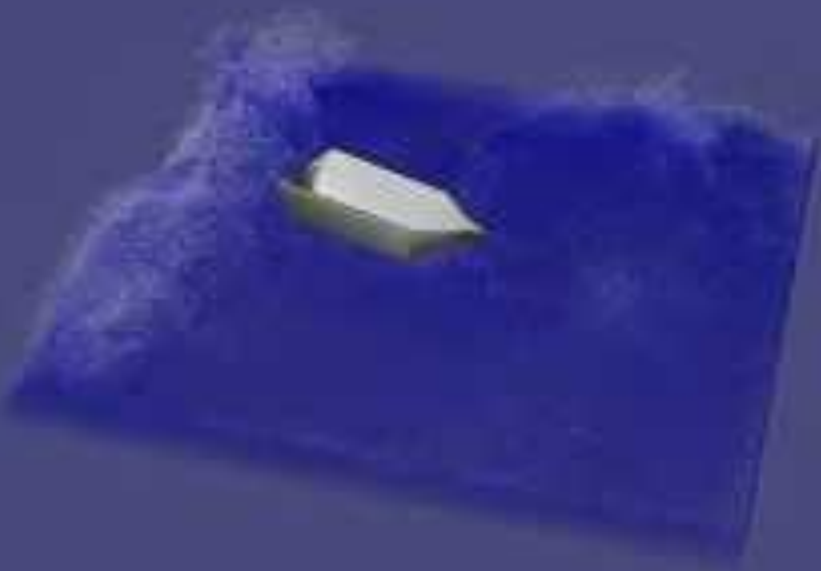
Project: Waterboat

Group: PhysIT
Phillip Løjmand, Daniil Emtsev, Anton Maksimov



Physics

1. Explicit WCSPH solver with boundary handling [1]

[1] PBS lecture on SPH + Eurographics Tutorial 2019, section “Boundary handling”

Physics

1. Explicit WCSPH solver with boundary handling [1]
2. Two-way rigid body-water coupling [2]

Repulsion $$F^{p}_{i\leftarrow j} = -m_i m_j \left(\frac{p_j}{\rho_i \rho_j}\right) \nabla W_{ij}$$

Friction $$F^{v}_{i\leftarrow j} = -m_i m_j \nu \left(\frac{max(v_{ij} x_{ij},0)}{|x_{ij}|^2 + \epsilon h^2}\right) \nabla W_{ij}$$

[1] PBS lecture on SPH + Eurographics Tutorial 2019, section “Boundary handling”

[2] Akinci et al., “Versatile Rigid-Fluid Coupling for Incompressible SPH”, 2012

Physics

1. Explicit WCSPH solver with boundary handling [1]
2. Two-way rigid body-water coupling [2]

Repulsion $$F^{p}_{i\leftarrow j} = -m_i m_j \left(\frac{p_j}{\rho_i \rho_j} \right) \nabla W_{ij}$$

Friction $$F^{v}_{i\leftarrow j} = -m_i m_j \nu \left(\frac{max(v_{ij} x_{ij}, 0)}{|x_{ij}|^2 + \epsilon h^2} \right) \nabla W_{ij}$$

1. Rigid body movement with boundary repulsion

[1] PBS lecture on SPH + Eurographics Tutorial 2019, section “Boundary handling”
[2] Akinci et al., “Versatile Rigid-Fluid Coupling for Incompressible SPH”, 2012

