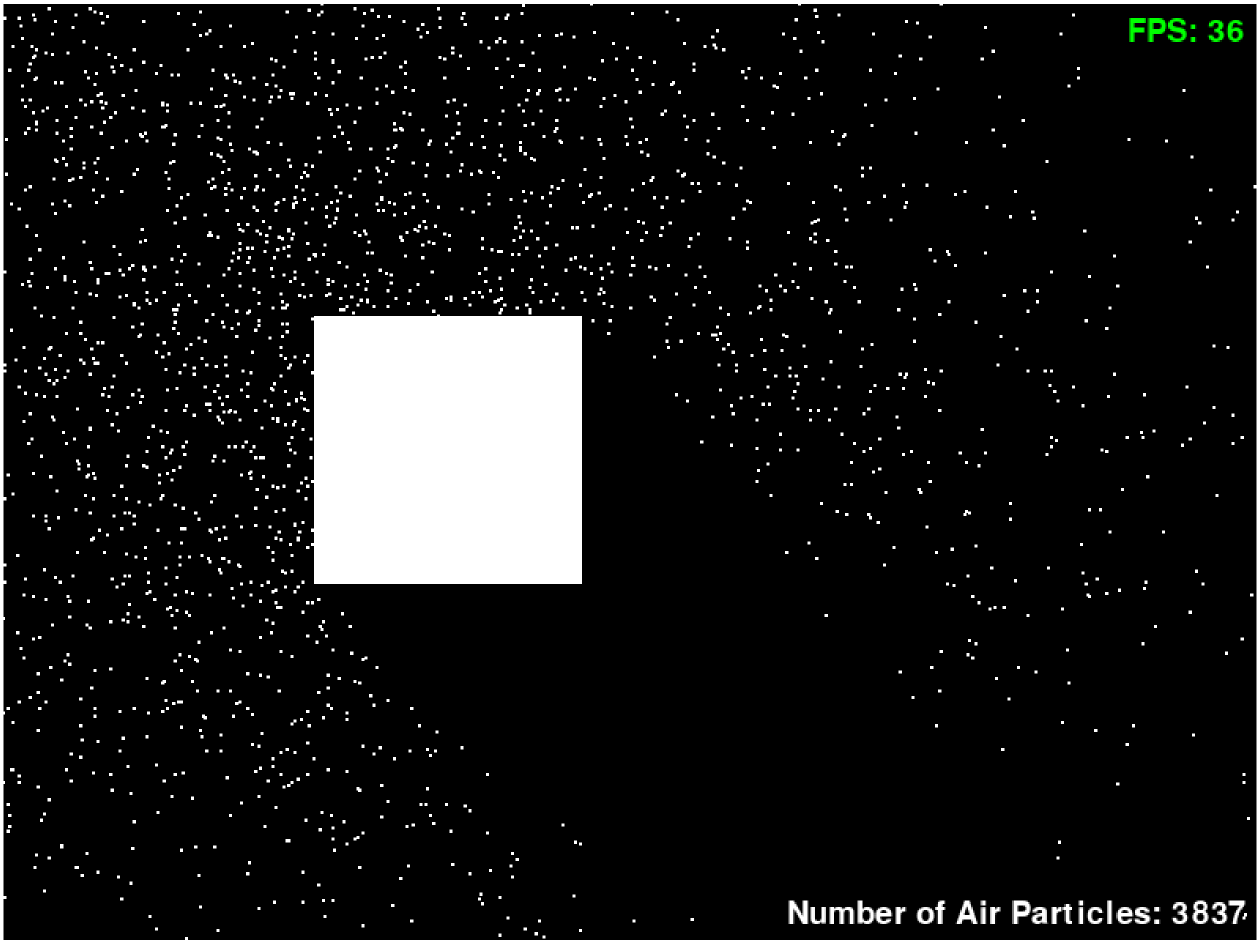
Results: Accurate Deflection and Momentum Change

- When an air particle collides with the spacecraft, our simulation accurately calculates:
    - The **angle of deflection** based on the collision angle.
    - The **change in momentum** using the impulse-momentum theorem.
    - The air particle's **updated velocity** to reflect realistic deflection.
- This behavior aligns with the **aerodynamic forces** experienced by spacecraft during atmospheric flight.

Discussion: Real-World Relevance and Computational Methods

- Our simulation approach mirrors the **computational methods** used in aerospace engineering.
- Specifically, we draw inspiration from NASA's **Advanced Supercomputing (NAS) Facility**.
- The **impulse-based collision response** we use is reflective of real-world physics engines employed in high-fidelity simulations.
- These simulations are essential for understanding and predicting aerodynamic forces during spacecraft entry, descent, and landing.

By combining our script with insights from these references, we've created a robust simulation that captures the essence of spacecraft aerodynamics. The collaboration between programming and real-world principles enhances our understanding and ensures the accuracy of our results.



Conclusion:

The simulation provides an educational tool for understanding the principles of spacecraft aerodynamics. While it does not encompass the full complexity of NASA's CFD tools, it serves as a foundational platform for exploring the effects of aerodynamic forces on spacecraft.

Future Work: Future enhancements could include the integration of 3D models, varying atmospheric conditions, and more complex aerodynamic calculations to increase the simulation's fidelity.

Acknowledgments: We acknowledge the contributions of the open-source community and the valuable insights provided by NASA's simulation and modeling resources, which have been instrumental in guiding this project.

This report integrates the programming project with theoretical concepts from reliable sources, emphasizing the simulation's relevance to real-world aerospace applications. The code snippets provided offer a glimpse into the programmatic structure that underpins the simulation, showcasing its educational value in the field of spacecraft aerodynamics.

References:

Interactive Simulations | Glenn Research Center | NASA

Cart3D (nasa.gov)

Researchers Test Control Algorithms for NASA SPHERES Satellites with a MATLAB Based Simulator - MATLAB & Simulink (mathworks.com)

NASA@SC19: Simulating Dream Chaser® Spacecraft Aerodynamics: Subsonic through Hypersonic

Simulation & Modeling - NASA

Free Software (nasa.gov)

Dream Chaser at Dawn - NASA

Dream Chaser – Space Station (nasa.gov)

NASA Advanced Supercomputing Division Website

Syllabus AEM 3101 - Mathematical Modeling and Simulation in Aerospace Engineering (umn.edu)

Aerospace Systems Design and Simulation | Aerospace Engineering | UIUC (illinois.edu)

Finite element methodology for modeling aircraft aerodynamics: development, simulation, and validation | Computational Mechanics (springer.com)

Impulse and Momentum Calculator (omnicalculator.com)

Mechanics Map - The Impulse-Momentum Theorem for a Particle (psu.edu)