# GAMES103: Intro to Physics-Based Animation

## Smoothed Particle Hydrodynamics

Huamin Wang

Jan 2022

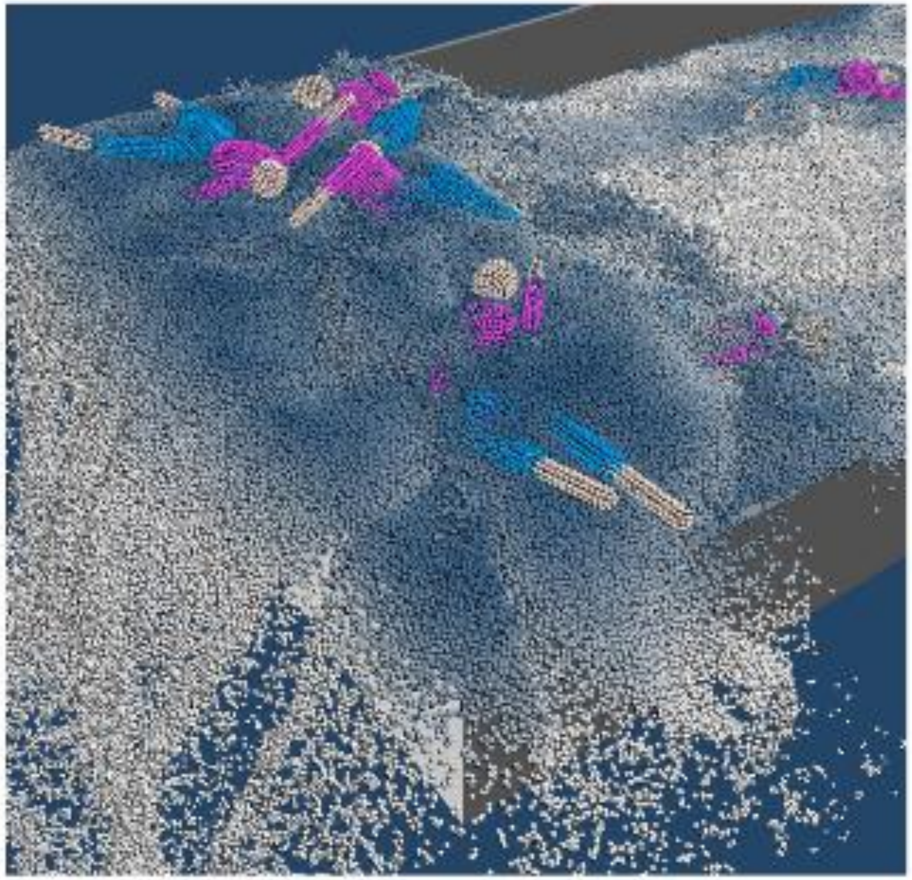
# Topics for the Day

- A SPH model
- SPH-based fluids

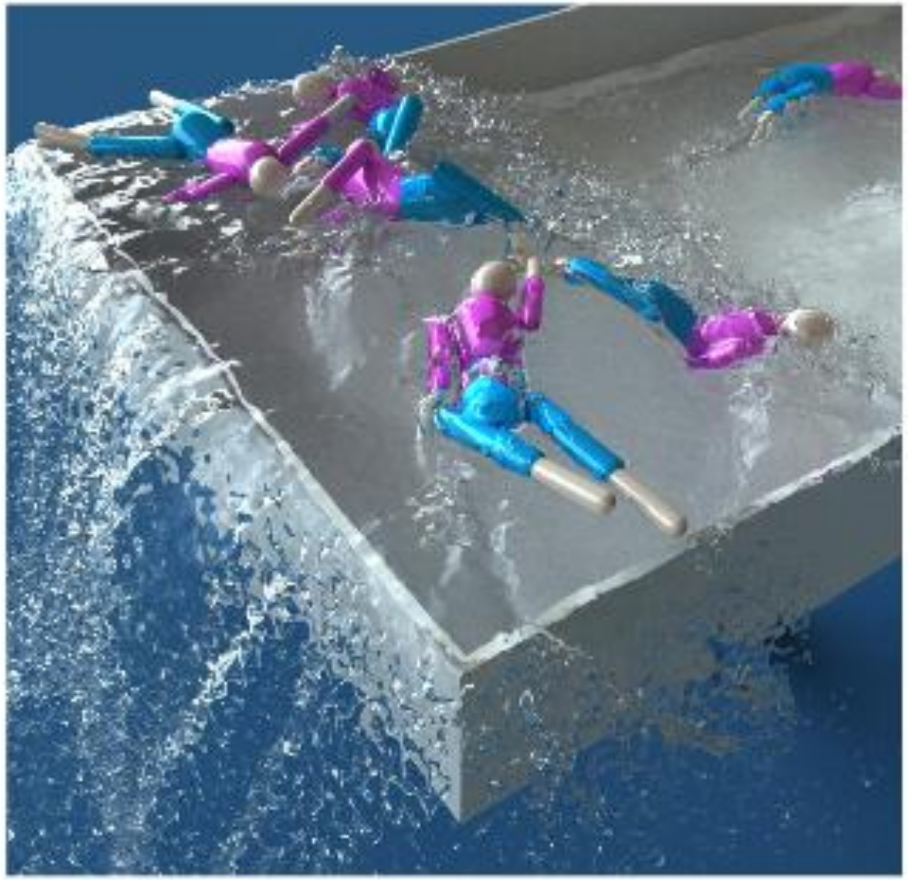
# A SPH Model

# A SPH Model

Consider a (Lagrangian) particle system: each water molecule is a particle with physical quantities attached, such as position $\mathbf{x}_i$, velocity $\mathbf{v}_i$, and mass $m_i$.



