Fluid Simulation

From an Eulerian Viewpoint

Andy Li

The University of British Columbia

Canadian Undergraduate Mathematics Conference, 2024

Overview

Vector Calculus Background

- Gradient : $\nabla f(x, y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right)$
- Divergence: $\nabla \cdot \vec{u} = \nabla \cdot (u, v) = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}$
- Curl: $\nabla \times \vec{u} = \nabla \times (u, v) = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}$
- Laplacian : $\nabla \cdot \nabla f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$

The Navier-Stokes Equations

Lets Introduce 2 PDEs that govern the motion of in-compressible Newtonian fluids.

Navier-Stokes Equations

The Navier-Stokes equations describe the motion of viscous fluid substances. They are a set of nonlinear partial differential equations given by:

$$\nabla \cdot \mathbf{u} = 0 \tag{1}$$

$$\rho \frac{d\mathbf{u}}{dt} = -\nabla p + \mu \nabla^2 \mathbf{u} + \rho \mathbf{F}, \tag{2}$$

The Imcompressibility Condition

A fluid is incompressible if its density remains constant over time.

Momentum Equation

The Incompressibility Condition from the Navier-Stokes equations is given by:

$$\nabla \cdot \mathbf{u} = 0,$$

where $\mathbf{u}$ is the velocity field.

What does this mean?

The Momentum Equation

Recall Newton's 2nd where $ma = \sum F$. A fluid's pressure, viscosity, and external forces dictates its movement.

Momentum Equation

The momentum equation of the Navier-Stokes equations is given by:

$$\rho \frac{d\mathbf{u}}{dt} = -\nabla p + \mu \nabla^2 \mathbf{u} + \rho \mathbf{F},$$

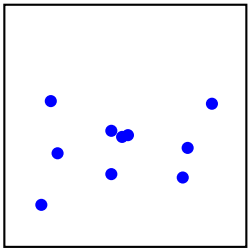
where $\mathbf{u}$ is the velocity field.

Lets break down each term qualitatively.

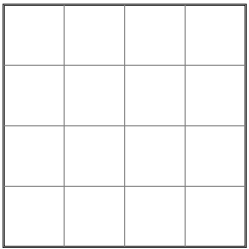
Lagrangian vs Eulerian Viewpoints

When we think about a continuum (like a fluid or a deformable solid) moving, there are two approaches to tracking this motion. Illustrated below are the two.

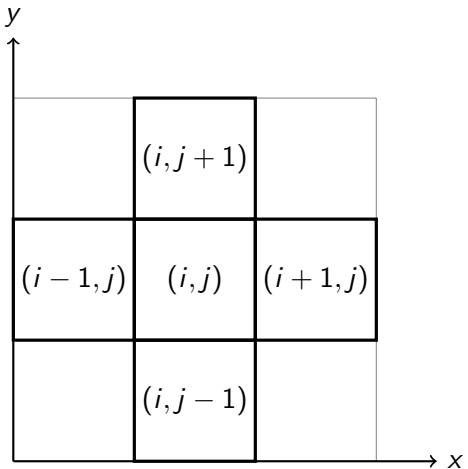
Lagrangian Viewpoint



Eulerian Viewpoint



Our Grid



- For each grid cell (i, j) we need to keep track of density and velocity.
- Both properties exhibit diffusion and advection.
- Our grid wraps around itself, so we have periodic boundary conditions.

Density Solver: An Unstable Approach to Diffusion

- Let $d(x, y)$ be the density of the square at (x, y).
- Let $\mu(x, y) = \frac{d(x+1,y)+d(x-1,y)+d(x,y-1)+d(x,y+1)}{4}$, the average densities of the 4 adjacent squares.
- Let $\square_c$ represent the value at our current time, and $\square_n$ to be the value after 1 time step.
- Let $\alpha = \Delta tnmD$, a constant from our time step, grid size, and diffusion coefficient.

Density Solver: An Unstable Approach to Diffusion

- Let $d(x, y)$ be the density of the square at (x, y).
- Let $\mu(x, y) = \frac{d(x+1,y)+d(x-1,y)+d(x,y-1)+d(x,y+1)}{4}$, the average densities of the 4 adjacent squares.
- Let $\square_c$ represent the value at our current time, and $\square_n$ to be the value after 1 time step.
- Let $\alpha = \Delta tnmD$, a constant from our time step, grid size, and diffusion coefficient.

A simple algorithm looks like this:

Unstable Diffusion Eq.

$$d_n(x, y) = d_c(x, y) + \alpha(\mu_c(x, y) - d_c(x, y))$$

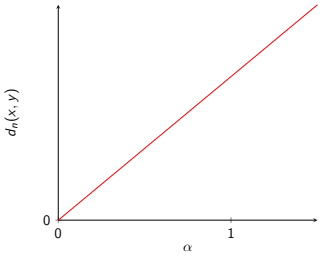
Density Solver: An Unstable Approach to Diffusion

- Let $d(x, y)$ be the density of the square at (x, y).
- Let $\mu(x, y) = \frac{d(x+1,y)+d(x-1,y)+d(x,y-1)+d(x,y+1)}{4}$, the average densities of the 4 adjacent squares.
- Let $\square_c$ represent the value at our current time, and $\square_n$ to be the value after 1 time step.
- Let $\alpha = \Delta t n m D$, a constant from our time step, grid size, and diffusion coefficient.

