COL865: Special Topics in Computer Applications

Physics-Based Animation

14 — Fluid simulation on grids II

Review

Navier-Stokes equations for fluid velocity $\mathbf{u}(\mathbf{x}, t)$:

$$\partial \mathbf{u}/\partial t + (\mathbf{u} \cdot \nabla)\, \mathbf{u} = \rho^{-1} (-\nabla p + \mu\, \nabla^2 \mathbf{u} + \mathbf{f}_{ext})$$

$$\nabla \cdot \mathbf{u} = 0$$

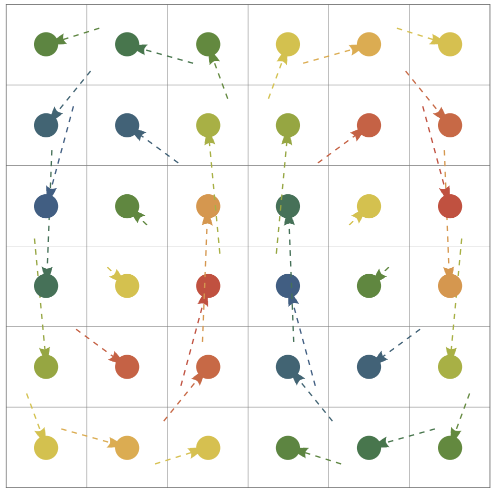
Solve on grid via ***splitting***:

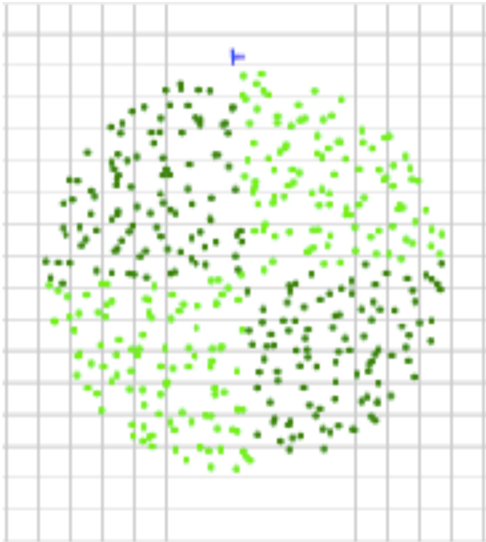
- ***Advection***: $\mathbf{u}^{(1)} = \text{advect}(\mathbf{u}^n, \mathbf{u}^n, \Delta t)$
- ***Body forces***: $\mathbf{u}^{(3)} = \mathbf{u}^{(2)} + \mathbf{f}_{ext}\, \Delta t$
- ***Viscosity***: $\mathbf{u}^{(2)} = \mathbf{u}^{(1)} + \nu\, \nabla^2 \mathbf{u}\, \Delta t$
- ***Pressure***: $\mathbf{u}^{n+1} = \mathbf{u}^{(3)} - \nabla p\, \Delta t$ so that $\nabla \cdot \mathbf{u}^{n+1} = 0$

Advection

$$\mathrm{D}\mathbf{u}/\mathrm{D}t \;=\; \partial\mathbf{u}/\partial t + (\mathbf{u}\cdot\nabla)\,\mathbf{u} \;=\; 0$$

- ***Finite differences***
  (time step limited by CFL condition)
- ***Semi-Lagrangian***
- ***Particle-based***: PIC, FLIP, APIC





Pressure

$$\mathbf{u}^{n+1} = \tilde{\mathbf{u}} - \nabla p \, \Delta t,$$
$$\nabla \cdot \mathbf{u}^{n+1} = 0$$

$$\Rightarrow \quad \nabla \cdot \tilde{\mathbf{u}} - \nabla^2 p \, \Delta t = 0$$

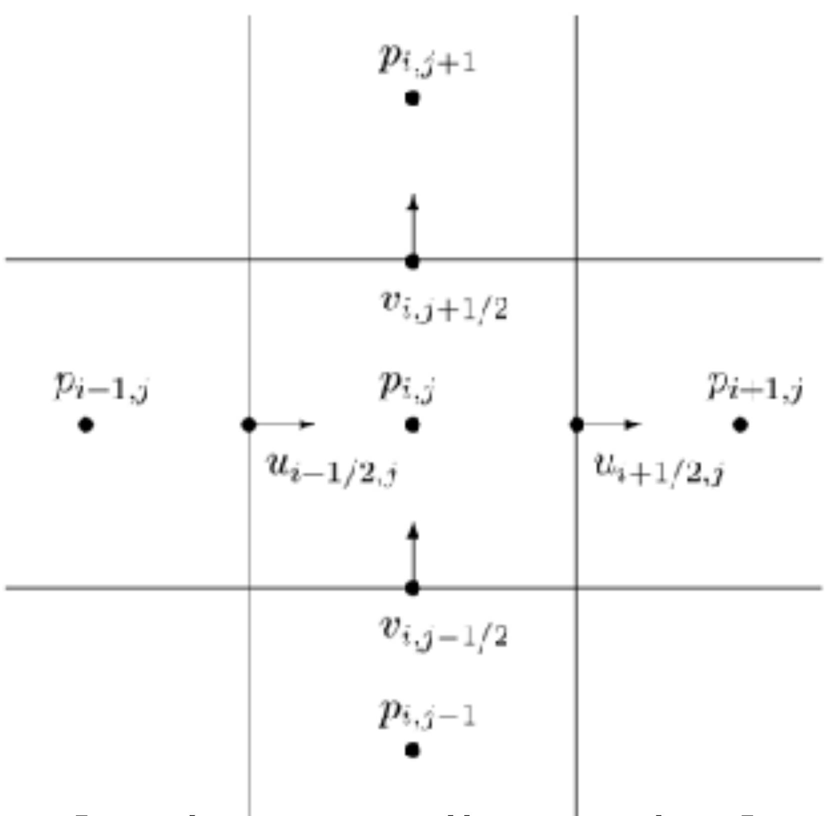
$\tilde{\mathbf{u}}$ = $\mathbf{u}^{n+1}$ + ∇p

[Tong et al. 2003]

Pressure on staggered grids

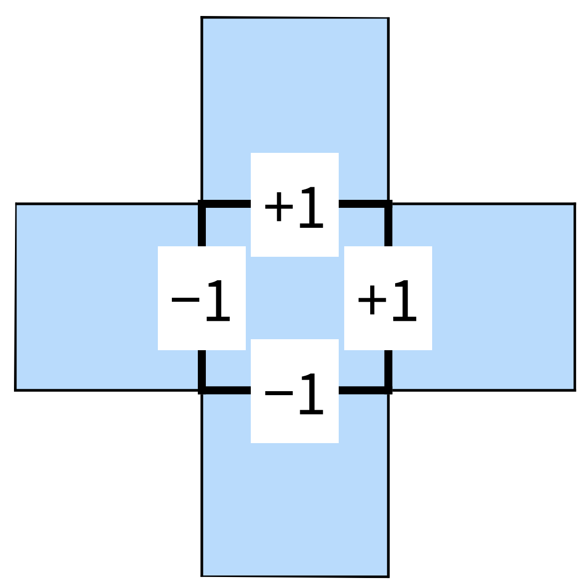
$$(\nabla \cdot \mathbf{u})_{i,j} \approx (u_{i+½,j} - u_{i-½,j})/\Delta x + (v_{i,j+½} - v_{i,j-½})/\Delta x$$

$$(\nabla^2 p)_{i,j} \approx (p_{i-1,j} + p_{i+1,j} + p_{i,j-1} + p_{i,j+1} - 4\, p_{i,j})/\Delta x^2$$