representation

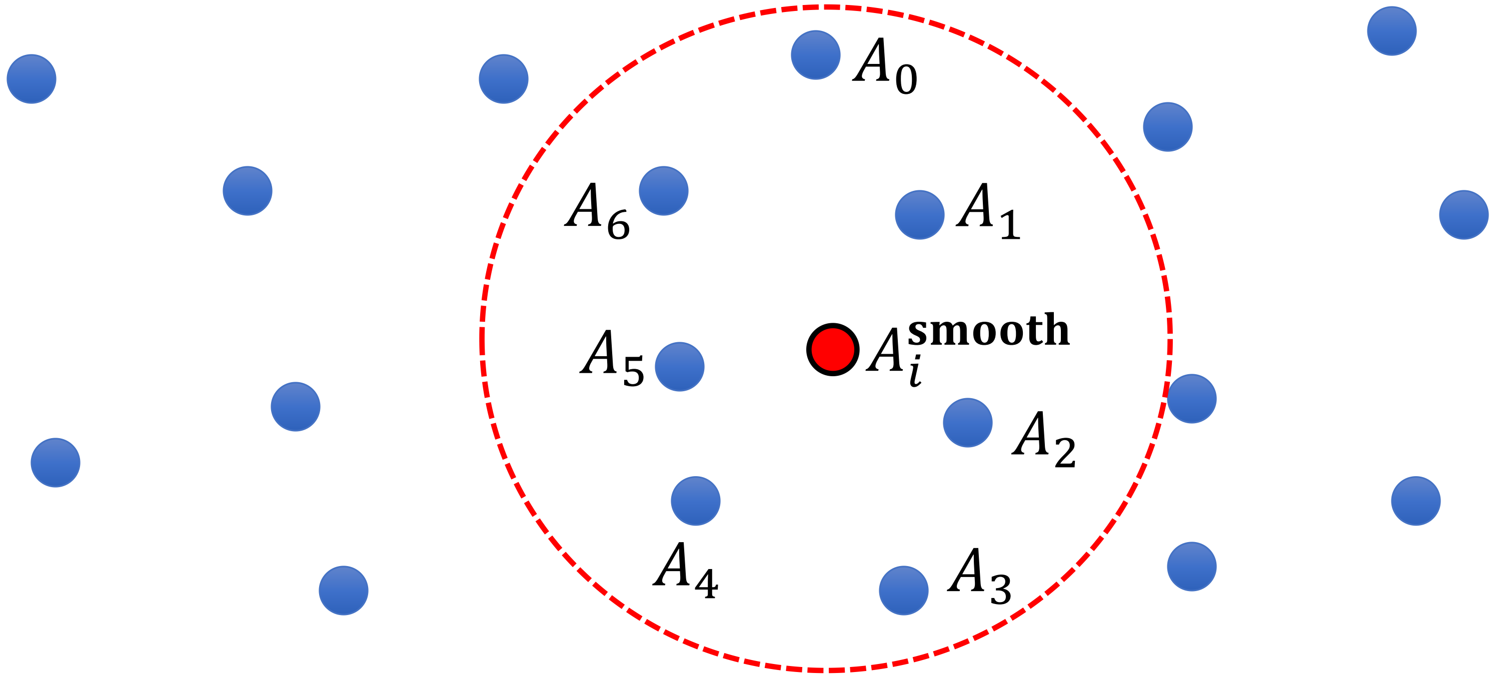


typical visualization

# Smoothed Interpolation – A Simple Model

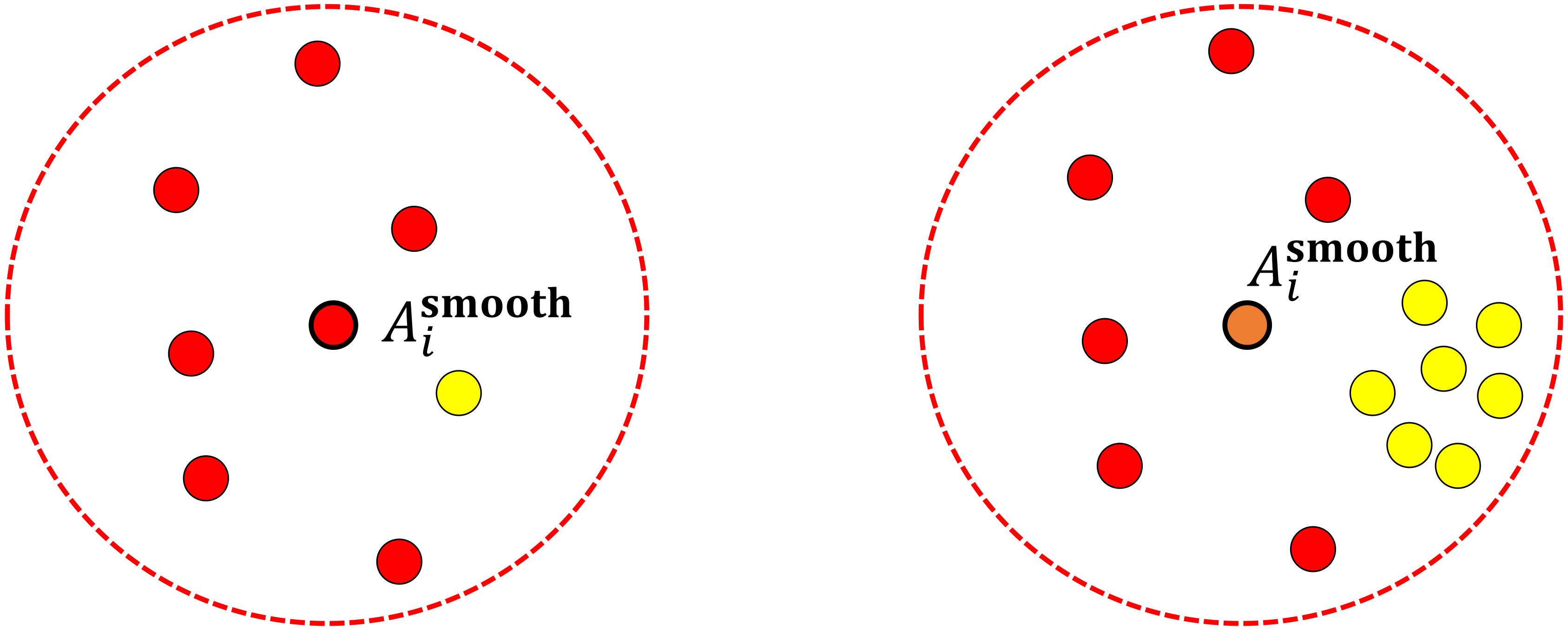
- Suppose each particle j has a **physical quantity** $A_j$.
- The quantity can be: velocity, pressure, density, temperature….
- How to estimate the quantity at a new location $\mathbf{x}_i$?

$$A_i^{\mathbf{smooth}} = \frac{1}{n}\sum_j A_j \qquad \text{For } \|\mathbf{x}_i - \mathbf{x}_j\| < R$$



# Problem with the Simple Model

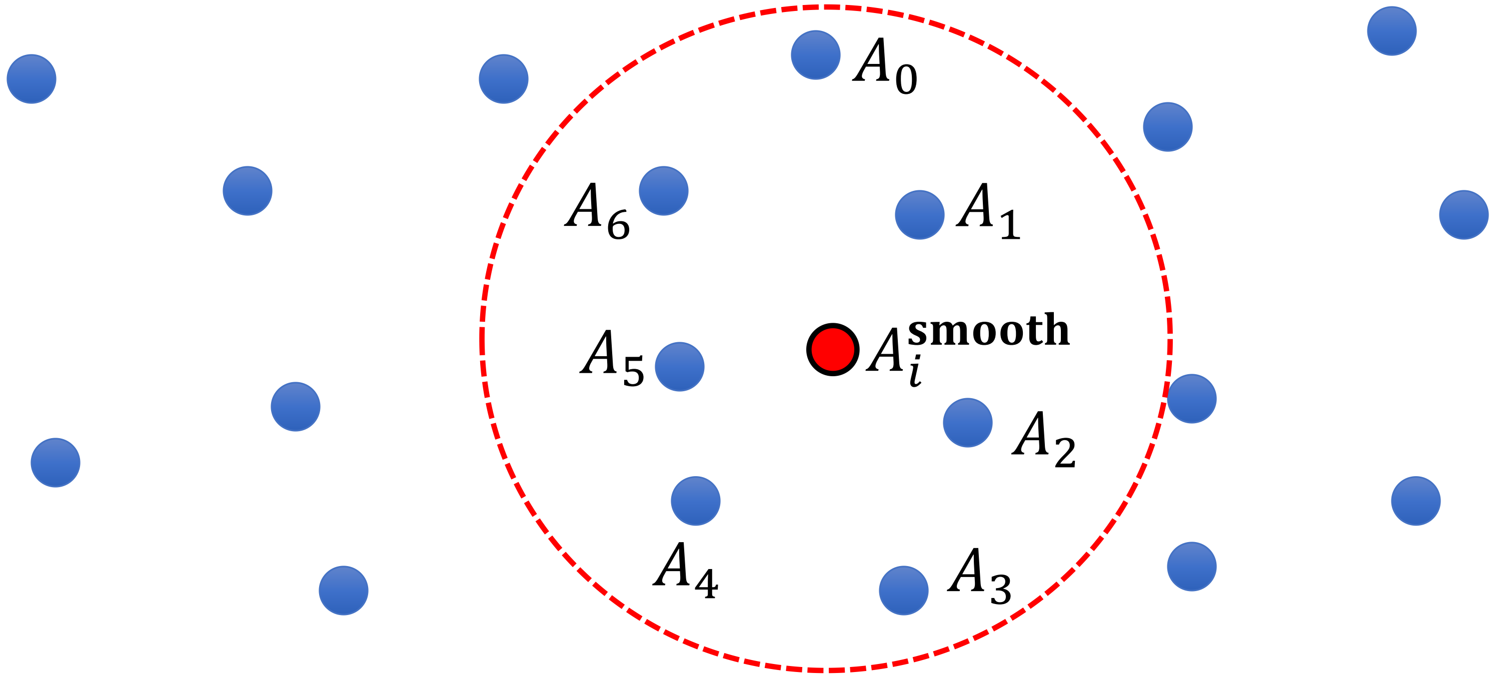
$$A_i^{\mathbf{smooth}} = \frac{1}{n}\sum_j A_j$$



# Smoothed Interpolation – A Better Model

- Let us assume each one represents a volume $V_j$.
- So a better solution is:

$$A_i^{\mathbf{smooth}} = \frac{1}{n}\sum_j V_j A_j \qquad \text{For } \|\mathbf{x}_i - \mathbf{x}_j\| < R$$

