Notes

- Error in last lecture slides: simplified level set reinitialization equation is
  \[ \phi_t + \text{sign}(\phi)\left|\nabla \phi \right| - 1 = 0 \]

- Decide on your final project
  - Talk to me about it preferably
  - However, I will not be around this afternoon, so email or waiting until tomorrow is fine

Particle-Level Set

- Last time - advocated marker particles (MAC) method for rough surfaces
- But if we want surface tension (which is strongest for rough flows!) or smooth water surfaces, we need a better technique
- Hybrid method: particle-level set
  - [Fedkiw and Foster], [Enright et al.]
  - Level set gives great smooth surface - excellent for getting mean curvature
  - Particles correct for level set mass (non-)conservation

Level set advancement

- Put marker particles with values of \( \phi \) attached in a band near the surface
  - We’re also storing \( \phi \) on the grid, so we don’t need particles deep in the water
  - For better results, also put particles with \( \phi > 0 \) (“air” particles) on the other side
- After doing a step on the grid and moving \( \phi \), also move particles with (extrapolated) velocity field
- Then correct the grid \( \phi \) with the particle \( \phi \)
- Then adjust the particle \( \phi \) from the grid \( \phi \)

Level set correction

- Look for escaped particles
  - Any particle on the wrong side (sign differs) by more than the particle radius \( |\phi| \)
  - Rebuild \( \phi < 0 \) and \( \phi > 0 \) values from escaped particles (taking min/max’s)
  - Merge rebuilt \( \phi < 0 \) and \( \phi > 0 \) by taking minimum-magnitude values
- Reinitialize new grid \( \phi \)
- Correct again
- Adjust particle \( \phi \) values within limits (never flip sign)
Artificial Compressibility

- Let’s make a quick detour…
- So far we’ve seen projection methods for enforcing divergence-free constraint
  - Means solving Poisson equation for pressure
  - Big, sparse linear system - it’s slow, it’s the bottleneck
  - Particularly on parallel architectures - global communication
  - Needs a weird staggered grid, or more complicated problems and fixes
- An alternative: artificial compressibility

Revisiting incompressibility

- Real fluids are not incompressible
- We just make the idealization of incompressibility
  - Water, air are very close unless material velocity comparable to sound speed (transonic or faster)
  - Simplifies math a lot
  - Means we can ignore sound waves in numerical methods - terrible time step limit
- But we could go the other way
  - Replace real compressible physics with fake ones that still have sound speed much faster than material velocity

Equation of state

- Turn hard constraint $\nabla \cdot u = 0$ into soft constraint
  - Allow the fluid to compress a little, but add restoring force to get it back
- Real compressible flow does this, but with all sorts of complications from thermodynamics
- We’ll fake it, simplify compressible flow
  - We don’t care about compressibility effects and ideally won’t even see them at all
- Artificial equation of state: $p = c^2 \rho$
- [Chorin ’67]

New equations

- Need to include density again (continuity eq. = conservation of mass)
  $$\rho_t + \nabla \cdot (\rho u) = 0$$
  $$\rho_t + u \cdot \nabla \rho = -\rho \nabla \cdot u$$
- And momentum equation
  $$u_t + u \cdot \nabla u + \frac{1}{\rho} \nabla p = g + \frac{1}{\rho} \nabla \cdot \mu \left( \nabla u + \nabla u^T \right)$$
- And the new equation of state
  $$p = c^2 \rho$$
What is c?

• [derive sound speed = c]
• We want to make sure that the maximum material speed (u) is much less than c
  • E.g. choose c at least 10 |u|_{max}
• Note that time step limit (for explicit methods) will have Δt<Δx/c
  • Hope is that 10+ times the number of steps is worth it for no pressure solve, easier programming, etc.

Where to now?

