SAND2012-2137C

The Leading Edge: Impact of ASC Investments in Large Scale Visualization and Data Analysis

David Rogers, Becky Springmeyer, James Ahrens
Peer-Timo Bremer, Eric Brugger, Hank Childs, Nathan Fabian, Berk Geveci,
Li-Ta Lo, Ming Jiang, Kenneth Moreland, Ron Oldfield, Valerio Pascucci,
John Patchett, Christopher Sewell, Brad Whitlock, Jonathan Woodring

Abstract— **We discuss the production tools and research projects for scalable data analysis and visualization that have resulted from the Advanced Simulation and Computing (ASC) Program's R&D investment in large data analysis. We give examples of resulting open source software projects that are have daily impact far beyond ASC domains.**

Keywords- Advanced Simulation and Computing; Parallel visualization; Large scale visualization; Large scale analysis; Big Data; Scalable analysis tools

I. Introduction

Advanced Simulation and Computing Program (ASC) investments in R&D for visualization and data analysis (VDA) have advanced scientific discovery at DOE laboratories, and have impacted non-DOE laboratories, academic institutions, and businesses since the program began. Established in 1995 to support the U.S. Defense Program's shift in emphasis from test-based confidence to simulation-based confidence[1], the ASC program has lead research and development of multi-platform, parallel visualization applications and toolkits which are in wide use today. In addition, the ASC program supports a visualization and analysis community that collaborates with ASCR and leverages ASCR research, increasing the scope and impact of scientific visualization and analysis for all large-scale simulation scientists.

The investments made by both ASC and ASCR have been – and continue to be – a fertile source of collaboration and technical impact across agencies. The result is an array of production tools and research projects that meet today's needs while also driving future research.

II. ASC Production Tools

ASC visualization R&D fundamentally changed large-scale scientific data analysis, as evidenced in software like ParaView, VisIt, EnSight Gold, Chromium, and leading edge hardware installations such as interactive theatres and facilities with large-format displays. Today, computational scientists around the world routinely use ASC-funded technology to analyze large scale datasets on parallel supercomputers and visualization clusters. These large-scale tools are key elements in computational science efforts supported and adopted by government programs including Office of Science (OSC), Advanced Scientific Computing Research (ASCR), the National Science Foundation (NSF) and the Department of Defense (DOD). In addition many academic, industrial, and commercial institutions are active users and contributors to these projects.

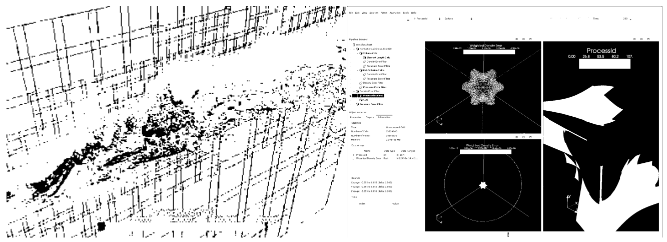


Figure 1: Two views from ParaView being used to visualize large data – one is a screen capture of a data view, and the other a snapshot of the entire application GUI

A. ParaView

In 1998, national labs at Los Alamos, Sandia and Livermore lead an effort to create an end-user visualization tool for large-scale data. This tool was ParaView - a parallel version of the open-source, multi-platform, visualization toolkit (VTK), and was created to analyze large data sets in a scalable manner. An early accomplishment – visualizing a petabyte dataset when computation was happening at terascale computing – demonstrated the scalability of the parallel toolkit.

ParaView enables scientific discovery from the massive data produced by high-performance DOE simulation computer codes. A driving goal of this effort was to effectively deliver the results of visualization and analysis research and development efforts directly to DOE simulation scientists. The delivery model for ParaView, which was available as an open-source project, significantly changed how large-scale, scientific visualization was carried out. Today, scientists use supercomputers in conjunction with ParaView to routinely visualize extremely large-scale datasets that could have never been analyzed in the past.