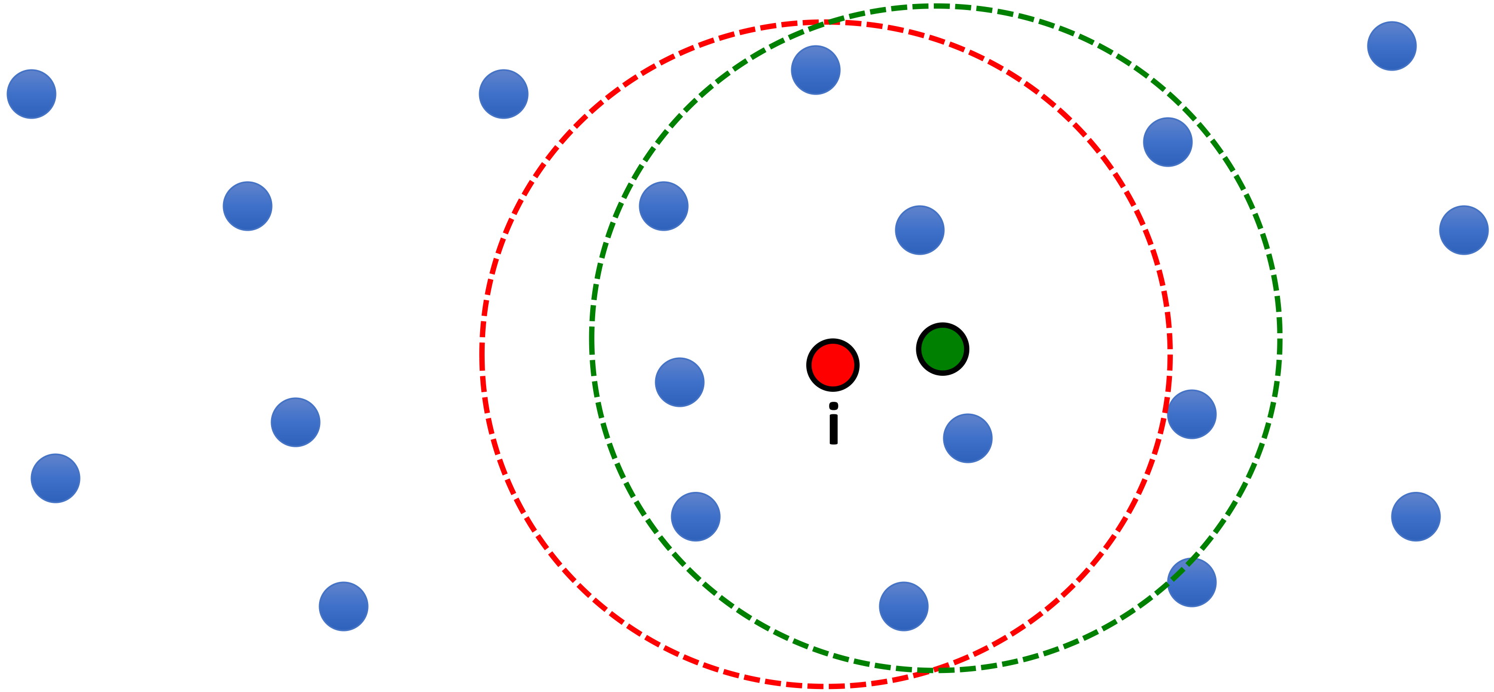


# Problem with the Better Model

- One problem of this solution:

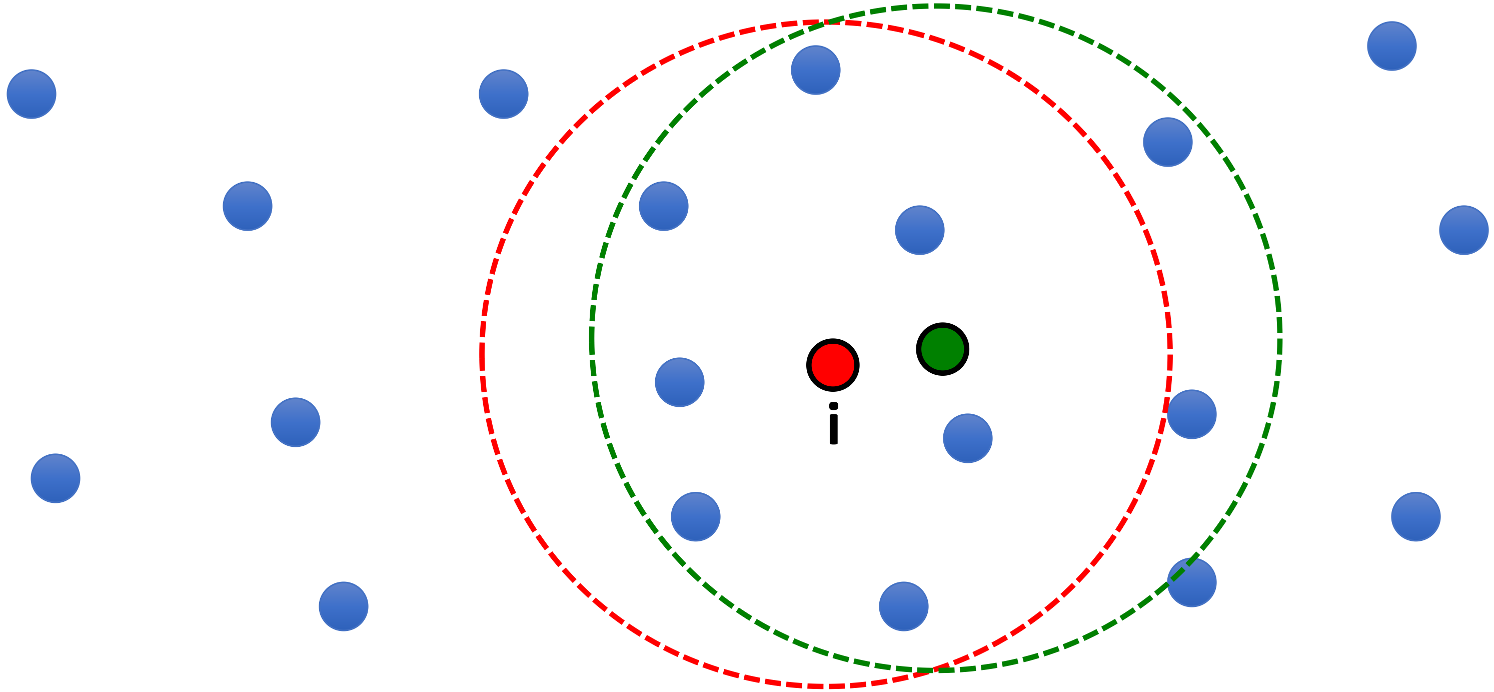
$$A_i^{\mathbf{smooth}} = \frac{1}{n}\sum_j V_j A_j \qquad \text{For } \|\mathbf{x}_i - \mathbf{x}_j\| < R$$

- Not smooth! (7 -> 9!)



# Smoothed Interpolation – Final Solution

- Final solution: $$A_i^{\mathbf{smooth}} = \sum_j V_j A_j W_{ij} \qquad \text{For } \|\mathbf{x}_i - \mathbf{x}_j\| < R$$
- $W_{ij}$ is called smoothing kernel.
- When $\|\mathbf{x}_i - \mathbf{x}_j\|$ is large, $W_{ij}$ is small.
- When $\|\mathbf{x}_i - \mathbf{x}_j\|$ is small, $W_{ij}$ is large.