• With (simplified) compressible flow, it’s all about advection
• Lagrangian particles handle advection brilliantly
  • Motivation for semi-Lagrangian method
• Let’s look at using real particles
• We’re moving in a spectrum from fully Eulerian (finite differences) to mixed Eulerian/Lagrangian, eventually to fully Lagrangian

PIC

• Let’s dig up some CFD history (for compressible flow)
• PIC = Particle-in-Cell [Harlow’64]
• Opposite of semi-Lagrangian advection:
  • Keep particles in the grid
  • At each step, interpolate grid values onto particles
  • Then move particles (advection)
  • Transfer back to the grid (weighted averages)
• Problem - way too much diffusion - but it did allow unthinkably complex physics early on

FLIP

• Fluid-Implicit Particle [Brackbill & Ruppel ‘85]
• Fixed PIC by making particles first class
  • Their values for u, p, etc. are not overwritten by grid interpolation
• Each step transfer from particles to grid
• Do the non-advection grid stuff (∇p, g, …)
  • Easy to handle non-advection stuff on a grid
• Update particle values by grid increments
  • Including positions - use grid velocity
• Eventually morphed into MPM (Material Point Method) [Sulsky et al ‘94]
**SPH**

- Smoothed Particle Hydrodynamics
- Get rid of the mesh altogether - figure out how to do $\nabla p$ etc. with just the particles
- Before we get there, let’s hack around a bit…

**Particle Systems Redux**

- Long ago mentioned particle systems are incredibly flexible when you allow forces to depend on other particles
- One example: fix springs between particles $\rightarrow$ simple elasticity model
- We can similarly rig up a simple fluid model
  - Each particle is a blurry chunk of fluid - may overlap
  - Instead of a fixed mesh, particles just interact with nearby particles

---

**Particle fluids**

- Basic qualitative behaviour of fluids: resist density changes
  - When particles get too close, add repulsion forces between them
  - When they get just a little too far, add attraction forces
  - When far, no force at all
- If we want viscosity too, add (essentially) velocity damping between nearby particles
  - A little tricky to conserve angular momentum as well…

**Getting specific**

- Each particle has a mass $m$, and a (blurry) radius $h$
- Force potential (for pressure)
  - $[\text{draw it}]$
  - $E_{ij} = g(|x_i - x_j|/h)$
  - $F_i = \sum_j \nabla_i E_{ij}$
- Boundaries: can treat the same way
  - If we have signed distance, plug it in
  - If not, just nail particles to the boundaries that the fluid particles can interact with
Mesh-free?

- Mathematically, SPH and particle-only methods are independent of meshes
- Practically, need an acceleration structure to speed up finding neighbouring particles (to figure out forces)
- Most popular structure (for non-adaptive codes, i.e. where $h =$ constant for all particles) is...

  a mesh (background grid)

SPH

- SPH can be interpreted as a particular way of choosing forces, so that you converge to solving Navier-Stokes
- [Lucy’77], [Gingold & Monaghan ’77], [Monaghan…], [Morris, Fox, Zhu ’97], …
- Similar to FEM, we go to a finite dimensional space of functions
  - Basis functions now based on particles instead of grid elements
  - Can take derivatives etc. by differentiating the real function from the finite-dimensional space

Kernel

- Need to define particle’s influence in surrounding space (how we’ll build the basis functions)
- Choose a kernel function $W$
  - Smoothed approximation to $\delta$
  - $W(x) = W(|x|)$ - radially symmetric
  - Integral is 1
  - $W=0$ far enough away - when $|x| > 2.5h$ for example
- Examples:
  - Truncated Gaussian
  - Splines (cubic, quartic, quintic, …)

Cubic kernel

- Use $W(x) = \frac{1}{h^3} f \left( \frac{|x|}{h} \right)$ where

  $f(s) = \frac{1}{\pi} \begin{cases} 
  1 - \frac{3}{4} s^2 + \frac{3}{4} s^3, & 0 \leq s \leq 1 \\
  \frac{1}{4} (2 - s)^3, & 1 \leq s \leq 2 \\
  0, & 2 \leq s 
\end{cases}$

- Note: not good for viscosity (2nd derivatives involved - not very smooth)
Estimating quantities

