Diving into SPH

Navier-Stokes Equations

(Still N-S!)

The key idea is unchanged. You want to approximate the N-S equations as before, but not discretize your solution on a mesh. This is like any Lagrangian method. To recap the equations for continuum flow:

- Continuity Equation:

$$\frac{D\rho}{Dt} = -\rho\nabla \cdot \mathbf{u}$$

- Momentum Equation:

$$\frac{D\mathbf{u}}{Dt} = -\frac{1}{\rho}\nabla P + \nu\nabla^2\mathbf{u} + \mathbf{f}$$

Smoothed Particle Hydrodynamics (SPH)

What makes it different

- SPH is mesh-free!
- Free surface flows that are challenging in Eulerian methods are more or less natural in SPH

Kernels

Mathematical Form

In 3D and 4th order (also what I've used)

$$W(q) = \frac{21}{64\pi h^3} \cdot \begin{cases} (2-q)^4(1+2q) & \text{if } 0 \leq q \leq 2 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where:

- Kernels just convolve over the function, and aim to be an approximation to the Dirac Delta function
- It has compact support, which means the tail is not infitine, it takes a certain range of particles around it
- A kernel function must be consistent, i.e.:

$$\int_{-\infty}^{\infty} W(q)\, dq = 1 \tag{2}$$

SPH Equations

Continuity

The approach in SPH is sto sum up contributions from neighbours.

$$\rho_a = \sum_b m_b \frac{W(\mathbf{r}_{ab}, h)}{\rho_b} \tag{3}$$

where:

- W is the smoothing kernel function.
- $\mathbf{r}_{ab}$ is the vector from particle a to b.
- h is the smoothing length.

SPH Equations

Momentum

The kernel functions can be differentiated - and this is used in the gradient for pressure in the momentum equation. Pressure is calculated from the equation of state.

$$\frac{D\mathbf{v}_a}{Dt} = -\sum_b m_b \left(\frac{P_a}{\rho_a^2} + \frac{P_b}{\rho_b^2} \right) \nabla_a W_{ab}(h) + \nu \nabla^2 \mathbf{v}_a + \mathbf{f}_a \tag{4}$$

where:

- P_a and P_b, ρ_a and ρ_b are the pressures and densities of the current particle and its neighbours
- $\nabla_a W_{ab}(h)$ is the gradient of the smoothing kernel between particles a and b with smoothing length h.

Numerical quantities

Kernel Weighted Interpolation

- Postprocessing of SPH would involve interpolation of the same nature to obtain a numerical quantity from particle quantities
- We renormalize the interpolation using a Shepherd summation in the denominator
- For instance, the numerical pressure P_a at particle a can be computed as a weighted sum of neighbour particles

$$P_a = \frac{\sum_b P_b \frac{m_b}{\rho_b} W_{ab}}{\sum_b \frac{m_b}{\rho_b} W_{ab}}$$

A Simple Solver

Pressure Force

```
for(const auto& other : particles)
{
  if(&p != &other)
  {
    double r = distance(p, other);
    if(r > 0)
    {
      double pj = calculatePressureTait(other);
      double rhoj = other.density;

      double w_press = wendlandGradient2D(r);

      pressureForceX +=
        other.mass * (pi / (rhoi * rhoi) + pj / (rhoj * rhoj)) * w_press;
      pressureForceY +=
        other.mass * (pi / (rhoi * rhoi) + pj / (rhoj * rhoj)) * w_press;
    }
  }
}
```

A Simple Solver

Density

```cpp
double calculateDensity(const std::vector<Particle>& particles, const
    Particle& p)
{
  double density = 0.0;
  for(const auto& other : particles)
  {
    double r = distance(p, other);
    density += other.mass * wendlandKernel2D(r);
  }
  return density;
}
```

A Simple Solver

Density

- hello