CS 230 Computer Graphics | **Jialiang Liu**
Project 3 Write-up | SID: X664851
Prof. Craig Shroeder | Due Date: 03/20/2019

A Fluid Simulation Based on SPH

Even though the topic I chose was an easy one, and indeed, the paper[1] gives most of the equations that are needed, which should make it not hard to understand and implement. But maybe it's due to my weakness in calculus and programming, it was hard for me to get started with this project. So I searched for lots of blogs and articles about implementing SPH, and most of them have ideas came from this paper of Müller. So I basically followed those articles, and some parts of my code came from others' work. I probably shouldn't do that, but since the topic isn't a hard one, I want to at least turn in something, and I can't do it by myself. The articles I read are listed in the references.

Accomplishment of Goals

These are the goals I was supposed to reach:

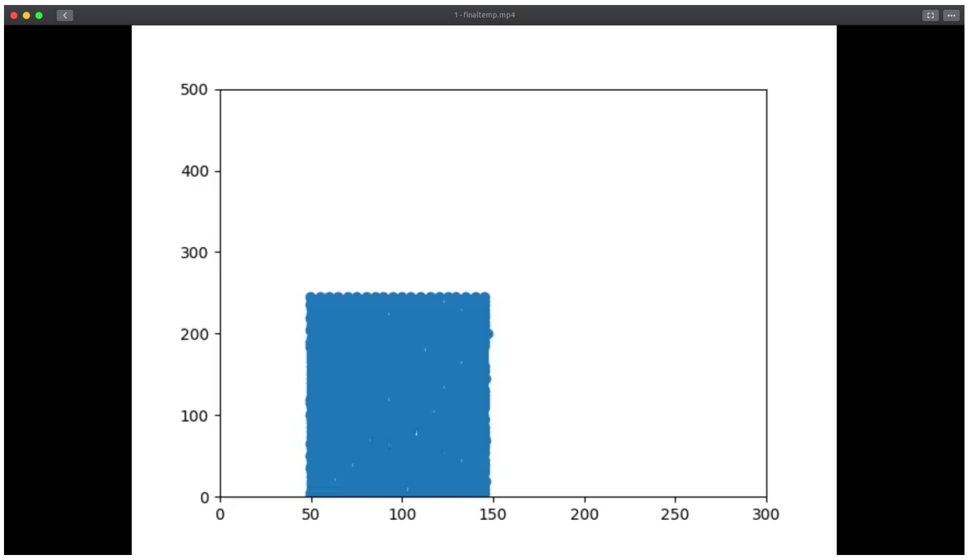
- Compute the properties for the particles. **(Fully reached)**

  Based on the equations from the paper, and an article called "SPH survival kit"[2], I programmed functions that can compute the density, pressure, force, velosity, and location of each particle.

- Render the particles with OpenGL, and render the fluid with POV-Ray. **(Partially reached in different way)**

  Since I chose Python as my programming language for this project, it's easier to use matplotlib to plot the particles instead of rendering them with OpenGL. For render particles to fluid, I didn't have enough time to do it.

  Though I didn't try, I know there are lots of work to do for rendering the fluid. First, the surface of fluid needs to be computed, which will involve Marching Cube. (I only know the name, haven't studied about it.) Second, fluids have refraction, reflection, and specular light, which all need to be implemented.



- Simulate a pre-programmed swirl to a glass of fluid, the swirl will be given by the mouse if I have time to implement. **(Failed)**

  Instead of doing interaction (a swirl) with the fluid, I implemented a ”dam break” simulation. The reason is explained in the ”Problems” section. I also tried giving the dam a initial velosity, but it’s not obvious as a swirl.

Problems

- Locate particles that near a specific particle.

  To reduce the time complexity of locating near particles, spatial grid is normally used. Müller also had a paper[4] on this topic. Unfortunately, I haven’t encountered spatial hashing before, and also had some problem understanding how to implement it. But this is an essential part for the fluid simulation. I managed to find a Python code[5] for spatial hashing online, but it was used for 2-D implementing with dam break. I tried to modify it to 3-D with simulating ”pouring” water, but never succeed. This is also the reason why I implemented a dam break instead of swirl.