- Say we want to estimate some flow variable \( q \) at a point in space \( x \)
- We'll take a mass and kernel weighted average
- Raw version: \( Q(x) = \sum_j m_j q_j W(x - x_j) \)
  - But this doesn't work, since sum of weights is nowhere close to 1
  - Could normalize by dividing by \( \sum_j m_j W_j \) but that involves complicates derivatives...
  - Instead use estimate for normalization at each particle separately (some mass-weighted measure of overlap)

\[
Q(x) = \sum_j m_j q_j W(x - x_j)
\]

Smoothed Particle Estimate

- Take the "raw" mass estimate to get density: \( \langle \rho(x) \rangle = \sum_j m_j W(x - x_j) \)
- [check dimensions]
- Evaluate this at particles, use that to approximately normalize:

\[
\langle q(x) \rangle = \sum_j q_j \frac{m_j W(x - x_j)}{\rho_j}
\]

Incompressible Free Surfaces

- Actually, I lied
  - That is, regular SPH uses the previous formulation
  - For doing incompressible flow with free surface, we have a problem
  - Density drop smoothly to 0 around surface
  - This would generate huge pressure gradient, compresses surface layer
- So instead, track density for each particle as a primary variable (as well as mass!)
  - Update it with continuity equation
  - Mass stays constant however - probably equal for all particles, along with radius

Continuity equation

- Recall the equation is
  \[
  \rho_i + u \cdot \nabla \rho = -\rho \nabla \cdot u
  \]
- We'll handle advection by moving particles around
- So we need to figure out right-hand side
- Divergence of velocity for one particle is
  \[
  \nabla \cdot \mathbf{v} = \nabla \cdot \left( v_j W(x - x_j) \right) = v_j \cdot \nabla W_j
  \]
- Multiply by density, get SPH estimate:

\[
\langle \rho \nabla \cdot \mathbf{v} \rangle = \sum_j m_j v_j \cdot \nabla W_j
\]
**Momentum equation**

- Without viscosity: \( u_t + u \cdot \nabla u = -\frac{1}{\rho} \nabla p + g \)
- Handle advection by moving particles
- Acceleration due to gravity is trivial
- Left with pressure gradient
- Naïve approach - just take SPH estimate as before

\[
\frac{dv_i}{dt} = \left( -\frac{1}{\rho} \nabla p \right) = -\sum_j m_j \frac{p_j}{\rho_j^2} \nabla W_{ij}
\]

**Conservation of momentum**

- Remember momentum equation really came out of \( F=ma \) (but we divided by density to get acceleration)
- Previous slide - momentum is not conserved
  - Forces between two particles is not equal and opposite
- We need to symmetrize this somehow

\[
\frac{dv_i}{dt} = -\sum_j m_j \left( \frac{p_i}{\rho_i^2} + \frac{p_j}{\rho_j^2} \right) \nabla W_{ij}
\]

- [check symmetry - also note angular momentum]

**SPH advection**

- Simple approach: just move each particle according to its velocity
- More sophisticated: use some kind of SPH estimate of \( v \)
  - keep nearby particles moving together, like PIC and FLIP
- XSPH

\[
\frac{dx_i}{dt} = v_i + \sum_j m_j \left( v_j - v_i \right) \frac{1}{\frac{1}{2} \left( \rho_i + \rho_j \right)} W_{ij}
\]

**Equation of state**

- Some debate - maybe need a somewhat different equation of state if free-surface involved
- E.g. [Monaghan’94]

\[
p = B \left( \frac{\rho}{\rho_0} \right)^7 - 1
\]

- For small variations, looks like gradient is the same [linearize]
  - But SPH doesn’t estimate \(-1\) exactly, so you do get different results…
Incompressible SPH

- Can actually do a pressure solve instead of using artificial compressibility
- But if we do exact projection get the same kinds of instability as collocated grids
  - And no alternative like staggered grids available
- Instead use approximate pressure solve
- [Cummins & Rudman ‘99]