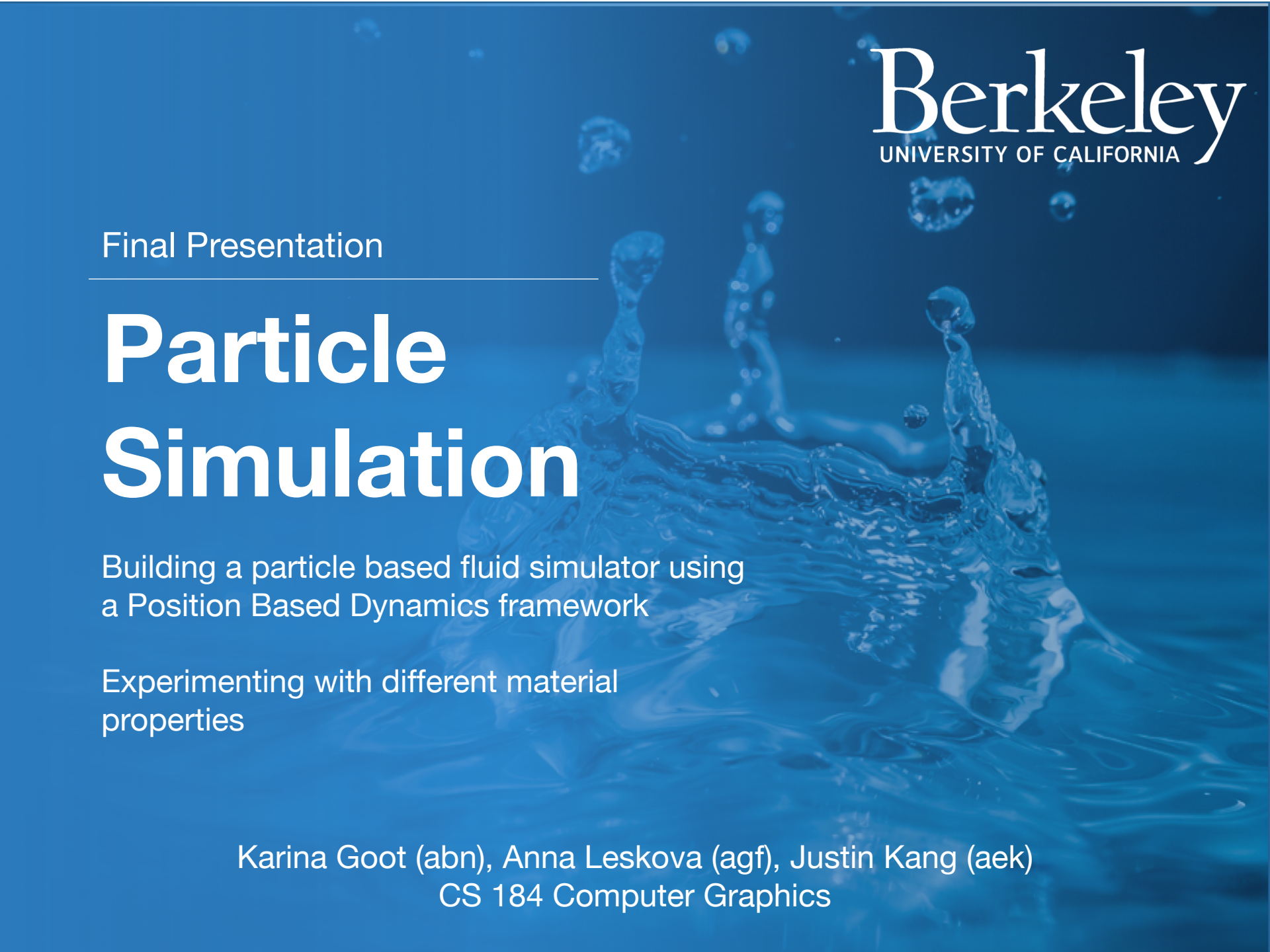


Final Presentation

Particle Simulation

Building a particle based fluid simulator using a Position Based Dynamics framework

Experimenting with different material properties

Karina Goot (abn), Anna Leskova (agf), Justin Kang (aek)
CS 184 Computer Graphics

Agenda

Demo

Results

Introduction

Implementation

Conclusion

Background

Simulating fluids has been a long standing challenge in computer graphics

- Liquids are super awesome and add interesting features to any scene
- Relevant in computer games, movies, and 9gag gifs!