Physics

1. Explicit WCSPH solver with boundary handling [1]
2. Two-way rigid body-water coupling [2]

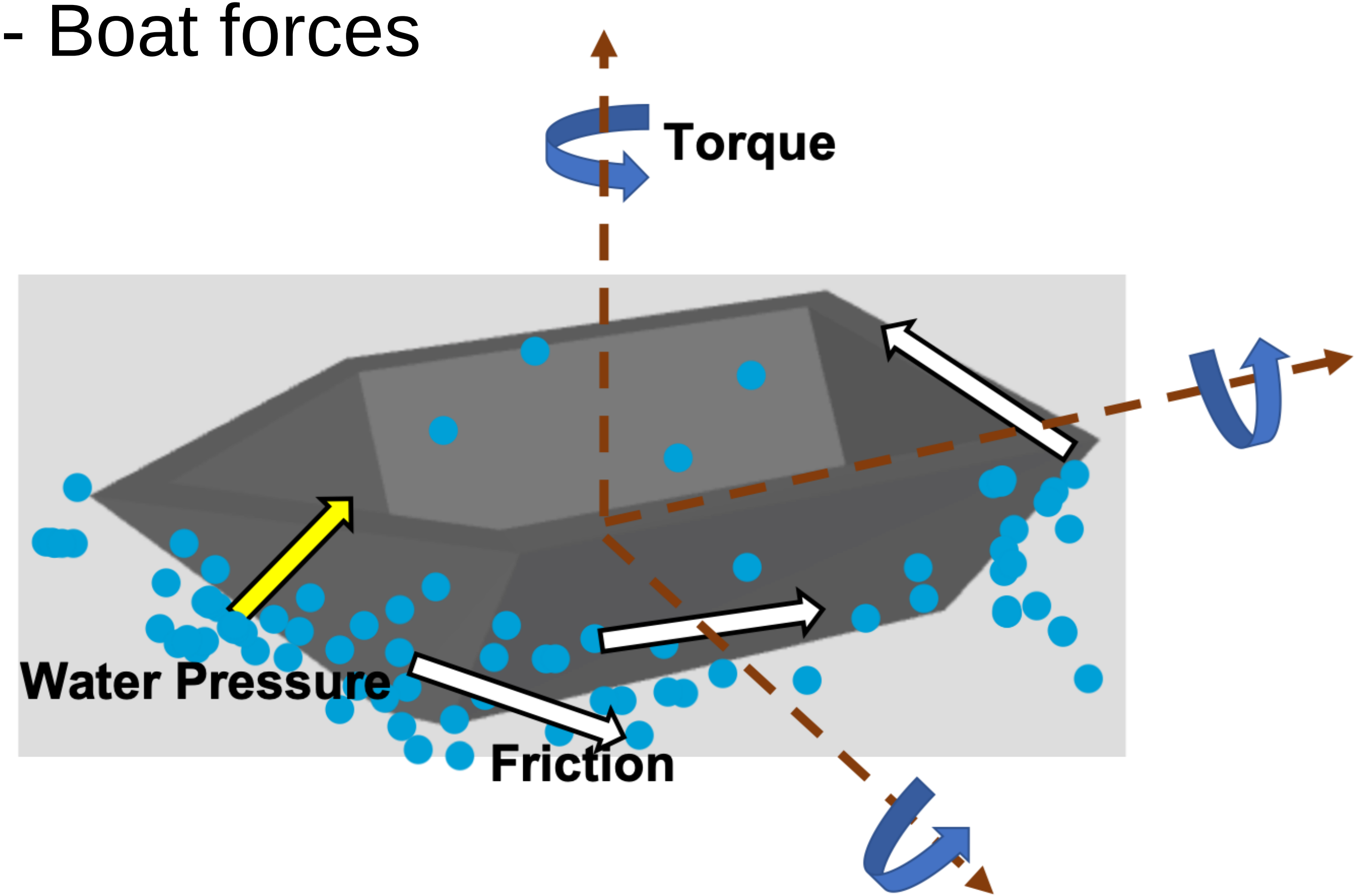
Repulsion $$F^{p}_{i\leftarrow j} = -m_i m_j \left(\frac{p_j}{\rho_i \rho_j}\right) \nabla W_{ij}$$

Friction $$F^{v}_{i\leftarrow j} = -m_i m_j \nu \left(\frac{max(v_{ij} x_{ij}, 0)}{|x_{ij}|^2 + \epsilon h^2}\right) \nabla W_{ij}$$

