Milestone Report: Powder Master

2D Particle Simulation Game

Roy Guo Christine Sheng Mingcheng Zhu

ABSTRACT

This project explores options to optimize real-time 2D particle simulation without sacrificing simulation accuracy. We simulate particle to particle interaction as well as external forces such as wind. We also add an interactive layer on top of the simulation, so the user can enjoy the excitement of creation and exploration.

1 Introduction

Our goal is to produce a 2D simulation game modeling various "powders" that can float around in a fluid as well as interact with each other. This will involve a grid-based fluid simulation using Navier-Stokes equations, as well as a particle system for the powders. The particles will interact with each other using a simple cellular automata.

Several challenges we will have to solve include how to simulate interactions between our particle system and our grid-based system, and how to optimize the simulation so that it can be run in real-time as a game. The simulation will likely have to deal with tens of thousands of particles, as well as thousands of cells in our fluid model, so there will have to be trade-offs made in how fine-grained we want the different parts of our simulation to be.

2 Related Work

The particle system is implemented referencing Position Based Fluids, by Macklin and Muller. We modify the Smoothed Particle Hydrodynamics, as described in the paper, for 2D simulation. We were also using Google's LiquidFun as a reference.

To accelerate the simulation, binning is also implemented. After binning, the simulation runtime complexity goes from O(n^2) to O(n).

The underlying fluid simulation is done using a eulerian approach. The algorithm is based off of Mick West's "Practical Fluid Mechanics" for game development.

3 METHOD

3.1 Particle System Simulation

As described in Position Based Fluids, the system has one constraint for density per particle, which is defined as follows for particle *pi*:

$$\frac{\rho i}{\rho 0} - Ci(p1...pn) = 1 \quad (1)$$

p1...pn are the neighbor particles of *pi*. ρi is the SPH density estimator:

$$\rho i = \sum_{j=1}^{n} W(pi - pj, h) \quad (2)$$

W is the weight function and *h* is particle radius. We use the poly6 kernel as our weight function:

$$Wpoly6(r, h) = \frac{315}{64\pi h^9}(h^2 - |r|^2)^3 \quad (3)$$

For every frame, our goal is to find a Δp for each particle such that $C(p + \Delta p) = 0$. Δp is found by a series of Newton steps along the constraint gradient:

$$\Delta p \approx \nabla C(p)\lambda \quad (4)$$

We need to solve for particle density constraint λ in order to find Δp, which is found by

$$\lambda i = -\frac{Ci(p1,...,pn)}{\sum k |\nabla pk Ci|^2} \quad (5)$$

Then we use it to update the particle location.

3.2 Grid-based Fluid Simulation

The simulation uses advection on a grid based model to simulate the underlying fluid ("air") in the game. The game space is divided into equally sized square cells, and each cell contains three fields: pressure, x velocity, and y velocity.

At each step in the simulation, each of these fields is advected by the velocity at the cell. That is, for each cell, the value in that cell is moved based on the velocity, and the value is distributed to the surrounding cells based on proximity. This is known as forward advection.

In our program, first the pressure is advected, and then the velocities are advected. However, forward advection alone does not simulate an incompressible fluid. We chose to implement pressure as a simple and fast way to simulate this. Based on the difference in pressure between neighboring cells, the velocity of

those two cells is changed to simulate the flow of the fluid from areas of high pressure to low pressure. This ideally causes the simulation to tend towards states where pressure is equal across all cells.

4 Progress

For user interaction, we have implemented a mouse click control for particle generation. The next step would be to be able to select different kinds of particles and tools.

For the particle system, we are able to simulate fluid-like particle-to-particle simulation. It is unoptimized and currently only supports around 300 particles to run in real-time. In addition, the bottom particle density seems higher than the upper parts.

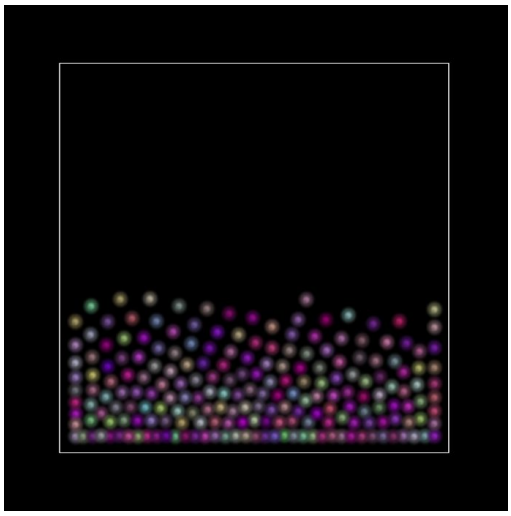

Fig1. Particle system with 200 particles

For the current fluid simulation, the restoring force caused by pressure differences is not tuned properly, so the fluid tends to have very chaotic behavior as time goes on. The velocity in the fluid seems to constantly accelerate as time goes on rather than stabilizing. We will have to experiment with other grid based simulation techniques so that the simulation is stable.

5 Reference

1. Efficient Spatial Binning on the GPU
2. LiquidFun
3. Position Based Fluids
4. Practical Fluid Mechanics