ParaView is an open-source, extensible software tool and framework for scientists to visualize and analyze large-scale data sets. It provides a graphical user interface for the creation and interactive execution of analysis tasks, in addition to scripted batch tasks. ParaView transparently supports the analysis of large data sets by executing these programs in parallel on shared or distributed memory computers. It supports hardware-accelerated parallel rendering, parallel analysis routines, and achieves interactive, human-in-the-loop performance. The software design balances and integrates a number of diverse requirements, including the ability to handle large data, scalability, interactivity, ease of use, and future extensibility by researchers and developers.

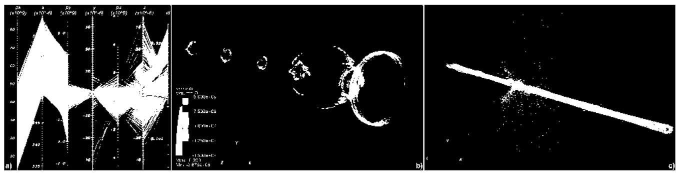


Figure 2: Screen captures showing VisIt visualizing large, high dimensional data.

B. VisIt

VisIt is an open source parallel visualization and graphical analysis tool for viewing and analyzing large-scale scientific data. It was originally developed at LLNL in the late nineties to serve the needs of ASC scientists for visualizing large-scale simulation data. VisIt's development has gone beyond its initial use case such that it currently has a broad DOE usage community, including the NNSA, the Office of Science ASCR and the Office of Nuclear Energy. This usage has extended outside of the DOE to be used by other large-scale simulation scientists as well, including governmental agencies like NSF and DoD, and by leading visualization academic institutions like UC Davis and the University of Utah, to name just a few.

VisIt's client-server architecture combined with its parallel server allows it to visualize and analyze data sets of unprecedented size on the supercomputing platforms without moving the data. VisIt supports five major use cases, including quantitative analysis, visual debugging of simulation codes, comparative analysis, data exploration and the creation of presentation graphics. Scaling studies show that VisIt can handle data sets up to a 4 trillion cells, which is more than an order of magnitude larger than the largest actual data sets encountered to date. VisIt also supports extensibility through the use of plugins that are dynamically loaded at run-time. The wide use and scalability of VisIt has enabled simulation scientists around the world to analyze their large-scale data, which would not have been previously possible without parallel tools like VisIt. It has also provided a development vehicle for future large-scale visualization and analysis research that is able to reach many users.

C. IceT

The Image Composition Engine for Tiles (IceT) is a high-performance, sort-last parallel rendering library. It has demonstrated scalability to today's petascale supercomputers, integrating some of the most recently developed parallel algorithms, such as radix-k from Argonne. With these improvements, IceT has demonstrated record-breaking rendering performance. IceT's nonintrusive image-based API simplifies integrating with any application by accepting any geometry, data-format, or rendering system. It is currently available for use in large-scale, high-performance visualization and graphics applications and is used directly in multiple production large-scale visualization tools, like ParaView and VisIt.

In addition to providing accelerated parallel rendering, IceT provides the unique ability to generate images for tiled large-format displays, in use by many DOE laboratories for their powerwalls and immersive display theatres. The overall resolution of the display may be several times larger than any viewport that may be rendered by a single machine. It manages the extra pixels of large-format displays by taking advantage of spatial decomposition of geometry, allowing tools like ParaView and VisIt to be used in large-scale immersive visualization. IceT uses this information to build load-balanced sparse images and reduce the overall computation and communication, providing interactive visualization.

III. ASC RESEARCH AND DEVELOPMENT

ASC has had many successes from the nineties to present day, and new research continues in large-scale visualization and analysis to prepare for exascale supercomputing and the new physics that will be enabled by large-scale simulations. In this era of supercomputing, there are many challenges for visualization and analysis, including massive-scale parallelism, storage and I/O bottlenecks, and increasingly complex results coming from high-performance simulations. To solve these challenges, particular areas of ASC research include frameworks for large-scale analysis application development, portable visualization and analysis codes for large-scale parallel architectures, and research into in situ (simulation run-time analysis) techniques, to name just a few.

