Extreme Multi-Resolution Visualization: A Challenge on Many Levels

Joanna Balme, Eric Brown-Dymkoski, Victor Guerrero, Stephen Jones, Andre Kessler, Adam Lichtl, Kevin Lung, William Moses, Ken Museth, Nathan Roberson, Danny Taller, and Tom Fogal (NVIDIA)

SpaceX

August 7, 2015

1 Introduction and Motivation

Accurate simulation of turbulent flows is a persistent problem, marked by the cost of directly resolving chaotic interactions between widely disparate scales. Energies interact between the largest bulk phenomena – the full aircraft, thrust chamber, or device of interest – down to the minute lengths where dissipative forces dominate. Applied cases easily encompass length and time scales ranging over six orders of magnitude for industrial combustors. Bulk statistical quantities can often suffice for design and analysis of some applications, such as aerodynamics; however, in rocket engines, mixing on the smallest scales drives combustion and is crucial for capturing the large scale dynamics which characterize stability and performance. In these cases, current computational approaches still fall short for predictive analysis: the difficulties in simultaneously capturing the large and small scales, and their interactions, significantly degrades capability and limits the utility of simulation in engineering design cycles.

Multiresolution wavelet analysis is an emerging method for dynamically adaptive simulation. Especially applicable to turbulent flows, it has the ability to efficiently capture wide scale ranges at high accuracy by using multidimensional signal compression to identify intermittent coherent structures. A many-core algorithm was implemented targeting graphics processing units (GPUs) in order to take full advantage of developments in single instruction, multiple thread (SIMT) parallelism. In particular, GPUs offer high memory bandwidth that can be exploited by programs with well-coalesced access. Complimentary to this, an efficient algorithm was developed for mapping the implicit wavelet topologies for visualization. This system serves both polygonal representation approaches and a common infrastructure to interface with movie-quality rendering and animation packages.

2 Adaptive Wavelet Simulation

Adaptive wavelet collocation is a general finite-difference framework for solving PDEs on an optimal, dynamic grid [2]. It permits high order, non-dissipative finite differencing and high order interpolation. Resolution levels are defined by a series of dyadic nested grids, with lower levels being the coarser. An interpolating wavelet transform identifies information-carrying grid nodes at each level, such that by identifying only these significant nodes and discarding the remainder the system maintains a sparse data representation as it evolves through time.

Figure 1 shows how a mesh adapts around a shear-flow vortex, in this case using 14 resolution levels to capture the finest structures. The resulting grid con-

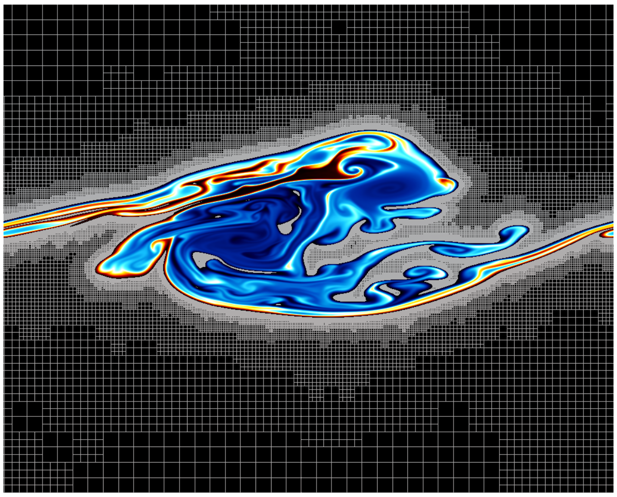

Figure 1: Periodic shear flow simulation (Re $=$ $80,000$, 14 levels of resolution), colored by fluid density and showing wavelet mesh refinement.

sists only of nodes that are required to reconstruct the global flow field at an error level which can be specified *a priori*. For homogeneous turbulent regions, data compression greater than $95-99\%$ can be achieved while resolving the very small, intermittent eddies and motions with less than 1% error [3].

These grids pose several challenges for high-performance computing: the data layout evolves continuously throughout the course of a simulation, and the complicated topology of orthogonal, multi-resolution stencils frustrates coalesced memory access. While the existence of stencil support is guaranteed by the wavelet transform, locating it in memory is non-trivial due to the adaptive, multi-level construction.

Taking advantage of the property that memory access patterns are not random, this algorithm avoids the need for auxiliary topological structures. Instead, it uses a series of permutations to optimize the grid for streaming access. For each step, the subset of compute nodes for a given level are extracted from the full grid along with all nodes from coarser resolutions. By sorting along each dimensional direction, the fact that stencil locations are well defined by integer strides on this subset means that the memory can be traversed in a parallel-friendly fashion with threads operating on consecutive memory addresses. The GPU implementation employs scalable, $O(N)$ parallel partitioning and radix sorts for efficient data rearrangement.

