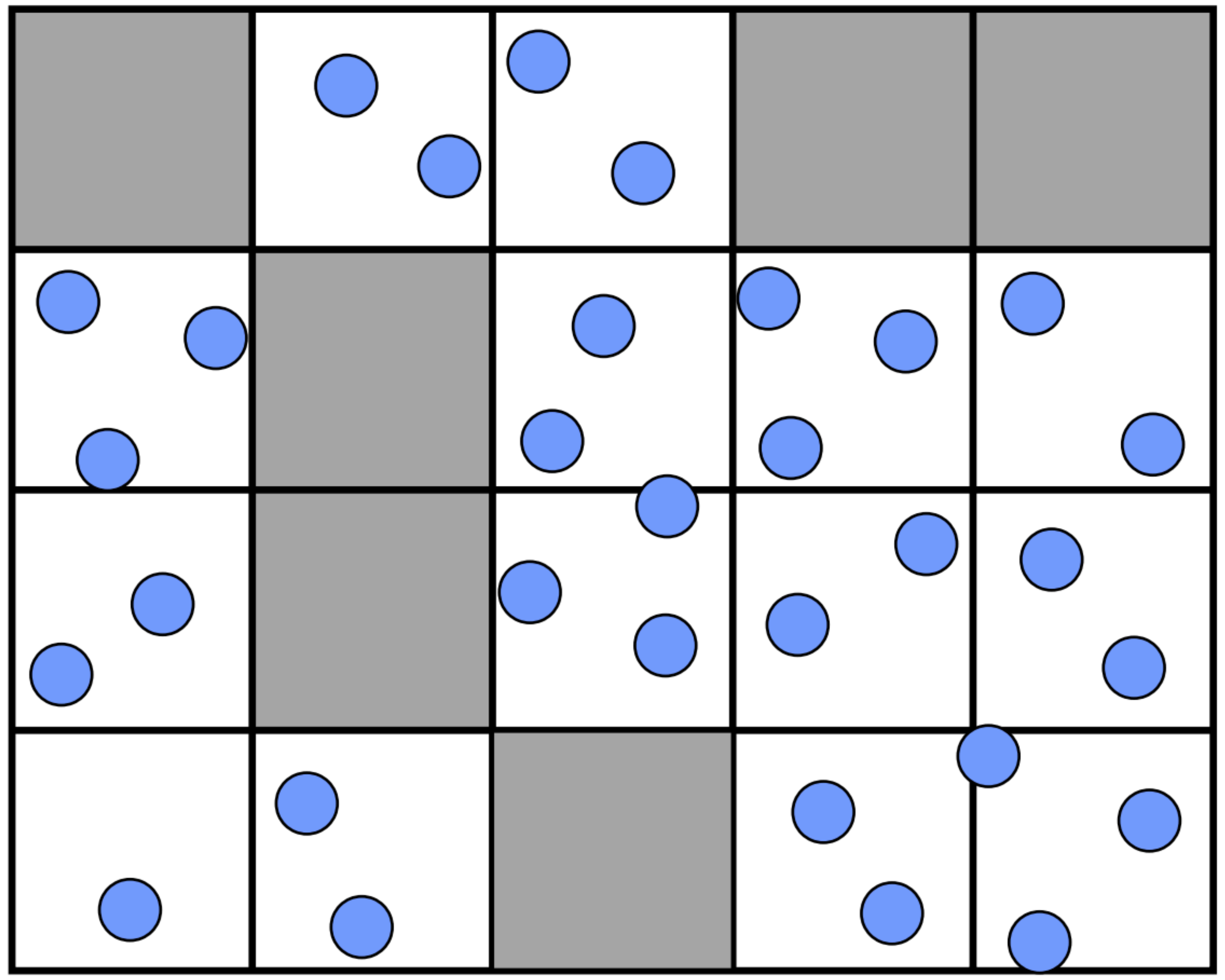
Instructor: Minchen Li

$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} + \frac{1}{\rho}\nabla p = \vec{g} + \nu \nabla \cdot \nabla \vec{u},$$
$$\nabla \cdot \vec{u} = 0.$$



Lec 16: Hybrid Lagrangian/Eulerian Methods

15-769: Physically-based Animation of Solids and Fluids (F23)

Recap: Fluid Simulation Fundamentals

Fluid as a special kind of solid, Eulerian View

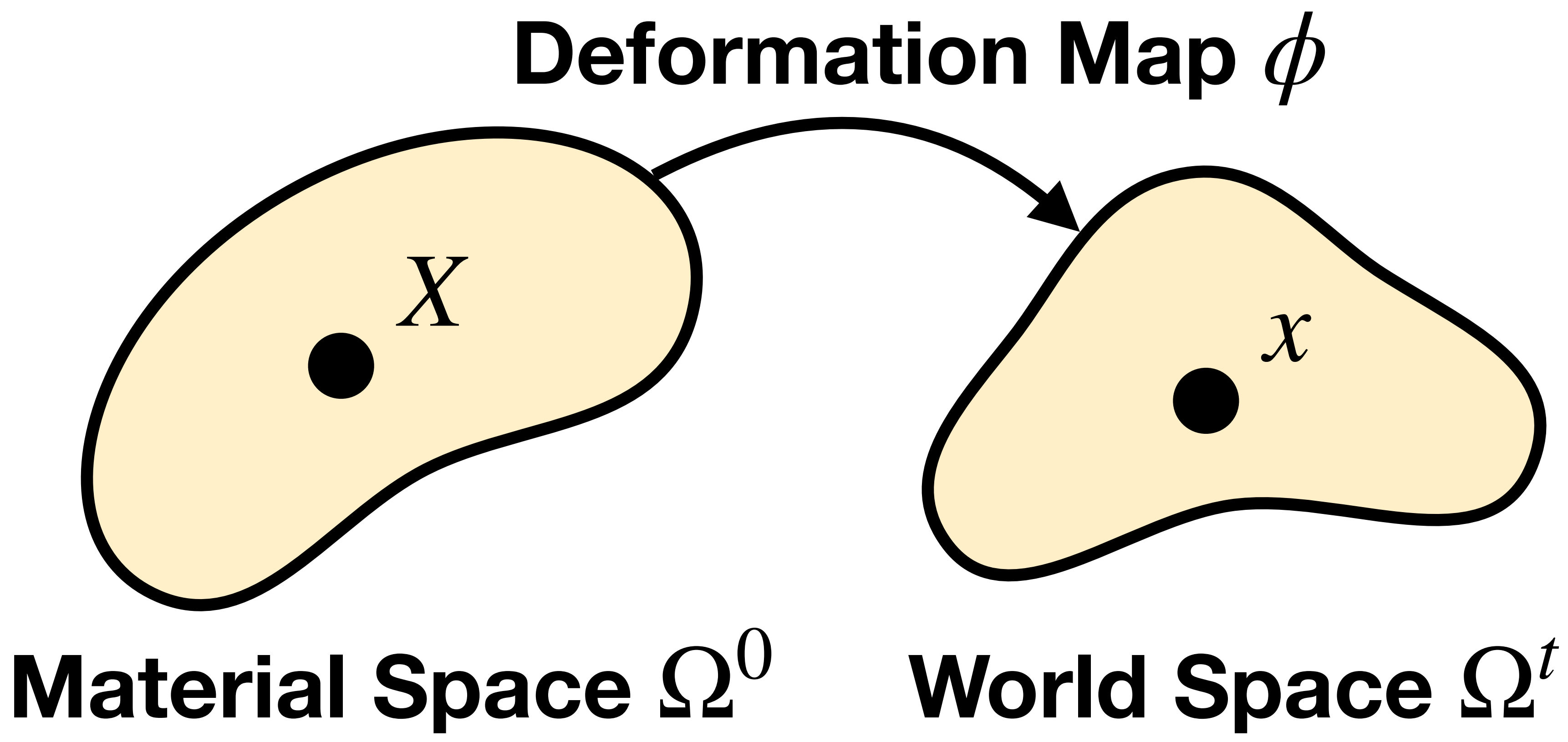
Fluid: a special kind of solid whose strain energy only penalizes volume change