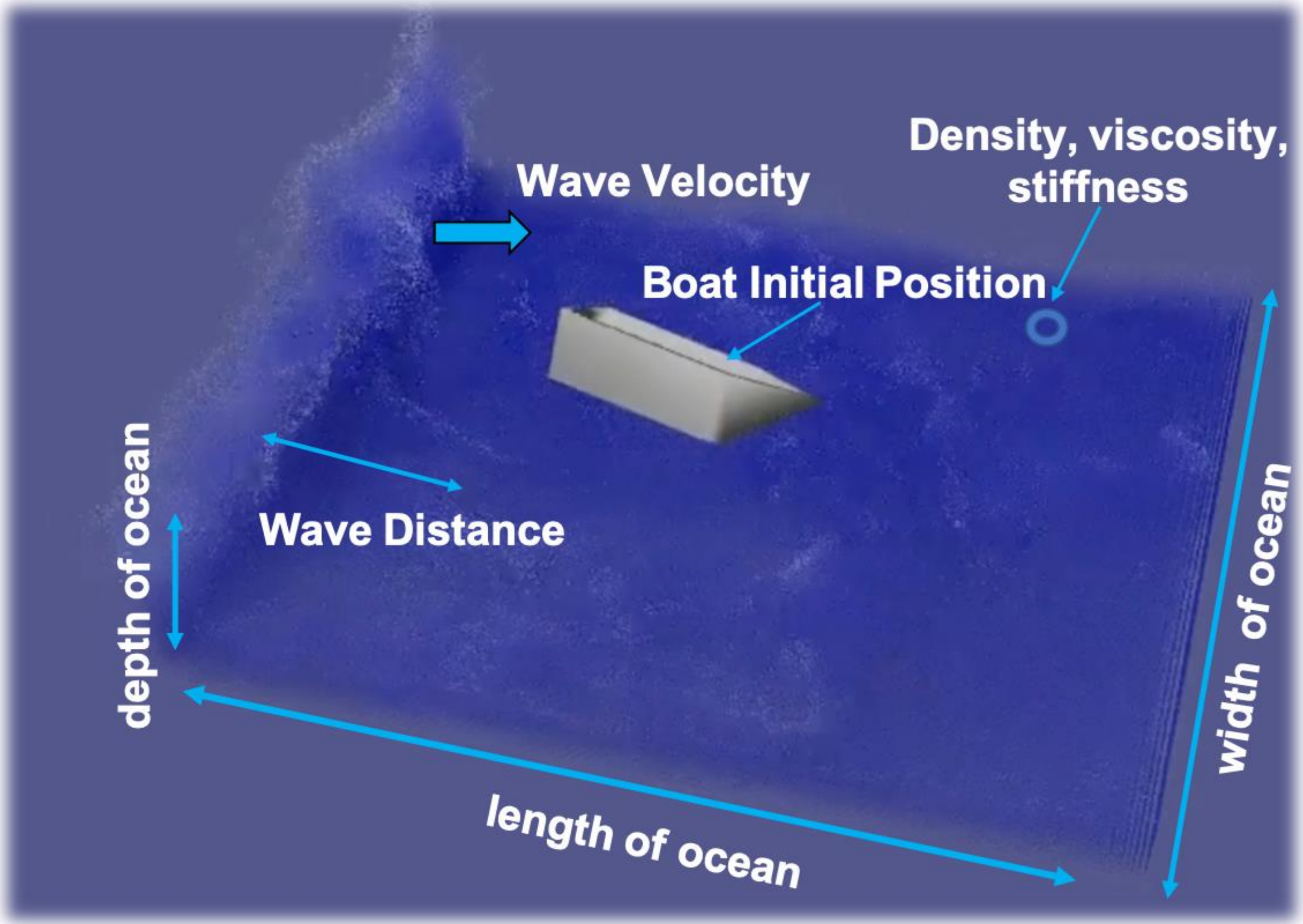
3. Rigid body movement with boundary repulsion
4. *Parameters tuning*

[1] PBS lecture on SPH + Eurographics Tutorial 2019, section “Boundary handling”
[2] Akinci et al., “Versatile Rigid-Fluid Coupling for Incompressible SPH”, 2012

