Fluids for Computer Graphics
Bedřich Beneš, Ph.D.
Purdue University
Department of Computer Graphics Technology

Overview
• Some terms…
• Incompressible Navier-Stokes Equations
• Boundary conditions
• Lagrange vs. Euler
• Eulerian approaches
• Lagrangian approaches
• Shallow water
• Conclusions

Some terms
• **Advect:** evolve some quantity forward in time using a velocity field. For example particles, mass, etc.

• **Convec**t: transfer of heat by circulation of movement of fluid.

Some terms
• **Lagrangian:** methods that move fluid mass (for example by advecting particles)

• **Eulerian:** fluid quantities are defined on a grid that is fixed (the quantities can vary over time)
Equations

Fluids are governed by the incompressible Navier-Stokes Equations

\[
\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} + \frac{1}{\rho} \nabla p = -\mathbf{g} + \mathbf{v} \cdot \nabla \mathbf{u} \tag{1}
\]
\[
\nabla \cdot \mathbf{u} = 0 \tag{2}
\]

Equations

• \( \mathbf{g} = (0, -9.81, 0) \) accel. due gravity \([m/s^2]\)
  (assuming: 
  \( z \) – points to you, \( y \) – is up, \( x \) – is right)

• \( \nu \) (upsilon) – *kinematics viscosity*
  how difficult it is to stir

Equations

• \( \mathbf{u} = (u, v, w) \) *velocity* of the fluid \([m/s]\)

• \( \rho \) (rho) fluid *density* \([kg/m^3]\)
  water \( \sim 1000 \)
  air \( \sim 1.3 \)

• \( p = \frac{\mathbf{F}}{A} \) *pressure* \([Pa]\)
  force per unit area that the fluid exerts

Equations

The momentum equation \((\mathbf{F} = m\mathbf{a})\)

How the fluid accelerates due to the forces acting on it

\[
\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} + \frac{1}{\rho} \nabla p = -\mathbf{g} + \mathbf{v} \cdot \nabla \mathbf{u} \]
Equations

The momentum equation \( \vec{F} = m \vec{a} \)

- **Balance of momentum.** Internal + external forces = change in momentum.

- **Conservation of energy.** Kinetic + internal energy = const.

Conservation of mass
Adveacting mass through the velocity field cannot change total mass.

\[
\nabla \cdot \vec{u} = 0
\]

\[
\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0
\]
Boundary Conditions

• Three types
  • Solid walls
  • Free surface
  • Other fluids

Solid boundaries
• Normal component of fluid velocity = 0
  \( \mathbf{u} \cdot \hat{n} = 0 \)

Ideal fluids (slip boundary)
• The tangential component is unchanged.

Viscous fluids (no-slip boundary)
• The tangential component is set to zero.

Free surface
• Interface between the fluid and “nothing” (air)
  • Volume-of-fluid tracking (for Eulerian)
  • Mesh tracking (tracks evolving mesh)
  • Particle fluids (Lagrangian)
  • Level sets: advects a signed distance function

Solutions
• Lagrangian:
  • The world is a particle system
  • Each particle has an ID and properties (position, velocity, acceleration, etc.)
• Eulerian:
  • The point in space is fixes
  • Measure stuff as it flows past
• Analogy - temperature:
  • Lagrangian: a balloon floats with the wind
  • Eulerian: on the ground wind blows past
Incompressibility

- Real fluids are compressible that is why we hear under water
- Not important for animation and expensive to calculate

Eulerian Approach

- Each wall stores the velocity vector
- Space discretization
  - Uses fixed 3D regular grid (voxels) (two of them)
  - Each cell has pressure $p$ in the middle
  - Each cell has its state:
    - FULL of fluid
    - SOLID material
    - EMPTY air
    - SURFACE boundary cell
- Discretization of the NS momentum equation gives new velocity $\vec{u}$:
Eulerian Approach

- Time step and the convergence condition
  \[ 1 > \max[u \frac{\partial t}{\partial x}, v \frac{\partial t}{\partial y}, w \frac{\partial t}{\partial z}] \]

- \( \partial x, \partial y, \partial z \) are given, so we can only decrease the time step.
- This causes the simulation to slow down over time.

\[
\mathbf{D}_{i,j,k} = -(1/\delta x)(u_{i+1/2,j,k} - u_{i-1/2,j,k})
+ (1/\delta y)(v_{i,j+1/2,k} - v_{i,j-1/2,k})
+ (1/\delta z)(w_{i,j,k+1/2} - w_{i,j,k-1/2}).
\]

