Grid-based fluid simulation

- Recap: **Eulerian** viewpoint
  - Grid is fixed, fluid moves through it
  - How does the velocity at a grid cell change over time?
Consider a weather balloon moving with the wind, measuring air temperature $T(x, t)$.
Eulerian and Lagrangian time derivatives

Consider a weather balloon moving with the wind. It measures air temperature as $T(x(t), t)$

$$
\frac{dT}{dt}(x(t), t) = \frac{\partial T}{\partial t} + \frac{\partial T}{\partial x} \frac{dx}{dt}
$$

$\frac{\partial T}{\partial t} + \nabla T \cdot \frac{dx}{dt}$

$T = \frac{\partial T}{\partial t} + u \cdot \nabla T$

Temperature as seen by balloon

Velocity of balloon

= velocity of air
Eulerian and Lagrangian time derivatives

$$\frac{DT}{Dt} = \frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T$$

Lagrangian derivative:
change seen by a point moving with the fluid

Eulerian derivative:
change at a fixed point

Change due to movement of fluid ("advection")
The fluid equations

• Newton’s second law

\[ \mathbf{a} = \mathbf{f}/\rho \]

• Forces:

\[ \mathbf{f} = \mathbf{f}^{\text{ext}} - \nabla p + \mu \nabla^2 \mathbf{u} \]

Reminder:
Velocity is \( \mathbf{u} \) now

• Acceleration is Lagrangian:

\[ \mathbf{a} = \left( \frac{\mathbf{D}\mathbf{u}}{Dt} \right) = \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} \]

Acceleration of fluid “molecule”
Change at a grid cell
Flow advects itself!
The fluid equations

\[ \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = \frac{1}{\rho} (f^{\text{ext}} - \nabla p + \mu \nabla^2 \mathbf{u}) \]

\[ p = ? \]

- The Navier-Stokes equations

Millenium prize: $1,000,000 to prove (or disprove) existence & smoothness of solutions

C.-L. Navier  
G.G. Stokes
Operator splitting

\[ \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = \frac{1}{\rho} \left( \mathbf{f}^{\text{ext}} + \nabla p + \mu \nabla^2 \mathbf{u} \right) \]

- Lots of different terms; hard to integrate in one go
- Deal with one term at a time
  - (ignoring all the others)
Operator splitting

\[ \frac{\partial u}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = 0 \quad \text{for time } \Delta t \]

\[ \frac{\partial u}{\partial t} = \frac{1}{\rho} \mathbf{f}^{\text{ext}} \quad \text{for time } \Delta t \]

\[ \frac{\partial u}{\partial t} = \frac{\mu}{\rho} \nabla^2 \mathbf{u} \quad \text{for time } \Delta t \]

\[ \frac{\partial u}{\partial t} = -\frac{1}{\rho} \nabla p \quad \text{for time } \Delta t \]

\[ \mathbf{u}^{n+1} \]
Advection

\[ \frac{\partial u}{\partial t} + u \cdot \nabla u = 0 \]

- Transport a quantity (in this case, \( u \)) via the velocity field \( u \)

- Confusing! Let’s transport something else first

\[ \frac{\partial A}{\partial t} + u \cdot \nabla A = 0 \]

- e.g. colour, temperature, concentration of ink in water / smoke in air / etc.
Advection

\[
\frac{\partial A}{\partial t} + \mathbf{u} \cdot \nabla A = 0
\]

Transport \( A \) by tracing **backwards** and looking up its value
Advection

- Input: initial grid $A^n$, velocity field $u^n$
- Output: final grid $A^{n+1}$
- For each grid cell $x_i$
  - Backtrace position, e.g. $x^{\text{back}} = x_i - u_i \Delta t$
  - Set output $A^{n+1}_i = \text{interpolate} \ A^n$ at $x^{\text{back}}$

To advect velocities, just feed in $u^n$ as the initial grid too.
External forces

- Gravity
- Buoyancy
- User interaction

Buoyancy strength

\[ f_{\text{ext}} = -\beta(T - T_{\text{amb}})g \]

Temperature of fluid

“Ambient” temperature
External forces

Lentine, Zheng, and Fedkiw, 2010
Pressure

Becker and Teschner, 2007
Incompressibility

- We will prohibit compressibility from our simulation.

\[ \text{Net flow into/out of region} = \int \int \mathbf{u} \cdot \hat{n} \, dA \]

\[ = \int \int \int \nabla \cdot \mathbf{u} \, dV \]  

[Divergence theorem]

We want net flow to be 0 for all possible regions, so...

\[ \nabla \cdot \mathbf{u} = 0 \text{ everywhere} \]
Pressure

\[ \frac{\partial u}{\partial t} = -\frac{1}{\rho} \nabla p \]

\[ p = ? \]

• Let’s just do forward Euler…
Pressure

\[ u^{\text{new}} = u - \frac{\Delta t}{\rho} \nabla p \]

\[ \nabla \cdot u^{\text{new}} = 0 \]

- Let’s just do forward Euler…

- …and choose the \( p \) which makes \( u^{\text{new}} \) divergence-free

\[ \nabla \cdot u - \frac{\Delta t}{\rho} \nabla^2 p = 0 \]

\[ \nabla^2 p = \frac{\rho}{\Delta t} \nabla \cdot u \]

Ignore the \( \rho/\Delta t \) (just a rescaling)
The Helmholtz-Hodge decomposition

- Equivalently: separate $\mathbf{u}$ into curl-free and divergence-free components
  ...then throw away the curl-free part

$\mathbf{u} = \nabla \rho + \mathbf{u}^{\text{new}}$
Pressure

• We just have to solve $\nabla^2 p = \nabla \cdot u$

• How?