Water - Boat forces



Interactive parameters



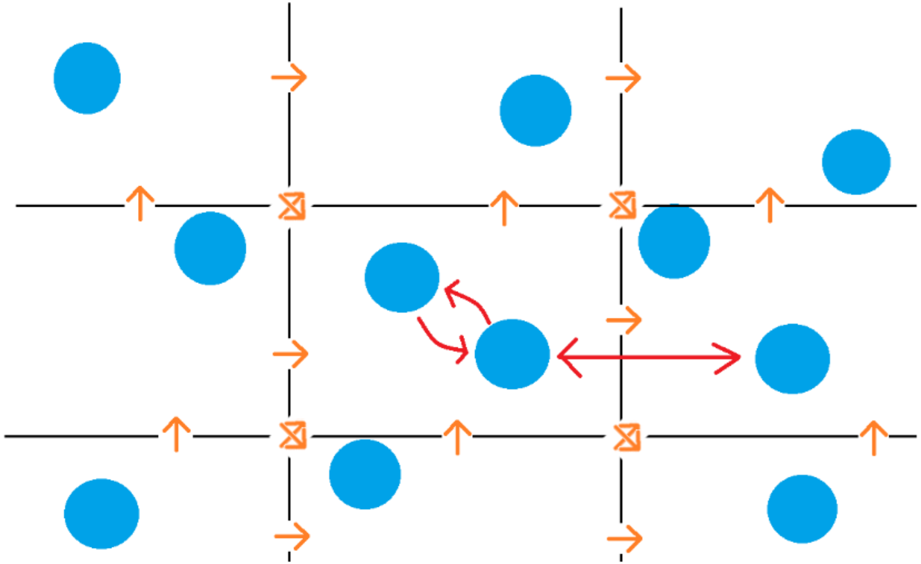
Optimizations

- **Cache locality (~2x)**
  - **Grid contains vector of particles and list of pointers to Neighbor-grids lists**
  - **Particle struct fit in cache-line. Using floats instead of doubles.**
  - **Bottleneck is particle interactions, not re-arrangements.**

An iteration of ~1.1M particles	Time (ms)
Reset densities and forces	27.803
Sum densities	(34.5%) 306.700
Update pressures	15.378
Sum forces	(48.3%) 429.419
Update velocities and positions	38.583
Re-arrange particles	(8%) 71.722
Total	889.605

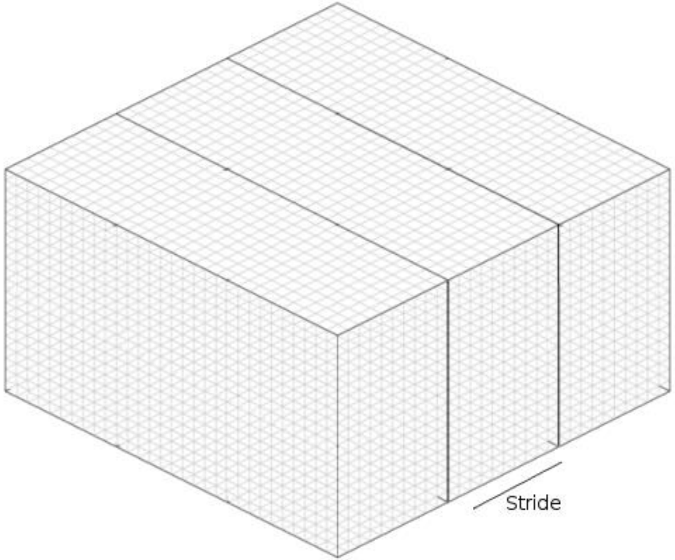
Optimizations

- Cache locality
  - Grid contains vector of particles and list of pointers to Neighbor-grids lists
  - Particle struct fit in cache-line. Using floats instead of doubles.
  - Bottleneck is particle interactions, not re-arrangements.
- **Double update (~1.5x)**
  - **Filters neighbor search**



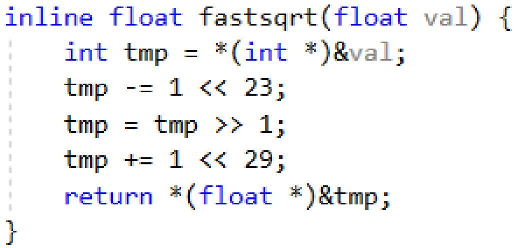
Optimizations

- Cache locality
  - Grid contains vector of particles and list of pointers to Neighbor-grids lists
  - Particle struct fit in cache-line. Using floats instead of doubles.
  - Bottleneck is particle interactions, not re-arrangements.
- Double update
  - Filters neighbor search
- **Parallelization (2 cores 4 threads, ~2x)**
  - **OpenMp**
  - **Reduced cache coherence overhead by strides**