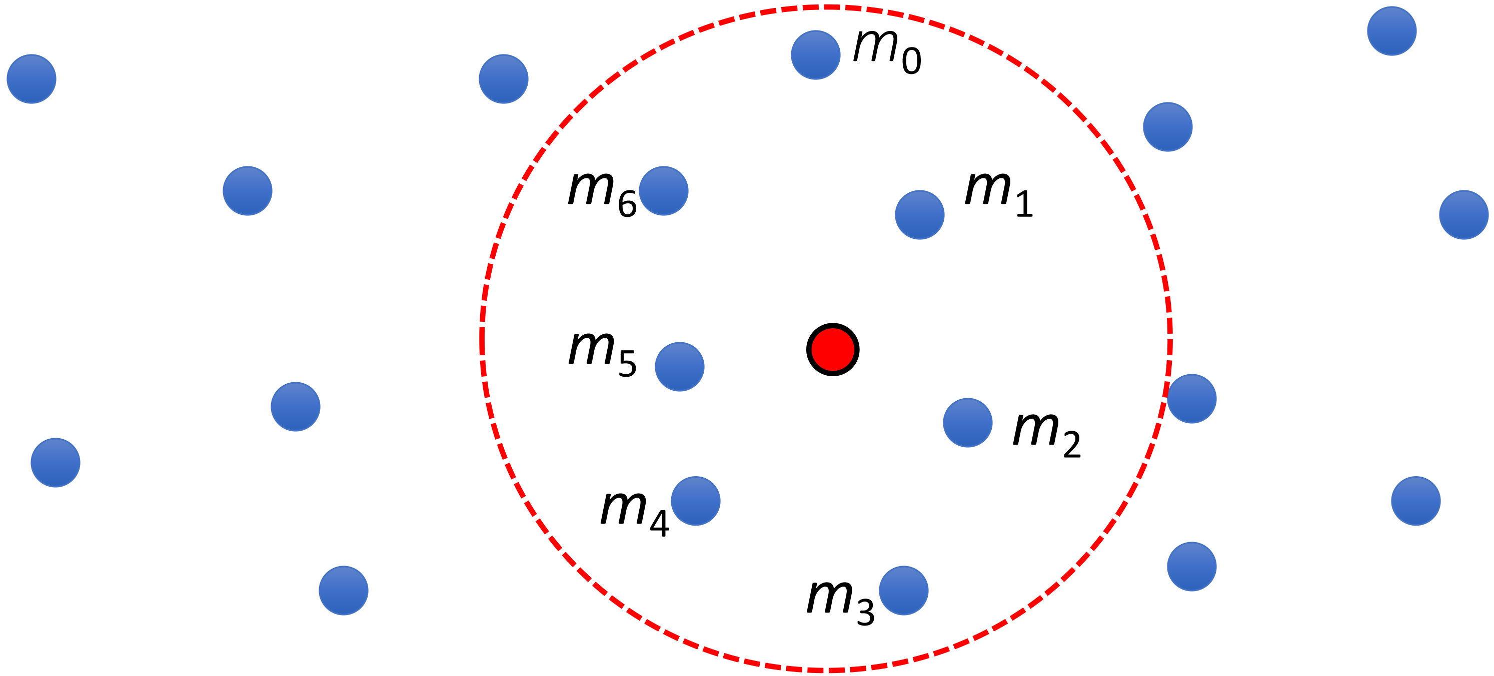
# Particle Volume Estimation

- But how do we get the volume of particle *i*?

$$V_i = \frac{m_i}{\rho_i}$$

$$\rho_i^{\mathbf{smooth}} = \sum_j V_j \rho_j W_{ij} = \sum_j m_j W_{ij}$$

$$\boxed{V_i = \frac{m_i}{\rho_i^{\mathbf{smooth}}} = \frac{m_i}{\sum_j m_j W_{ij}}}$$



# Smoothed Interpolation – Final Solution

- So the actual solution is:

$$A_i^{\textbf{smooth}} = \sum_j V_j A_j W_{ij}$$

$$V_i = \frac{m_i}{\sum_j m_j W_{ij}}$$

$$A_i^{\textbf{smooth}} = \sum_j \frac{m_j}{\sum_k m_k W_{jk}} A_j W_{ij}$$

# Why Smoothed Interpolation?

- We can easily compute its derivatives:
  - Gradient

$$A_i^{\mathbf{smooth}} = \sum_j V_j A_j W_{ij} \qquad \nabla A_i^{\mathbf{smooth}} = \sum_j V_j A_j \nabla W_{ij}$$

  - Laplacian

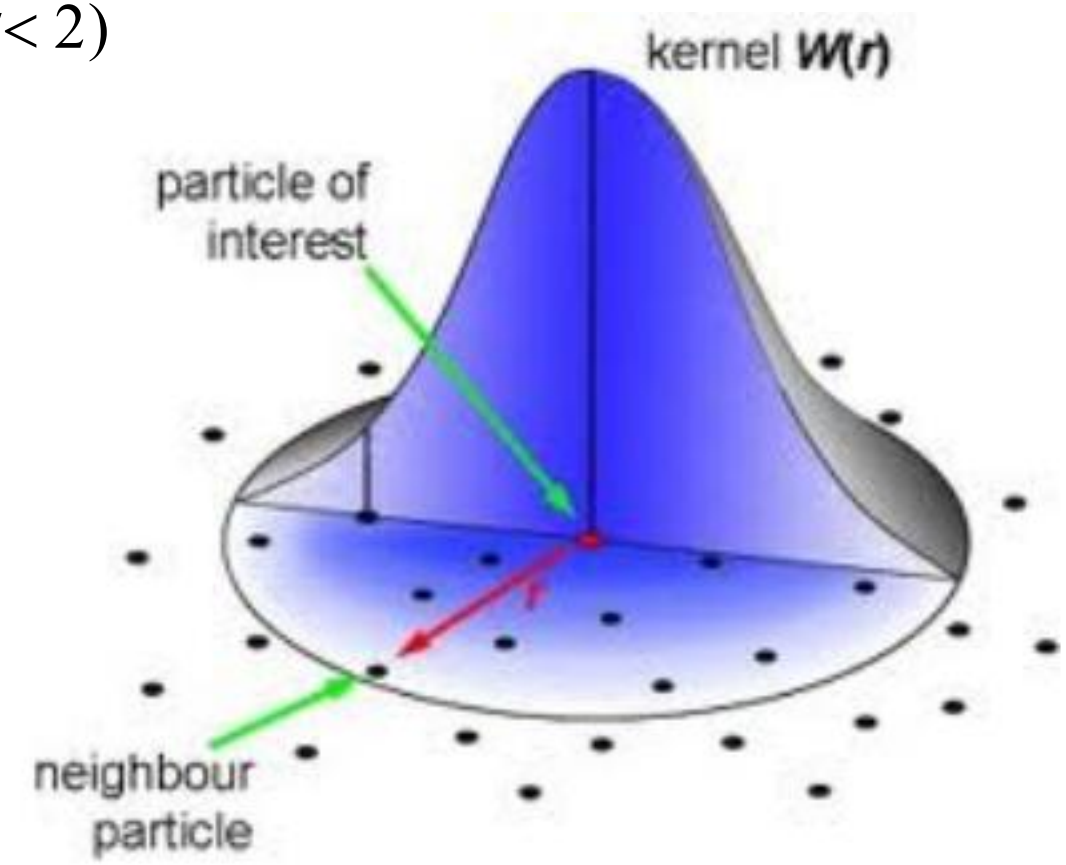
$$A_i^{\mathbf{smooth}} = \sum_j V_j A_j W_{ij} \qquad \Delta A_i^{\mathbf{smooth}} = \sum_j V_j A_j \Delta W_{ij}$$

# A Smoothing Kernel Example

$$W_{ij} = \frac{3}{2\pi h^3} \begin{cases} \frac{2}{3} - q^2 + \frac{1}{2} q^3 & (0 \le q < 1) \\ \frac{1}{6}(2 - q)^3 & (1 \le q < 2) \\ 0 & (2 \le q) \end{cases}$$

$$q = \frac{\left\| \mathbf{x}_i - \mathbf{x}_j \right\|}{h}$$

*h* is called smoothing length



# Kernel Derivatives

- Gradient at particle i (a vector)

$$\boxed{\nabla_i W_{ij} = \begin{bmatrix} \dfrac{\partial W_{ij}}{\partial x_i} \\ \dfrac{\partial W_{ij}}{\partial y_i} \\ \dfrac{\partial W_{ij}}{\partial z_i} \end{bmatrix} = \frac{\partial W_{ij}}{\partial q} \nabla_i q = \frac{\partial W_{ij}}{\partial q} \frac{\mathbf{x}_i - \mathbf{x}_j}{\left\| \mathbf{x}_i - \mathbf{x}_j \right\| h}}$$

$$q = \frac{\left\| \mathbf{x}_i - \mathbf{x}_j \right\|}{h}$$

$$W_{ij} = \frac{3}{2\pi h^3} \begin{cases} \dfrac{2}{3} - q^2 + \dfrac{1}{2} q^3 & (0 \le q < 1) \\ \dfrac{1}{6}(2-q)^3 & (1 \le q < 2) \\ 0 & (2 \le q) \end{cases}$$

$$\frac{\partial W_{ij}}{\partial q} = \frac{3}{2\pi h^3} \begin{cases} -2q + \dfrac{3}{2} q^2 & (0 \le q < 1) \\ -\dfrac{1}{2}(2-q)^2 & (1 \le q < 2) \\ 0 & (2 \le q) \end{cases}$$

# Kernel Derivatives

- Laplacian at particle i (a scalar)

$$\boxed{\nabla_i W_{ij} = \frac{\partial^2 W_{ij}}{\partial x_i^2} + \frac{\partial^2 W_{ij}}{\partial y_i^2} + \frac{\partial^2 W_{ij}}{\partial z_i^2} = \frac{\partial^2 W_{ij}}{\partial q^2}\frac{1}{h^2} + \frac{\partial W_{ij}}{\partial q}\frac{2}{h}} \qquad q = \frac{\left\|\mathbf{x}_i - \mathbf{x}_j\right\|}{h}$$

