CSCI 5980/8980: Special Topics in Computer Science

Physics-Based Animation

13 — Fluid simulation with grids

October 20, 2015
Today

- Presentation schedule
- Fluid simulation with grids
- Course feedback survey (again)
<table>
<thead>
<tr>
<th>Oct 20</th>
<th>Fluids on grids</th>
<th>Thu 22</th>
<th>Liquid surfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oct 27</td>
<td>Elasticity</td>
<td>Thu 29</td>
<td>Finite element method</td>
</tr>
<tr>
<td>Nov 3</td>
<td>Sound (Bryan)</td>
<td>Thu 5</td>
<td>Cloth and hair</td>
</tr>
<tr>
<td>Nov 10</td>
<td>Cloth (Alex &amp; Morgan)</td>
<td>Thu 12</td>
<td>Fluids (Ran &amp; Jingying)</td>
</tr>
<tr>
<td>Nov 17</td>
<td>Fire (Devin &amp; Lei)</td>
<td>Thu 19</td>
<td>TBD</td>
</tr>
<tr>
<td>Nov 24</td>
<td>Solids (George &amp; Jie)</td>
<td>Thu 26</td>
<td>Thanksgiving</td>
</tr>
<tr>
<td>Dec 1</td>
<td>Real-time stuff (Zahra)</td>
<td>Thu 3</td>
<td>Real-time stuff (Dan &amp; Matt)</td>
</tr>
<tr>
<td>Dec 8</td>
<td>Wrap-up day</td>
<td>Thu 10</td>
<td>Project presentations</td>
</tr>
<tr>
<td>Dec 15</td>
<td>Project presentations</td>
<td>Thu 10</td>
<td>Project presentations</td>
</tr>
</tbody>
</table>
Course feedback survey
Representing continua
Particles (SPH)

- Values on moving particles
- Approximate continuous field by weighted averaging
  \[ A(\mathbf{x}) = \sum A_i \frac{m_i}{\rho_i} W(\mathbf{x} - \mathbf{x}_i, h) \]
- Derivatives by differentiating weighting kernel
Grids

- Values on nodes of rectilinear grid
- Easy to interpolate using only 4 (in 2D) or 8 (in 3D) nearest values
Grids

class Grid {
    int m, n, o;
    Vec3d origin;
    double dx;
    Type *values;
    Type get(int i, int j, int k);
    ...
};
**Grids (finite differences)**

\[
\left( \frac{\partial c}{\partial x} \right)_{11} \approx \frac{c_{21} - c_{11}}{\Delta x}
\]

\[
\left( \frac{\partial c}{\partial x} \right)_{11} \approx \frac{c_{11} - c_{01}}{\Delta x}
\]

\[
\left( \frac{\partial c}{\partial x} \right)_{11} \approx \frac{c_{21} - c_{01}}{2\Delta x}
\]

\[
\begin{array}{ccc}
0 & -1 & 1 \\
-1 & 1 & 0 \\
-\frac{1}{2} & 0 & \frac{1}{2}
\end{array}
\]

\[
\begin{bmatrix}
c_{02} & c_{12} & c_{22} & \cdots \\
c_{01} & c_{11} & c_{21} & \cdots \\
c_{00} & c_{10} & c_{20} & \cdots \\
\vdots & \vdots & \vdots & \ddots
\end{bmatrix}
\]
\[ \nabla^2 c \approx \frac{1}{h^2} \]

**Grids (finite differences)**

\[
\begin{array}{ccc}
    & 1 & \quad \quad \quad \quad \\
1 & -4 & 1 \\
1 & \quad \quad \quad \quad & \\
\end{array}
\]

\[
\begin{array}{cccc}
    c_{02} & c_{12} & c_{22} & \cdots \\
    c_{01} & c_{11} & c_{21} & \cdots \\
    c_{00} & c_{10} & c_{20} & \cdots \\
    \vdots & \vdots & \vdots & \ddots \\
\end{array}
\]
Meshes (FEM)

- Values on nodes of (not necessarily regular) mesh
- Evaluate derivatives on “elements” (triangles/tetrahedra/etc.)
Fluids on grids
Grids vs. particles

Particles
- Mass, velocity, pressure, etc. stored on particles
- Particles move with fluid

Grids
- Velocity, pressure, etc. stored on grid cells
- Grid doesn’t move
Advection

Grid doesn’t move, fluid flows through grid

“Lagrangian” \[ \frac{Dc}{Dt} = 0 \] but \[ \frac{\partial c}{\partial t} \neq 0 \] “Eulerian”
Advection

\[
\frac{Dc}{Dt} = \frac{\partial c}{\partial t} + \mathbf{u} \cdot \nabla c
\]

