A practical visualization strategy for large-scale supernovae CFD simulations

Conference Paper · December 2011

DOI: 10.1145/2077378.2077397

6 authors, including:

Derek K Gerstmann
University of Western Australia
7 PUBLICATIONS **228** CITATIONS

SEE PROFILE

Paul Bourke
117 PUBLICATIONS **1,020** CITATIONS

SEE PROFILE

Andreas Wicenec
University of Western Australia
135 PUBLICATIONS **1,226** CITATIONS

SEE PROFILE

A Practical Visualization Strategy for Large-Scale Supernovae CFD Simulations

Derek K. Gerstmann *
ICRAR, University of Western Australia

Toby Potter
ICRAR, University of Western Australia

Michael Houston
Advanced Micro Devices, Inc.

Paul Bourke
iVEC, University of Western Australia

Kwan-Liu Ma
University of California, Davis

Andreas Wicenec
ICRAR, University of Western Australia

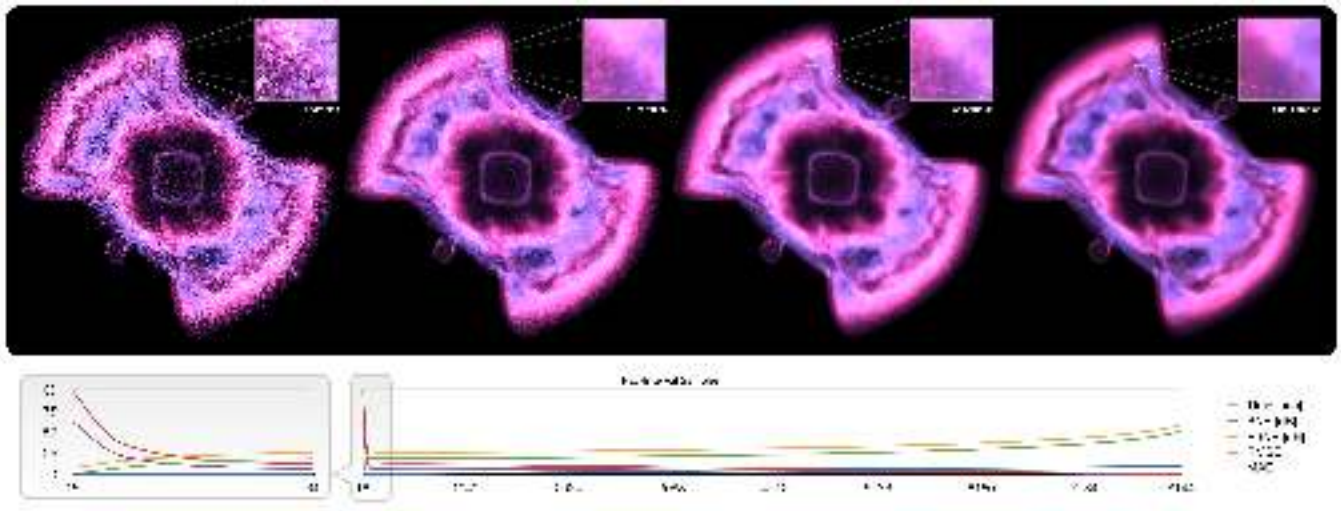

Figure 1: *Progressive Volume Rendering of Radio Frequency Emission Derived from an Expanding Type II Supernova CFD Simulation.*

Top (from left to right): *16 ray-interval steps (0.17 sec), 64 steps (0.45 sec), 256 steps (1.56 sec), 8192 steps (47.81 sec).* **Bottom:** *Quality metrics showing convergence over time. Left graph corresponds to highlighted zoomed in section. Note the quality achieved in the first 64 ray-interval samples (a total of 8 passes). Times reported are total seconds elapsed for rendering a 1k image for a single time-step from a resampled* 256^3 *dense grid w/32-bit float-point density values. Target system was an Apple MacBook Pro 2011 w/8GB RAM, Intel Core i7 2.2GHz, AMD Radeon 6750M 1GB running Mac OSX Lion v10.7.0. Dataset courtesy of Toby Potter from the International Centre for Radio Astronomy Research (ICRAR), University of Western Australia (UWA).*