$$x^{n+1} = arg\min_x \frac{1}{2}\|x - \tilde{x}^n\| + h^2 \sum P(x)$$

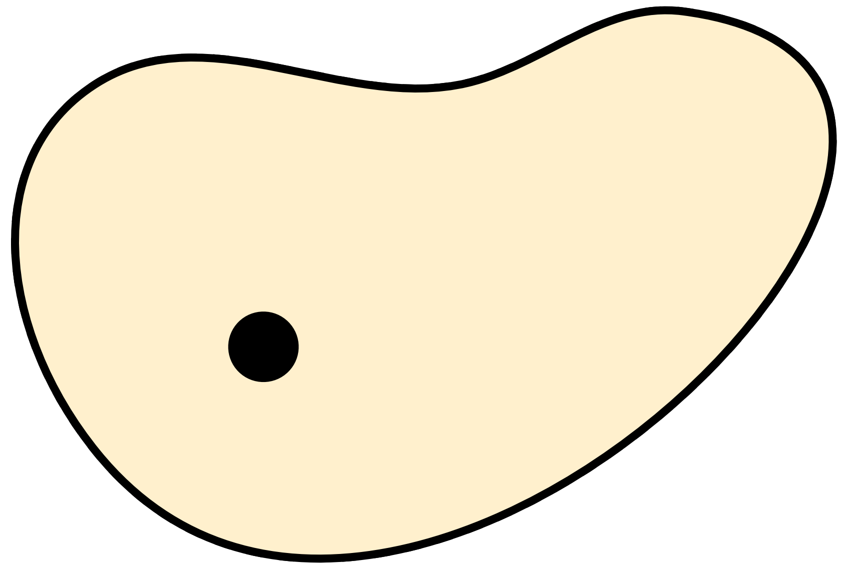
e.g. $P_{fluid}(x) = \sum_e V_e^0 \frac{\kappa}{2}(\det(\mathbf{F}_e(x)) - 1)^2$

Fluid changes topology rapidly:

- **Use particles to track/represent fluid regions,**
- **Use shape functions in world-space (Eulerian View)**



$$\mathbf{x} = \mathbf{x}(\mathbf{X}, t) = \phi(\mathbf{X}, t)$$



Eulerian view:
Quantity measured at a point in space

Recap: Fluid Simulation Fundamentals

Push Forward and Pull Back, Material Derivatives

$$\mathbf{x} = \mathbf{x}(\mathbf{X}, t) = \phi(\mathbf{X}, t) \qquad \mathbf{X} = \phi^{-1}(\mathbf{x}, t)$$

Push forward: $\mathbf{Q}(\mathbf{X}, t) = \mathbf{Q}(\phi^{-1}(\mathbf{x}, t), t) \equiv \mathbf{q}(\mathbf{x}, t)$

Pull back: $\mathbf{q}(\mathbf{x}, t) = \mathbf{q}(\phi(\mathbf{X}, t), t) \equiv \mathbf{Q}(\mathbf{X}, t)$

$$\mathbf{V}(\mathbf{X}, t) = \frac{\partial \phi}{\partial t}(\mathbf{X}, t)$$

$$\mathbf{A}(\mathbf{X}, t) = \frac{\partial^2 \phi}{\partial t^2}(\mathbf{X}, t) = \frac{\partial \mathbf{V}}{\partial t}(\mathbf{X}, t).$$

$$\boldsymbol{v}(\boldsymbol{x}, t) = \mathbf{V}(\phi^{-1}(\boldsymbol{x}, t), t),$$

$$\boldsymbol{a}(\boldsymbol{x}, t) = \mathbf{A}(\phi^{-1}(\boldsymbol{x}, t), t).$$

$$\mathbf{V}(\mathbf{X}, t) = \boldsymbol{v}(\phi(\mathbf{X}, t), t),$$

$$\mathbf{A}(\mathbf{X}, t) = \boldsymbol{a}(\phi(\mathbf{X}, t), t).$$

$$\mathbf{A}(\mathbf{X}, t) = \frac{\partial}{\partial t}\mathbf{V}(\mathbf{X}, t) = \frac{\partial \boldsymbol{v}}{\partial t}(\phi(\mathbf{X}, t), t) + \frac{\partial \boldsymbol{v}}{\partial \boldsymbol{x}}(\phi(\mathbf{X}, t), t)\frac{\partial \phi}{\partial t}(\mathbf{X}, t).$$

$$A_i(\mathbf{X}, t) = \frac{\partial}{\partial t}V_i(\mathbf{X}, t) = \frac{\partial v_i}{\partial t}(\phi(\mathbf{X}, t), t) + \frac{\partial v_i}{\partial x_j}(\phi(\mathbf{X}, t), t)\frac{\partial \phi_j}{\partial t}(\mathbf{X}, t).$$

$$a_i(\boldsymbol{x}, t) = A_i(\phi^{-1}(\boldsymbol{x}, t), t) = \frac{\partial v_i}{\partial t}(\boldsymbol{x}, t) + \frac{\partial v_i}{\partial x_j}(\boldsymbol{x}, t)v_j(\boldsymbol{x}, t)$$

$$\boxed{a_i(\boldsymbol{x}, t) \neq \frac{\partial v_i}{\partial t}(\boldsymbol{x}, t).} \qquad \boxed{\mathbf{a}(\mathbf{x}, t) = \frac{D\mathbf{v}(\mathbf{x}, t)}{Dt}}$$

(Material Derivative)

Recap: Fluid Simulation Fundamentals

Deriving Incompressible Navier-Stoke's Equation for Newtonian Fluids

Momentum Equation (Lagrangian View):

$$R(\mathbf{X},0)\frac{\partial \mathbf{V}}{\partial t}(\mathbf{X},t) = \nabla^{\mathbf{X}} \cdot \mathbf{P}(\mathbf{X},t) + R(\mathbf{X},0)\boldsymbol{g}$$

