Bending Fire with Plume, a CUDA-Based 3d Fluid Solver

Olivier Maury Dan Piponi Florent Andorra Craig Hammack
Industrial Light & Magic*

Figure 1: *Industrial Light & Magic. All rights reserved.*

1 Introduction

Central to the animated series "Avatar: The Last Airbender" are characters called "benders" who control the elements. For the movie adaptation, Industrial Light & Magic was tasked with translating the element bending effects into live action. The challenge of developing a methodology for fire bending is that it epitomizes the conflict between the needs for both creative freedom and realism. In one of our recent shows, we had already demonstrated that we could reliably leverage the power of GPUs for both simulation and final rendering in production [Horvath and Geiger 2009]. However, the frustum-based solution developed for that show was inappropriate for the camera motion, the fire dynamics and rendering requirements of the new show. In response, we developed Plume, a more general purpose GPU-based 3d fluid solver integrated with a volume renderer. Although originally developed for fire bending, Plume was also used throughout the show for air bending dynamics as well as the simulation and rendering of dust, smoke and mist.

2 Contributions

The contributions of this new fluid tool to our effects pipeline were threefold. First, we achieved an order of magnitude speed up compared to our routinely used CPU-based solution. This speed up allowed artists to produce many iterations of their work each day rather than leaving simulations to run overnight. It also allowed them to work at higher resolutions. Second, the integration of the simulation and rendering in the same tool allowed creative decisions to be made based on the final render rather than on intermediate simulation visualization. Third, this integration also allowed non-effects artists to use the tool productively without having to first learn the many steps of a typical flexible effects pipeline.

3 Simulation & Rendering

Because Python is embedded in our proprietary 3d application, we wrote Plume as a Python extension module, making it very easy for artists to access the tool from a familiar environment. All the heavy lifting is carried out on the graphics board using CUDA on high end graphics cards. The solver core is based on a traditional Eulerian smoke solver [Fedkiw et al. 2001] enhanced with an artist friendly combustion model. We chose algorithms that map particularly well to the GPU, including a full geometric multigrid for the projection step and an adaptive semi-Lagrangian scheme for the advection step. Collisions with obstacles were handled through the simple yet very effective Iterative Orthogonal Projection [Molemaker et al. 2008]. Temperature and fuel grids can be seeded in various ways. The most popular method turned out to be from particles driven by our effects authoring module. Through the course of production, we also discovered that artists required greater control over the grid values as the simulation progressed. This realization prompted us to write a virtual machine to evaluate expressions directly on the GPU, using Python to produce byte code. This feature allowed users to author high performance simulation stages in Python.

On the rendering side, we wrote a CUDA-based ray-marcher. Since we quickly discovered that I/O became a bottleneck for high resolution simulations, this ray-marcher was designed to render the simulated grids either from disk or directly from memory. This renderer provides an extensive set of features that control how simulated quantities map to final colors and densities, mostly through the use of ramps and procedural noise. As we were developing the tool, we tested different detail enhancement techniques. The most successful solution was to develop tools to facilitate the partitioning of simulations into smaller pieces, enabling very high effective resolution.

Plume was conceived as a solution to a specific problem on "The Last Airbender". Once in production it was quickly adopted for use on "Iron Man 2" and continues to be used on other projects across the facility.

Special thanks to Daniel Pearson, John Hansen, Mark Chataway, and Vick Schutz

References

FEDKIW, R., STAM, J., AND JENSEN, H. W. 2001. Visual simulation of smoke. In *Proceedings of SIGGRAPH 2001*, ACM Press / ACM SIGGRAPH, E. Fiume, Ed., Computer Graphics Proceedings, Annual Conference Series, ACM, 15–22.

HORVATH, C., AND GEIGER, W. 2009. Directable, high-resolution simulation of fire on the gpu. *ACM Transactions on Graphics 28*, 3 (July), 41:1–41:8.

MOLEMAKER, J., COHEN, J. M., PATEL, S., AND NOH, J. 2008. Low viscosity flow simulations for animation. In *2008 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, 9–18.

*e-mail: {omaury, dpiponi, fandorra, craigh}@ilm.com

SIGGRAPH 2010, Los Angeles, California, July 25 – 29, 2010.
ISBN 978-1-4503-0210-4/10/0007