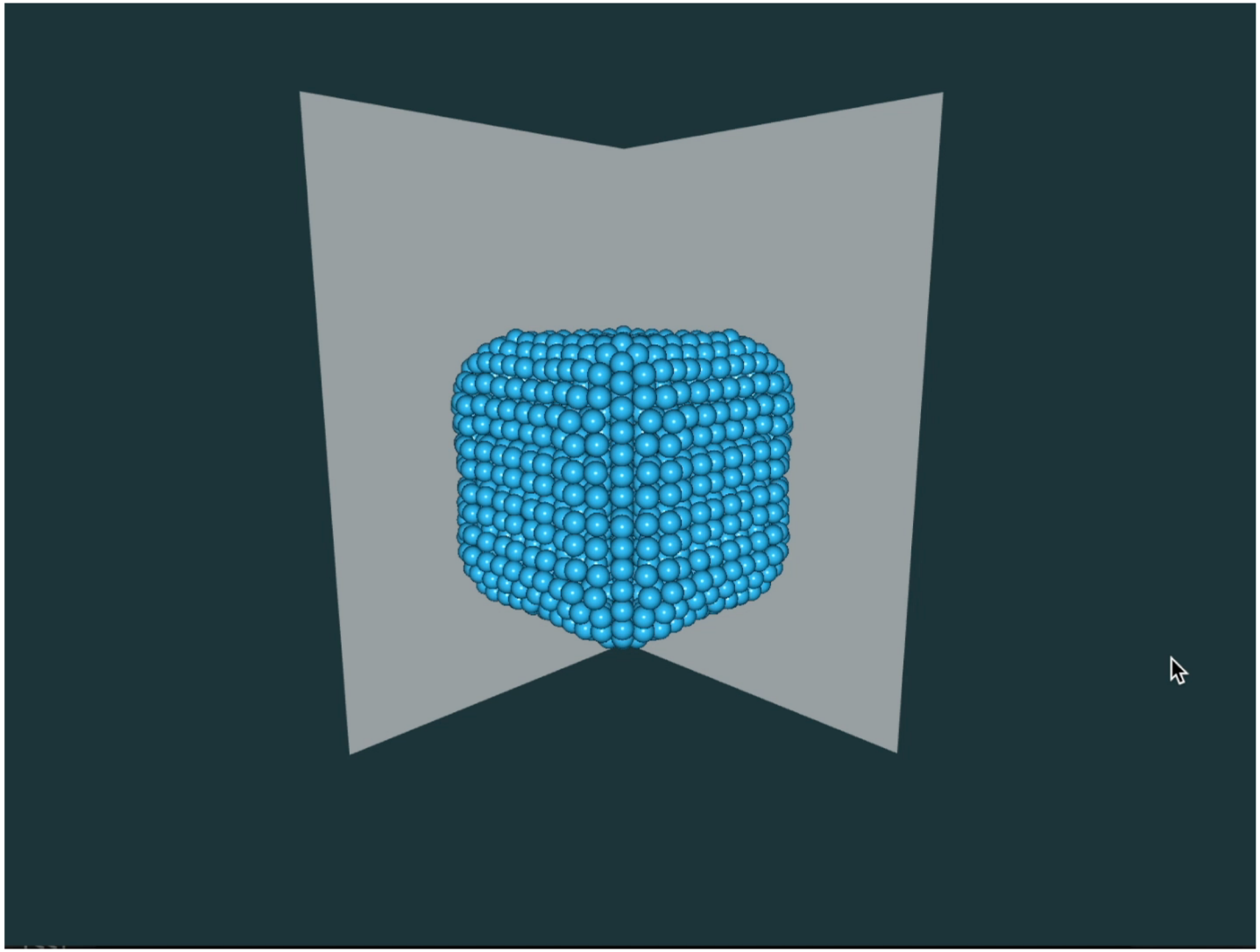
The goal is to build a particle based method for fluid simulation in real time