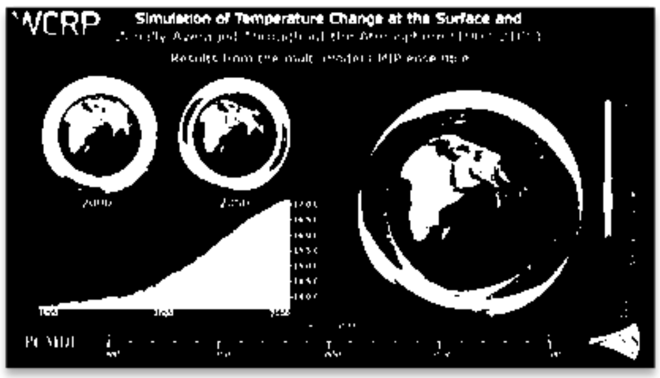


Figure 3: A screen capture from ViSUS, showing a view of an ensemble of climate simulation data.

A. ViSUS

ViSUS (Visualization Streams for Ultimate Scalability) is a project aimed at providing lightweight visualization and analysis of extreme scale data. Based on some initial advances in cache-oblivious data layouts and out-of-core algorithms as a project at LLNL, ViSUS has become a general software platform for fast development and deployment of task driven solutions for science, intelligence, and engineering applications.

The above images show how the ViSUS agile deployment approach enables quick adaptation to a specific set of computational and networking resources available while directly addressing the user need. The above images show the analysis of an ensemble of climate modeling simulations to understand and explain possible causes of global warming trends. Over 10 TB of raw simulation data was explored and summarized, which required developing a streaming pipeline for moving and processing the information from its storage source to the science investigation environment.

At the same time, climate scientists used this analysis and visualization environment to study atmospheric warming patterns and highlight for the first time important signatures in a way that can also presented to the general public and decision makers. In particular, scientists were able to see how the global warming is surprisingly structured in a cooling top layer combined with a stronger warming of the lower atmospheric layer, matching expectations related to anthropogenic causes. ViSUS also showed cloud activity as projected by averaging of a multiplicity of climate models and presented in a flat projection facilitation the analysis of more detailed information with reduced 3D occlusion.

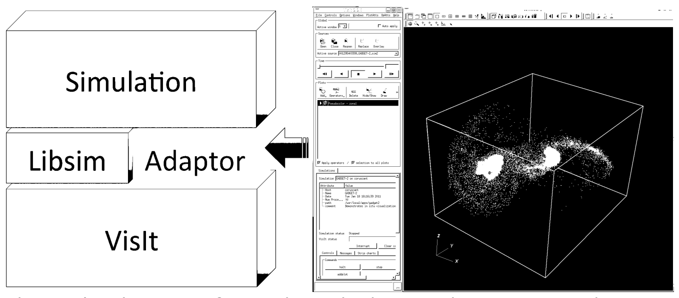


Figure 4: Diagram of the VisIt Libsim architecture, showing how VisIt attaches to a running simulation by running data through an adaptor module.

B. *VisIt Libsim*

VisIt Libsim is a library for instrumenting simulation codes to perform visualization operations in situ while data are being generated. In situ processing is meant to address the inherent I/O bottleneck in today's supercomputers and those planned in the near future. Current and future supercomputers provide far more compute capacity than I/O bandwidth. Post-processing applications that will read that data will face similarly slow I/O performance. To avoid the penalty associated with doing I/O twice, visualization and analysis must be able to execute at simulation run-time. VisIt Libsim enables data analysis and visualization at massively parallel scales without the customary I/O operations associated with writing and then reading data during offline analysis.

Libsim uses the paradigm of message passing, based on the MPI library whereby the in situ processing lives in the same process space as simulations. It consists of a small C library, callable from multiple languages that can be used to instrument simulations at the cost of adding a few tens of kilobytes to their executable size. At runtime, the simulation periodically calls Libsim functions to listen for inbound VisIt client connections. Once that connection has been made, Libsim dynamically loads the rest of VisIt's compute server library and the interactive VisIt client can request any of the simulation's meshes and arrays for use by any of VisIt's functions. The meshes and data are passed to VisIt through an adaptor layer. The adaptor allows simulation arrays to be bundled into higher-level constructs and sent to VisIt where it uses the simulation data to initialize VTK data structures. When the internal format of the simulation's data structures is compatible with those of VTK, zero-copy is achieved, lowering the memory overhead.