  I also encountered some indexing problems with how to use this code I found. But it got solved by reading the code and simply trying.

- Smoothing kernels from the paper.

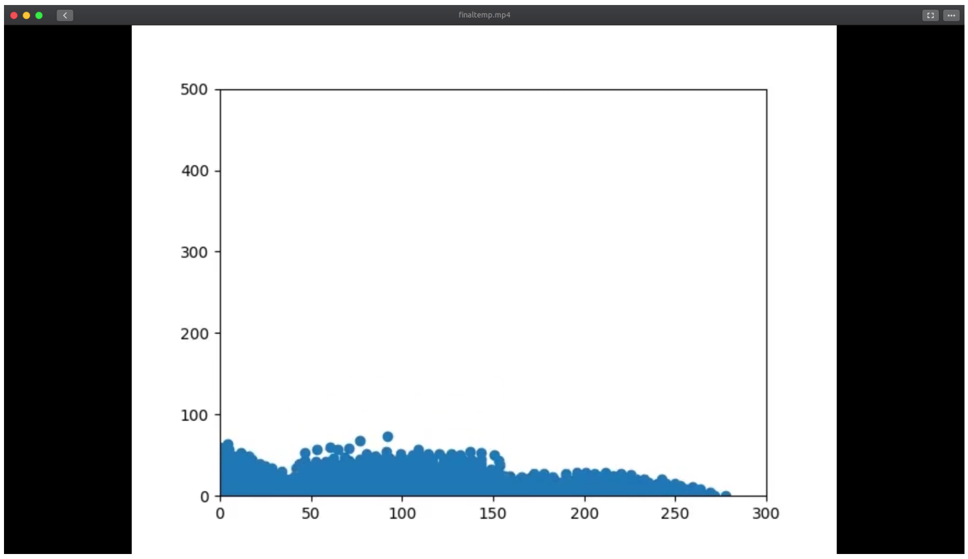
  In the paper, three smoothing kernels were provided for computing pressure and force. But I didn’t realize that I need to use the gradient and laplacian form for two of those kernels. But then I saw it from one of the articles I found and fixed my code.

- Time complexity.

  As the properties of a particle is not only decided by the particle itself, but also affected by its neighboring particles, the computation cannot benefit from the broadcasting from Python, and also the time complexity is determined by the efficiency of spatial hash. I can’t improve it, but I think it has to be fixed for real-time simulation.

Things I’ve Learned

- In class, we mainly talked about how to deduce the equations. With this project, I learned how to use them in practice to compute the results I need.
- Although I’m still not quite sure about how to implement spatial hashing, but I’ve gotten some basic ideas of it.
- I learned how to use matplotlib to create an animation, since I only knew it can plot images. As Python is my favorite language, it’s nice to know it.



If I'm going to do this project again from the begining, the first thing I would do is choosing C++ as my programming language. Not mention most of the articles I found was for programming under C++, it's more likely to use C++ in graphic field. And I will learn about spatial hashing and marching cubes, in order to implement and render the 3-D fluid.

References

[1] M. Müller; B. Solenthaler; R. Keiser; M. Gross (2005). "Particle-based fluid-fluid interaction". *In Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation* (pp. 237-244). ACM.

[2] "SPH survival kit", accessed on 13 March 2019. Available on https://www8.cs.umu.se/kurser/TDBD24/VT06/lectures/sphsurvivalkit.pdf

[3] "Implementing SPH in 2D", 8 July 2017, accessed on 28 February 2019. https://bigtheta.io/2017/07/08/implementing-sph-in-2d.html

[4] M. Teschner; B. Heidelberger; M. Mller; D. Pomerantes; M. H. Gross (2003, November). "Optimized Spatial Hashing for Collision Detection of Deformable Objects". *In Vmv* (Vol. 3, pp. 47-54).

[5] "Summary of Fluid Simulation Based on SPH", 27 October 2013, accessed on 13 March 2019. https://blog.csdn.net/changbaolong/article/details/13172079