- Draw inspiration from an NVIDIA paper by Macklin and Muller
- Study how incompressible flow can be simulated using a Position Base Dynamics (PBD) framework

We started with Dillon Yao's code found in the f2 repository

- It contained a really nice project directory structure and make files that were beautifully set up
- GUI that build an empty scene where particles can reside

Demo



Implementation Overview

Position Based Fluids by Miles Macklin and Matthias Muller

Position Based Dynamics

With an iterative density solver to converge to smoothed particle hydrodynamic solvers

Incompressible Flow

1. Using particle density constraints and constraint gradients to find particle lambda values

$$\lambda_i = -\frac{C_i(\mathbf{p}_1, \ldots, \mathbf{p}_n)}{\sum_k \left|\nabla_{\mathbf{p}_k} C_i\right|^2 + \varepsilon},$$

2. Utilize this lambda value to calculate the next particle positior

$$\Delta \mathbf{p}_i = \frac{1}{\rho_0} \sum_j (\lambda_i + \lambda_j) \nabla W(\mathbf{p}_i - \mathbf{p}_j, h).$$

Vorticity Confinement and Viscosity

1. Estimate particle voriticity to find a corrective force for particle

$$\mathbf{f}_i^{vorticity} = \varepsilon (\mathbf{N} \times \omega_i).$$

2. Apply XSPH viscosity to calculate new particle velocity

$$\mathbf{v}_i^{new} = \mathbf{v}_i + c \sum_j \mathbf{v}_{ij} \cdot W(\mathbf{p}_i - \mathbf{p}_j, h)$$