3 Sparse Dataset Visualization

Sparse data representations are also crucial for efficient and high fidelity visualization at extreme levels of detail. Unfortunately, the wavelet grid is ill-suited for this purpose. Lack of explicit connectivity information, while a major computational advantage for simulation, complicates all but trivial point rendering techniques. Many visualization approaches require the topological information of the mesh to be known, since it facilitates efficient interpolation kernels for ray-integration and volume rendering. Sparse octrees, with fully populated leaf nodes, are a common representation that naturally defines this connectivity.

Although they have been derived independently and for differing purposes, octrees and wavelet grids are intrinsically aligned. Both feature a level-by-level dyadic construction that terminates at a well defined root, and our algorithm exploits this topological similarity to efficiently construct a forest of octrees from the simulation-optimized wavelet grid.

Initially, the process builds a VDB tree [1] from the adaptive wavelet structure. While OpenVDB offers constant-time ($O(1)$) threaded random access iterators for sparsely sampled domains, we primarily use this VDB tree as an intermediate acceleration structure to index and query point-neighborhood information. As wavelet transform stencils ensure that internal branch nodes in the octree exist, top-down tree building is efficient and allows for early termination of searches. Nodes added in order to fully populate octree leaves have their accompanying data filled in using tri-linear interpolation–though low order, this is fast and ensures boundedness of the simulation data. The wavelet transform and finite difference stencil support present a topology such that minimal nodes need to be added, minimizing artifacts and maintaining the integrity of the visualization.

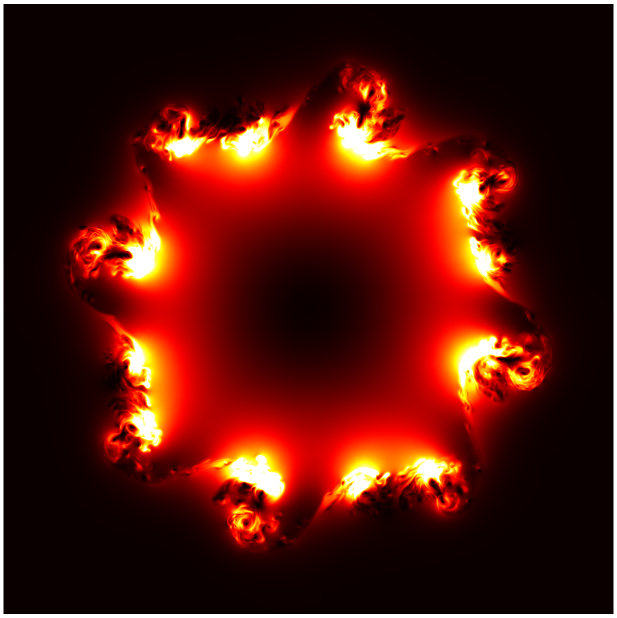

Figure 2: Vortices develop at the interface between counter-rotating flows, simulated on a multi-resolution grid using 17 levels of detail. Retaining the sparse structure during rendering maintains an effective compression ratio of 500x and allows each timestep to be converted and rendered in less than one second.

The resulting forest of octrees, once constructed, enable two separate visualization approaches: polygonalization for rasterizing, or volumetric rendering and ray-tracing.

Figure 2 shows a multi-resolution rendering of polygonalized data from a simulation spanning 17 levels of detail, where the smallest feature is just 15 microns within an overall domain spanning 2 meters. Direct rendering of the original point cloud takes more than 60 seconds per frame using one common visualization tool; by comparison, the VDB-based sparse-data algorithm developed here both generates topology and renders the scene in less than one second.

Volumetric rendering and ray-tracing may be performed either in-place, or by populating a second sparse VDB representation. This is distinct from the VDB tree that is used previously in tree construction: it is a self-compressed structure, employing a controlled-fidelity approach similar to wavelets, and populating interpolated voxel information from the octrees. This derived representation is natively supported by, and allows direct interfacing with, most major movie-quality production and animation packages, including Houdini™, RenderMan™, Arnold™ and NVIDIA OptiX™.

4 Professional & Production-Quality Visualization

Along with the benefits of leveraging the technology, taking a movie-industry approach to rendering reintroduces a palpable element often lost in the more common scientific visualization techniques. It does not replace such post-processing, but rather is complimentary, offering an intuitive view. It enables a very human and tangible approach that is more prevalent in experimental physics.