Push forward on integral form

Eulerian View: $$\rho(\mathbf{x},t)\frac{D\mathbf{v}}{Dt}(\mathbf{x},t) = \nabla^{\mathbf{x}} \cdot \sigma(\mathbf{x},t) + \rho(\mathbf{x},t)\mathbf{g}$$

$$\sigma = \frac{1}{J}\mathbf{P}\mathbf{F}^T = \frac{\partial \Psi}{\partial J}\mathbf{I} = -pI$$

Navier Stoke's Equation (Inviscid): $$\frac{\partial \vec{u}}{\partial t} + \vec{u}\cdot\nabla\vec{u} + \frac{1}{\rho}\nabla p = \vec{g}$$

Bulk modulus $\rightarrow \infty$

Incompressible Navier-Stoke's Equation (Inviscid):

$$\frac{\partial \vec{u}}{\partial t} + \vec{u}\cdot\nabla\vec{u} + \boxed{\frac{1}{\rho}\nabla p} = \vec{g}$$

$$\nabla \cdot \vec{u} = 0.$$

Lagrange multiplier term

Adding viscosity for Incompressible Newtonian fluids

Incompressible Navier-Stoke's Equation

$$\frac{\partial \vec{u}}{\partial t} + \vec{u}\cdot\nabla\vec{u} + \frac{1}{\rho}\nabla p = \vec{g} + \nu\nabla\cdot\nabla\vec{u},$$

$$\nabla \cdot \vec{u} = 0.$$

Strain rate tensor: $\mathbf{D} = \frac{1}{2}(\nabla\mathbf{v} + \nabla\mathbf{v}^T)$

Newtonian fluids: $\sigma_{viscosity} = 2\mu\mathbf{D} + \lambda \mathrm{tr}(\mathbf{D})\mathbf{I}$

Recap: Smoothed Particle Hydrodynamics (SPH)

Basic Idea

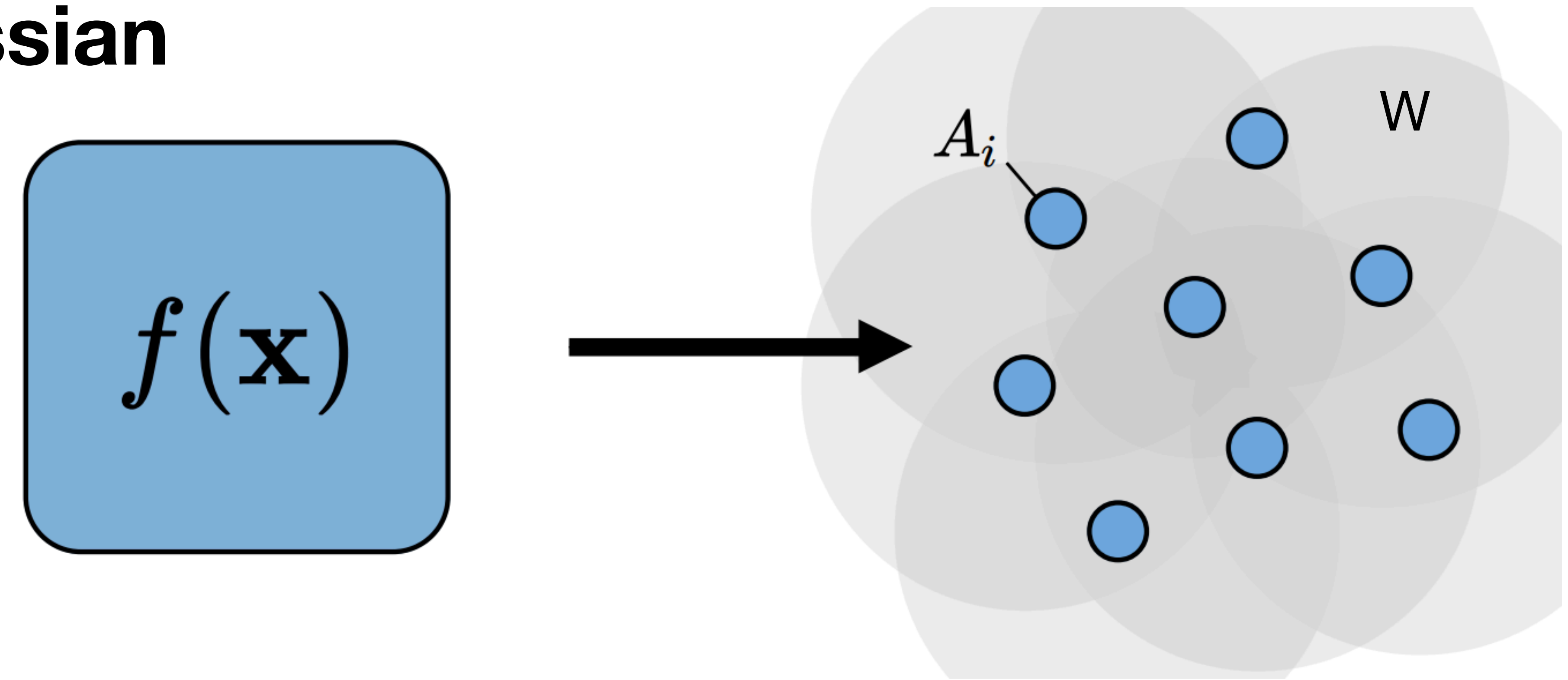
Given a field A and a smoothing kernel function W, e.g. Gaussian

A smoother version of A as an approximation of it is

$$A(\mathbf{x}) \approx (A * W)(\mathbf{x}) = \int A(\mathbf{x}')W(\mathbf{x}-\mathbf{x}',h)\,dv'$$

Favored properties of W:

$\int_{\mathbb{R}^d} W(\mathbf{r}',h)\,dv' = 1$	(normalization condition)
$\lim_{h'\to 0} W(\mathbf{r},h') = \delta(\mathbf{r})$	(Dirac-δ condition)
$W(\mathbf{r},h) \geq 0$	(positivity condition)
$W(\mathbf{r},h) = W(-\mathbf{r},h)$	(symmetry condition)
$W(\mathbf{r},h) = 0$ for $\lVert\mathbf{r}\rVert \geq \hbar,$	(compact support condition)



Discretization using particles:

$$(A * W)(\mathbf{x}_i) = \int \frac{A(\mathbf{x}')}{\rho(\mathbf{x}')} W(\mathbf{x}-\mathbf{x}',h)\,\underbrace{\rho(\mathbf{x}')\,dv'}_{dm'}$$

$$\approx \sum_{j\in\mathcal{F}} A_j \frac{m_j}{\rho_j} W(\mathbf{x}_i - \mathbf{x}_j, h) =: \langle A(\mathbf{x}_i)\rangle$$

The kernel needs to involve a large number of neighbors for accurate estimation!