1 Overview

Simulating the expansion of a Type II supernova using an adaptive computational fluid dynamics (CFD) engine yields a complex mixture of turbulent flow with dozens of physical properties. The dataset shown in this sketch was initially simulated on iVEC's EPIC supercomputer (a 9600 core Linux cluster) using FLASH [Fryxell et al. 2000] to model the thermonuclear explosion, and later post-processed using a novel integration technique to derive the radio frequency emission spectra of the expanding shock-wave front [Potter et al. 2011]. Model parameters have been chosen to simulate the asymmetric properties of the SN 1987A remnant [Potter et al. 2009].

This offline workflow takes several hundred machine-hours to complete, and results in a volumetric time-series dataset stored on an adaptive mesh refinement grid with a total storage allocation of $>=$ 10 terabytes. This dataset consists of several thousand time-steps (adaptively outputted at non-linear time intervals), each containing a dozen simulation variables stored as floating-point vector fields. Due to the intricate nature of the flow, visualizing these datasets requires a rendering engine capable of high-quality image reconstruction in order to maintain the underlying visual complexity.

In this sketch, we describe a practical approach we've developed which enables explorative visualization for studying large-scale time-series astrophysical CFD simulations. This is part of an ongoing data-intensive research project within our group to support the visualization of large-scale astrophysics datasets for the scientists at the International Centre for Radio Astronomy Research (ICRAR).

*e-mail: derek.gerstmann@icrar.org

SIGGRAPH Asia 2011, Hong Kong, China, December 12 – 15, 2011.
ISBN 978-1-4503-0807-6/11/0012

In particular, we discuss the application of progressive stochastic sampling and adjustable workloads to insure a consistent response time and a fixed frame-rate to guarantee interactivity. The user is permitted to adjust all rendering parameters while receiving continuous visual feedback, facilitating explorative visualization of our complex volumetric time-series datasets.

2 Approach

The main goal of our system was to enable users to quickly search and isolate specific features within our large-scale time-series datasets, while providing an accurate and appropriate representation of the underlying data. Our system uses a progressive rendering approach and combines this with stochastic sampling to enable a high-quality rendering and interactive control over all render parameters including camera properties, clip-planes and transfer function editing.

The image sequence in Fig. 1 demonstrates the capabilities of our stochastic rendering approach in the context of our radio-frequency simulation workflow. The figure highlights the quality of our system for a given number of ray-interval samples and the corresponding graphs summarize the rate of convergence for the listed quality metrics. The rendered images provide a unique view of the radio frequency emission at 843 MHz for day 4136 after the SN 1987A explosion, as represented by the radio power per unit volume (in log-scale), modulated by the spectral index from the aforementioned simulation dataset.

While other progressive and stochastic sampling techniques for volume rendering exist (eg [Csebfalvi and Szirmay-kalos 2003]), our

system differs from prior systems in that consistent performance was a key requirement. The user is thus provided with immediate visual feedback during interactions and presented with a high-quality approximation of a final rendered image. This eliminates the time-consuming process of iteratively adjusting render parameters, and waiting for new images to get rendered.

The unique aspects of our approach are as follows:

- Progressive volumetric ray-marching with stochastic sampling which provides a high-quality volume rendering with continuous refinement.
- Guaranteed interactivity by adopting deadline scheduling and adjustment of workloads to limit frame computation time.
- Continuous refinement of the visual representation which converges to a high-quality image over time. Rendering can be interrupted and restarted at any time.

The system uses OpenGL for displaying final images to the user, and either OpenCL or OpenGL for computing the intermediate frame-buffers, depending on the target platform. Each time-step has been reduced to a 256^3 dense grid for the target machine (see Fig. 1 caption), and is stored in memory as a floating-point 3D texture.

3 Progressive Stochastic Volume Rendering

An overview of our sampling approach is outlined below:

- Per ray-interval hash is used to generate pseudo random numbers in parallel to stochastically sample ray origins, ray interval locations, and 3d texture coordinates.
- Intermediate frame-buffer values are generated from N-samples per ray-interval and stochastically sampled per pixel in screen space per-pass.
- Final images are composed from a weighted average of intermediate frame-buffers and displayed after each pass.