• Q1: How to evaluate $\nabla \cdot$ and $\nabla^2$ on a grid

• Q2: How to store $p$ and $u$ on the grid in the first place
Finite differences in 1D

\[ A'(x) = \lim_{h \to 0} \frac{A(x + h) - A(x)}{h} \]

- On a grid, we only have samples at grid spacing \( \Delta x \)

\[ A_i' \approx \frac{A_{i+1} - A_i}{\Delta x} \]

or

\[ A_i' \approx \frac{A_i - A_{i-1}}{\Delta x} \]

or

\[ A_i' \approx \frac{A_{i+1} - A_{i-1}}{2\Delta x} \]
Finite differences in 2D

\[ \nabla^2 A = \frac{\partial^2 A}{\partial x^2} + \frac{\partial^2 A}{\partial y^2} \]

• Apply forward and backward differences

\[ \left( \frac{\partial^2 A}{\partial x^2} \right)_{i,j} \approx \frac{A_{i-1,j} - 2A_{i,j} + A_{i+1,j}}{\Delta x^2} \]

\[ \left( \frac{\partial^2 A}{\partial y^2} \right)_{i,j} \approx \frac{A_{i,j-1} - 2A_{i,j} + A_{i,j+1}}{\Delta x^2} \]

Five point stencil for the Laplacian
Staggered (marker-and-cell) grids

- Store pressure at cell centers, but velocity at cell faces

\[(\nabla \cdot \mathbf{u})_{i,j} \approx \frac{u_{i+1,j} - u_{i,j}}{\Delta x} + \frac{v_{i,j+1} - v_{i,j}}{\Delta x}\]

- Finite differences line up

- \(\nabla \cdot \mathbf{u}\) and \(p\) at cell centers

- Components of \(\nabla p\) and \(\mathbf{u}\) at cell faces
Boundary conditions

• At solid obstacles, \( \mathbf{u} \cdot \hat{n} = 0 \)

• Fluid cannot flow into or out of obstacles

• At free surface, \( p = 0 \)

• Air applies negligible force on water
Pressure solve

• Build a linear system with one equation per cell

\[(\nabla^2 p)_{i,j} = (\nabla \cdot u)_{i,j}\]

• Whole system: \(Ax = b\), where

  • \(x\) is a vector containing all the \(p_{i,j}\)
  • \(b\) is a vector containing \((\nabla \cdot u)_{i,j}\)

  Rows of \(A\) contain stencil for \(\nabla^2\)

• Be careful about boundaries!
Pressure solve

• Solve $Ax = b$ to get pressure values $p$
  
  • $A$ is large, sparse, symmetric, positive (semi)definite
  
  • Use e.g. preconditioned conjugate gradient method
  
  • Update velocities: $u^{\text{new}} = u - \nabla p$

Refer to Bridson & Müller-Fischer 2007 for full details
Viscosity

\[ \frac{\partial \mathbf{u}}{\partial t} = \frac{\mu}{\rho} \nabla^2 \mathbf{u} \]

- Often ignored: advection causes enough diffusion already

- For high-viscosity fluids, just use implicit integration

\[ \mathbf{u}^{\text{new}} = \mathbf{u}^{\text{old}} + \frac{\mu}{\rho} \nabla^2 \mathbf{u}^{\text{new}} \Delta t \]
Viscosity

Carlson, Mucha, Van Horn, and Turk, 2002
Smoke simulation

Lentine, Zheng, and Fedkiw, 2010
Surface tracking

• How to represent the surface of a liquid?

Option 1: Level set method

• Store the **signed distance** to surface $\phi(x)$ on grid cells
  
  • Advect forward at each time step
  
  • Surface is the level set (isosurface) at $\phi = 0$
Surface tracking

• How to represent the surface of a liquid?

Option 2: Particles (easier)

• Keep lots of particles in the fluid, passively advected with the flow

• Reconstruct surface as in SPH

More options: level set + particles, volume-of-fluid, meshes, …
Liquid simulation

English, Qiu, Yu, and Fedkiw, 2013
Real-time liquid simulation

Chentanez and Müller, 2011
References

- Bridson and Müller-Fischer, “Fluid Simulation for Computer Animation”, SIGGRAPH 2007 course notes

- Stam, “Stable Fluids”, 1999

- Enright, Marschner, and Fedkiw, “Animation and Rendering of Complex Water Surfaces”, 2002

- Zhu and Bridson, “Animating Sand as a Fluid”, 2005 (for particle-based surface tracking, and more)