Recap: SPH Fluid Simulation

Just need to approximate the differential operators, and relate velocity to pressure via constitutive models to solve

$$\frac{\partial \vec{u}}{\partial t} + \vec{u} \cdot \nabla \vec{u} + \frac{1}{\rho} \nabla p = \vec{g} + \nu \nabla \cdot \nabla \vec{u},$$

$$\nabla \cdot \vec{u} = 0.$$

Time Splitting

For each time step n:

$u^a \leftarrow$ **Solve** $\frac{\partial u}{\partial t} + u \cdot \nabla u = 0$ **(advection)**

$u^b \leftarrow$ **Solve** $\frac{\partial u}{\partial t} = g$ **(apply external force)**

$u^c \leftarrow$ **Solve** $\frac{\partial u}{\partial t} = \nu \nabla \cdot \nabla u$ **(diffusion)**

$u^{n+1} \leftarrow$ **Solve** $\nabla \cdot u = 0$ **(pressure projection)**

Direct discretization are not accurate and can lead to instability:

$$\nabla \mathbf{A}_i \approx \sum_j \frac{m_j}{\rho_j} \mathbf{A}_j \otimes \nabla W_{ij} \quad \nabla \cdot \mathbf{A}_i \approx \sum_j \frac{m_j}{\rho_j} \mathbf{A}_j \cdot \nabla W_{ij}$$

Difference and symmetric formula are often used:

$$\nabla A_i \approx \langle \nabla A_i \rangle - A_i \langle \nabla 1 \rangle$$

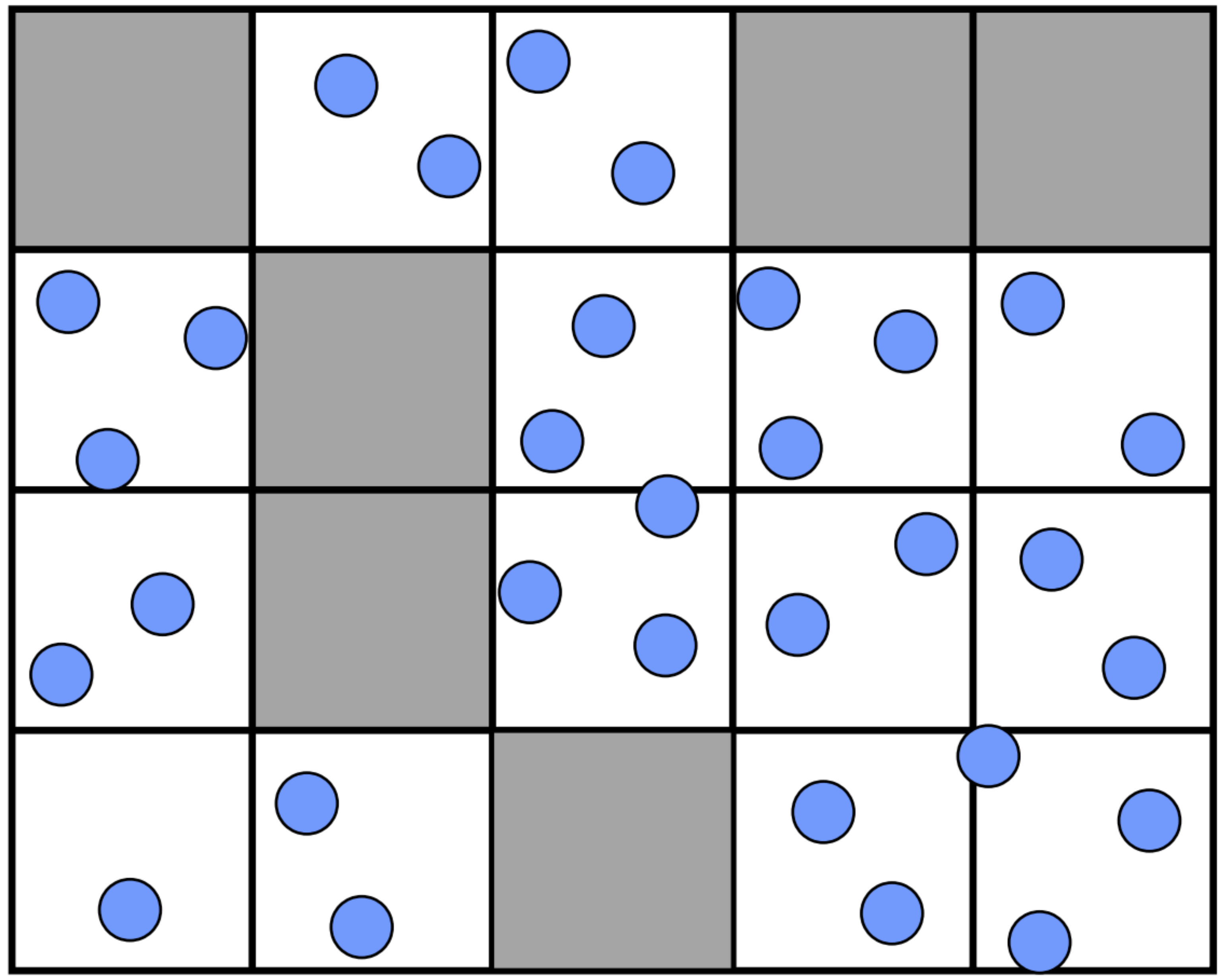
$$= \sum_j \frac{m_j}{\rho_j} (A_j - A_i) \nabla_i W_{ij}\,.$$

$$\nabla A_i \approx \rho_i \left(\frac{A_i}{\rho_i^2} \langle \nabla \rho \rangle + \langle \nabla \left(\frac{A_i}{\rho_i} \right) \rangle \right)$$

$$= \rho_i \sum_j m_j \left(\frac{A_i}{\rho_i^2} + \frac{A_j}{\rho_j^2} \right) \nabla_i W_{ij}\,.$$

Today: Hybrid Lagrangian/Eulerian Methods

Basic Idea



Introduce a background Eulerian Grid, and measure quantities on the grid nodes

Transfer information between the particles and grid

Time Splitting

For each time step n:

Using particles

$$u^a \leftarrow \textbf{Solve } \frac{\partial u}{\partial t} + u \cdot \nabla u = 0 \textbf{ (advection)}$$

Using the grid

$$u^b \leftarrow \textbf{Solve } \frac{\partial u}{\partial t} = g \textbf{ (apply external force)}$$

$$u^c \leftarrow \textbf{Solve } \frac{\partial u}{\partial t} = \nu \nabla \cdot \nabla u \textbf{ (diffusion)}$$

$$u^{n+1} \leftarrow \textbf{Solve } \nabla \cdot u = 0 \textbf{ (pressure projection)}$$

— take advantage of both representations

Particle Advection

$u^a \leftarrow$ **Solve** $\dfrac{\partial u}{\partial t} + u \cdot \nabla u = 0$

— fluids are moving,
resulting in Eulerian velocity changes.

