DD::Fluid::Solver::SolverFire (sketches_0438)

Nafees bin Zafar nafees@d2.com
Henrik Falt hfalt@d2.com
Chamberlain Fong cfong@d2.com
Mir Zafar Ali mzali@d2.com

Digital Domain

This paper describes an implementation of a fire simulation tool used by artists at Digital Domain. We discuss choices of computational techniques, practicalities of initiating a fire, user controls, and useful data outputs to aid rendering.

Keywords: fluid dynamics, fire simulation, visual effects.

Simulation Method

We required a simulator which could produce complex and turbulent flames. Artifacts such as flame sheet separation, or multiple flame interactions, are difficult to produce with single-phase incompressible Navier-Stokes solvers. We chose to employ the multi-phase fire simulation scheme introduced by Nguyen et al [2002]. The reacting vaporized fuel, or blue core, is simulated like a liquid, and tracked with a particle corrected level set surface. Temperatures and densities are generated around the blue core to drive the gaseous phase. The two phases are modelled separately, and subsequently coupled by enforcing mass and momentum conservation.

The blue core level set can undergo interesting topology changes, which greatly adds to the turbulent look of the simulation. Boundary and solid fuel objects are also represented with level sets. We utilize a new stable semi-lagrangian level set advection technique [Enright et al. 2004] which provides significant performance gains over the traditional method by taking fewer timesteps than prescribed by the CFL limit. We maintain fields such as velocity and temperature separately for the two phases. This data separation and the numerically independent nature of the phase coupling provides us an opportunity to perform the linkage calculations in parallel.

Fuel and Ignition

Our simulator introduces a new vaporized fuel type, in addition to solid fuels described by Nguyen et al. Ignition involves generating a blue core level set from a burning fuel, and setting corresponding initial velocities. The user specifies regions at ignition temperature, and the simulator starts a reaction zone if fuel occupies those regions. The ignition regions can be animated, thus allowing artists to direct initiation and progression of a flame. The user may also directly model a blue core to start a fire.

A solid fuel object can be ignited by placing the burning voxels in a temporary level set interior. The reinitialized level set is then inflated by a user specified expansion term, and clipped against all boundary objects via CSG operations. Velocities in this new region are set to the expansion term along the normal vector. The vaporized fuel ignition involves additional velocity terms to model the explosion-like ignition. This subject is covered in further detail in a separate presentation.

Simulation Controls

The hot gaseous product generation is driven by a reaction coordinate field. This concept proposed by Nguyen et al. uses a scalar term representing the amount of time a fluid sample has been away from the blue core. Since their formulation inversely relates the coordinate with reaction progression it is more intuitive to provide the user a direct time parameter ($t = Y_{initial} - Y$, where Y is the reaction coordinate).

A fire will affect all of a small simulation space very quickly. If the simulation space extents are boundaries, then undesirable artifacts will occur from the feedback of forces from these walls. We allow for "wall-less" simulations by explicitly setting the staggered velocity component on the wall equal to the component on the opposing voxel face. This is performed for all voxels on the simulation bounds prior to the divergence correction step.

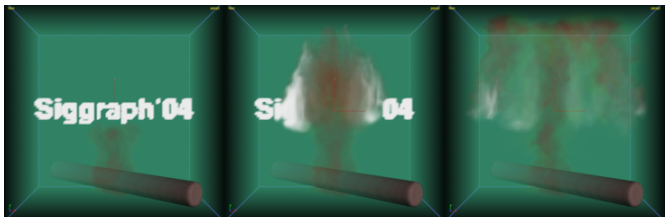

Figure 1: Previsualization of a fire with solid and vapor fuels.

Renderer Support

Velocity, temperature, and density fields are sufficient for photorealistic rendering of a single frame, but over a sequence spatial coherence of large scale procedural noise patterns must be established. Fluid fields provide insufficient data to achieve this, so we resort to flowing particles along the simulation with each internal timestep. Particles are birthed by the simulator in areas specified by the reaction time parameter, and only live for a limited time. Each particle carries the internal fluid fields, and an orientation vector. Particle orientation can be calculated by accumulating the rotational component of the fluid (Cauchy-Stokes decomposition) along the particle's path.

References

ENRIGHT, D., LOSASSO, F., AND FEDKIW, R. 2004. A fast and accurate semi-lagrangian particle level set method. *http://graphics.stanford.edu/~fedkiw/papers/stanford2003-10.pdf*. In review.

NGUYEN, D., FEDKIW, R., AND JENSEN, H. 2002. Physically based modelling and animation of fire. *ACM Transaction on Graphics 21*, 721–728.