VisIt's Libsim has been deployed in its current form for 2 years and has been incorporated into several of LLNL's ASC codes, among them Ale3D and Kull. Libsim and VisIt are undergoing additional development to make them more suitable for large-scale in situ operations that take place in batch at higher levels of concurrency. For instance, this will allow Libsim to not require a VisIt client in order to create plots and post-process data. Other future work will consist of extending Libsim to create reduced datasets for traditional post-processing.

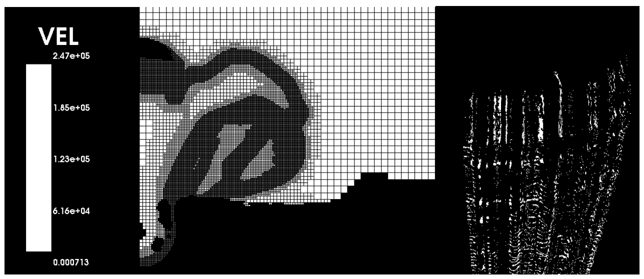


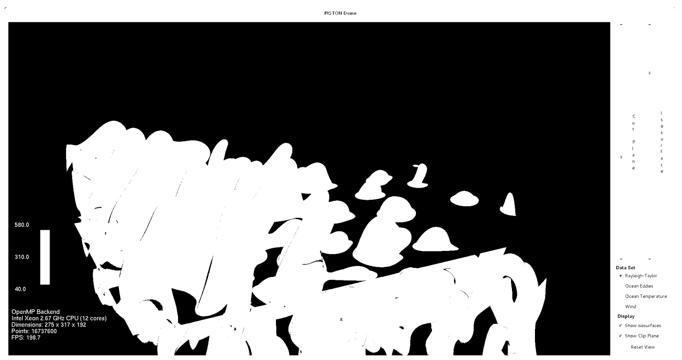
Figure 5: Results of using the ParaView Coprocessing library. On the left is an image written directly to disk during a simulation, and on the right is geometry that was generated during a running simulation.

C. *ParaView Coprocessing*

ParaView Coprocessing is an in situ library, originally developed at Sandia and Kitware beginning in 2008, which provides the numerous visualization and analysis algorithms for simulation run-time analysis from ParaView and VTK. Instead of the traditional "reader" component that reads simulation data from disk, memory adapter components are used to transform the simulation data model appearance into the VTK data model. ParaView pipeline codes then run in the same process space as the simulation and have direct access to the simulation data structures, but with a VTK appearance around the data. Perioidically, the simulation calls to ParaView to execute an analysis pipeline and the memory adapter allows VTK and ParaView to run transparently with simulation data, as if it were VTK data in a VTK pipeline. By utilizing the extreme computational capability of the supercomputer, we are able to reduce I/O time increasing the overall performance and throughput of large-scale simulations.

CTH is an Eulerian shock physics code that uses an adaptive mesh refinement (AMR) data model using the ParaView Coprocessing library. We show the simulation of an exploding pipe bomb shown in the upper-left figure. Using an algorithm that finds water-tight fragment isosurfaces at over each material volume fraction at simulation run-time, we can find fragments that separate from the original mesh and measure various quantities of interest in these fragments without storing the raw data. The result is a polyhedral mesh surface that contains no gaps and can be many orders of magnitude smaller than the original AMR mesh. In some cases, where an analyst is only concerned with a histogram of fragment quantities, the data written out can be on the order of bytes.

ParaView Coprocessing integration is also being developed for use in LANL simulation codes. One example of this is the LCROSS impact simulation in XRAGE shown in the upper-right image. The visualization and analysis algorithms are being integrated into XRAGE such that the simulation scientist will be able to generate images and plots that are directly specified from the simulation input deck. After the simulation is complete, images and plots, generated by ParaView Coprocessing are immediately available for analysis. This saves simulation time by not writing out raw data to disk and scientist time by not having to read back in the raw data for analysis in the post-processing tool.