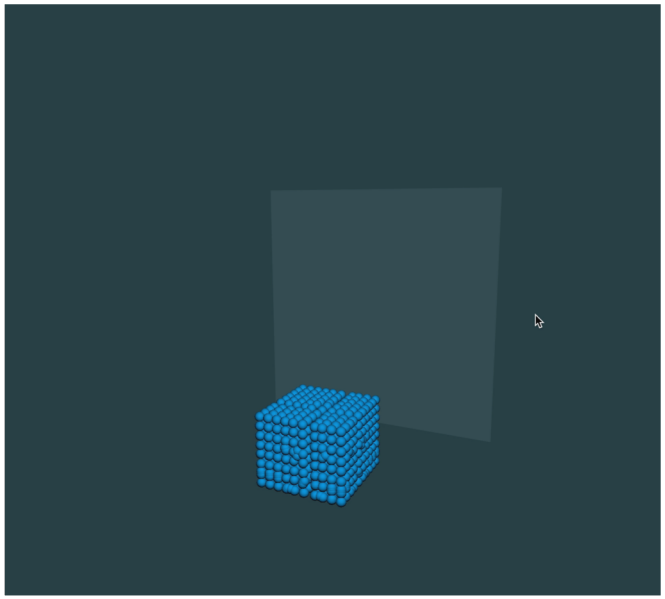
Algorithm 1 Simulation Loop

for all particles i **do**
  apply forces $\mathbf{v}_i \Leftarrow \mathbf{v}_i + \Delta t \mathbf{f}_{ext}(\mathbf{x}_i)$
  predict position $\mathbf{x}_i^* \Leftarrow \mathbf{x}_i + \Delta t \mathbf{v}_i$
end for
for all particles i **do**
  find neighboring particles $N_i(\mathbf{x}_i^*)$
end for
while $iter < solverIterations$ **do**
  for all particles i **do**
    calculate λ_i
  end for
  for all particles i **do**
    calculate $\Delta \mathbf{p}_i$
    perform collision detection and response
  end for
  for all particles i **do**
    update position $\mathbf{x}_i^* \Leftarrow \mathbf{x}_i^* + \Delta \mathbf{p}_i$
  end for
end while
for all particles i **do**
  update velocity $\mathbf{v}_i \Leftarrow \frac{1}{\Delta t}(\mathbf{x}_i^* - \mathbf{x}_i)$
  apply vorticity confinement and XSPH viscosity
  update position $\mathbf{x}_i \Leftarrow \mathbf{x}_i^*$
end for

Implementation Hurdles

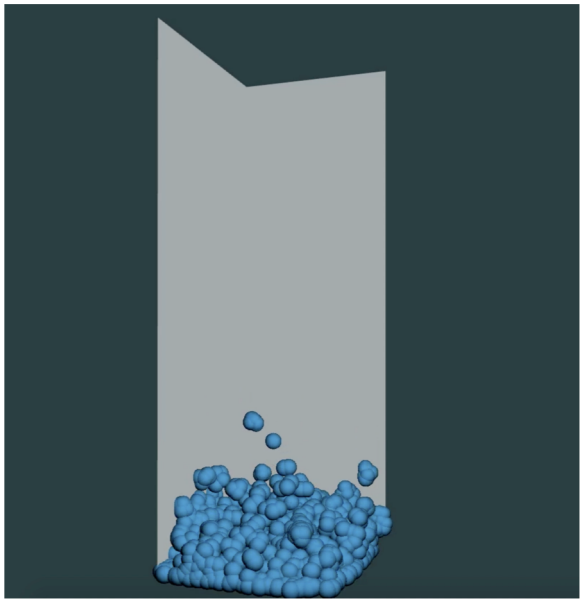
Box Set-Up

Plane collisions were not set up correctly and particles would escape corners and edges



