Harish Tella

March 16, 2005

Math198

Illigas

I propose the creation of a interactive and realistic gas simulation. The project will be implemented in GLUT/OpenGL and will then be ported to the Cave/Cube. The algorithms and methods used will be derived from the paper on realistic modeling of fluid systems by Jos Stam. In his paper Jos Stam presents a method to create a visually convincing simulations of a gas based on the Navier Stokes equations. He claims that his method departs from traditional approaches used in computational physics and focuses on speed and visual quality. “Unlike physically accurate solvers which have strict bounds on their time steps, our algorithms are stable, and never 'blow up'”. The implementation starts with dividing a 3D space into grid points. There will be four arrays created that represent the density, x velocity, y velocity, and z velocity at each grid point. The extreme grid points of the 3D space are treated specially because they represent the boundary layer conditions of the simulation. Before the simulation is started the user selects grid points to add density to. Once the simulation has started the user can add additional density and of course add forces with the mouse/cave controller.

There is a series of calculations that occur during each time step. They can be easily grouped into two sets of functions. The first set is the density step. The density step contains three steps within itself. First, the program checks if the user is adding density to the simulation currently in progress. If this is so the program modifies the density array

depending on where the user desires to add density. Then the program handles the diffusion of the gas(density). Each grid point looses some of its density to the cell directly above it, below it, right of it, to the left of it, in front of it, and behind it, but the cell also gains density from these same cells. Lastly the density is affected by a vector field that remains constant during the density step phase. This moves the density values along the vector field.

The second major set of calculations is the velocity step. The steps that take place within the velocity step are similar to the density step. The first step is to check if the user is adding forces. If so the program modifies the three velocity arrays respectively. Then the program handles the viscous diffusion of the velocity field. This is where each grid point of the velocity field looses some of its magnitude to its neighbors, but also gains magnitude from its neighbors. This is similar to the diffusion step of the density except it is being applied to the velocity arrays. Lastly the velocity step handles the self advection of velocity field. Jos Stam believes that this is best understood as “the velocity field moving along itself”.

There is one additional step that is part of the velocity step and is not part of the density step. The steps within the velocity step are not mass conserving so there is an additional step to compensate for this. It is based on the Hodge Decomposition which says that every velocity field is composed of a mass conserving field and a gradient field. The mass conserving step subtracts the gradient field from the velocity field which gives us the mass conserving field.

Program Sequence

input initial densities
(loop for each time step)
- velocity step
  - adding forces?
  - viscous diffusion
  - mass-conservation step
  - self-advection
  - mass conservation step
- density step
  - adding density?
  - diffusion of density
  - advection of density
- draw density

Navier Stokes Equations

$$\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} \cdot \nabla)\mathbf{u} + \nu \nabla^2 \mathbf{u} + \mathbf{f}$$

$$\frac{\partial \rho}{\partial t} = -(\mathbf{u} \cdot \nabla)\rho + \kappa \nabla^2 \rho + S$$

The Navier-Stokes equations: The first equation is the evolution of a velocity field. The second equation is the evolution of a density field in a velocity field. The three terms in each of these equations correspond to the three steps within the density step and the velocity step of the program.

The density generated will be drawn as 2 dimensional alpha blended sprites. When the user holds a button on the Cube controller and moves it (or holds a mouse button and moves the mouse for the desktop version) the vector field is modified which affects the density field. This is the basic setup of the program. If time permits there can be additions such as constant sources of particles and objects that constantly affect the

vector field, such as a fan.

Sources:

Stam, Jos. Real-Time Fluid Dynamics for Games. Alias | Wavefront.

<http://www.dgp.toronto.edu/people/stam/reality/Research/pdf/GDC03.pdf>.