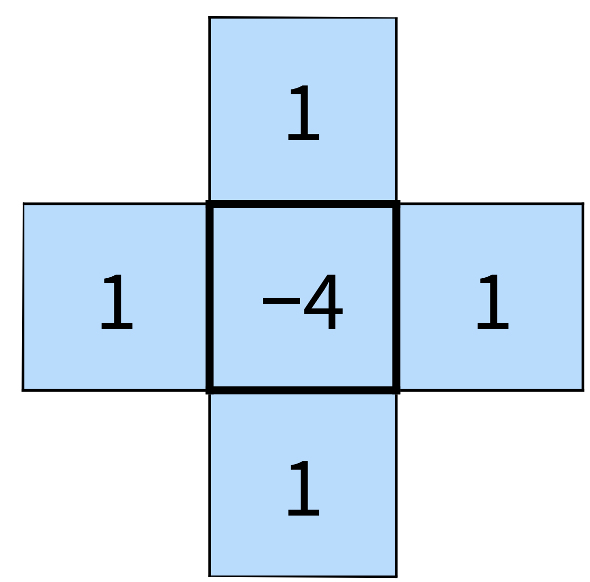


[Bridson & Müller-Fischer]

$\nabla \cdot \mathbf{u} \approx$



$\nabla^2 p \approx$



The pressure system

$$\nabla^2 p \, \Delta t = \nabla \cdot \tilde{\mathbf{u}}$$

1. Compute $(\nabla \cdot \tilde{\mathbf{u}})_{i,j} = d_{i,j}$ on cell centers
2. Put pressure values in a vector $\mathbf{p}$:
   $(\nabla^2 p \, \Delta t)_{i,j}$ becomes a linear operator $\mathbf{A}\,\mathbf{p}$

$$\begin{bmatrix} \ddots & & & & & & \\ & 1 & & 1 & -4 & 1 & & 1 & \\ & & & & & & & & \ddots \end{bmatrix} \begin{bmatrix} \vdots \\ p_{i-1,j} \\ \vdots \\ p_{i,j-1} \\ p_{i,j} \\ p_{i,j+1} \\ \vdots \\ p_{i+1,j} \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ (\nabla \cdot \mathbf{u})_{i,j} \\ \vdots \end{bmatrix}$$

(taking $\Delta t = \Delta x = 1$)

3. Solve $\mathbf{A}\,\mathbf{p} = \mathbf{d}$ for $\mathbf{p}$

The pressure system

$$\nabla^2 p \, \Delta t = \nabla \cdot \tilde{\mathbf{u}}$$

1. Compute $(\nabla \cdot \tilde{\mathbf{u}})_{i,j} = d_{i,j}$ on cell centers
2. Put pressure values in a vector $\mathbf{p}$:
   $(\nabla^2 p \, \Delta t)_{i,j}$ becomes a linear operator $\mathbf{A}\,\mathbf{p}$

$$\begin{bmatrix} \ddots & & & & & & \\ & -1 & & -1 & 4 & -1 & & -1 & \\ & & & & & & & & \ddots \end{bmatrix} \begin{bmatrix} \vdots \\ p_{i-1,j} \\ \vdots \\ p_{i,j-1} \\ p_{i,j} \\ p_{i,j+1} \\ \vdots \\ p_{i+1,j} \\ \vdots \end{bmatrix} = \begin{bmatrix} \vdots \\ -(\nabla \cdot \mathbf{u})_{i,j} \\ \vdots \end{bmatrix}$$

Negate for positive definiteness

3. Solve $\mathbf{A}\,\mathbf{p} = \mathbf{d}$ for $\mathbf{p}$

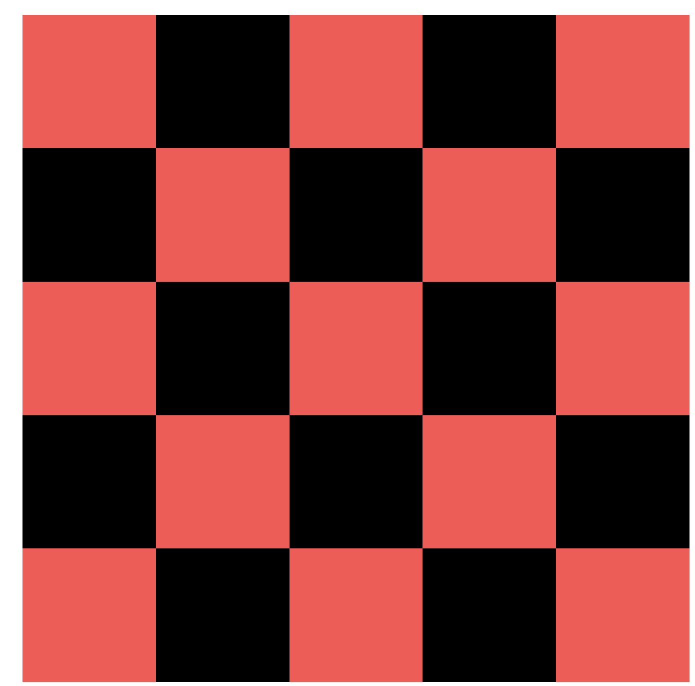
Solving the pressure system

- Easy way: ***Gauss-Seidel iterations***

$$p_{i-1,j} + p_{i+1,j} + p_{i,j-1} + p_{i,j+1} - 4\, p_{i,j} = (\nabla \cdot \tilde{\mathbf{u}})_{i,j}$$

$$\Rightarrow \quad p_{i,j} = \tfrac{1}{4}\left(p_{i-1,j} + p_{i+1,j} + p_{i,j-1} + p_{i,j+1} + (\nabla \cdot \tilde{\mathbf{u}})_{i,j}\right)$$

  Parallelization via ***red-black ordering***



- Better way: ***Preconditioned conjugate gradient method***

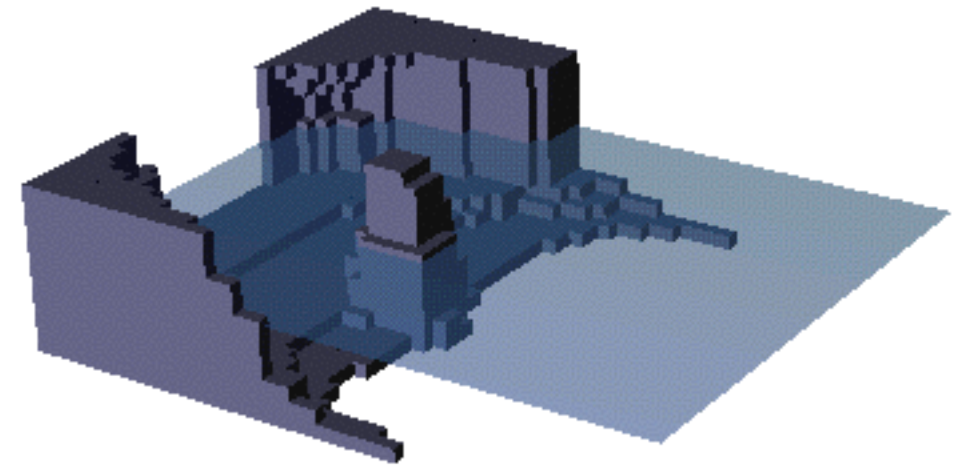
  See Bridson & Müller-Fischer, Ch 4.3

Boundaries

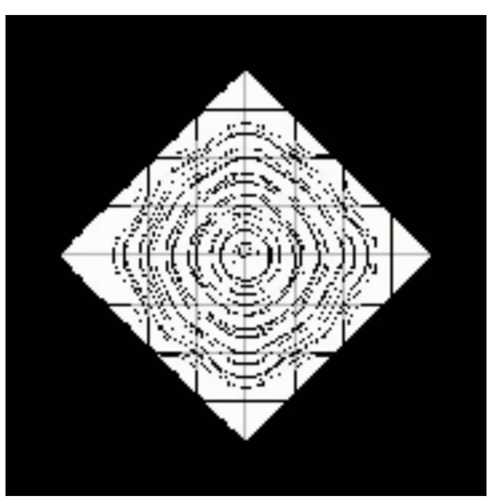
Static obstacles:

- Solid faces have $\mathbf{u} \cdot \mathbf{n} = 0$, do not contribute to $\nabla \cdot \mathbf{u}$
- Pressure shouldn't change $\mathbf{u} \cdot \mathbf{n}$, so $\nabla p \cdot \mathbf{n} = 0$

Limitation: Sloped boundaries are jagged

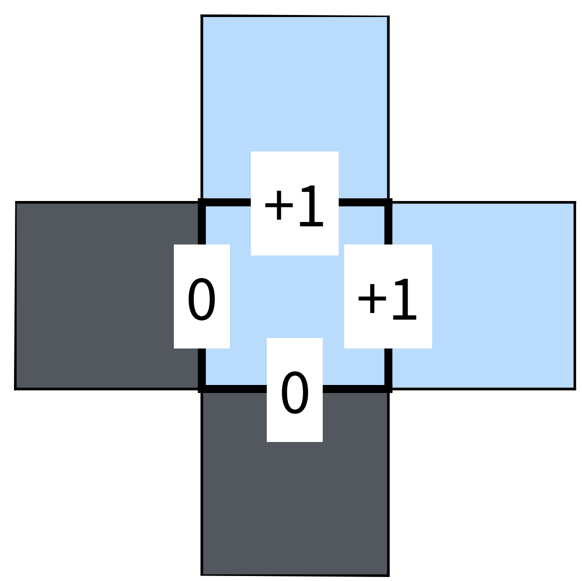


[Foster & Metaxas 1996]

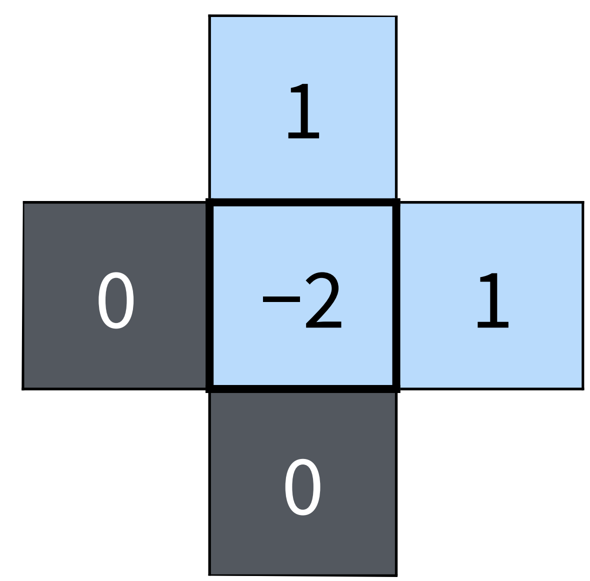


[Batty et al. 2007]

$\nabla \cdot \mathbf{u} \approx$



$\nabla^2 p \approx$



Putting it all together

A basic fluid simulator

Create staggered grid for domain, flag cells as solid/fluid

For each time step:

1. Compute $\tilde{\mathbf{u}} = \text{advect}(\mathbf{u}^n, \mathbf{u}^n, \Delta t)$
2. Add body forces: $\tilde{\mathbf{u}} \mathrel{+}= \rho^{-1}\,\mathbf{f}_{\text{ext}}\,\Delta t$
3. Do viscosity step if desired
4. Set $\mathbf{u}^{n+1} = \text{project}(\tilde{\mathbf{u}})$

Suppose $\mathbf{u}^0 = \mathbf{0}$, $\mathbf{f}_{\text{ext}} = \rho\,\mathbf{g}$… What happens?

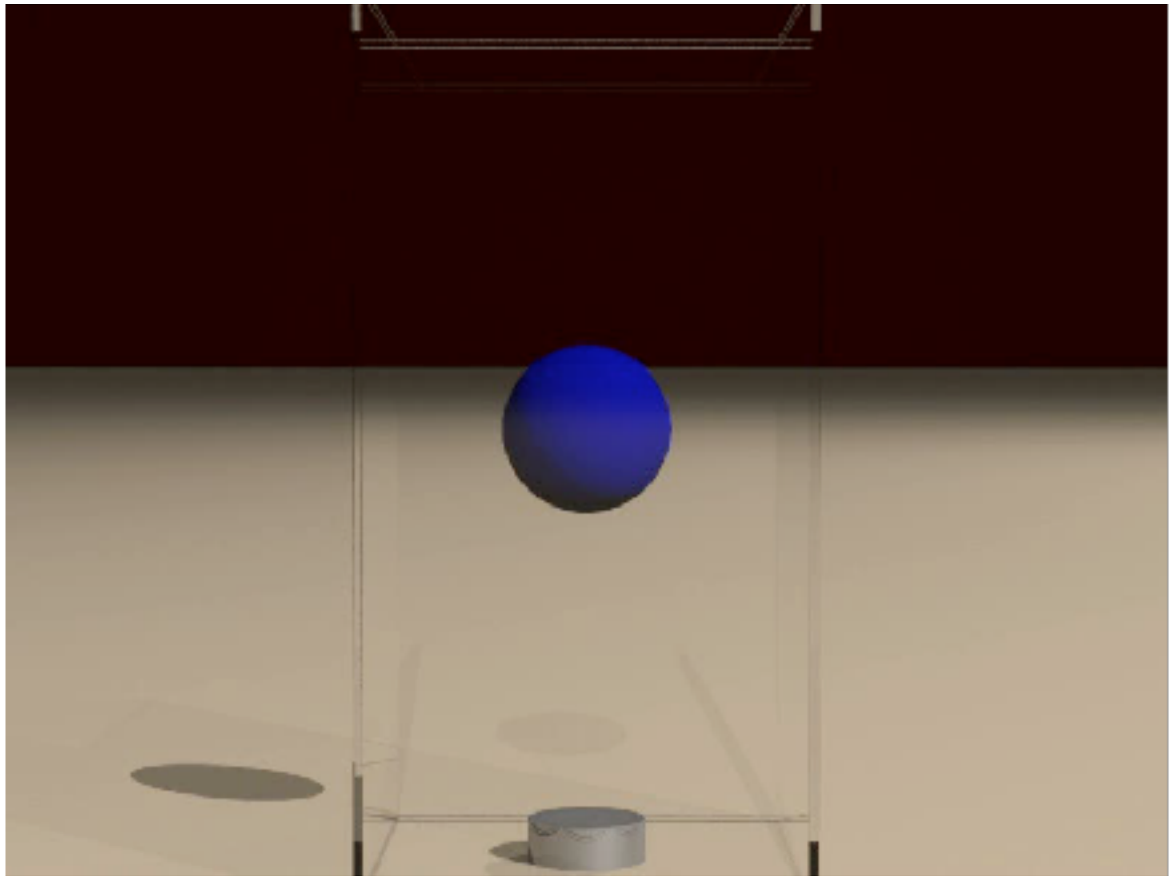
Smoke simulation

Scalar fields c : smoke density, T : (relative) temperature

1. Update $c^{n+1} = \text{advect}(c^n, \mathbf{u}^n, \Delta t)$,
   $T^{n+1} = \text{advect}(T^n, \mathbf{u}^n, \Delta t)$