— derived from $\dfrac{du(\phi(\mathbf{X}, t), t)}{dt} = 0$

Our particles are Lagrangian particles!
— each particle marks a fixed region in material space
(Forces are evaluated in an Eulerian view)

Recall that $\mathbf{V}(\mathbf{X}, t) = \mathbf{v}(\phi(\mathbf{X}, t), t)$, **so the advection equation becomes** $\dfrac{\partial \mathbf{V}(\mathbf{X}, t)}{\partial t} = 0$

Solving advection using particles,
we just need to move the particles based on the current velocity!

Forward Euler: $\mathbf{x}_p \leftarrow \mathbf{x}_p + h\mathbf{u}(\mathbf{x}_p, t)$

Can use explicit Runge-Kutta, e.g. RK3, for higher accuracy.

Particle-Grid Transfer

Grid to particle is easy, can just use e.g. bilinear interpolation:

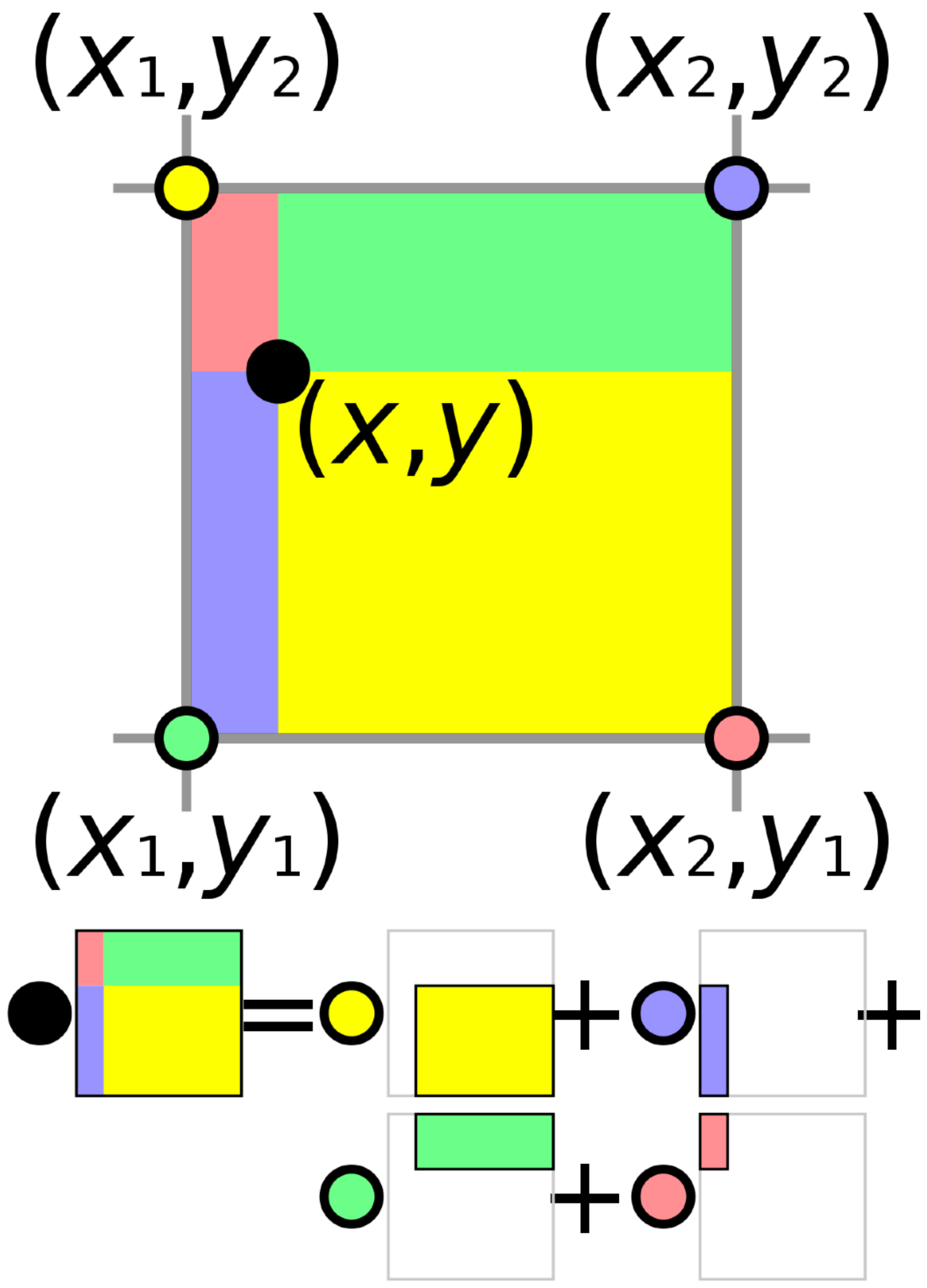
$x_p = Px_i$ **,** $P \in \mathbb{R}^{dn_p \times dn_i}$ **stores the interpolation weights**

Particle to grid: inverse interpolation?

$$x_i = \arg\min_x \frac{1}{2} \|Px - x_p\|^2$$ **Too expensive!**

Instead:

$x_i = N^{-1} P^T x_p$ **,** **where** $N_{ij} = \delta_{ij} \sum_k P_{ki}$ **is for normalization**