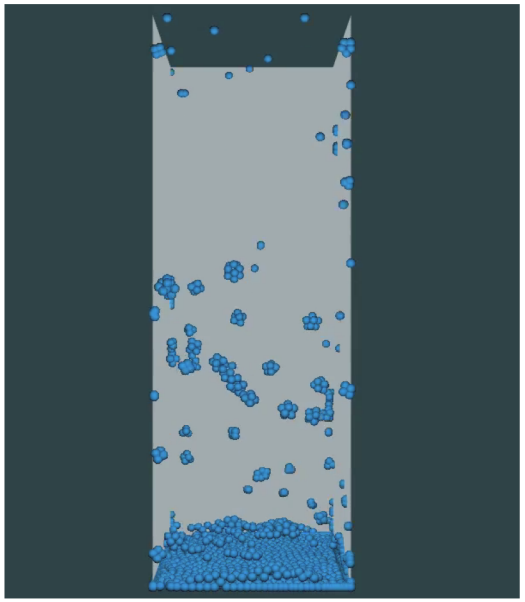
Boiling Water

Bugs in particle neighbors resulted in particles pushing each other away

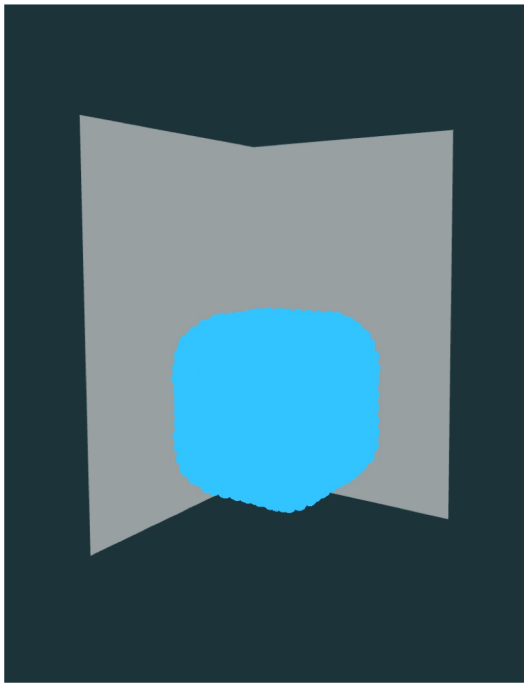


Vorticity

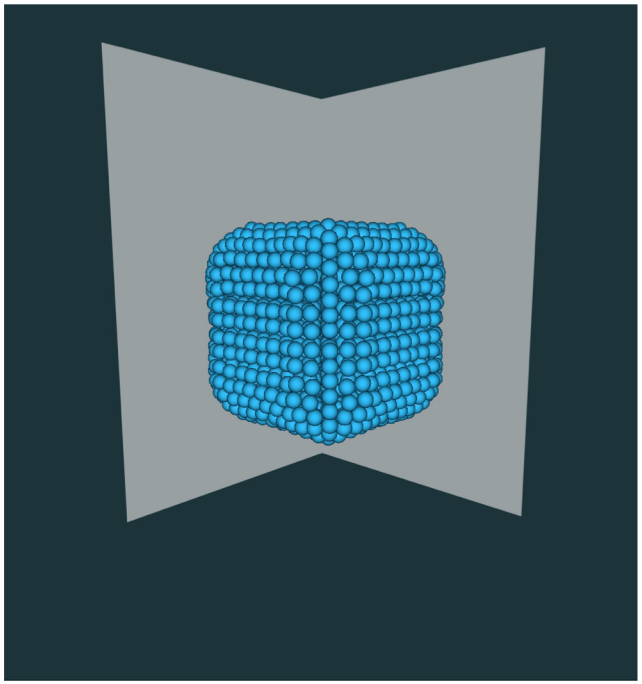
Bugs in our vorticity and viscosity implementation led to funky clumps and anti-gravity