OpenMP Backend
Intel Xeon 2.67 GHz CPU (12 cores)
Dimensions: 275 x 317 x 192
Points: 16737600
FPS: 198.7

Figure 6: Screen captures showing PISTON visualization of a large dataset.

D. PISTON

Developing and maintaining a library performance visualization and analysis operators that can take advantage of the multi-core parallelism available in each of the changing set of existing hardware platforms currently requires frequently re-writing algorithms. To address the pressing need for portability, PISTON is being developed to generate visualization and analysis operators that can portably achieve high performance on a wide variety of parallel hardware platforms. In the PISTON programming model, the visualization and analysis implementer is restricted to using only data-parallel primitives, such as scans, transforms, reductions, stream compactions, scatter / gathers, etc., each of which can be implemented efficiently in the backend compiler for the target architectures. This model allows the exact same operator code to compile and run efficiently on any of the many- and multi-core platforms.

Thus far, we have shown high-performance isosurface, cut surface, and threshold operators using PISTON for structured grid data sets. Using NVIDIA's Thrust library, whose backends compile to OpenMP and in CUDA, we have achieved fast, scalable performance on NVIDIA GPUs and on multi-core CPU architectures with the same implementation code for visualization and analysis. For future portability, we are also exploring backends for additional platforms, such as AMD GPUs using OpenCL, and multi-node parallelism through integration with MPI-based distributed-memory parallel libraries, like VTK and DIY. Further improvements include support for unstructured grids and additional operators, like glyphs and portable high-performance, multi-core rendering.

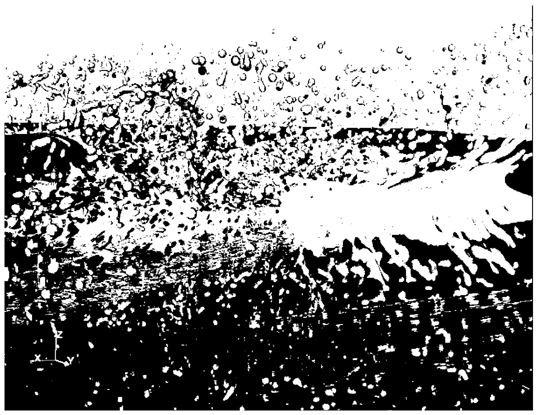


Figure 7: Image rendered using MantaView in ParaView. Note reflections and rendered shadows.

E. Manta View in ParaView

A problem for DOE computational science is managing hardware costs and serving the workflow needs of computational scientists that require large data visualization and analysis. Separate visualization clusters are expensive to buy and maintain, and it would be convenient if visualization and analysis could perform efficiently on the supercomputing platform instead of moving data to a separate resource. Modern supercomputers are currently composed of individual nodes with large core counts and large memory.

Performance of interactive rendering is a computationally intensive portion of the visualization process. For terascale platforms, commodity clusters with graphics processors (GPUs) have traditionally been used for interactive rendering. For petascale platforms, visualization and rendering is able to run efficiently on the supercomputer platform itself. By using the Manta Ray Tracer software from University of Utah, we were able to show that exceedingly large data sets could be interactively rendered on the supercomputer. This is achieved through the Manta rendering plugin in ParaView, which provides a 3D view using the Manta raytracer for rasterization of large datasets on parallel machines.

Over the past two decades, advances in both CPU hardware performance and ray tracing implementations have made interactive ray tracing feasible. In practice, Manta software rendering is faster in than software OpenGL rendering and it is also faster than hardware OpenGL for large triangle counts. For large-scale data on the supercomputer, Manta is used in parallel on individual nodes as the rasterizer, which is composited together using IceT. When rendering only on a single computer, the Manta view has the additional benefit of bringing realistic optical effects such as shadows, translucency and reflection. Software rendering is fast, but also very flexible, because it allows for portable code that can run on nearly any multi-core supercomputing hardware in the future.