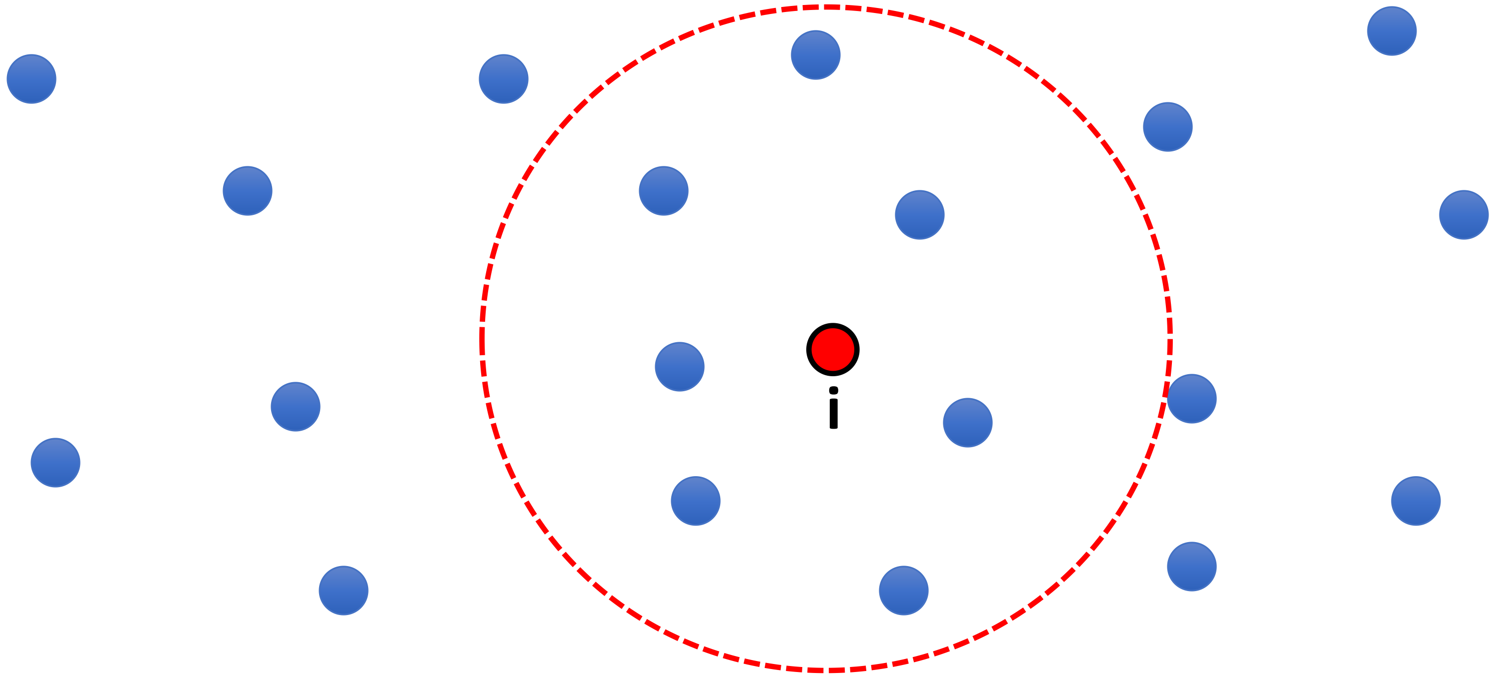
$$\frac{\partial W_{ij}}{\partial q} = \frac{3}{2\pi h^3}\begin{cases} -2q + \frac{3}{2}q^2 & (0 \le q < 1) \\ -\frac{1}{2}(2-q)^2 & (1 \le q < 2) \\ 0 & (2 \le q) \end{cases}$$

$$\frac{\partial^2 W_{ij}}{\partial q^2} = \frac{3}{2\pi h^3}\begin{cases} -2 + 3q & (0 \le q < 1) \\ 2 - q & (1 \le q < 2) \\ 0 & (2 \le q) \end{cases}$$

# SPH-Based Fluids

# Fluid Dynamics

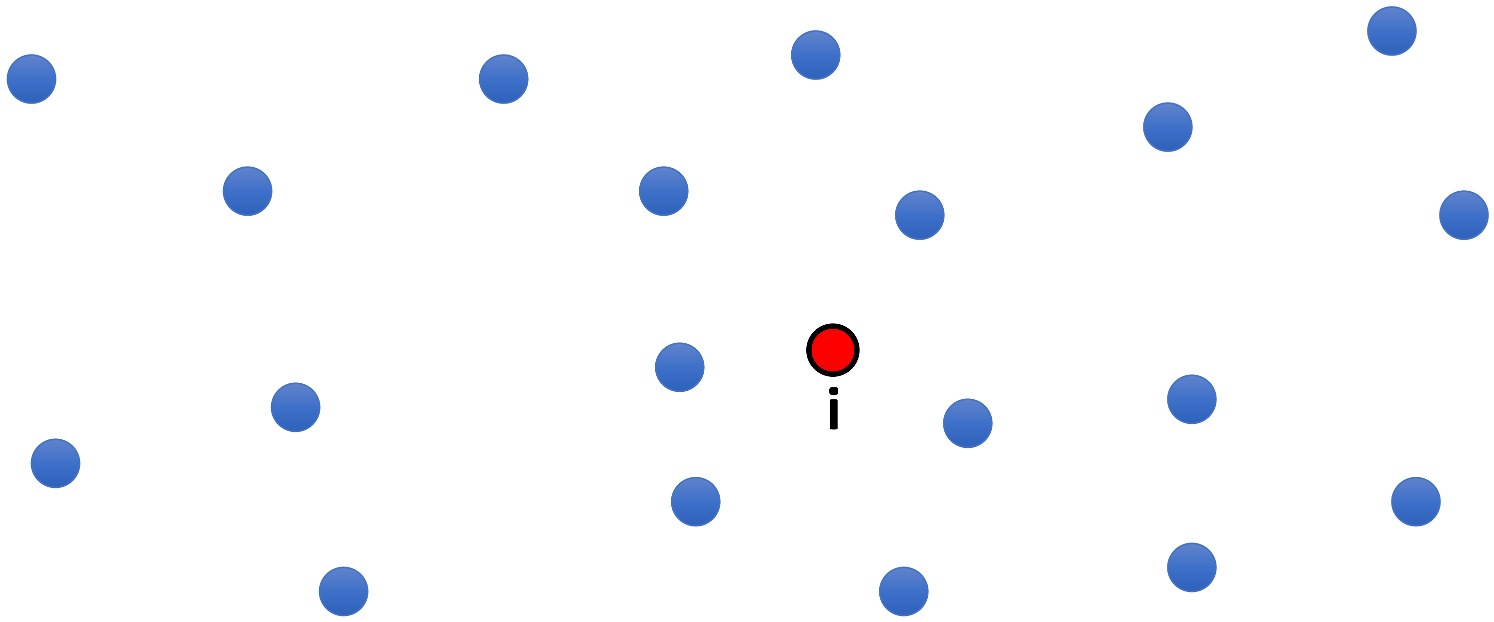
- We model fluid dynamics by applying three forces on particle i.
  - Gravity
  - Fluid Pressure
  - Fluid Viscosity



