CSE 528 Final Project Proposal

Russell Bentley

Wednesday 16th October, 2024

1 Introduction

Kinetic methods for solving fluid flow, like the *Lattice Boltzman method*(LBM) offer several advantages over traditional methods like finite volume or smooth particle hydrodynamics. In particular, they are highly parallelizable and by construction solve for compressible fluids which negates the need for a global pressure solve. LBM works by modelling the probability distribution of particles traveling between nodes on a fixed set of discretized velocities. LBM does a single time step with two operators. First a streaming operator moves the particle distributions along the lattice. Then a collision operator relaxes the velocities at a given node towards an equilibrium state.

Some recent advances in LBM have focused on increasing accuracy, an issue with traditional LBM implementations. In particular, a lack of *Galilean invariance* was identified as a critical issue [3]. That is, traditional collision operators like BGK will have different behavior depending on the bulk velocity across a given node. BGK in particular relaxes all the moments of velocity at a uniform rate. The central-moment method [1–4] seeks to fix this by both finding moments of velocity relative to the bulk velocity and relaxing these moments at independent rates.

Two recent papers in particular have advanced the use of LBM for accurate treatment of fluids in Computer Graphics [4, 5]. These two papers collectively describe how to find appropriate relaxation rates, handle conversion from real units to LBM units, and an accurate treatment of fluid-solid coupling.

2 Proposed Work

My proposal is to implement an LBM solver framework based on the central-moment principle, as well as implement fluid-solid interation. I do not intend to handle full fluid-solid coupling as described in [5], but I would like to immerse fixed geometry into the fluid flow.

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.

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.

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 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

References

[1] Alessandro De Rosis. "Nonorthogonal central-moments-based lattice Boltzmann scheme in three dimensions". In: *Physical Review E* 95.1 (2017), p. 013310.

[2] Alessandro De Rosis and Kai H Luo. "Role of higher-order Hermite polynomials in the central-moments-based lattice Boltzmann framework". In: *Physical Review E* 99.1 (2019), p. 013301.

[3] Martin Geier, Andreas Greiner, and Jan G Korvink. "Cascaded digital lattice Boltzmann automata for high Reynolds number flow". In: *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics* 73.6 (2006), p. 066705.

[4] Wei Li et al. "Fast and scalable turbulent flow simulation with two-way coupling." In: *ACM Trans. Graph.* 39.4 (2020), p. 47.

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