IV. ASC AND ASCR COLLABORATION

ASC visualization and analysis research extends beyond the direct relationship with ASC scientists and the Tri-Labs. Frameworks and tools like VisIt and ParaView have had a wide reaching impact to the scientific visualization community. Their role as a research and development vehicle has allowed to research for ASC and ASCR visualization and analysis, and to foster the collaboration between computational and visualization research scientists. Likewise, the research performed by ASCR scientists in visualization and analysis has enabled and augmented ASC research beyond its initial impact. The multiplicative, compounding efforts between ASC and ASCR research results have had, and continue to have, far and wide reaching impact for simulation and visualization science.

A. ParaView and ASCR

Scientists at all the major DOE laboratories and on all DOE supercomputers use ParaView to help them understand challenging and complex problems. ParaView brings visualization to thousands of users (over 7000 downloads per month for the past two years) while maintaining a world-wide open source community with over 100 contributors. ParaView is a multi-institutional project, involving joint efforts between industry (Kitware, Inc.), national laboratories (Los Alamos, Sandia, Argonne, Army Research Laboratory), universities (University of Utah) and international collaborators (Commissariat à l'Énergie Atomique (CEA)). ParaView is in use around the world. Many national agencies and centers (for example, NSF and DOD sites) use ParaView, as it is one of a few tools that can reliably visualize the massive datasets produced by today's large-scale simulations.

In 2006, the SciDAC Institute for Ultrascale Visualization adopted ParaView as its primary deployment platform for petascale visualization research. ParaView is being used by the DOE ASCR community in areas including climate, cosmology, astrophysics, plasma, materials and nuclear engineering, by the DOE NNSA ASC community in the areas of nuclear weapons simulations as well as by the larger community including members of the DoD, NIH, and NSF communities. A testament to the long-reaching scientific impact of ParaView is the strategic decision by two of the three ASCR exascale co-design centers to use Paraview for scalable, in situ data analysis and visualization. ParaView is a primary delivery vehicle for the SciDAC Scientific Data Management, Analysis and Visualization Institute.

B. VisIt and ASCR

VisIt is deployed to many users in both ASC and ASCR. As such, development performed by either organization immediately benefits the other. Collaborations between the development teams range from the mundane, such as bug fixes, library upgrades, porting to new platforms, and release management, to more major initiatives, such as the design of a new system for efficient particle advection in a parallel setting. The new research for large-scale particle advection brings the state-of-the-art in vector field visualization to both ASC and ASCR visualization researchers and scientists. Further developments into large-scale particle advection are currently underway, building upon this research that was developed in VisIt.

The most significant collaborations on VisIt between ASC and ASCR have dealt with support for large data. A formal study occurred in 2009, when a pair of experiments were run to assess and demonstrate VisIt's capabilities for scalability and large data. In the first experiment as seen in the left figure, VisIt's infrastructure and some of its key visualization algorithms were demonstrated to support weak scaling. This demonstration led to be VisIt being selected as a "Joule code," a formal certification process by the US Office of Management and Budget (OMB) to ensure that programs running on high end supercomputers are capable of using the machine efficiently. In the second experiment shown by the right figure, VisIt was scaled up to tens of thousands of cores and used to visualize data sets with trillions of cells per time slice. This study found VisIt itself to scale well, although overall performance was limited by the supercomputer's I/O bandwidth.

C. ViSUS and ASCR

Over the past decade ViSUS has received funding from ASC, ASCR, and numerous other government agencies and has been part of major DoE research efforts such as the Visualization and Analytics Center for Enabling Technology and the Center for Exascale Simulation of Combustion in Turbulence (ExaCT). A different combination of ViSUS data streaming infrastructure and S3D High Performance Computing technologies enabled the real time monitoring of a detailed study of high resolution combustion chemistry supported executed on the HOPPER2 supercomputer in NERSC and rendered on a client at Center for Extreme Data Management Analysis and Visualization in Salt Lake City in the left figure. In fact, scientists can monitor such simulations at any location using a handheld device such as an iPhone, as seen in the middle figure. The right figure shows the full ViSUS pipeline for such remote data streaming and visualization for a Uintah simulation of an explosion within a metal container from simulation code from C-SAFE.