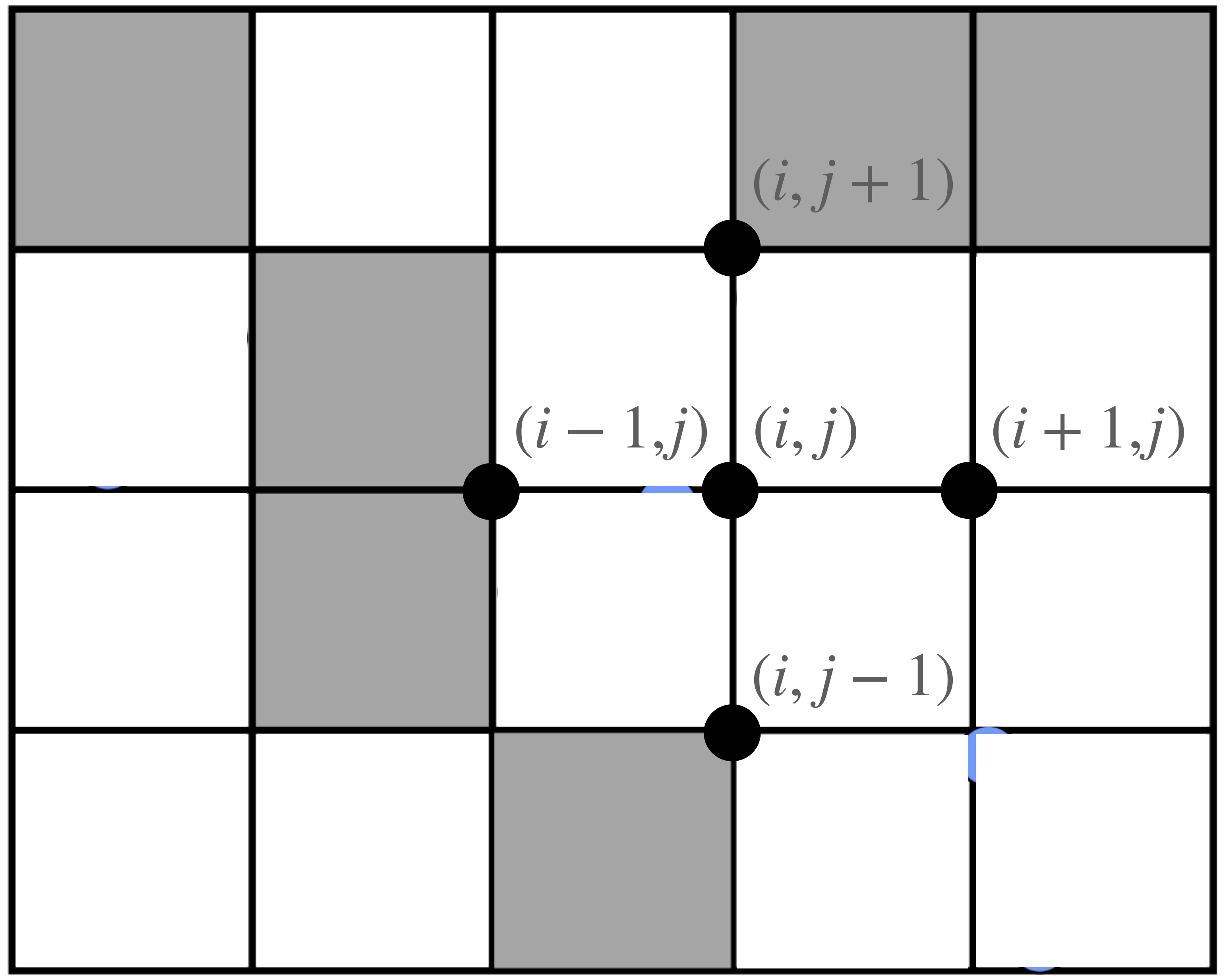


Grid-based Viscosity (Diffusion)

$u^c \leftarrow$ **Solve** $\dfrac{\partial u}{\partial t} = \nu \nabla \cdot \nabla u$

- **Independent per dimension:** $\dfrac{\partial u_k}{\partial t} = \nu(\nabla \cdot \nabla u)_k = \nu\left(\dfrac{\partial^2 u_k}{\partial x^2} + \dfrac{\partial^2 u_k}{\partial y^2}\right)$

$$\approx \nu\left(\frac{u_k(i+1,j) + u_k(i-1,j) - 2u_k(i,j)}{\Delta x^2} + \frac{u_k(i,j+1) + u_k(i,j-1) - 2u_k(i,j)}{\Delta x^2}\right)$$



Use implicit Euler for stability!

Pressure Projection on Eulerian Grid

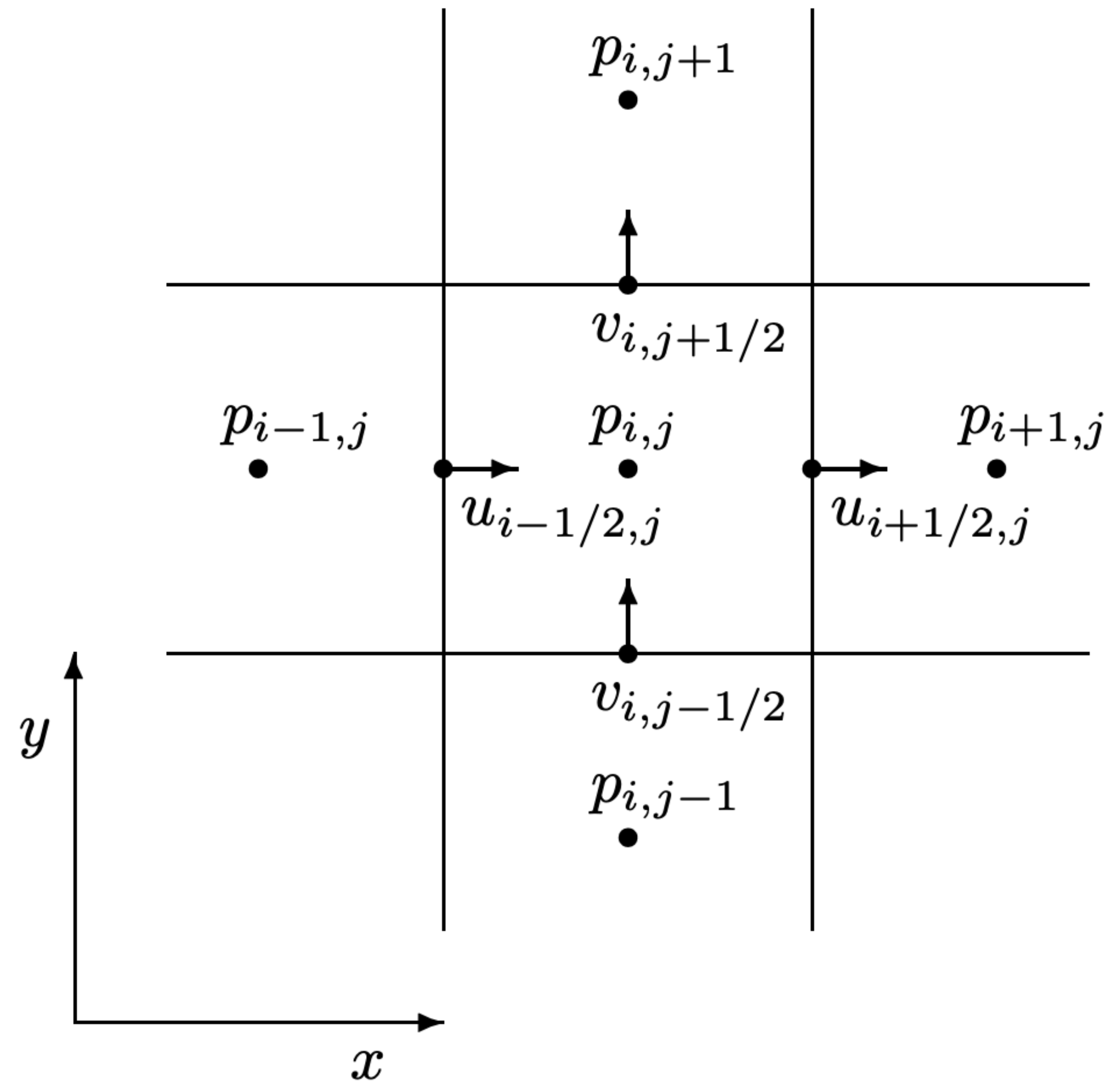
$u^{n+1} \leftarrow$ **Solve** $\dfrac{\partial u}{\partial t} = -\dfrac{1}{\rho}\nabla p$ **s.t.** $\nabla \cdot u = 0$