# Gravity Force

- Gravity Force is:

$$\mathbf{F}_i^{gravity} = m_i\mathbf{g}$$



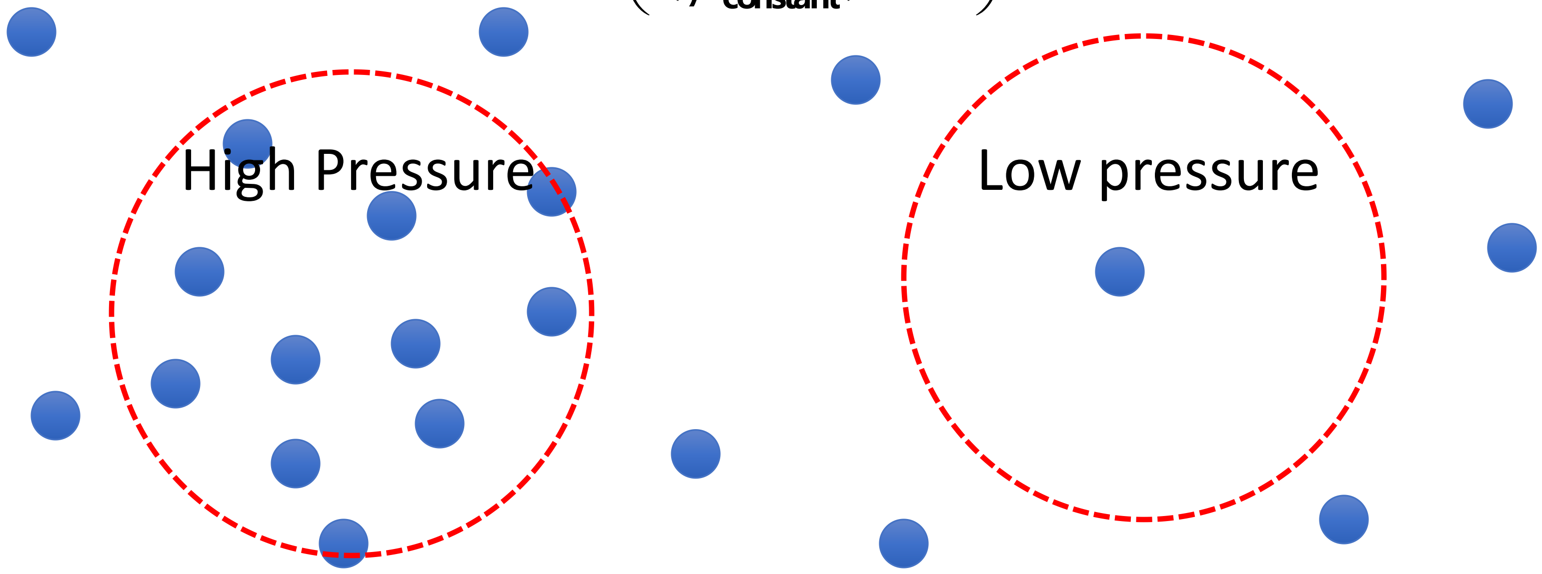
# Pressure Force

- Pressure is related to the density
  - First compute the density of Particle i:

$$\rho_i = \sum_j m_j W_{ij}$$

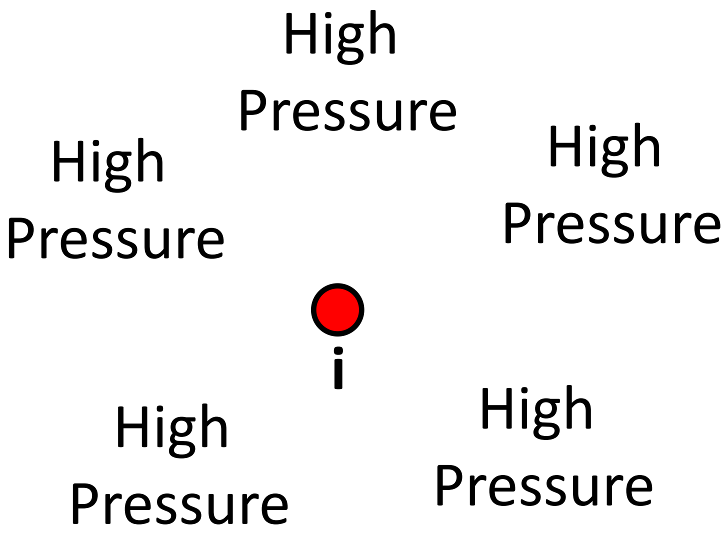
  - Convert it into pressure (some empirical function):

$$P_i = k\left(\left(\frac{\rho_i}{\rho_{\text{constant}}}\right)^7 - 1\right)$$

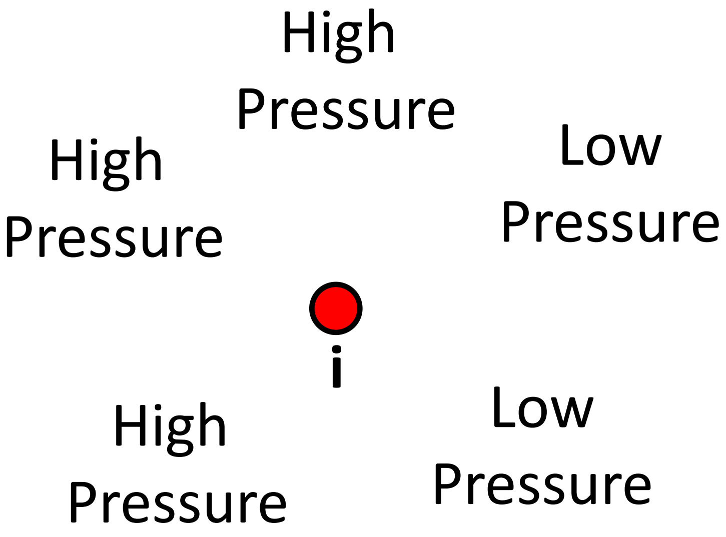


# Pressure Force

- **Pressure force depends on the difference of pressure:**




**No pressure force!**




**Pressure force!**

# Pressure Force

- Mathematically, the difference of pressure => Gradient of pressure.

$$\mathbf{F}_i^{pressure} = -V_i \nabla_i P^{smooth}$$

- To compute this pressure gradient, we assume that the pressure is also smoothly represented:

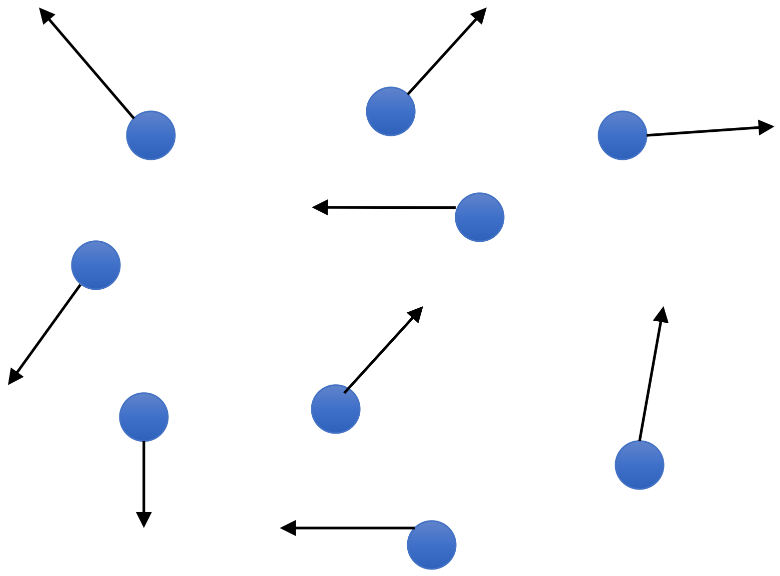
$$P_i^{smooth} = \sum_j V_j P_j W_{ij}$$

- So:

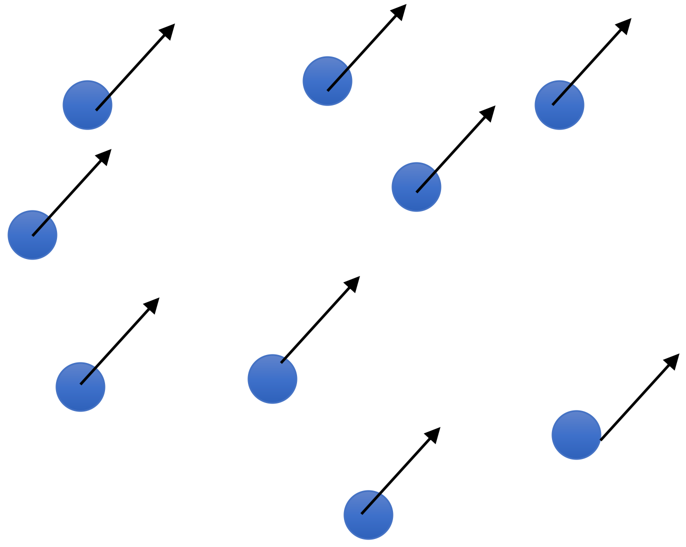
$$\mathbf{F}_i^{pressure} = -V_i \sum_j V_j P_j \nabla_i W_{ij}$$

# Viscosity Force

- Viscosity effect means: *particles should move together in the same velocity*.
- In other words, minimize the difference between the particle velocity and the velocities of its neighbors.



