Navier-Stokes & Flow Simulation

Last Time?
- Implicit Surfaces
- Marching Cubes/Tetras
- Collision Detection & Response
- Conservative Bounding Regions

Today
- Flow Simulations in Computer Graphics
  - water, smoke, viscous fluids
- Navier-Stokes Equations
  - incompressibility, conservation of mass
  - conservation of momentum & energy
- Fluid Representations
- Basic Algorithm
- Data Representation

Flow Simulations in Graphics
- Random velocity fields
  - with averaging to get simple background motion
- Shallow water equations
  - height field only, can’t represent crashing waves, etc.
- Full Navier-Stokes

  *note: typically we ignore surface tension and focus on macroscopic behavior*

Heightfield Wave Simulation

Flow in a Voxel Grid
- conservation of mass:
  \[
  \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0
  \]
Navier-Stokes Equations

- conservation of momentum:
  \[
  \frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} + \frac{\partial u}{\partial y} + \frac{\partial u}{\partial z} = \nu \left( \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} \right) + \frac{\partial p}{\partial x} + g_x + \nu \left( \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} \right)
  \]

  \[
  \frac{\partial v}{\partial t} + \frac{\partial v}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial v}{\partial z} = \nu \left( \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial z^2} \right) + \frac{\partial p}{\partial y} + g_y + \nu \left( \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\partial^2 v}{\partial z^2} \right)
  \]

  \[
  \frac{\partial w}{\partial t} + \frac{\partial w}{\partial x} + \frac{\partial w}{\partial y} + \frac{\partial w}{\partial z} = \nu \left( \frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} + \frac{\partial^2 w}{\partial z^2} \right) + \frac{\partial p}{\partial z} + g_z + \nu \left( \frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} + \frac{\partial^2 w}{\partial z^2} \right)
  \]

- gravity & other external forces
- pressure
- viscosity
- acceleration
- Convection: internal movement in a fluid (e.g., caused by variation in density due to a transfer of heat)

Modeling the Water Surface

- Volume-of-fluid tracking
  - a scalar saying how “full” each cell is
- Particle In Cell (PIC)
  - the particles have mass
  - impractical, until recently: Smoothed-Particle Hydrodynamics (SPH)
- Marker and Cell (MAC)
  - the particles don’t effect computation, just identify which cells the surface passes through

Comparing Representations

- How do we render the resulting surface?
- Are we guaranteed not to lose mass/volume? (is the simulation incompressible?)
- How is each affected by the grid resolution and timestep?
- Can we guarantee stability?

Demos

- Nice Marker and Cell (MAC)

videos at:

http://panoramix.ift.uni.wroc.pl/~maq/eng/cfdthesis.php

http://www.uwaterloo.ca/~fslien/free_surface/free_surface.htm

Reading for Today

- “Realistic Animation of Liquids”, Foster & Metaxas, 1996
Today

- Flow Simulations in Computer Graphics
- Navier-Stokes Equations
- Fluid Representations
- Basic Algorithm
- Data Representation

Each Grid Cell Stores:

- Velocity at the cell faces (offset grid)
- Pressure
- List of particles

Initialization

- Choose a voxel resolution
- Choose a particle density
- Create grid & place the particles
- Initialize pressure & velocity of each cell
- Set the viscosity & gravity
- Choose a timestep & go!

At each Timestep:

- Identify which cells are Empty, Full, or on the Surface
- Compute new velocities
- Adjust the velocities to maintain an incompressible flow
- Move the particles
  - Interpolate the velocities at the faces
- Render the geometry and repeat!

Empty, Surface & Full Cells

At each Timestep:

- Identify which cells are Empty, Full, or on the Surface
- Compute new velocities
- Adjust the velocities to maintain an incompressible flow
- Move the particles
  - Interpolate the velocities at the faces
- Render the geometry and repeat!
Compute New Velocities

\[ u_{i+1/2,j,k} = u_{i+1,1,1} + \frac{1}{6} \left( \frac{1}{\delta x} \left[ (u_{i+1,1,1} - u_{i+1,1,0})^2 - (u_{i+1,1,0})^2 \right] 
- (1/\delta y)(w_{i+1/2,j+1/2,k} - w_{i+1/2,j-1/2,k}) 
+ (1/\delta z)(u_{i+1,1,j+1/2,k} - u_{i+1,1,j-1/2,k}) + p + (1/\delta z)(p_{i+1,1,1} - p_{i+1,1,0}) + (\nu/\delta x)^2 u_{i+1,1,1} 
- 2u_{i+1,1,1} + u_{i+1,1,1} + (\nu/\delta y)^2 u_{i+1,1,1} + u_{i+1,1,1} + (\nu/\delta z)^2 u_{i+1,1,1} + u_{i+1,1,1} \right) \]

Note: some of these values are the average velocity within the cell rather than the velocity at a cell face.

At each Timestep:

- Identify which cells are Empty, Full, or on the Surface
- Compute new velocities
- Adjust the velocities to maintain an incompressible flow
- Move the particles
  - Interpolate the velocities at the faces
- Render the geometry and repeat!

Adjusting the Velocities

- Calculate the divergence of the cell (the extra in/out flow)
- The divergence is used to update the pressure within the cell
- Adjust each face velocity uniformly to bring the divergence to zero
- Iterate across the entire grid until divergence is < \( \varepsilon \)

Handing Free Surface with MAC

- Divergence in surface cells:
  - Is divided equally amongst neighboring empty cells
  - Or other similar strategies:
  - Zero out the divergence & pressure in empty cells

Velocity Interpolation

Image from Foster & Mataxas, 1996
Stable Fluids

Efficient Smoke Simulation
“Visual Simulation of Smoke”
Fedkiw, Stam & Jensen
SIGGRAPH 2001

Solid/Liquid: Time-varying viscosity
“Melting and Flowing”
Carlson, Mucha, Van Horn III & Turk
Symposium on Computer Animation 2002

Ron Fedkiw
http://physbam.stanford.edu/~fedkiw/

Reading for Friday 2/26:
• “Real-Time Hand-Tracking with a Color Glove”