After time discretization: $\dfrac{u^{n+1} - u^n}{h} = -\dfrac{1}{\rho}\nabla p$

$$u^{n+1} = u^n - h\frac{1}{\rho}\nabla p$$

We want $\nabla \cdot u^{n+1} = 0$**,**

or equivalently, $\nabla \cdot \nabla p = \dfrac{\rho}{h}\nabla \cdot u^n$ **— a Poisson Equation**



$$u^{n+1}_{i+1/2,j} = u_{i+1/2,j} - \Delta t\frac{1}{\rho}\frac{p_{i+1,j} - p_{i,j}}{\Delta x},$$

$$v^{n+1}_{i,j+1/2} = v_{i,j+1/2} - \Delta t\frac{1}{\rho}\frac{p_{i,j+1} - p_{i,j}}{\Delta x},$$

To avoid non-trivial null space of central difference,
we use MAC grid.

Boundary Conditions (BC)

Solid wall:

No-stick BC (for inviscid fluids): $\vec{u} \cdot \hat{n} = \vec{u}_{\text{solid}} \cdot \hat{n}.$

No-slip BC (for viscos fluids): $\vec{u} = \vec{u}_{\text{solid}}.$

Free surface:

$$M_{\hat{a}b} \frac{x^n_{b|\hat{i}} - (x^{n-1}_{b|\hat{i}} + hV^{n-1}_{b|\hat{i}})}{\Delta t^2} = \int_{\partial\Omega^0} N_{\hat{a}}(\mathbf{X}) \overset{=0}{T_{\hat{i}}(\mathbf{X}, t^n)} ds(\mathbf{X}) - \int_{\Omega^0} N_{\hat{a},j}(\mathbf{X}) P_{\hat{i}j}(\mathbf{X}, t^n) d\mathbf{X}$$

In Eulerian View: $\sigma \cdot n = 0$

For inviscid fluids: $p = 0$

The Particle-In-Cell Method

Time Splitting

For each time step n:

$u^a \leftarrow$ **Solve** $\dfrac{\partial u}{\partial t} + u \cdot \nabla u = 0$ **(advection)** — **Using particles**

Transfer velocity from particles to grid

$u^b \leftarrow$ **Solve** $\dfrac{\partial u}{\partial t} = g$ **(apply external force)**

$u^c \leftarrow$ **Solve** $\dfrac{\partial u}{\partial t} = \nu \nabla \cdot \nabla u$ **(diffusion)**

$u^{n+1} \leftarrow$ **Solve** $\nabla \cdot u = 0$ **(pressure projection)**

Using the grid

Transfer velocity from grid to particles

Extending to Hybrid Lagrangian/Eulerian Solid Simulation

The Material-Point Method

For each time step n: *Time Splitting*

$u^a \leftarrow$ **Solve** $\frac{\partial u}{\partial t} + u \cdot \nabla u = 0$ **(advection)** Using particles

Transfer velocity from particles to grid

$u^b \leftarrow$ **Solve** $\frac{\partial u}{\partial t} = g$ **(apply external force)**

$u^c \leftarrow$ **Solve** $\frac{\partial u}{\partial t} = \nu \nabla \cdot \nabla u$ **(diffusion)** Using the grid

$u^{n+1} \leftarrow$ **Solve** $\nabla \cdot u = 0$ **(pressure projection)**

Transfer velocity from grid to particles

The Particle-In-Cell Method

For each time step n: *Time Splitting*

$u^a \leftarrow$ **Solve** $\frac{\partial u}{\partial t} + u \cdot \nabla u = 0$ **(advection)** Using particles

Transfer information from particles to grid

$u^{n+1} \leftarrow$ **Solve** $\frac{\partial u}{\partial t} = \nabla \cdot \sigma + g$ Using the grid

Transfer information from grid to particles

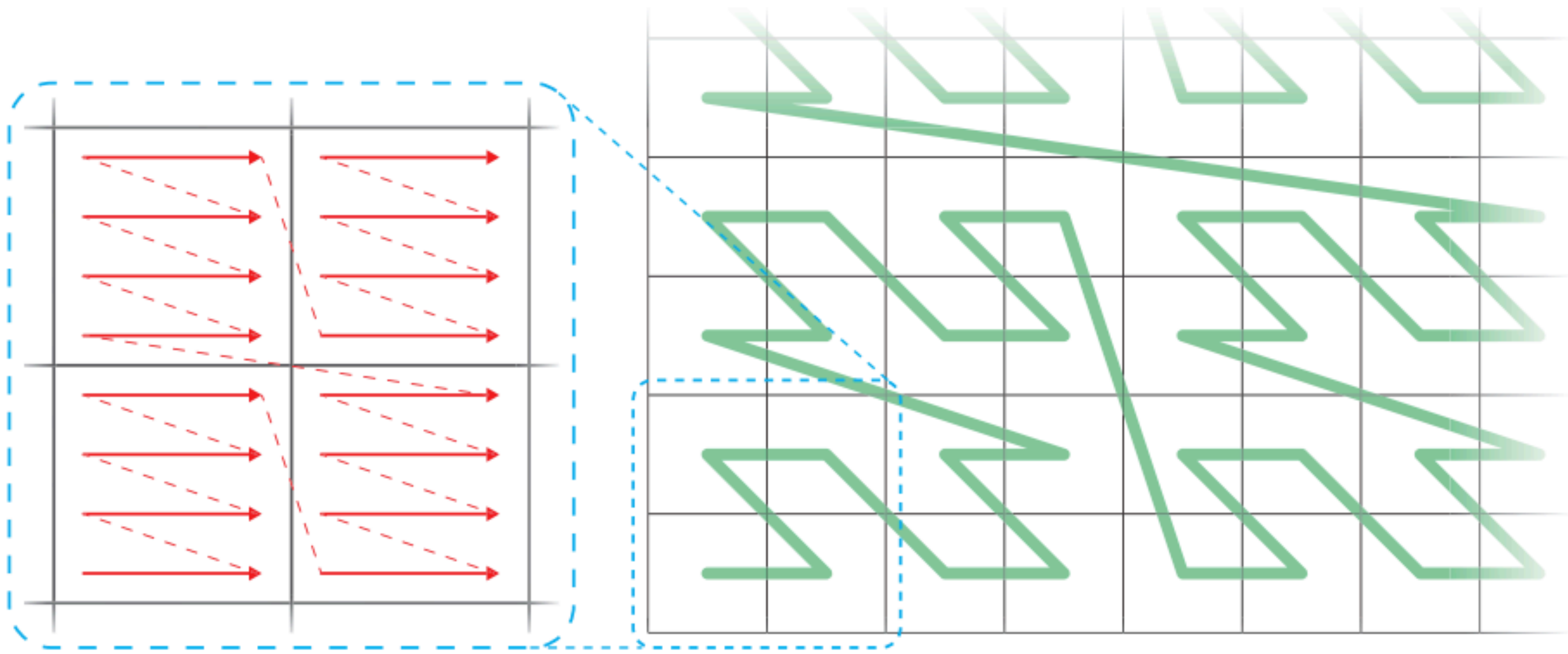
- **Needs to track deformation gradient per particle using updated Lagrangian:** $\mathbf{F}^{n+1} \approx \mathbf{F}^n + h\frac{\partial \mathbf{F}}{\partial t}$