2. Add buoyancy force:
   $\mathbf{f}_{\text{ext}} \mathrel{+}= (-\alpha\, c^{n+1} + \beta\, T^{n+1})\, \hat{\mathbf{z}}$

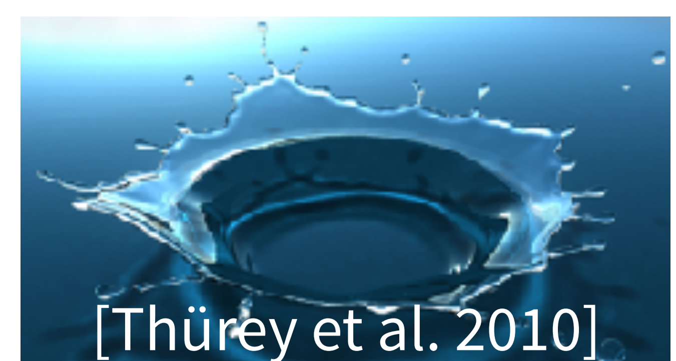
[Lentine et al. 2010]



Issues

- ***Numerical dissipation***
  - Diffusion in advection (almost eliminated by FLIP, APIC)
  - Energy loss in projection [see Zhang et al. 2015, Zehnder et al. 2018]
- ***Boundary handling***
  - Sloped solid boundaries [Batty et al. 2007]
  - Liquid surfaces (next up)
  - Small features (thin boundaries, fluid sheets, splashes) require hybrid methods

[Losasso et al. 2008]



[Thürey et al. 2010]

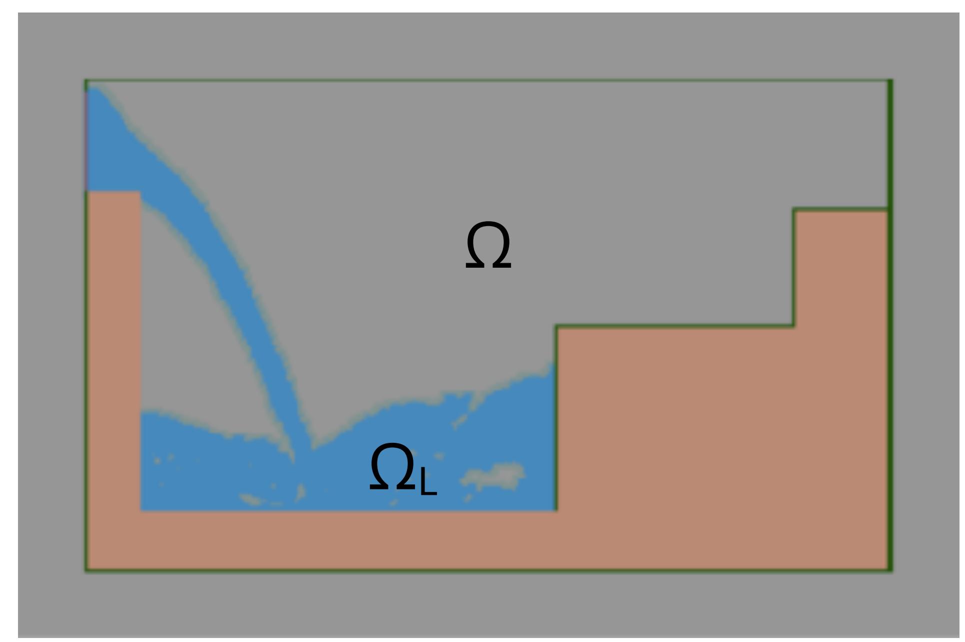
Liquids

Liquids

What's the difference between liquids and gases (in our model)?

Liquid region does not fill entire domain.
Liquid/air boundary is a ***free surface***
that moves with fluid velocity **u**

- How to represent liquid region?
- How to modify eqs. of motion?



[Foster & Metaxas 1996]

Surface tracking

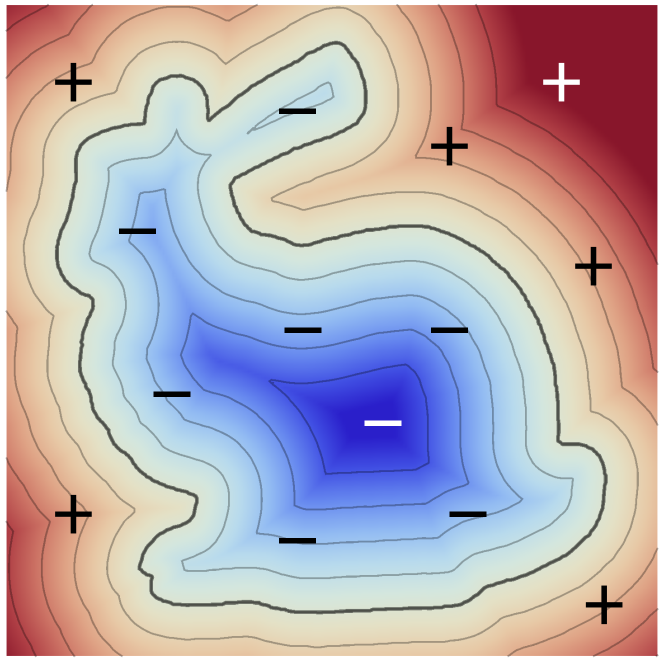
What we need:

- Determine whether cell is ***inside*** / ***outside*** liquid
- ***Advect*** through velocity field
- ***Reconstruct*** surface for rendering

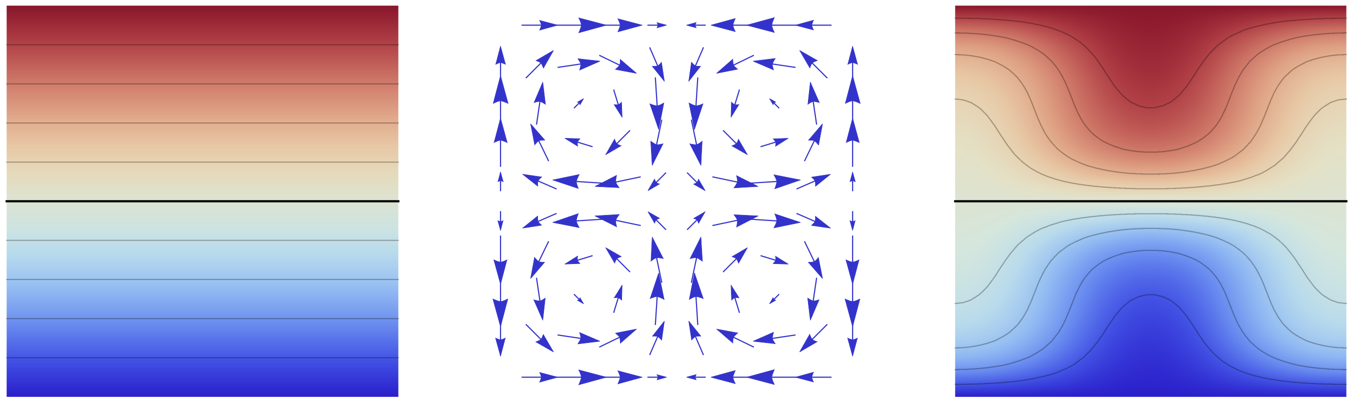
Level sets

[Foster & Fedkiw 2001, Bridson & Müller-Fischer Ch 6.2]