A simple algorithm looks like this:

Unstable Diffusion Eq.

$$d_n(x, y) = d_c(x, y) + \alpha(\mu_c(x, y) - d_c(x, y))$$



Density Solver: A Stable Approach to Diffusion

Lets take different approach that results in a stable interpolation. We find the densities which when diffused backward in time yield the densities we started with.

$$d_c = d_n - \alpha(\mu_n - d_n),$$
$$d_c = d_n(\alpha + 1) - \alpha\mu_n.$$

Density Solver: A Stable Approach to Diffusion

Lets take different approach that results in a stable interpolation. We find the densities which when diffused backward in time yield the densities we started with.

$$d_c = d_n - \alpha(\mu_n - d_n),$$
$$d_c = d_n(\alpha + 1) - \alpha\mu_n.$$

Rearrange this to get:

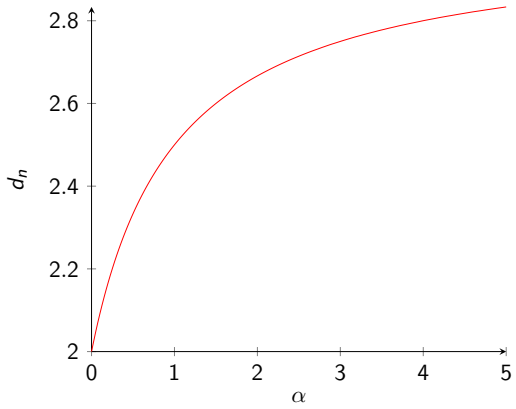
Stable Diffusion Eq.

$$d_n = \frac{d_c + \alpha\mu_n}{\alpha + 1}$$

Graph of Stable Diffusion Equation

Let $d_c = 2$, $\mu_c = 3$. Notice that:

$$\lim_{\alpha \to \infty} Diff(d_c, \mu_c, \alpha) = \mu_c$$



Density Solver: Iterative Methods

Recall:

Stable Diffusion Eq.

$$d_n = \frac{d_c + \alpha \mu_n}{\alpha + 1},$$

Where $\mu_n = \frac{d_n(x+1,y)+d_n(x-1,y)+d_n(x,y-1)+d_n(x,y+1)}{4}$.

Density Solver: Iterative Methods

Recall:

Stable Diffusion Eq.

$$d_n = \frac{d_c + \alpha \mu_n}{\alpha + 1},$$

Where $\mu_n = \frac{d_n(x+1,y)+d_n(x-1,y)+d_n(x,y-1)+d_n(x,y+1)}{4}$.

Options to solve this linear system include:

- Build a matrix, then use a matrix inversion routine.
- simpler iterative technique like Jacobi or Gauss-Seidel Method.

Density Solver: Advection pt1.

How can we make the density follow a given velocity field?

- Use an iterative technique.
- Use a semi-Lagrangian viewpoint.

Density Solver: Advection pt1.

How can we make the density follow a given velocity field?

- Use an iterative technique.
- Use a semi-Lagrangian viewpoint.

The sketch of our algorithm:

1. Treat each grid cell center as a particle.
2. Trace backwards over a time step to find the particle p that end up at our current cell c.
3. Let particle p be represented by its 4 closest cells.
4. Interpolate the density of p with these 4 cells and assign this to be our new density for c.

Density Solver: Advection pt2.

We will use the following functions:

- floor(x)=$\lfloor x \rfloor$
- fract(x) = x - floor(x)
- $L(a, b, s) = a + s(b - a)$

Density Solver: Advection pt2.

We will use the following functions:

- floor(x)=$\lfloor x \rfloor$
- fract(x) = x - floor(x)
- $L(a, b, s) = a + s(b - a)$

Example

Consider grid cell $c = (x, y)$. Let the particle that ends up at c after Δt be $p = (x', y') = (x, y) - v(x, y)\Delta t$.

Lets find our new density for c,

Satisfying the Incompressibility Condition

Helmholtz Decomposition

Any sufficiently smooth, rapidly decaying vector field $\mathbf{F}$ defined on all of $\mathbb{R}^3$ can be decomposed into a curl-free (irrotational) component and a divergence-free (solenoidal) component. In our case:

$$\mathbf{u} = -\nabla\varphi + \nabla \times \mathbf{A}$$

where:

- φ is a scalar potential function.
- $\mathbf{A}$ is a vector potential function.

Satisfying the Incompressibility Condition Cont'd.

We can find the scalar potential function φ:

$$\nabla \cdot v(x,y) = \frac{v_x(x+1,y) - v_x(x-1,y)}{2} + \frac{v_y(x,y+1) - v_y(x,y-1)}{2},$$
$$\varphi(x,y) = \frac{[\varphi(x+1,y) + \varphi(x-1,y) + \varphi(x,y+1) + \varphi(x,y-1)] - \nabla \cdot v(x,y)}{4}.$$
$$\nabla\varphi = \left(\frac{\varphi(x+1,y) - \varphi(x-1,y)}{2}, \frac{\varphi(x,y+1) - \varphi(x,y-1)}{2}\right).$$

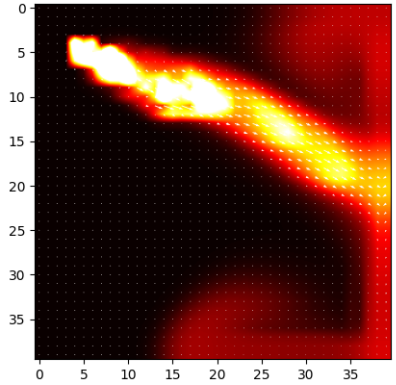
The curl of a gradient field $= 0$.

Results.

This can be implemented in around 100 lines of readable C code.

Results.

This can be implemented in around 100 lines of readable C code.



References

Bridson, R., & Müller-Fischer, M. (2007).
Fluid Simulation: SIGGRAPH 2007 Course Notes.
University of British Columbia, AGEIA Inc., Vancouver, Canada and Zurich, Switzerland, August 10.

Stam, J., (2003).
Real-Time Fluid Dynamics for Games.
Alias — wavefront, Toronto, Canada, May 25.

Questions?