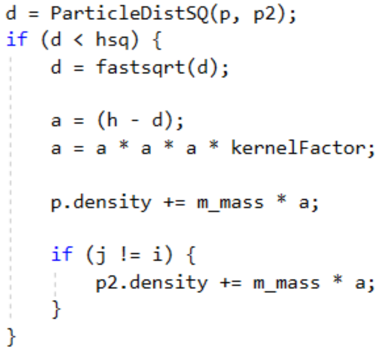


Optimizations

- Cache locality
  - Grid contains vector of particles and list of pointers to Neighbor-grids lists
  - Particle struct fit in cache-line. Using floats instead of doubles.
  - Bottleneck is particle interactions, not re-arrangements.
- Double update
  - Filters neighbor search
- Parallelization
  - OpenMp
  - Reduced cache coherence overhead by strides
- **Low - level optimizations (~2x)**
  - **Avoid pow and sqrt**
  - **Reuse part of calculations in temp variables**



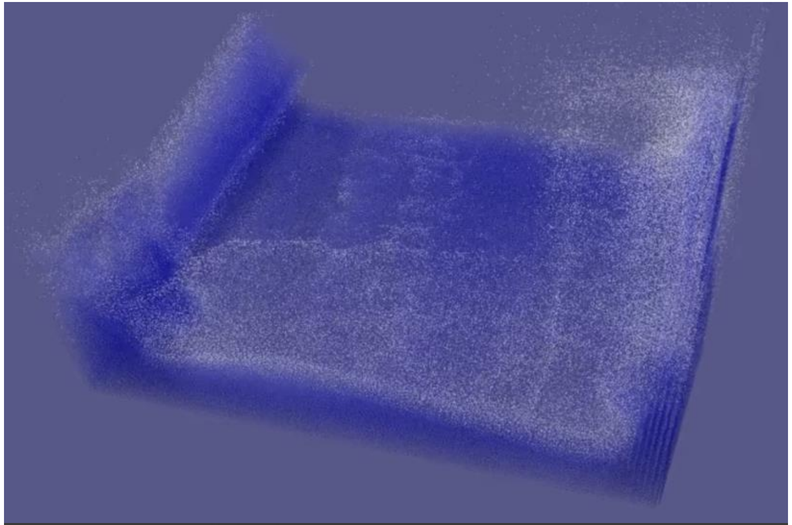
```
inline float fastsqrt(float val) {
    int tmp = *(int *)&val;
    tmp -= 1 << 23;
    tmp = tmp >> 1;
    tmp += 1 << 29;
    return *(float *)&tmp;
}
```

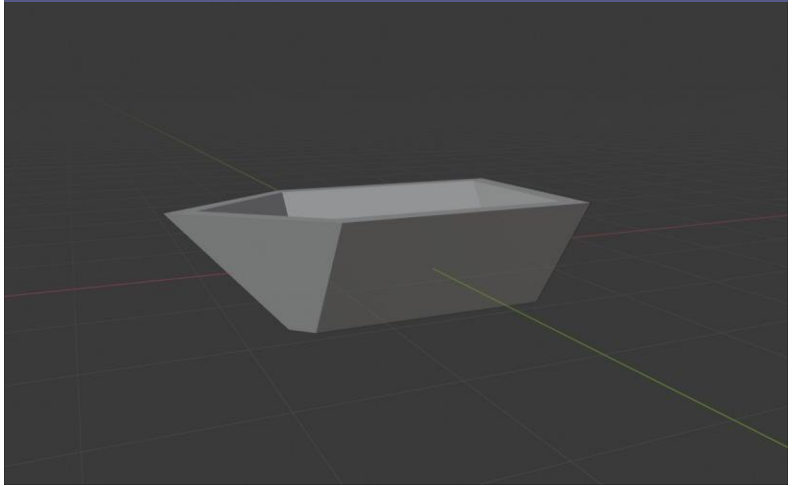


```
d = ParticleDistSQ(p, p2);
if (d < hsq) {
    d = fastsqrt(d);

    a = (h - d);
    a = a * a * a * kernelFactor;

    p.density += m_mass * a;

    if (j != i) {
        p2.density += m_mass * a;
    }
}
```