$$\frac{\partial}{\partial t}\mathsf{F}(\mathbf{X}, t^{n+1}) = \frac{\partial \mathbf{V}}{\partial \mathbf{X}}(\mathbf{X}, t^{n+1}) = \frac{\partial \mathbf{v}^{n+1}}{\partial \mathbf{x}}(\phi(\mathbf{X}, t^n))\mathsf{F}(\mathbf{X}, t^n)$$

Sparse Grid for Better Efficiency

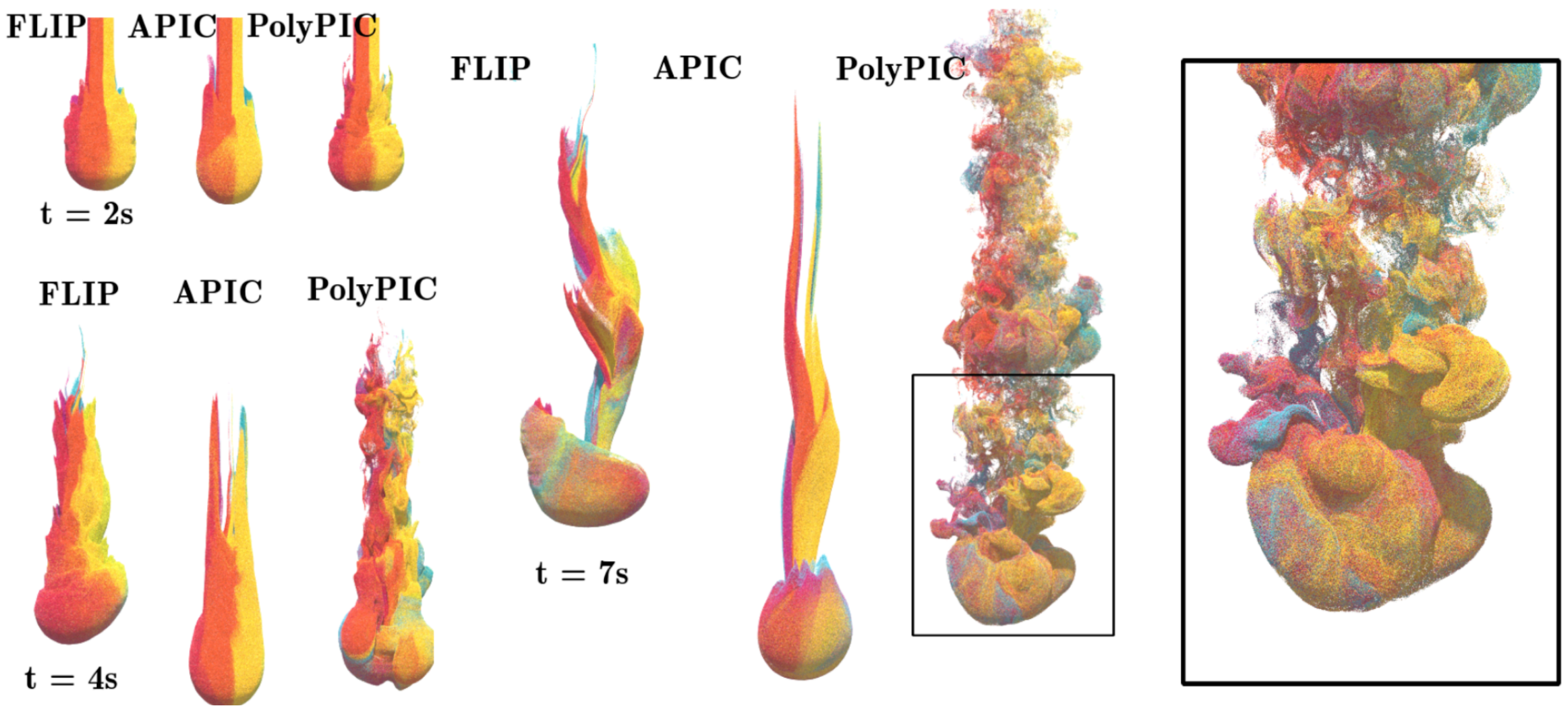
Only place grid cells at region of interests:

Z-order indexing with better data locality:



Improving Accuracy

- Particle-grid transfer:
    - FLIP, APIC, PolyPIC, PowerPIC, …
- Advection:
    - BiMocq, Covector Fluids, …
- Pressure Projection:
    - Advection-Reflection Solver
    - Cut-cell methods
    - …



(a) MC+R [Zehnder et al. 2018]
(b) BiMocq [Qu et al. 2019]
(c) CF (Covector Fluids)+MCM (Ours)

More Fluid Simulation Research

- Vortex methods *(Daniel will present a relevant paper)*
- Lattice Boltzmann methods (LBM)
  - Based on statistical physics
  - Well-suited for efficient simulation of turbulent flows
- Reduction:
  - modeling fluids as height fields
  - Applying model reduction *(Olga will present a relevant paper)*
- Solid-fluid coupling *(Sarah will present a relevant paper)*

This is the last lecture

Next Week

- Nov 28: Paper Presentation
  - Yin et al. Fluid Cohomology. SIGGRAPH 2023 **(Presenter: Daniel Zeng)**
  - Panuelos et al. PolyStokes: A Polynomial Model Reduction Method for Viscous Fluid Simulation. SIGGRAPH 2023 **(Presenter: Olga Guțan)**
- Nov 30: Paper Presentation
  - Fei et al. A Multi-Scale Model for Simulating Liquid-Hair Interactions. SIGGRAPH 2017 **(Presenter: Sarah Di)**
  - Rioux-Lavoie and Sugimoto et al. A Monte Carlo Method for Fluid Simulation. SIGGRAPH Asia 2022 **(Presenter: Daniel Zeng)**

Image Sources

- https://sph-tutorial.physics-simulation.org/
- https://en.wikipedia.org/wiki/Bilinear_interpolation
- https://docs.taichi-lang.org/docs/sparse
- https://orionquest.github.io/papers/SSPGASS/paper.html
- https://dl.acm.org/doi/pdf/10.1145/3130800.3130878
- https://cseweb.ucsd.edu/~viscomp/projects/SIG22CovectorFluids/paper/CovectorFluids.pdf