Represent region as sublevel set of scalar field φ, usually ***signed distance function***

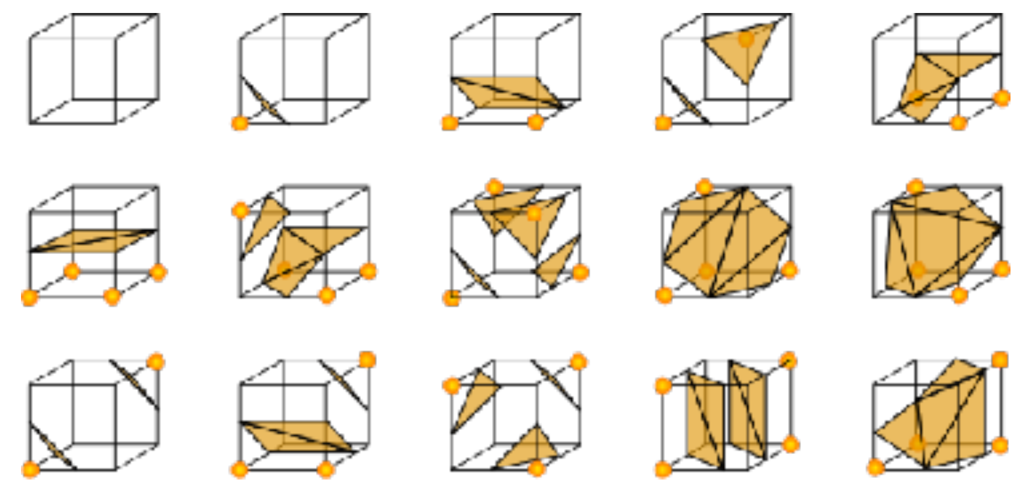


- $\varphi < 0$ inside, $\varphi > 0$ outside,
  $|\varphi|$ = distance to surface
- ***Inside/outside***: Just check sign of φ
- ***Advection***: Advect φ as scalar field, then do “redistancing”
  - Fast marching, fast sweeping methods



Level sets

- ***Reconstruction***: Marching cubes



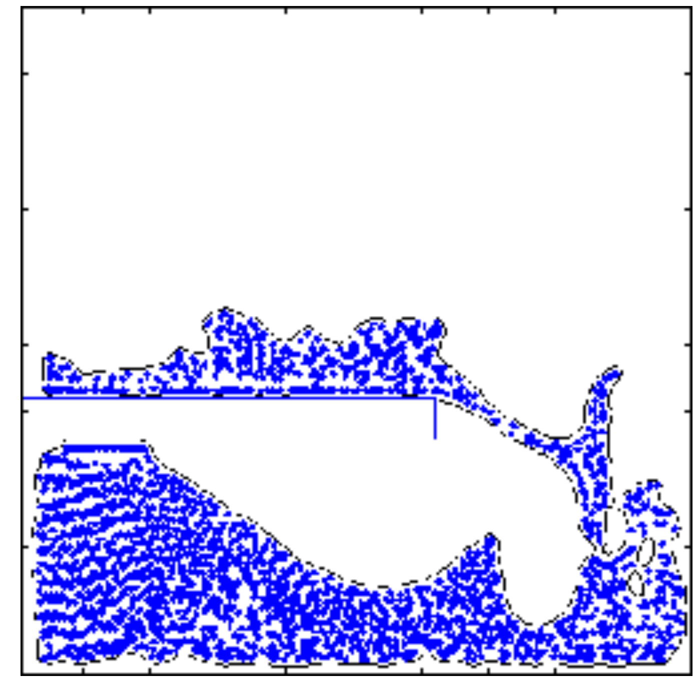
Advantages:

- Automatically handles topology changes

Disadvantages:

- Diffusion causes loss of volume & surface detail
- Requires periodic redistancing

Particles



[Zhu & Bridson 2005]

Place particles around the surface [Enright et al. 2002] or everywhere in liquid [Foster & Metaxas 1996, Zhu & Bridson 2005]

- ***Advection*** is trivial: just move particles
- ***Inside/outside***: Mark cell as fluid if it contains ***any*** particles
- ***Reconstruction***: Construct an SDF φ, then marching cubes (or directly render with ray tracing)
  - Most common approach: distance from "average neighbour" $\varphi(\mathbf{x}) = \|\mathbf{x} - \bar{\mathbf{x}}\| - \bar{r}$ [Zhu & Bridson 2005, Adams et al. 2007]