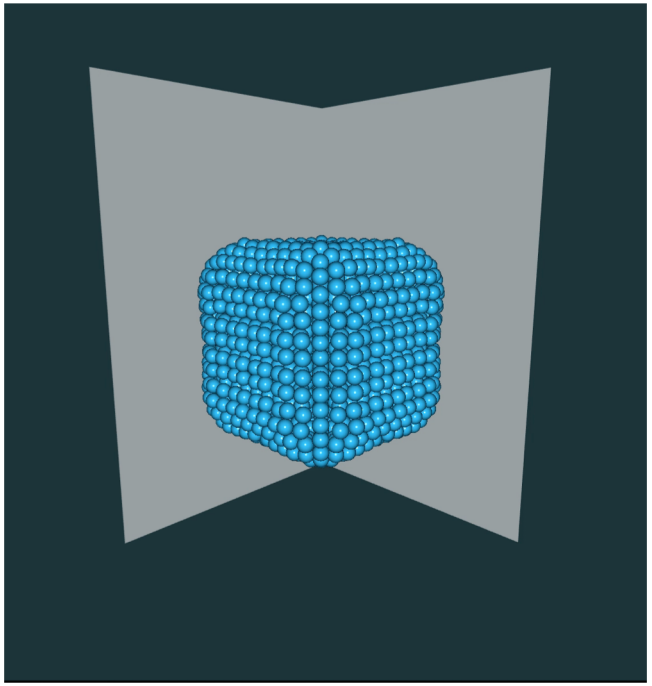
Results -- Water



No particle shading

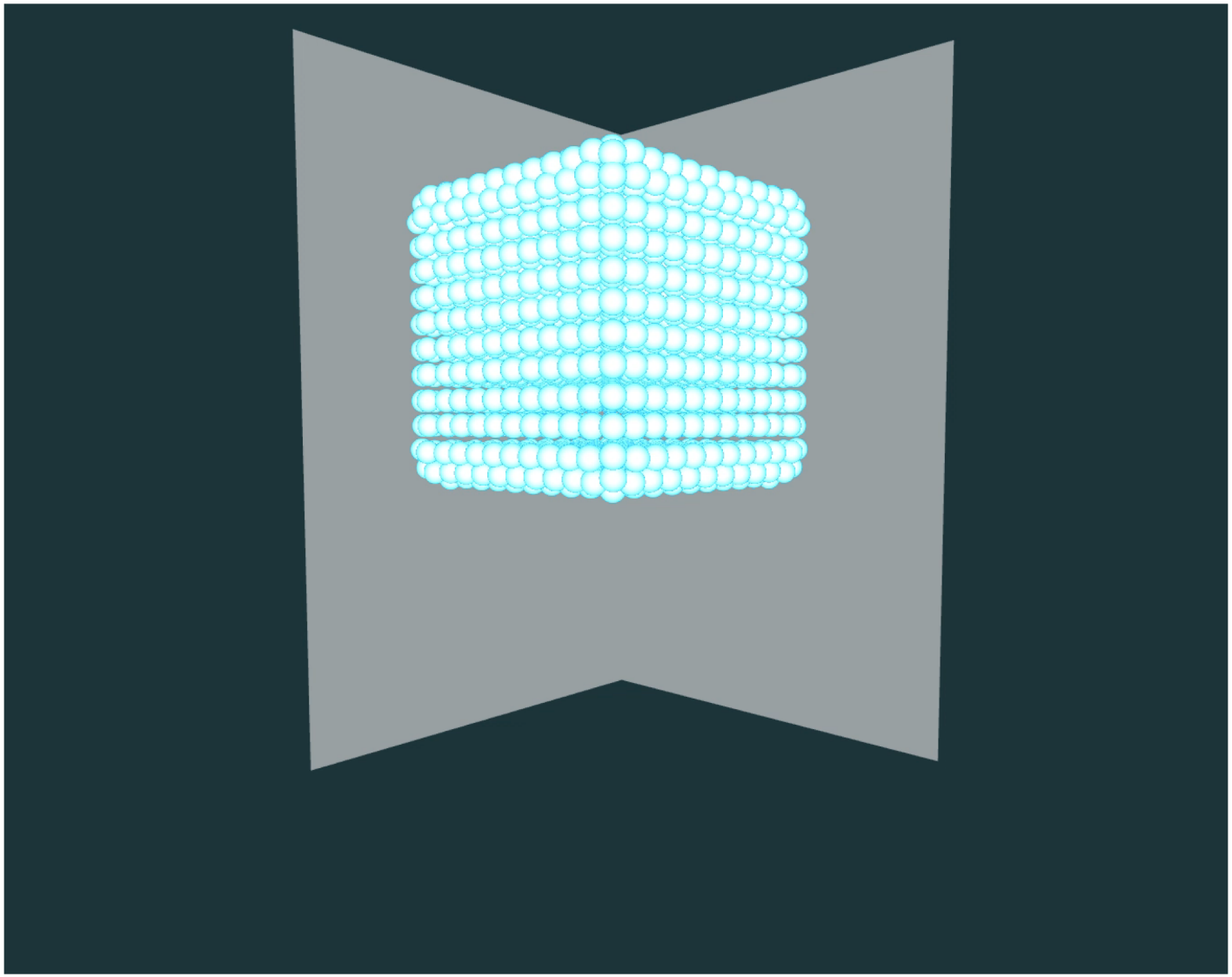


Blinn Phong Diffuse Shading

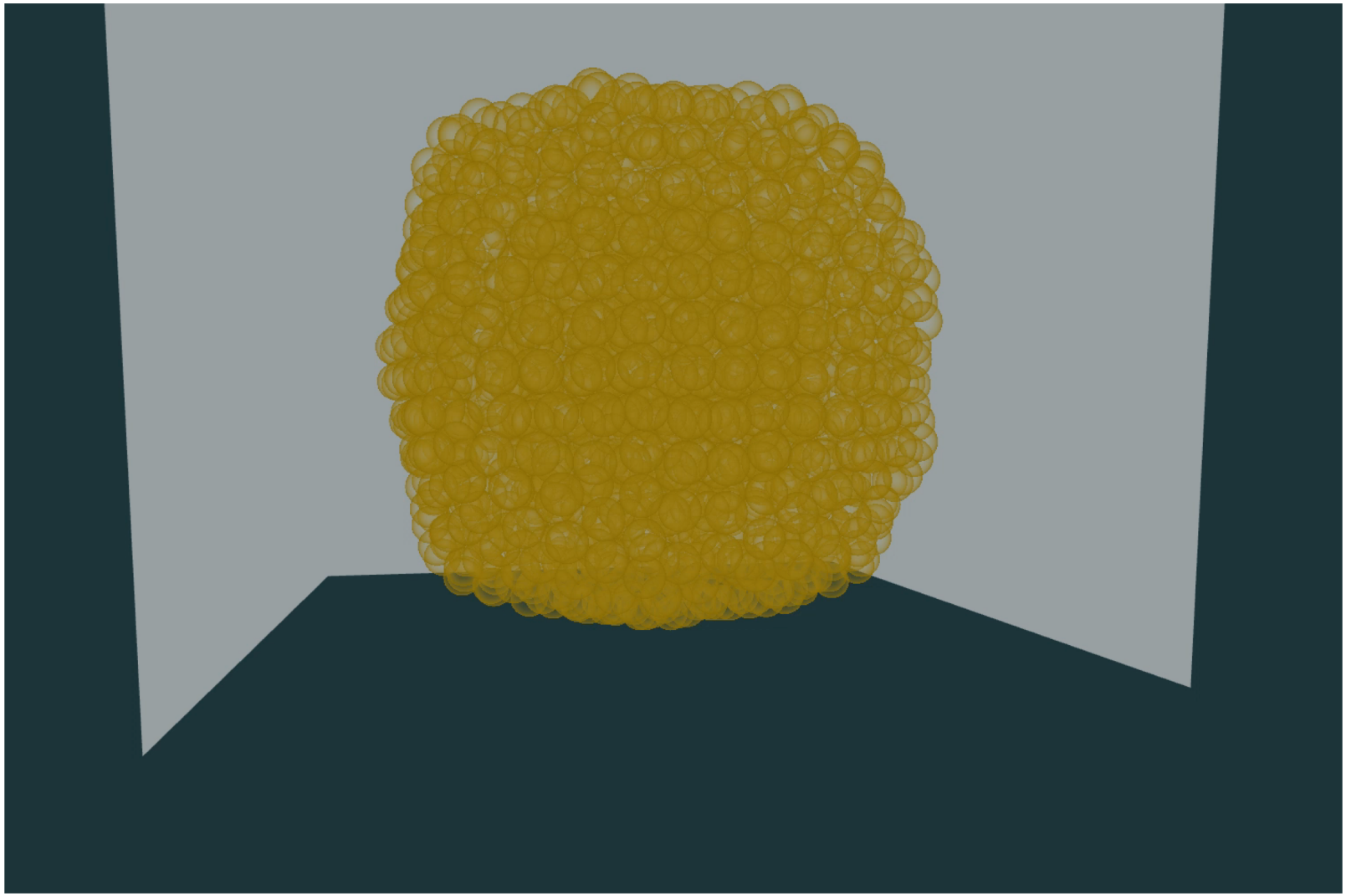


Bling Phong Diffuse & Specular

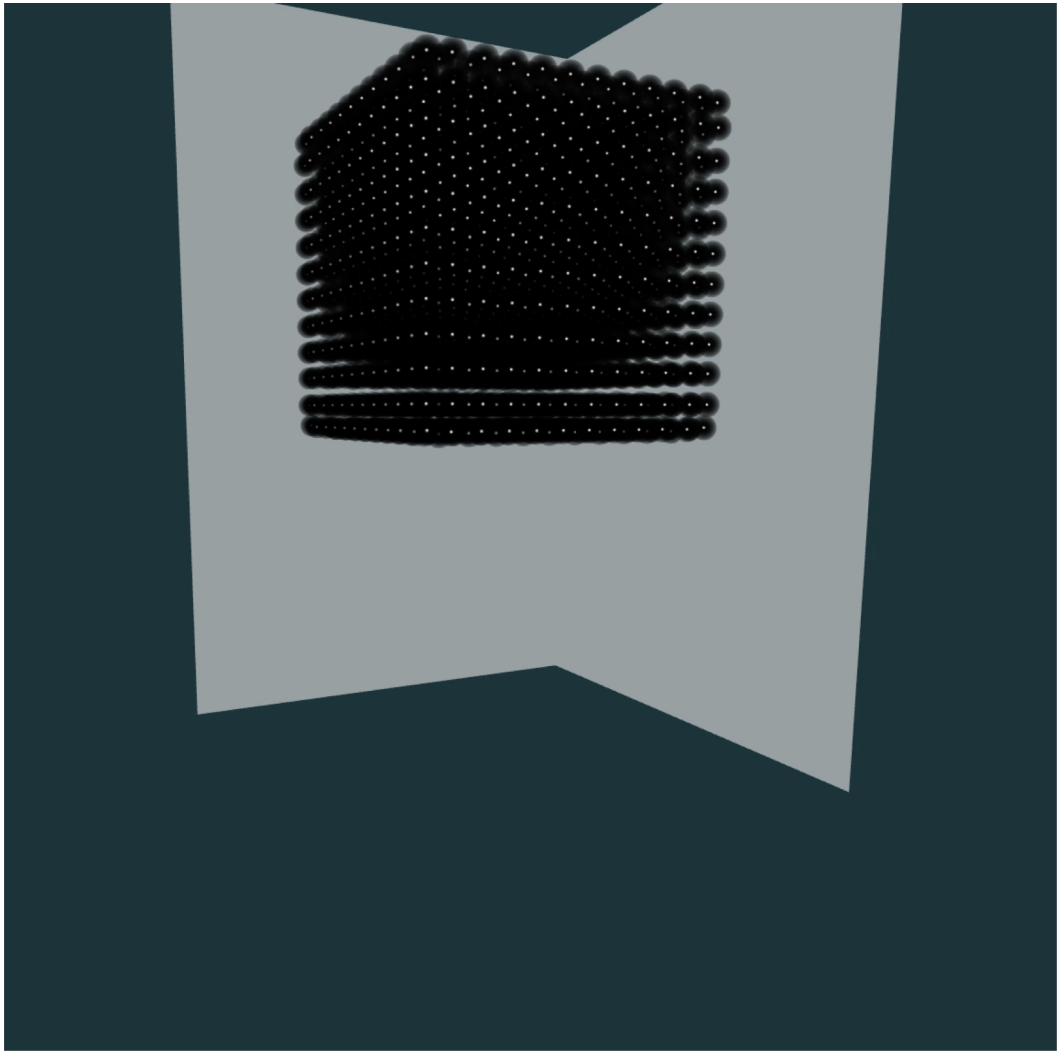
Results – Styrofoam Balls



Results – Oil



Results – Boba!



Boba particle simulation uses a velocity based color changing scheme

Faster particles are more transparent

Conclusion

Lessons Learned:

- Fluid Dynamics is fascinating and we wish we had time to take physics while in college
- Working across three different computers in parallel can be challenging and exciting
- Optimizing runtime by sacrificing memory can have tremendous improvements to over all performance

Our Favorite Part:

Seeing Justin's face when he finally got vorticity to work

Thank You:

Thanks Dillon for the countless debugging advice, and ridiculous patience