Computational Fluid Dynamics With a Paper Airplane

Tasada, Daniel Tse, Nathan

October 14, 2024

Abstract

In this paper, we investigate the relationship between an airplane's shape and its performance. Our results show that ...

Contents

Chapter 1

Introduction

This thesis covers the simulation of the aerodynamics of an airplane, using our own Computation Fluid Dynamics (or CFD) model.

The goal is to calculate the ideal shape of an airplane for its aerodynamic performance. We aim to do this by creating a Machine Learning model, and feed it the results of the CFD, which should result in our final model.

The thesis questions are the following:

- How do the different aspects of fluid dynamics work and how do we implement it in a computer program?
- How do we dynamically generate 3D models?
- How does an airplane's wing shape influence its performance?

Chapter 2

Execution

2.1 Methodology

To achieve our goal, we've had first had to learn about particle dynamics. It took a while to get an understanding, but here's what we found:

2.1.1 The Math

Particle collision

Part one of the fluid dynamics simulation is particle collision. Fluid dynamics, in essence is particle physics, where every air molecule is a "particle". We start with a container, and a bunch of air molecules, all of which interact with each other to create a fluid. This interaction is effectively the collisions between particles. It turns out that simulating particle dynamics in two dimensions is pretty straightforward. But when you throw in that Z-axis, it gets a lot harder. The main article I used for this is Rigid Body Collision Resolution (Hakenberg, 2005).

Here is a set of formulas necessary to perform the calculations that describe the behavior of a pair particle before and after their collision:

The following variables are necessary to perform the calculations:

Inertia tensor I $(kg \cdot m^2)$:	$L = \frac{L}{\omega}$;
Angular momentum L $(kg \cdot m^2/s)$:	$L = mvr$;
Angular velocity ω (rad/s) :	$\omega = \frac{\Delta\theta}{\Delta t}$;

Collision normal($n \in \mathbb{R}^3$) in world coordinates away from body 1;
Point of contact ($r_i \in \mathbb{R}^3$) in world coordinates with respect to p_i;
Orientation ($R_i \in SO(3)$) transforming from object to world coordinates;

Where i represents one of two particles in a given collision:

Velocity after collision	$\tilde{v}_i$,
Angular velocity after collision	$\tilde{\omega}_i$,
Constant	λ,

The following formulas represent the relation between particles:

$$\tilde{v}_1 = v_1 - \frac{\lambda}{m_1}n;$$
$$\tilde{v}_2 = v_2 + \frac{\lambda}{m_2}n;$$
$$\tilde{\omega}_1 = \omega_1 - \Delta q_1;$$
$$\tilde{\omega}_2 = \omega_2 + \Delta q_2;$$
$$\text{where } q_i := I_i^{-1} \cdot R_i^{-1} \cdot (r_i \times n),$$
$$\text{and } \lambda = 2\frac{nv_1 - nv_2 + \omega_1 I_1 q_1 - \omega_2 I_2 q_2}{(\frac{1}{m_1} + \frac{1}{m_2})n^2 + q_1 I_1 q_1 + q_2 I_2}$$

2.1.2 The Code

The Structure

We decided to write the simulator in Go, for its simplicity and development ergonomics. We used raylib for the 3D rendering, as it's a simple and easy-to-use library. The `github.com/dtasada/paper` package contains the main entry point of the program, which then uses the `github.com/dtasada/paper/engine` package to perform the interactions between particles.

TODO: Explain container system, maybe show code samples

Chapter 3

Results

Present your experimental or theoretical results in this section. Use tables and figures to illustrate important points.

3.1 Conclusion

Summarize the main findings of the paper. Mention potential future work or research directions.

Bibliography

Ash, M. (2006). Fluid simulation for dummies. www.mikeash.com/pyblog/fluid-simulation-for-dummies.html

Gradience. (2024). *Teaching myself c so i can build a particle simulation.* www.youtu.be/NorXFOobehY

Hakenberg. (2005). Rigid body collision resolution. www.hakenberg.de/diffgeo/collision/rigid_body_collision_resolution.pdf

Lague, S. (2023). *Coding adventure: Simulating fluids.* www.youtu.be/rSKMYc1CQHE

raysan5. (2013). *Raylib.* www.raylib.com

Stam, J. (2012). Real-time fluid dynamics for games. www.dgp.toronto.edu/public_user/stam/reality/Research/pdf/GDC03.pdf

Train, T. C. (2019). *Coding challenge #132 fluid simulation.* www.youtu.be/alhpH6ECFvQ

Unknown. (2012). Lattice-botlzmann fluid dynamics. www.physics.weber.edu/schroeder/javacourse/LatticeBoltzmann.pdf

Work, P. (2023). *Writing a physics engine from scratch - collision detection optimization.* www.youtu.be/9IULfQH7E90