- Is a finite difference approx. of \( \nabla \cdot \mathbf{u} = 0 \)

\[
\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0
\]

Eulerian Approach

- Change of the mass causes change of the pressure

\[
\partial p = \beta D
\]

\[
\beta = \beta_0/2\delta t \left( \frac{1}{\delta x^2} + \frac{1}{\delta y^2} + \frac{1}{\delta y^2} \right)
\]

\( \beta_0 \in [1,2] \)
Eulerian Approach

• The face vortices are then updated
  \[ w_{i+1/2,j,k} = w_{i+1/2,j,k} + (\frac{\partial t}{\partial x}) \delta p, \]
  \[ w_{i-1/2,j,k} = w_{i-1/2,j,k} - (\frac{\partial t}{\partial x}) \delta p, \]
  \[ v_{i,j+1/2,k} = v_{i,j+1/2,k} + (\frac{\partial t}{\partial y}) \delta p, \]
  \[ v_{i,j-1/2,k} = v_{i,j-1/2,k} - (\frac{\partial t}{\partial y}) \delta p, \]
  \[ w_{i,j,k+1/2} = w_{i,j,k+1/2} + (\frac{\partial t}{\partial z}) \delta p, \]
  \[ w_{i,j,k-1/2} = w_{i,j,k-1/2} - (\frac{\partial t}{\partial z}) \delta p. \]

Putting this all together:

1. Scene definition (material, sources, sinks)
2. Set initial pressure and velocity
3. In a loop
   I. Compute \( \tilde{u}, \tilde{v}, \tilde{w} \) for all Full cells.
   II. Pressure iteration for all Full cells.

Eulerian Approach

• Cell pressure is updated
  \[ \tilde{p} = p + \delta p \]

• Where is the free level?
  • Use marching cubes
  • Use Marker Particles – particles that are advected with the velocity Markers and Cells (MAC)
Stability

- Slowing down because of stability
- Some iterations for divergence needed
- Demo

Follow up works


Follow up works


Follow up works

- Erosion simulation
Follow up works

• Erosion

Lagrangian approaches

• *Lagrangian*: methods that move fluid mass (for example by advecting particles)

Smoothed Particle Hydrodynamics (SPH)


SPH

- Fluid is divided into discrete particles
- Particles advect mass
- The values of different properties in the space where no particles are present are calculated by using smoothing functions

\[ \phi = \sum_j m_j \frac{\phi_j}{\rho_j} W(x - x_j, h) \]

\( \phi \) – the physical value at location \( x \)
\( m_j \) - mass of the \( j \)-th particle
\( \phi_j \) – the physical value of the \( j \)-th particle
\( x_j \) – location of the \( j \)-th particle
\( W \) – the smoothing kernel

\( h \) – radius of influence

Boundary

- Can be represented as a triangular mesh
- or as boundary particles (slip, no-slip)
Varying viscosity

http://www.youtube.com/watch?feature=player_embedded&v=6bdIHFTfTdU

http://www.youtube.com/watch?feature=player_embedded&v=Kt4oKhXngBQ

http://www.youtube.com/watch?feature=player_detailpage&v=UYIPg8TEMmU
**SPH**

- **Pros**
  - No problems with losing material
  - Implicitly saves data
    (particles are where the fluid is)
  - Splashes are easy…
- **Cons**
  - Large number of particles is necessary
  - Not as good for GPU as Eulerian
    (but still pretty good)

**Shallow Water Simulation**

- The fluid is a simple 2-D grid with layers
- Neighbor cells are connected by pipes
- Cannot simulate splashes and overhangs
- Good enough for near-still water with boats

**Algorithm:**
1) Get acceleration from unequal levels
2) Calculate flow between cells
3) Change levels of water
4) Go to 1)

**Acceleration**

\[ a = \frac{g \Delta h}{l} \]

Distance between cells
Shallow Water Simulation

- New flow

\[ f^{t+\Delta t} = f^t + \Delta t C_a \]

New flow, Old flow, Cross-sectional area of the pipe

- Update levels

Shallow Water Simulation

© Bedrich Beneš

Shallow Water Simulation

© Bedrich Beneš

Shallow Water Simulation

© Bedrich Beneš

Shallow Water Simulation

© Bedrich Beneš
Conclusions

• Fluid simulation is a complex topic
• Fluid simulation for CG uses simplifications that are aimed at
  • Speed
  • Visual quality
• Still an open problem
• lot of work to do…

Reading

• Robert Bridson
  Fluid Simulation for
  Computer Graphics

• Siggraph proceedings