Position Based Fluids optimized using Machine Learning

Utkarsh Singhal, Yiding Jiang, Gauthier Dieppedalle
CS184/284A Final Project, Spring 2018

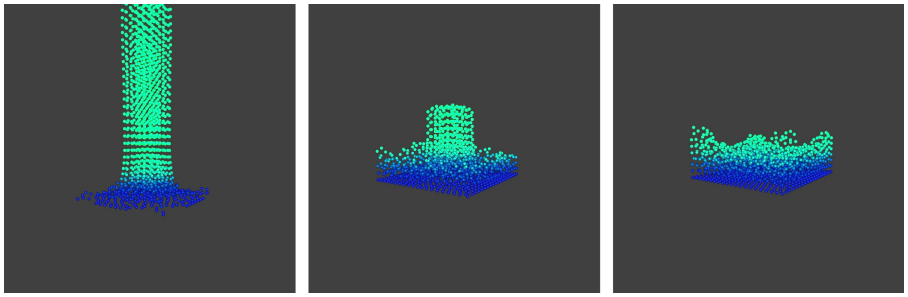

Figure 1: *Rendering of particles of water using Position Based Fluids*

Abstract

Fluid simulation is an interesting topic that has many important applications in computer graphics. We are interested in producing photo realistic simulation of water and understanding the underlying physics of water; therefore, we think particle based simulation is an ideal candidate. It is important to have physically realistic simulation of water because many engineering applications require accurate simulations that reflect the actual physical interaction. Over the course of the past couple of weeks we have been implementing the Position Based Fluids paper written by Nvidia [Macklin and Müller 2013]. Currently, we have setup the project environment, implemented the physics of the particles of the fluid, and started exploring different surfacing techniques. In the next two weeks we will finish implementing the surfacing and use TensorFlow for the Machine Learning aspect of the project. [Jeong et al. 2015] We are currently considering using a neural network with the neighbors position, the velocity, and the direction of the particles to estimate the state of the fluid.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Physically based modeling I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation;

Keywords: fluid simulation, SPH, PCISPH, constraint fluids, position based dynamics, machine learning

1 Setup of the environment

The initial structure of the project has been taken from Project 4: Cloth Simulation. We are using OpenGL to render the particles as well as nanogui for the buttons that adjust the parameters in the scene. All of the parameters in the scene are initialized by a JSON file, such as the number of particles, the initial position of the particles, and the size of the box. We implemented our own particle class with properties like position, velocity, viscosity etc. Each particle is rendered using the OpenGL particle class. We then added particle collision between the plane and the particles so that they would stay in the box create. We are currently in the process of implementing collisions between particles and general polygons, so that we could import any .dae file in our scene.

2 Physics of the Particles

To implement the Position Based Fluid paper, we needed to get the nearest neighbors for each particle in order to ensure that the liquid has a constant density. At first we used a grid based approach in which each neighbor particles would be in cells next to each other. The grid would be represented by a hash table. We ran into efficiency issues and decided to use the FLANN (a KD Tree implementation) library to quickly find neighbor particles. Using FLANN has made our algorithm much faster.

We then were able to represent the physics behind the particles using the Position Based Fluids paper. We first had to enforce constant density in the fluid to enforce incompressibility by solving a system of non-linear constraints. We implemented tensile instability to add clustering between particles of our fluid. Vorticity confinement and viscosity were then added to introduce turbulent motion between the particles.

3 Surfacing and Rendering

We started to look into implementing the surfacing and rendering for our fluid. Initially we wanted to use the Volume Rendering Plugin from Mitsuba to render water. We have written a script that converts particles to voxels into a format that is accepted by Mitsuba. However, we were not able to make a surface that looks like water as a BSDF cannot be simply applied to a medium in Mitsuba. The plugin that we tried using abstracted too many functionalities that Mitsuba offered. Our plan for the next week would be to implement the Marching Cube algorithm in our C++ code and import the mesh directly in Mitsuba [Lorensen and Cline 1987]. A BSDF representing water can then be easily applied to a mesh in Mitsuba.

4 Slides

`https://docs.google.com/presentation/d/1WFHYo0yR2pkavF79elH-taTa5xkzhFRqttoH_YvyOIA/edit?usp=sharing`

5 Video

`https://youtu.be/GPTqNQEA31k`

Acknowledgements

We wanted to say thank you to the CS184/284A staff for helping us throughout the past couple of months answering clarifying question about the implementation of the papers.

References

JEONG, S., SOLENTHALER, B., POLLEFEYS, M., GROSS, M., ET AL. 2015. Data-driven fluid simulations using regression forests. *ACM Transactions on Graphics (TOG) 34*, 6, 199.

LORENSEN, W. E., AND CLINE, H. E. 1987. Marching cubes: A high resolution 3d surface construction algorithm. In *ACM siggraph computer graphics*, vol. 21, ACM, 163–169.

MACKLIN, M., AND MÜLLER, M. 2013. Position based fluids. *ACM Transactions on Graphics (TOG) 32*, 4, 104.