CSE 528 Final Project Proposal

Russell Bentley

Wednesday 25th September, 2024

1 Introduction

My proposal is to implement a solver framework for modeling solid-fluid coupling and apply this framework to simulating several real life phenomenon. Recent work like [2] (published in SIGGRAPH ASIA 2021) has greatly improved the robustness of solid-fluid coupling with kinetic methods like Lattice Boltzmann. My framework will look re-implement the methods described there. Using this framework, I can build different simulations and generate datasets that I can render separately. The final result should be generated images that capture complex interactions between fluids and solids.

2 Proposed Implementation

This project is composed of two components, the solver framework and the rendering pipeline. My aim is to spend most of my time on the solver framework, and largely use off the shelf tools for rendering.

2.1 Solver

This kinetic portion of the solver will largely run on the GPU. While solvers of this nature are typically implemented in CUDA, my goal is to target wgpu instead. Wgpu is a rust library that implements the WebGPU API. Here's an outline of why I want to do it this way:

- I would like to write the host side in Rust, which I prefer to Cpp / C.
- This implementation would be more portable.
  - I want to support Apple GPUs, which often have access to more memory than NVIDIA GPUs.
  - I would like to support GPUS from AMD.
  - Future versions could be deployed to web-browsers.
- This would be trying something new.

The technique I want to implement use point samples from the solid's geometry. There solver framework will also need to handle geometry-preprocessing in order to generate these samples. In particular, I will need to implement poisson sampling over triangular meshes. However, I may also investigate other geometric representations that could be supported. For example, nTop can produce random surface samplings for implicit geometry.

2.2 Rendering

Generating high quality images is an important part of the project. My initial idea is to utilize OpenVDB as an intermediate form for the volumetric data the solver will generate. From there, I would like to use Blender to create volumetric renders. This pipeline will be partially automated, but will also involve some per-scene manual setup.

My backup option for rendering is to use VTK and my intermediate format, and to render with Paraview. For contour and vector plots this may be my best option, so it will likely be supported regardless.

3 Proposed Simulations

I don't want to claim that I can achieve all of these, but I did want to brainstorm some possible ideas on how to demonstrate these techniques.

- **Thin Plates**
  This technique is supposed to better support very thin structures. Demonstrating stable flow around thin plates without any pass-through artifacts would be great.
- **Wind Turbine**
  In [2] they demonstrate their two-way coupling by using a constant flow to spin up a wind turbine with friction.
- **Wing Tip Vortices**
  This phenomenon is well described in aerodynamics literature. It would be interesting to demonstrate them in simulation, as well how the effectiveness of different designed at mitigating them.
- **Terminal Velocity**
  The two-way coupling would allow to study the terminal velocity of different objects and match experimental results.

4 Proposed Timeline

- Demonstrate VDB based volumentric Rendering
  - Within next 2 weeks.

- 2D Fluid Solver no-solids
  - Within the 4 weeks.
- 3D Fluid solver no-solids
  - Within the next 5 weeks.
- Random Surface Sampling
  - Within the next 6 weeks.
- Fluid-Solid coupling
  - Within the next 7 weeks.
- Fluid-solid rendering
  - With final Deliverable

5 Future Work

Combining the coupling techniques from [2] could be combined with other kinetic techniques. In particular, I would be interested in exploring how these coupling techniques could handle thermal, and multi-fluid flows [1]. The goal would be to expand these techniques to useful design problems like heat exchangers.

Another direction would be to explore alternative data structures for the discretization. As the relative volume fraction of the solid grows, a dense grid layout becomes less efficient. There are a variety of sparse data structures supported by the GPU that could be applicable to this problem.

Power diagrams are another avenue to explore. Kinetic solvers are typically implemented on regular discretizations, however a power diagram could offer a more flexible approach with a wider dynamic range.

Lastly, building a renderer into the framework would be a great learning exercise, and would allow for some interesting use-cases like getting deployed via web browsers (see wgpu section.)

References

[1] Wei Li, Kui Wu, and Mathieu Desbrun. "Kinetic Simulation of Turbulent Multifluid Flows". In: *ACM Transactions on Graphics (TOG)* 43.4 (2024), pp. 1–17.

[2] Chaoyang Lyu et al. "Fast and versatile fluid-solid coupling for turbulent flow simulation". In: *ACM Transactions on Graphics* 40.6 (2021), p. 201.