Volumetric ray-marching is utilized with jittered sample locations for integrating the volume. A standard bounding volume estimation is performed to determine the extents of the ray-interval.

Rendering begins by selecting a global step size based on the number of samples per pass and the spatial extents of the volume. This number is adjusted based on the deadline time and the average time it took to render the previous frames. If the previous average time took longer than the deadline time, the sample count is reduced to maintain interactivity.

A random seed is generated per frame, which is then used to generate separate globally unique identifiers per ray-interval. These identifiers are used to seed a pseudo-random number generator for generating uniform random values in the stochastic sampling routines [Thomas et al. 2009].

The volume is integrated by stepping along the ray-interval at the globally fixed step size. To reduce banding artifacts, we perform a limited binary search from the current sample position to the next half-step forward, using a thresholded value range. If the next sampled density value is outside the acceptable value range, we split our half-step interval and combine the values based on their distance from the primary sample position.

For each 3d texture fetch, we jitter the coordinates by sub-texel offsets prior to sampling. Ray-intervals are condensed down to per-pixel samples stored in a floating-point frame buffer and are accumulated after each pass using a rolling average. The accumulation step also jitters the texel offsets at sub-pixel intervals in screen-space to further reduce sampling artifacts.

4 Discussion

Other researchers have looked at sparse data structures, as well as streaming and out-of-core algorithms to handle larger-than memory volumetric datasets (eg [Fogal and Krüger 2010]). Our technique is largely orthogonal to this work, and could be combined to provide a high-quality, out-of-core, progressive rendering using more sophisticated level-of-detail routines, and more advanced load balancing schemes. We leave these ideas to future work.

5 Conclusion

We have presented a practical approach for generating a high-quality volumetric rendering with a guaranteed level of interactivity. Within this framework we have adopted a single scattering approximation which generates natural looking images without requiring a significant amount of expertise or artistic control. The approach naturally accentuates the denser regions of the dataset using absorption, extinction, lighting and shadows. We combine this with an arbitrary transfer function to give an expert even more fine-grained control. This system has proved extremely useful for explorative visualization and has been beneficial in our work on studying large-scale astrophysical simulations which have complex time-varying features.

Acknowledgements

This work was funded by UWA, UC Davis, and ICRAR, and was supported by iVEC through the use of advanced computing resources located at UWA, ARRC and Murdoch.

References

CSEBFALVI, B., AND SZIRMAY-KALOS, L. 2003. Monte carlo volume rendering. In *In Proc. of IEEE Visualization*, 449–456.

FOGAL, T., AND KRÜGER, J. 2010. Tuvok: an architecture for large scale volume rendering. In *VMV'10*, 139–146.

FRYXELL, B., OLSON, K., RICKER, P., TIMMES, F. X., ZINGALE, M., LAMB, D. Q., MACNEICE, P., ROSNER, R., TRURAN, J. W., AND TUFO, H. 2000. Flash: An adaptive mesh hydrodynamics code for modeling astrophysical thermonuclear flashes. *The Astrophysical Journal Supplement Series 131*, 1, 273.

POTTER, T. M., STAVELEY-SMITH, L., NG, C.-Y., BALL, L., GAENSLER, B. M., KESTEVEN, M. J., MANCHESTER, R. N., TZIOUMIS, A. K., AND ZANARDO, G. 2009. High resolution 36 GHz imaging of the supernova remnant of SN 1987A. *The Astrophysical Journal 705*, 1, 261.

POTTER, T., STAVELEY-SMITH, L., KIRK, J., REVILLE, B., BICKNELL, G., SUTHERLAND, R., WAGNER, A., ZANARDO, G., AND GERSTMANN, D. 2011. Asymmetric supernova remnants! 3D simulations of the expanding remnant of SN 1987A may shed light on the puzzle. In *SCSS4: Supernovae and their Host Galaxies*.

THOMAS, D. B., HOWES, L., AND LUK, W. 2009. A comparison of CPUs, GPUs, FPGAs, and massively parallel processor arrays for random number generation. In *Proceedings of FPGA*.