**Before**



**After**

# Viscosity Force

- Mathematically, it means:

$$\mathbf{F}_i^{viscosity} = -\nu m_i \Delta_i \mathbf{v}^{smooth}$$

- To compute this Laplacian, we assume that the velocity is also smoothly represented:

$$\mathbf{v}_i^{smooth} = \sum_j V_j \mathbf{v}_j W_{ij}$$

- So:

$$\mathbf{F}_i^{viscosity} = -\nu m_i \sum_j V_j \mathbf{v}_j \Delta_i W_{ij}$$

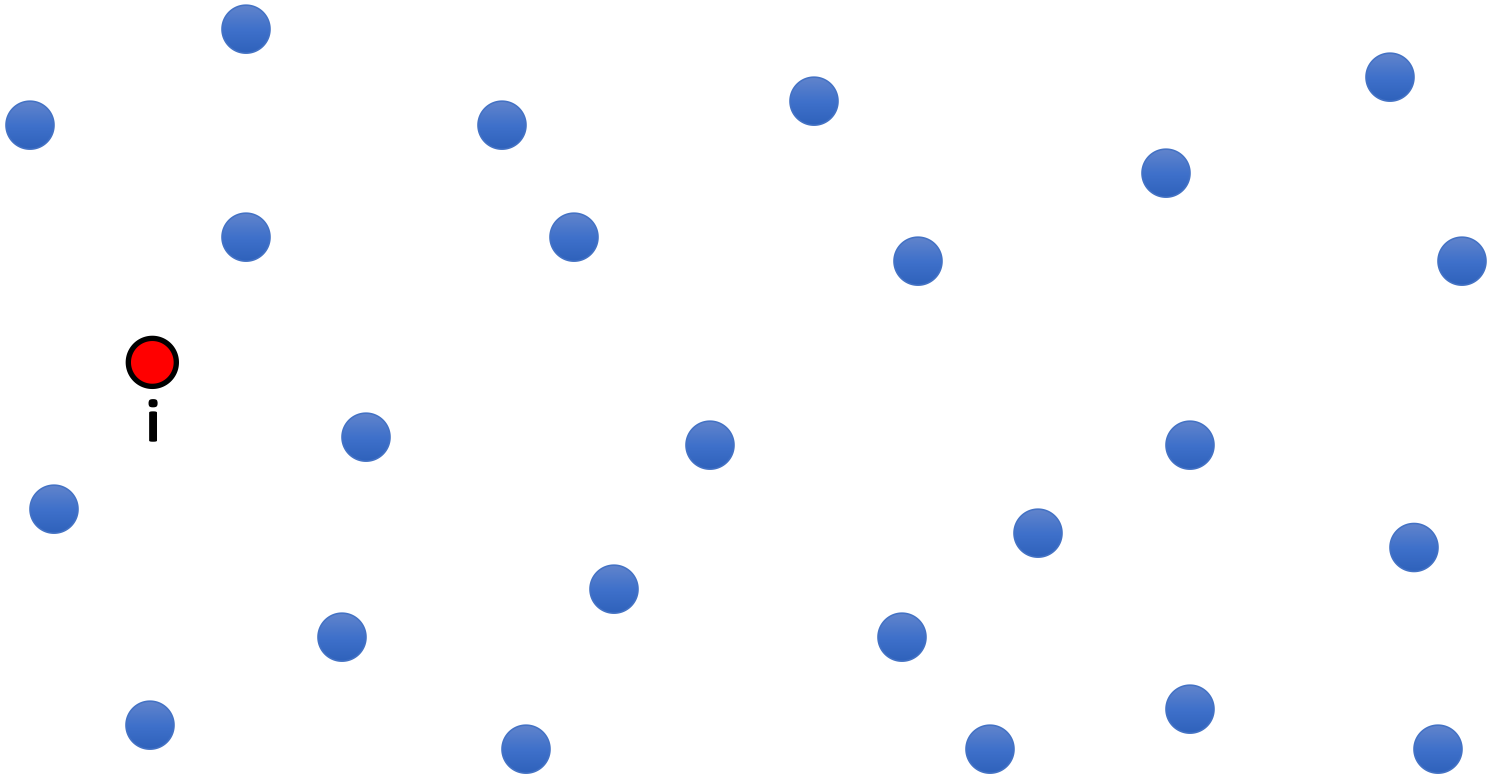
# Algorithm

- For every particle i
  - Compute its neighborhood set
  - Using the neighborhood, compute:
    - Force = 0
    - Force + = The gravity force
    - Force + = The pressure force
    - Force + = The viscosity force
  - Update $v_i = v_i + t * Force / m_i$;
  - Update $x_i = x_i + t * v_i$;

What is the bottleneck of the performance here?

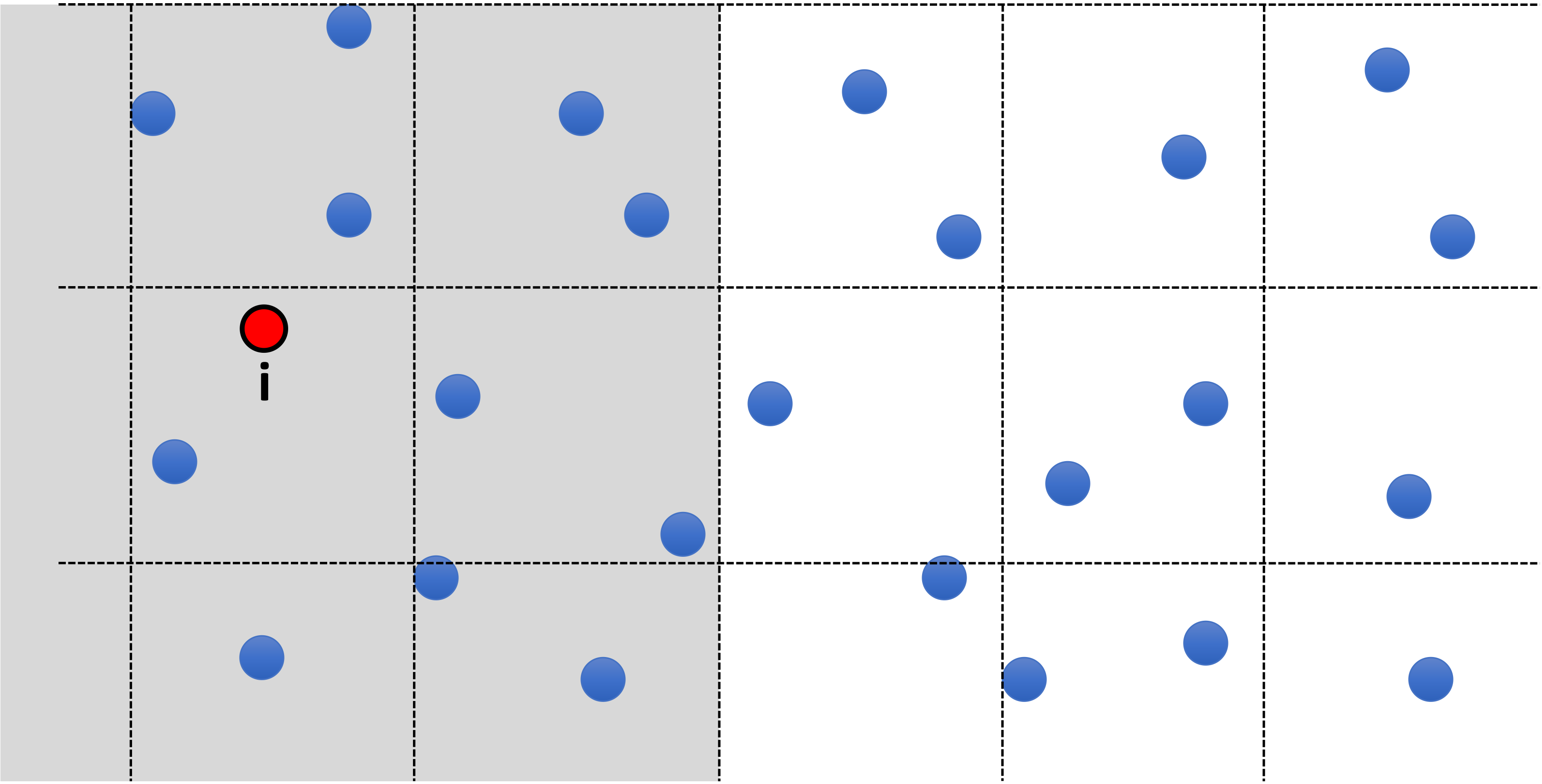
# Exhaustive Neighborhood Search

- Search over every particle pair? $O(N^2)$
- 10M particles means: 100 Trillion pairs…



