Smoke Simulation

Josh Lawrence

May 7, 2018

1 Introduction

The github page http://www.github.com/loshjawrence/FluidSim was done as a final project for CIS-563 Physically Based Animation at University of Pennsylvania. It's implementation is based on *Visual Simulation of Smoke* [Jen01], *Stable Fluids* [Sta99], and the 2007 SIGGRAPH course titled *Fluid Simulation* [MF07]. Other resources include Kahn Academy animations of the vector calculus concepts gradient, divergence, curl, and Laplacian. Some of the explanations that follow are pulled directly from these resources with some elaboration. It is written in a note-to-self style in an attempt to provide myself with a quick refresher for future implementations and/or extensions.

2 Math and Jargon Overview

Fluid simulation for computer graphics are based on the Navier-Stokes equations (two equations describing conservation of mass and conservation of momentum). Some knowledge of vector calculus is required to manipulate these equations to suit the needs of computer graphics, say for example, to make it less accurate to speed up computation without sacrificing too much visual fidelity. Sections 2.1-2.3 describe the some of the main vector calculus concepts used in describing a fluid material mathematically.

2.1 Divergence

Divergence is a scalar function (returns single value for a point in space) that operates on a vector field (such as a set of velocity vectors, either defined explicitly and regularly spaced in a grid or defined as a parametric function given a time value and a point in space). A divergence value greater than 0 means that if there were particles in the region then after a timestep t they would leave that region, i.e. the total number of particles in that region is decreasing over time. Therefore a positive divergence means that the field is indeed diverging. A divergence value less than 0 means the region is converging and particles are accumulating in that region overtime. A divergence value of 0 means that it is divergence free and the number of particles in that region remains the same overtime (desirable for fluid simulation as we must maintain incompressibility, i.e. no divergence, to simplify the math).

2.2 Curl

Curl also operates on a vector field and returns a vector and tells you how much it is swirling at a point in the field given by the right hand rule and the general swirling motion of the vectors in the vicinity. If you imagine a little ball centered at the point in space you are interested in, the 3d curl is the spin on the ball due to the vectors in the region. Mathematically, it is the right hand rule vector spin along the thumb. The magnitude of curl is magnitude of rotation.

2.3 Laplacian

The Laplacian is sort of like the second derivative. It is can be intuitively defined as the divergence of the gradient of a function. The gradient which I have yet to mention is nothing more than the slope of steepest ascent, similar to a first derivative, and gives you a vector field as set of directions that you should "walk" to reach the "top of the hill" (if you visualize the slope as a height field). The Laplacian (or divergence of the gradient) will be positive in "valleys" and negative at "peaks"

(again if we visualize the slope as a height field) and close to zero in between. Valleys will be areas of high pressure, pushing fluid away, and peaks are will be areas of low pressure, a place for water to go. Another way to think of the Laplacian is that it measures how far a quantity is from the average around it.

2.4 Lagrangian vs Eulerian Views of a Material

The Lagrangian view of a material is a soup of particles each with a position and velocity. The Eulerian approach, it looks at fixed points in space (where the simulation is tracked, in a grid) and measures fluid quantities like density, velocity, temperature, density, etc. at each time step. For example, as a warm fluid moves past followed by a cold fluid, the temperature at the fixed point in space will go down - even though the temperature of any individual particle in the fluid is not changing. In addition the fluid variables can be changing in the fluid, contributing the other sort of change that might be measured at a fixed point, i.e. the temperature at a fixed point in space might decrease as the fluid everywhere cools off.

An analogy contrasting the two: you're performing a weather report. In the Lagrangian viewpoint you're in a balloon floating along with the wind, measuring the pressure and temperature and humidity, etc. of the the air that's flowing alongside you. In the Eulerian viewpoint, you're stuck on the ground, measuring the pressure and temp and humidity etc. of the air that's flowing past you. The reason Eulerian is useful is that it's easier to work with spatial derivatives like pressure gradient and viscosity with a fix grid (even regular spacing). It is also easier to interpolate/approximate those spatial derivatives on a fixed Eulerian grid than on a cloud of arbitrarily moving particles. The key to connecting the two viewpoints is the material derivative. material derivative is how fast a quantity q (some physical attribute pressure, velocity, etc.) is changing for a particle related to how fast that q is changing at a point in space.

An example in 1D: we can have an expression for T as T(x) = 10x for initial grid state (0 at origin and 100 at x=10) and a wind blowing to the right expressed as u=c. We'll assume the temp of the particles aren't changing, so from a Lagrangian viewpoint the material derivative is 0. however from the Eulerian view point the rate of change (see the equations on page 18) is -10c. Meaning fixed points in space are dropping in temp at a rate of -10c. So the Eulerian derivative can be anything depending on how fast and what direction the flow is moving.

2.5 Semi-Lagrangian

Semi-Lagrangian is a term used to describe a specific style of simulation scheme that uses both Eulerian and Lagrangian views of a material. The Eulerian view comes in the form of a fixed grid where the simulation takes place. Marker and Cell grids (or MAC grids for short) are setup as such: material properties like pressure, temperature, density are usually defined at the grid centers and velocity is componentized and defined on cell faces. MAC grids are constructed this way to make interpolation unbiased and more accurate. The Lagrangian view comes from the implicit advection step used to update the grid values at the time t+deltaT. Grid points are temporarily treated as particles (not actually created in the computer) and traced backwards in time to a place in the grid where they originated from. The grid-interpolated value at this origination point becomes the new value at the point in the grid where we time traveled from.