The visualization shown in Figure 3 was created using NVIDIA OptiX. OptiX is a ray tracing framework for CPUs and GPUs that greatly simplifies the process of writing custom ray tracers and shaders. OptiX uses bounding volume hierarchies internally and groups similar rays automatically, allowing the developer to focus on using ray tracing to describe their scene and interact with their data. It is an excellent fit for scientific visualization as one can define the rendering in the best way to communicate the data's features. This can use the physical properties of light, or modified forms, such as virtual Schlieren imaging.

The OptiX renderer is similar to ray-guided renderers introduced in recent years [5, 6, 7]. It uses ray marching, which is considerably faster than the marching cubes alternative, to identify an isosurface of interest. The data from this simulation was converted into VDB format as previously outlined. Since OpenVDB was designed for a CPU, some modifications were made to use the computational power of a GPU to achieve real-time rendering with advanced

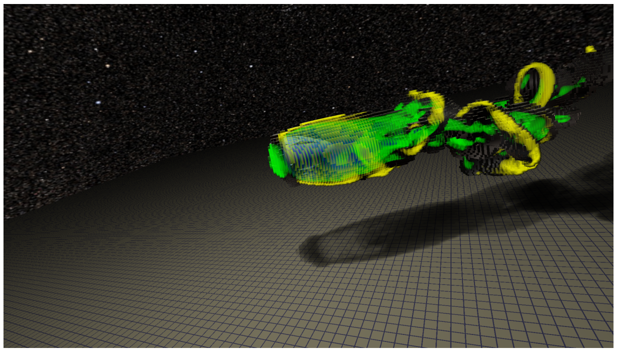

Figure 3: Ray-traced image from NVIDIA OptiX^{TM}, using refraction to visualize wake formation around a capsule during late re-entry (Mach = 0.33)

ray-tracing effects. This involved expanding VDB's internal tile and brick sizes by a factor of 64 to improve GPU access times. The data structure conversion takes about 2 seconds per timestep for a 120 megabyte simulation file.

OptiX has allowed experimentation with more advanced rendering techniques. Domain scientists are interested in validating the multi-resolution structure of their grid, but the sharp discontinuities can be jarring in communicative visualization. The visualization of Figure 3 uses refraction and reflection to detail fluid structure as a glassy material. This deemphasizes the 'blocky' discontinuities that can be highlighted with typical Phong shading, although it tends to drown out data at fine scales. Naive viewing the data strictly as an isosurface hides the 'core' of the shockwave and the structure that occurs inside the surface itself. The solution is to do restricted ray marching at the surface intersection point, accumulating color using a standard volume rendering approach. The volume rendering is blended with the color information from the obscuring isosurface computation. This unique method of rendering leads to an interesting focus+context style of visualization, with the volume rendering emphasizing fine-scale features while the semi-transparent isosurface gives large-scale context information.

5 Conclusion

Predictive simulations at the scale and complexity required by rocket combustor physics has prompted the development of a high-performance, adaptive methods. This approach exploits the similarity of the simulation and visualization sparse data representations to maintain a compressed form throughout the CFD pipeline.

References

[1] Museth, K.: *VDB: High-Resolution Sparse Volumes with Dynamic Topology*, ACM Transactions on Graphics **32**(3), 2013. Presented at SIGGRAPH 2013.

[2] Vasilyev, O.V., Bowman, C *Second Generation Wavelet Collocation Method for the Solution of Partial Differential Equations*, J. Comp. Phys., **165**, 660-693 (2000)

[3] Nejadmalayeri, A., Vezolainen A., Vasilyev, O. V. *Reynolds Number Scaling of CVS and SCALES*, Physics of Fluids, **25**, 110823, 660-693 (2013)

[4] Nejadmalayeri, A., Vezolainen A., Brown-Dymkoski, E., Vasilyev, O. V. *Parallel Adaptive Wavelet Collocation Method for PDEs*, J. Comp. Phys., **298**, 237-253 (2015)

[5] Crassin, C., Fabrice, N., Lefebvre, S., Eisemann, E *Gigavoxels: Ray-Guided Streaming For Efficient And Detailed Voxel Rendering*, ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D) (2009)

[6] Hadwiger, M., Beyer, J., Jeong, K., Pfister, H. *Interactive Volume Exploration Of Petascale Microscopy Data Streams Using A Visualization-Driven Virtual Memory Approach*, Proceedings of IEEE Visualization 2012

[7] Fogal, T., Schiewe, A., Krueger, J. *An Analysis of Scalable GPU-Based Ray-Guided Volume Rendering*, Large Data Analysis and Visualization, IEEE Visualization (2013)