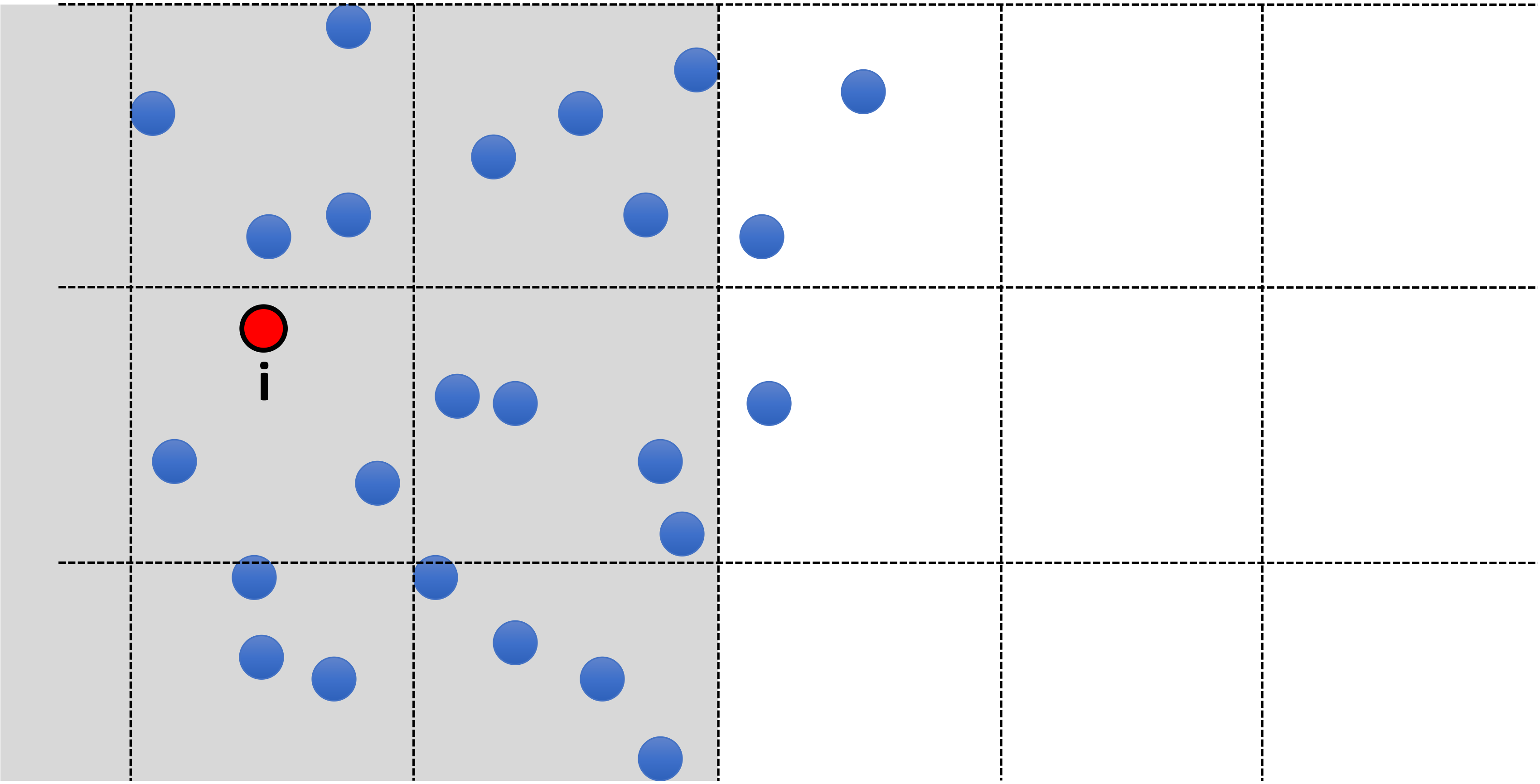
# Solution: Spatial Partition

- Separate the space into cells
- Each cell stores the particles in it
- To find the neighborhood of i, just look at the surrounding cells



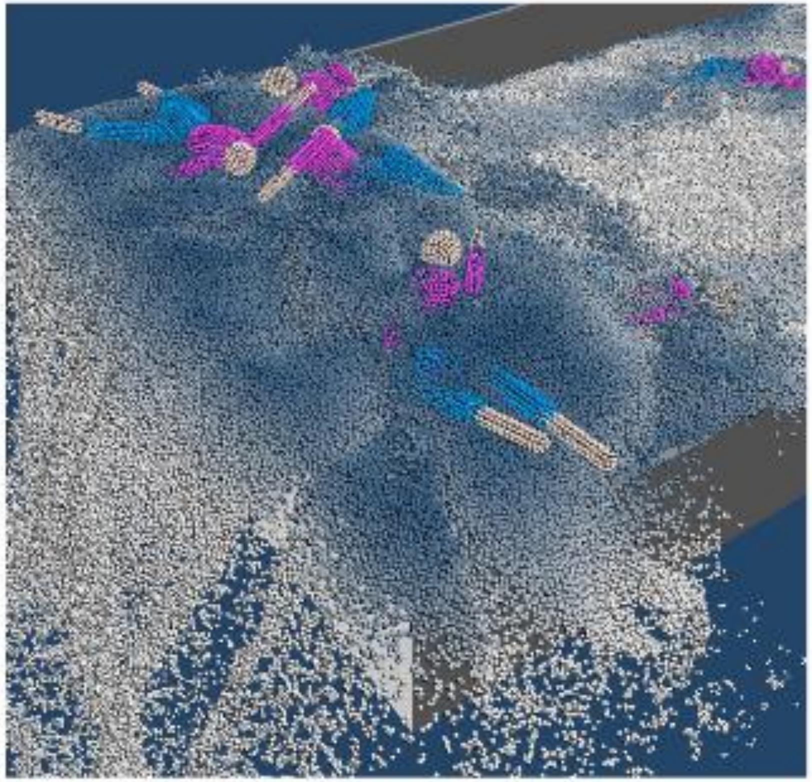
# Spatial Partition

- What if particles are not uniformly distributed?
- **Solution**: Octree, Binary Spatial Partitioning tree…

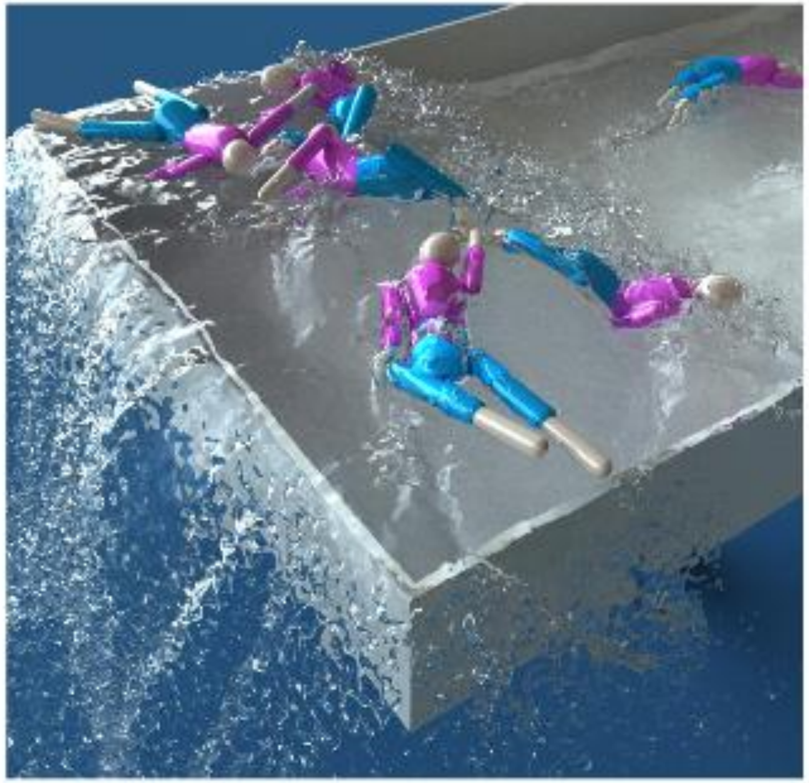


# Fluid Display

- Need to reconstruct the water surface from particles!



representation



typical visualization

# Ongoing Research

√ conference: SIGGRAPH
journal: TOG

差一点 IEEETPCG

- How to make the simulation more efficient?
- How to make fluids incompressible?
- When simulating water, only use water particles, no air particles. So particles are sparse on the water-air boundary. How to avoid artifacts there?
- Using AI, not physics, to predict particle movement?