Lagrangian derivative

Change due to movement of fluid ("advection")

Eulerian derivative

Proof: differentiate \( c(x(t), t) \) with respect to \( t \)
The fluid equations

\[ \rho \frac{D\mathbf{u}}{Dt} = \mathbf{f}_{\text{ext}} - \nabla p + \mu \nabla^2 \mathbf{u} \]

\[ \frac{D\mathbf{u}}{Dt} = \frac{\mathbf{f}_{\text{ext}}}{\rho} - \frac{1}{\rho} \nabla p + \nu \nabla^2 \mathbf{u} \]

\[ \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = \frac{\mathbf{f}_{\text{ext}}}{\rho} - \frac{1}{\rho} \nabla p + \nu \nabla^2 \mathbf{u} \]
Operator splitting

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

- Lots of different terms, hard to integrate safely in one step
- Deal with one term at a time, ignoring all the others
  - (e.g. IMEX)
Operator splitting

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

Integrate for time $\Delta t$

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

Integrate for time $\Delta t$

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

Integrate for time $\Delta t$

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

Integrate for time $\Delta t$

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

**Goal:** Integrate $\partial c/\partial t + u \cdot \nabla c = 0$ for time $\Delta t$

- **Interpretation:** $c$ moves with speed $u$ for time $\Delta t$

- **Solution:** To get the value of $c^{n+1}$ at any point, figure out where it came from and take the value of $c^n$ from there
Advection

Trace *backwards* through $u$ and look up values

“Semi-Lagrangian advection”
Advection

**Input:** initial grid $c^n$, velocity field $u^n$

**Output:** final grid $c^{n+1}$

- For each grid cell $x_i$
  - Backtrace position, e.g. $x^{\text{back}} = x_i - u_i \Delta t$
  - Set output $c_i^{n+1} = \text{interpolate} \; c^n \; \text{at} \; x^{\text{back}}$

To advect velocities, just use $u^n$ as the initial grid too.
Pressure
Pressure

In SPH, density $\rho$ was used for two things:

1. Normalizing for uneven particle distribution
2. Computing pressure forces

With grids, we don’t need to track $\rho$. But what about pressure? Treat it as a constraint force:

$$\rho = \text{const} \quad \Rightarrow \quad \frac{d\rho}{dt} = 0$$
Incompressibility

Net flow into/out of region

$$= \int\int \mathbf{u} \cdot \mathbf{n} \, dA$$
$$= \iiint \nabla \cdot \mathbf{u} \, dV$$

[Divergence theorem]

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

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

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

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

**Constraint:** \( u^{\text{new}} \) should be divergence-free, find the “constraint response” \( p \) that makes it so

- (analogous to computing \( \lambda \) so that \( J\mathbf{v} = 0 \))

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

Ignore the \( \rho/\Delta t \)

(just a rescaling)
Staggered 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

- Solid obstacles: \( \mathbf{u} \cdot \mathbf{n} = 0 \)
  - Fluid cannot flow into or out of obstacles
- Free surface: \( p = 0 \)
  - Air applies negligible force on water

\[ p = 0 \]
Pressure solve

\[ \nabla^2 p = \nabla \cdot u \]

Build a big linear system \( A p = b \) and solve it

• Rows of \( A \) contain stencil for \( \nabla^2 \)
  (except at boundaries, be careful!)

• \( b \) contains values of \( \nabla \cdot u \)

Then update \( u \leftarrow u - \nabla p \)
The other bits

External forces

• Gravity, buoyancy, user interaction, ...

• Explicit integration is fine

Viscosity

• Often ignored because numerical damping is enough

• For really thick fluids, use implicit integration

\[ \mathbf{f} = -\beta (T - T_0) \mathbf{g} \]
Operator splitting

\[ \begin{align*}
  &\text{Integrate } \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} = 0 \text{ for time } \Delta t \\
  \Rightarrow &\text{Integrate } \frac{\partial \mathbf{u}}{\partial t} = \frac{1}{\rho} \mathbf{f}_{\text{ext}} \text{ for time } \Delta t \\
  \Rightarrow &\text{Integrate } \frac{\partial \mathbf{u}}{\partial t} = \frac{\mu}{\rho} \nabla^2 \mathbf{u} \text{ for time } \Delta t \\
  \Rightarrow &\text{Integrate } \frac{\partial \mathbf{u}}{\partial t} = -\frac{1}{\rho} \nabla p \text{ for time } \Delta t \\
\end{align*} \]

\[ \mathbf{u}^{n+1} \]
Further reading

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

Course feedback survey
Next class

Even more fluids

• Hybrid (particles + grid) techniques for advection

• Liquid surface reconstruction