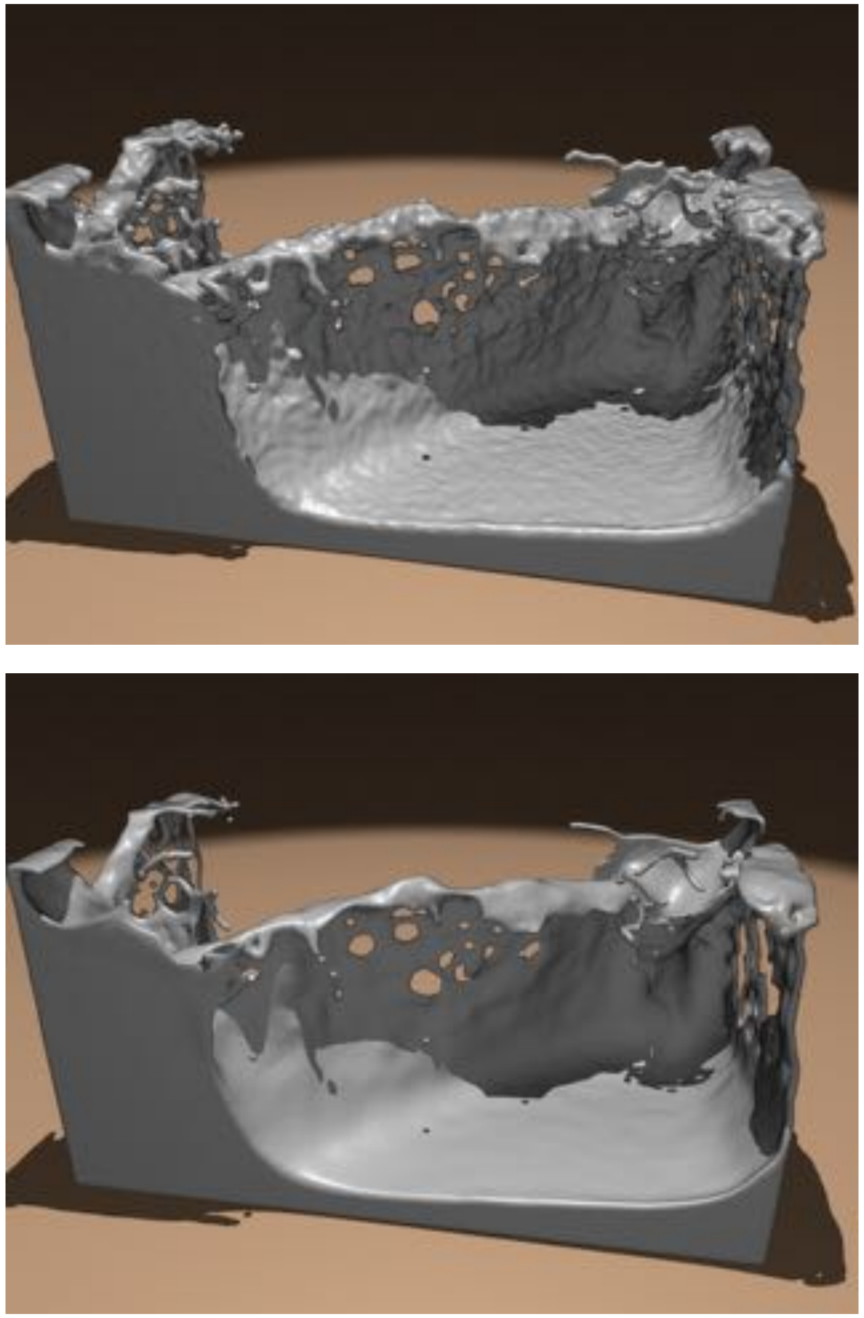
Particles

Advantages:

- Handles topology changes
- Better preserves volume
- Automatically produces droplets at splashes

Disadvantages:

- Bumpy surfaces
- Sheets tend to break up into droplets

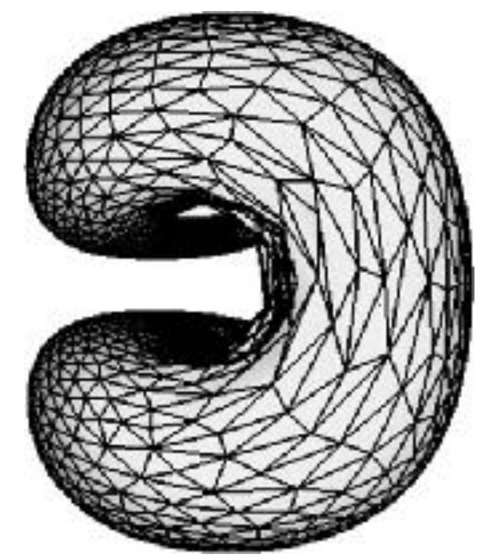


[Yu & Turk 2010]

Meshes

Store surface explicitly as triangle mesh [Wojtan et al. 2011]: no reconstruction necessary

- ***Inside/outside***: Ray casting
- ***Advection***: Move vertices (easy), then improve mesh (hard!)
  - Modify stretched/squashed triangles, deal with merging and splitting



[Brochu & Bridson 2009]

[Wojtan et al. 2009]

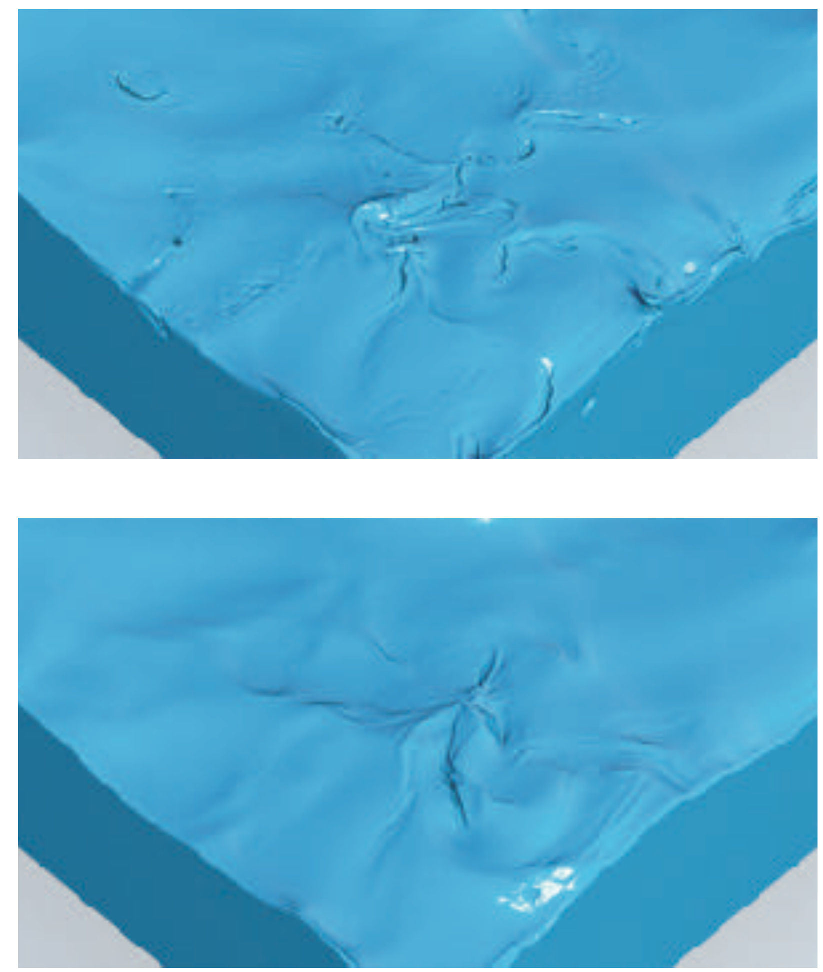
Meshes

Advantages:

- Highly accurate surfaces, great for surface tension effects
- Liquid sheets well preserved

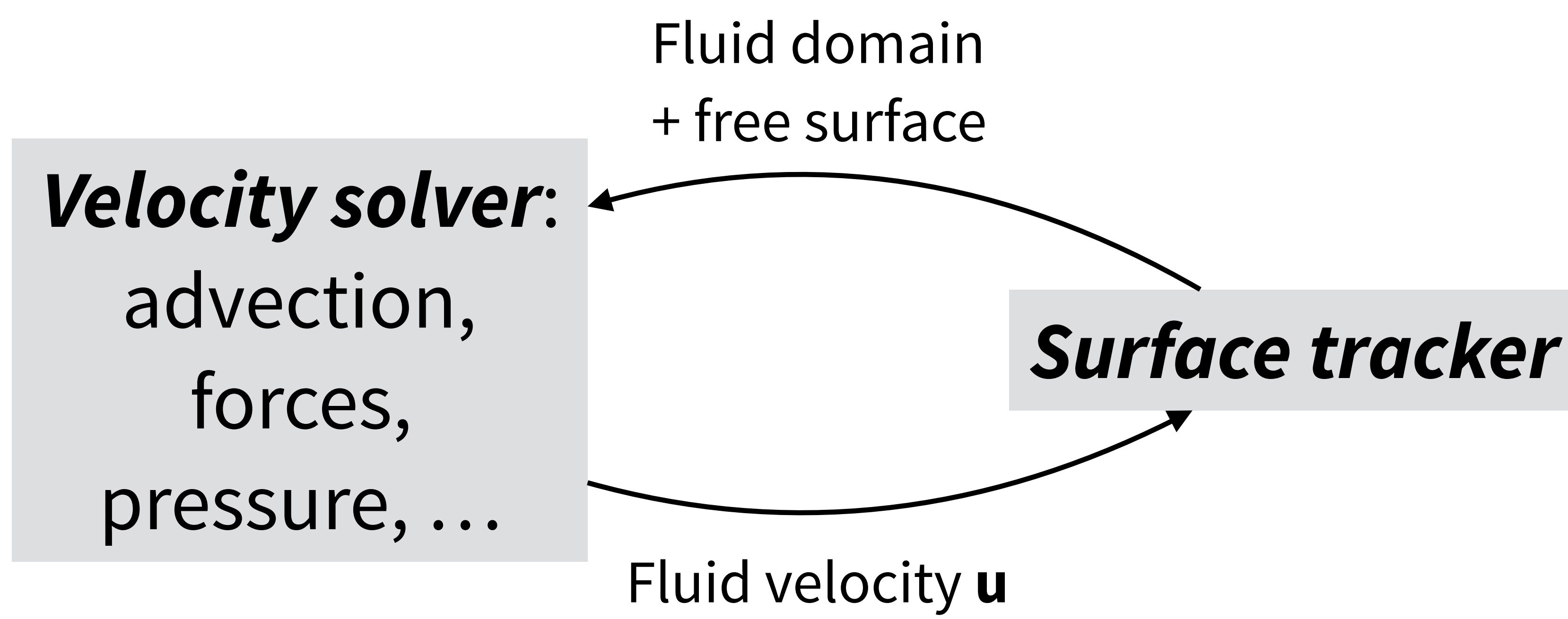
Disadvantages:

- Much more complicated to implement
- Grid dynamics may not “see” all the surface details



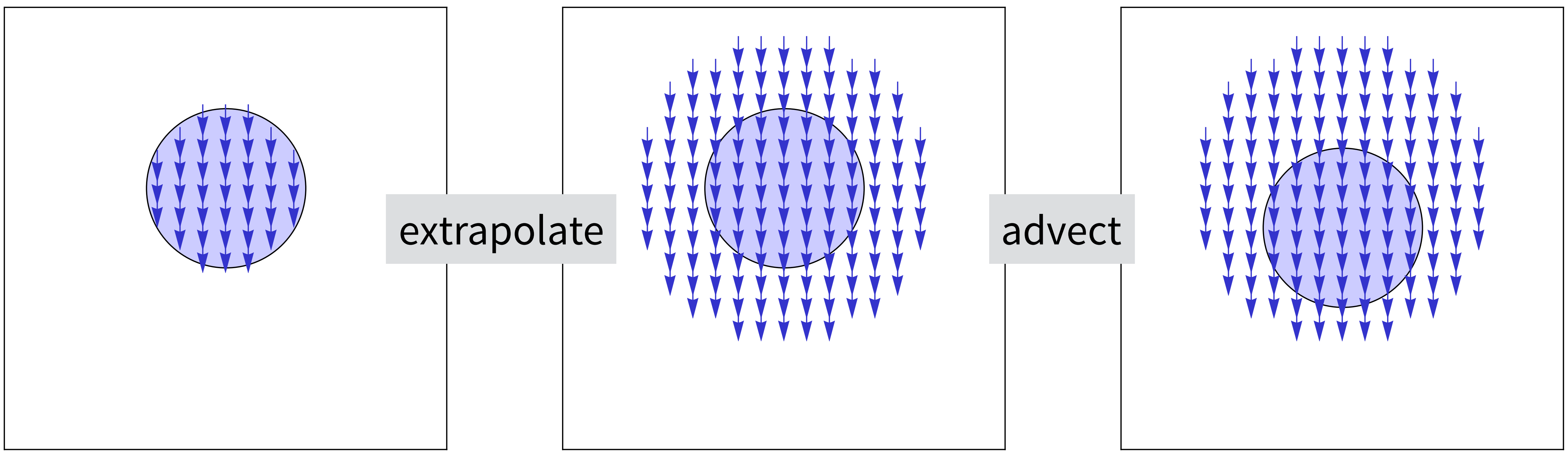
[Goldade et al. 2016]

Surface dynamics



Velocity extrapolation

Advection may query velocities ***outside*** current liquid region



Set **u**(air) = **u**(nearest fluid cell), similar to fast marching

Free surface boundary conditions

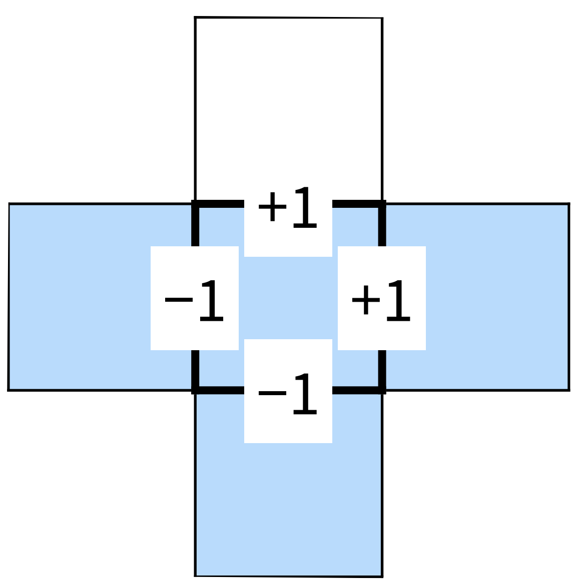
Assume air is at ***constant*** atmospheric pressure $p = p_{\text{atm}}$ (Dirichlet boundary condition)

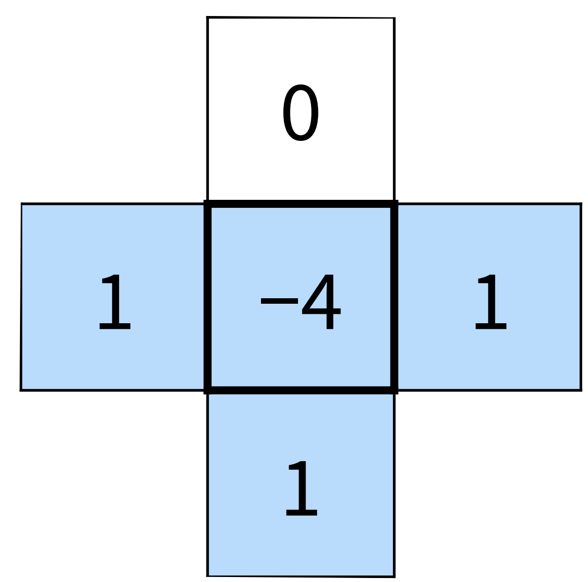
- Can assume $p_{\text{atm}} = 0$ (Why?)

Air cells drop out of Laplacian formula, e.g.

= 0

$$(\nabla^2 p)_{i,j} \approx (p_{i-1,j} + p_{i+1,j} + p_{i,j-1} + p_{i,j+1} - 4\,p_{i,j})/\Delta x^2$$