2.6 Boundary Conditions

For a solid wall boundary, velocity must have no component in the normal direction of the wall. This is similar to the gram-schmidt orthogonalization done in zero friction mass-spring solid body collision project. Or if the solid body is moving through the fluid, the fluid must match the velocity in the normal of the surface. For fluids with viscosity, you can simply just set the velocity to zero (not moving solid) or to the velocity of the solid (moving solid).

For air/fluid interface we can set the air fluid pressure to an arbitrary constant: 0. then a free surface (i.e. water meets air) is one where pressure = 0 and we don't control the velocity in any particular way. For smoke in open air, at the boundaries the fluid continues, allow fluid to enter and exit the simulation volume as it wants, so the boundary is a free surface p = 0.

For surface tension in a fluid simulation, at the free surface of water we add a pressure that acts to enforce surface tension. It is pressure = gamma*kappa, where gamma is surface tension coefficient, (for water gamma=0.073N/m) and kappa is the mean curvature, measure in 1/meters.

Chapter 6 of the SIGGRAPH Fluid Simulation course covers how to measure kappa, the mean curvature of a surface. If you want to model air bubbles in water (they'll collapse with the 0 pressure assumption) you need what's called a two phase flow (because there are two phases or types of fluid). This is not covered in any of the sources I am using.

2.7 Smoke Fluid Body Forces

In smoke simulation, the external forces that act on the fluid are derived from buoyancy, vorticity confinement, and pressure imbalance. We ignore forces due to viscosity for inviscid fluids like smoke. We will still get some look of viscosity from the numerical dissipation that occurs during interpolation/averaging operations that smooth out. This results in smoothing out fine features of the smoke. This dissipation is injected back into the fluid at locations where fine features are likely to be lost: vortexes of smoke. This is done using vorticity confinement. We compute the curl of the velocity field at each cell center and determine the gradient of this curl field. Using these two pieces of information we can calculate a force to keep the spin alive.

To model buoyancy we need two extra fluid variables: temperature and concentration (the thing we can actually see). In reality, temperature changes cause the air to change density however these changes are very small so we will use what is called the "boussinesq" approximation and assume constant density as before but replace the gravity force with a buoyancy force that is derived from these variations. To simplify even further, we assume the buoyancy force depends linearly on temperature and concentration. Temperature and concentration are advected around the and also slowly diffuse. Just as we dropped the viscosity since the numerical dissipation mimics it anyhow, we need not explicitly model heat and smoke diffusion; they'll happen in our numerical methods whether we want them or not. For boundaries, temperature of the solid object can be its own specific temperature or the ambient air temperature. Smoke concentration doesn't exist inside objects, except if its pouring out with a non-zero boundary velocity, in which case we can set a source value for smoke inside the object. Otherwise, we should extrapolate the smoke concentration into the object when asked for smoke inside an object, we interpolate from the closest values outside.

3 Algorithm Details

3.1 Outline

Basic fluid algorithm: Start with initial divergence-free velocity field u(t=0). Determine a good time step deltaT. Update the velocity field by performing the implicit advection scheme described previously and save the result to a scratch MAC grid data structure for working on intermediate calculations. Update this scratch grid with any forces that might be present on the fluid, in the case of smoke simulation these are buoyoancy and vorticity. Each force is added one by one as newval = oldval + deltaT*force. Next perform the projection step to solve for pressure values and update the non-srcatch copy of the MAC grid. Use this non-scratch copy to advect cell-centered fluid properties using the implicit advection scheme. Repeat from "Determine a good time step".

3.2 Time Step

Determining a good time step just means make sure it is not past the end of the frame time and it is suitable for not projecting any Lagrangian particles too far in the simulation volume. Fedkiw uses 5 cell width, so deltaT should be less than 5 cell widths divided by max velocity in the grid. A more robust estimate takes into account the velocities that might be induced due to acceleration from body forces like buoyancy and vorticity confinement. You can set maximum velocity to the max velocity magnitude in the grid plus the square root of 5 times the cell width * the max force in the grid.

3.3 Projection

Part of the computation speed up for fluid simulation comes from the incompressibility assumption for the fluid material. This simply means the simulation volume is divergence free. The projection step will subtract off the pressure gradient form the intermediate velocity field u so that the result

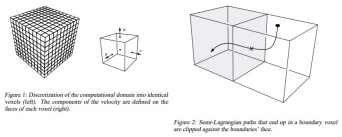

Figure 1: This is the MAC grid setup. AABB velocity components are defined at cell faces and other fluid properties defined at the center of the cells.

satisfies incompressibility inside the fluid and satisfies solid wall boundaries. for this, we need to be able to approximate the pressure gradient and divergence on a MAC grid. MAC grid makes accurate central differences robust, for example, where we need to subtract the d/dx component of the pressure gradient from the u component of velocity, there are two pressure values lined up perfectly on either side of the u component just waiting to be differenced. The full implementation of projection is lengthy and is detailed in [MF07] in chapter 4.

3.4 Resulting animations

The github page http://www.github.com/loshjawrence/FluidSim shows some resulting animations of the implemented smoke simulation using these resources.

References

[Jen01] Ronald Fedkiw; Jos Stam; Henrik Wann Jensen. Visualization of smoke. *SIGGRAPH*, 2001.

[MF07] Robert Bridson; Matthias Muller-Fischer. Fluid simulation: 2007 siggraph course notes. *SIGGRAPH*, 2007.

[Sta99] Jos Stam. Stable fluids. *SIGGRAPH*, 1999.