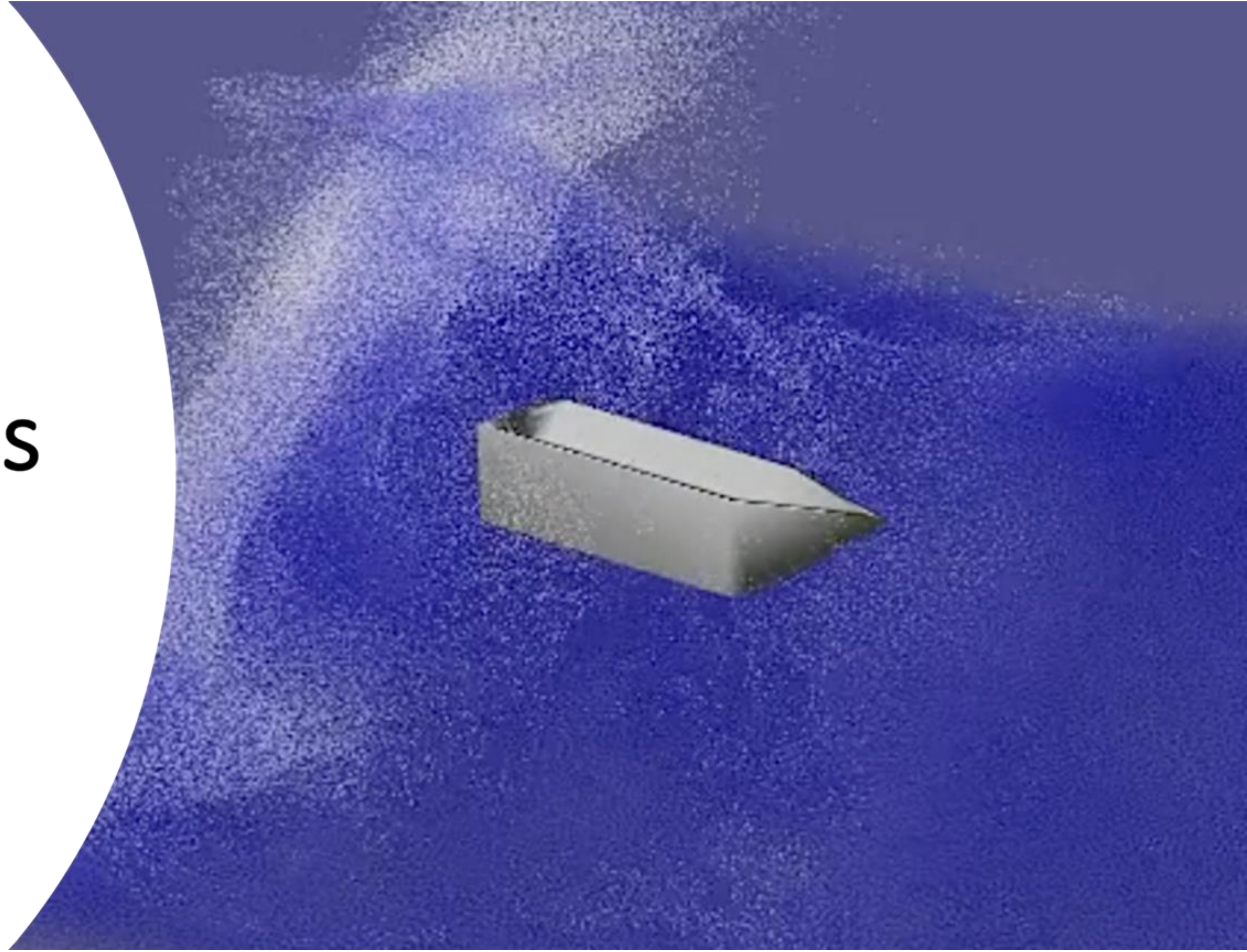
Visualization

- Dot position/color matrices
- Color based on density - foam imitation
- OpenGL dot transparency
- Blender used to make boat .obj file
- Render to bitmap @ 30fps
- FFmpeg converts bitmap to video





Questions



Parameters

1. Spiky kernel

$$W_{spiky}(r,h) = \frac{15}{\pi h^6}\begin{cases}(h-r)^3, & \text{if } 0 \leq r \leq h\\ 0, & \text{otherwise}\end{cases}$$

1. Pressure

$$p_i = k\left(\left(\frac{\rho_i}{\rho_0}\right)^7 - 1\right)$$

$$\mathrm{k} = 1000$$

1. Viscosity

$$\nu = 0.5$$