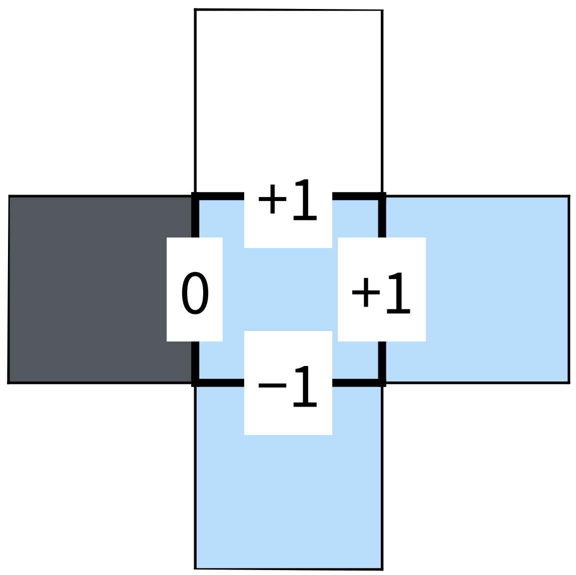
$\nabla \cdot \mathbf{u} \approx$ 

$\nabla^2 p \approx$ 

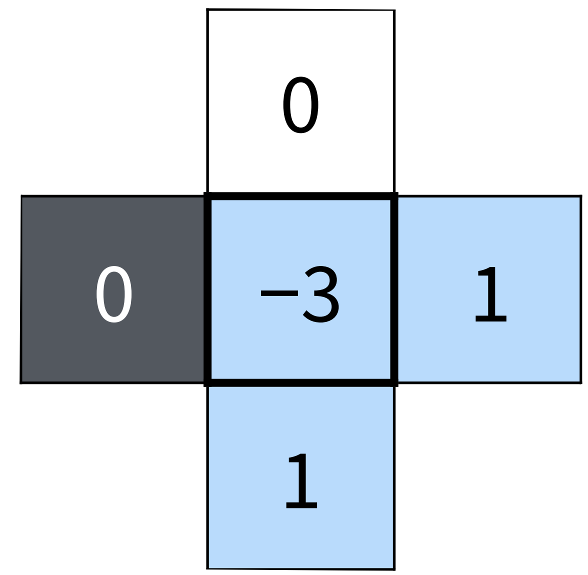
Free surface boundary conditions

With both solid and air neighbours:

$\nabla \cdot \mathbf{u} \approx$



$\nabla^2 p \approx$



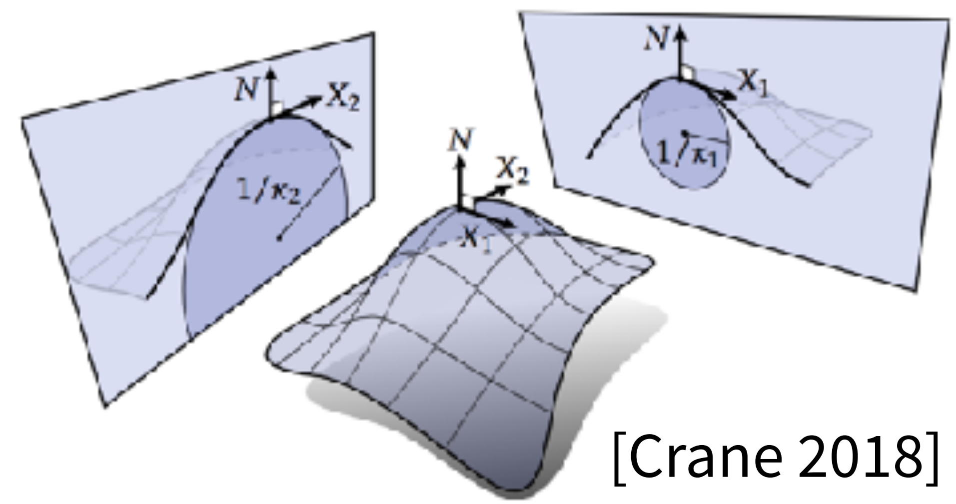
Sloped surfaces: [Gibou et al. 2002]
See Bridson & Müller-Fischer Ch 4.5.1

[Goldade et al. 2016]

Surface tension

surface tension coefficient → γ

Theory: force per unit area = $2\,\gamma\, H\, \hat{\mathbf{n}}$
where $H = (\kappa_1 + \kappa_2)/2$: mean curvature

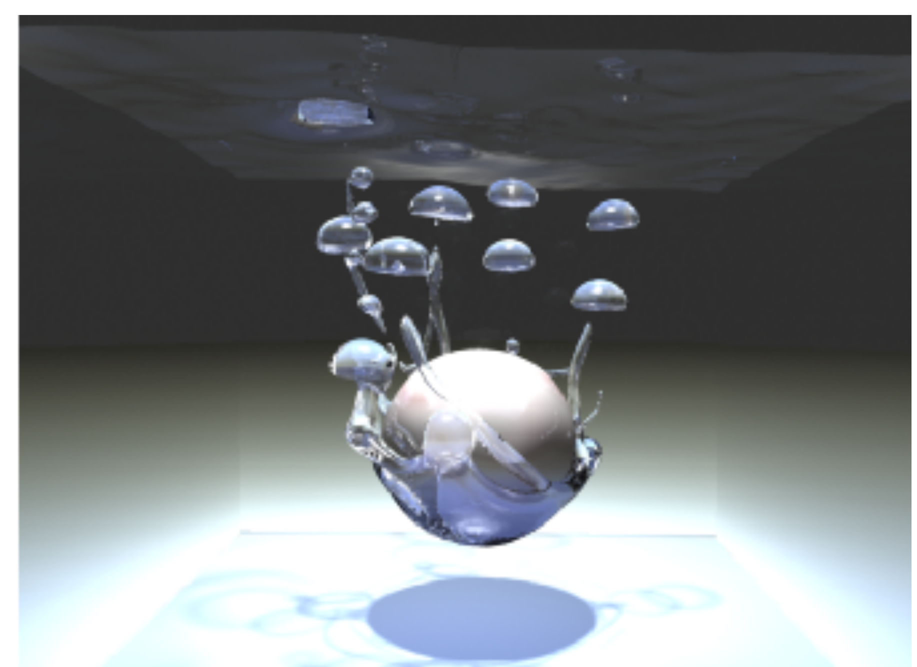


[Crane 2018]

One approach [Hong & Kim 2005]:

- Compute κ from SDF
- Apply pressure boundary condition
  $p = p_{\text{atm}} + 2\,\gamma\, H$

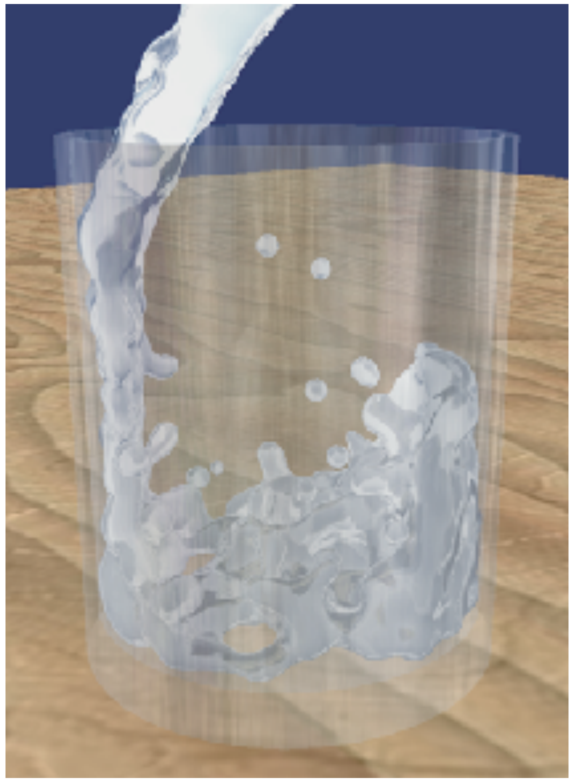
Problem: surface tension forces computed explicitly
⇒ time step restriction



[Hong & Kim 2005]

Next class

Fluid simulation with particles alone:
Smoothed particle hydrodynamics



Readings:

- Müller et al., “Particle-Based Fluid Simulation for Interactive Applications”, 2003
- Becker & Teschner, “Weakly Compressible SPH for Free Surface Flows”, 2007