D. *PISTON and Dax*

PISTON is able to provide portability and reasonable performance for the currently implemented visualization and analysis operations, but there are areas in which PISTON can benefit from similar efforts in Dax. First of all, there is no well-defined model for the different kinds of data needed by the scientific visualization and analysis community. This makes it difficult to support new data structures, such as unstructured grids, and for users to easily to write their own operators. Secondly, the execution model in PISTON is currently static. The visualization pipeline is coded as parameters in a C++ template that can only be defined at compile time. This makes building large visualization software, and run-time pipelines, very difficult and inflexible. The research that is being done in Dax can help in addressing these issues.

Where PISTON is a bottom-up approach, finding the best algorithms to run on today's hardware, Dax is a top-down approach, simplifying parallel programming by identifying common algorithm constructs aimed at exascale and by using these constructs to build a flexible visualization toolkit. The danger of Dax's top-down development is the possibility of missing important optimization opportunities that get lost in an intricate toolkit design. The Dax development is currently using PISTON as an exemplar of well-formed and optimized algorithms. Its development is directly benefiting from the research, implementation, and analysis of the PISTON library. Furthermore, PISTON serves as a baseline, without which Dax could not provide evidence that its constructs truly provide efficient programming.

E. *Kitware and ASCR SBIR*

ParaView was originally developed to support the needs of ASC scientists and engineers. As the project progressed, its user community and funding sources diversified to include DOE ASCR, DoD, as well as commercial organizations. Various ASCR SBIR programs have been among the ParaView funding sources.

For example, Kitware originally developed in-situ analysis and co-processing capability with ParaView by leveraging Phase I and Phase II SBIR funding from DoD. This capability is now in use by ASC at Sandia and Los Alamos and the development continues using both ASC and DoD funding. The in-situ analysis capability is also an essential component for the SciDAC SDAV institute and will be leverage and improved through ASCR funding. Furthermore, Kitware was recently awarded a DOE ASCR Phase I SBIR for the in-situ analysis of cosmological simulations. As part of this project, Kitware will collaborate with scientists from ANL to deploy in-situ analysis in simulations codes developed at LANL and ANL.

As part of another Phase II SBIR from DOE ASCR, Kitware has been developing capabilities to efficiently analyze AMR datasets within ParaView. This project includes demand-driven execution to minimize IO as well as remote streaming of AMR datasets to provide interactive analysis capabilities. It has been applied to Enzo and Flash output, in use by DOE ASCR, as well as Sandia's CTH in use by DOE ASC.

F. *Distance Visualization and In Situ*

The distance visualization project, funded by ASCR, improves the state-of-the-art for analysis and visualization of large-scale simulation data at remote supercomputing sites. Remote visualization, even at local supercomputing sites, is a difficult challenge because domain scientists have limited interaction and ability to visualize and analyze large-scale simulation data. This is due to the network performance bottlenecks associated with moving data. This distance visualization project has developed methods for remote multi-resolution visualization, quantified data compression, and statistical sampling techniques for large-scale remote data to manage data movement.

These distance techniques are currently being researched and deployed in ParaView. The development of ParaView as a stable research and production tool allows for fast R&D for the distance visualization project and rapid deployment to end-user scientists, in both ASC and ASCR. The ASCR distance visualization techniques are also contributing directly back to ASC research in situ visualization. Important components are methods for simulation bandwidth reduction and data error quantification that are influenced by ASCR research for large-scale data movement over a distance and scaling down data for wide-area networks.

V. CONCLUSION

ASC visualization and analysis research has had a significant impact on large-scale simulation science and continues to do so, by pushing the edge of visualization and analysis research. Early successes, like ParaView and VisIt, have had long-term and wide reaching impact that are enabling scientific discovery at scale. Without visualization and analysis research into these and other production tools, progress in simulation science would have been severely hindered. ASC research continues with the preparation for exascale supercomputing and the related challenges for simulation science. Research into portable codes, frameworks, and in situ analysis will enable scientific discovery and insight at scale. ASC research is further advanced by its close relationship to fundamental research by ASCR scientists. The close collaboration between ASC and ASCR scientists and research achieves accelerated results in visualization and analysis, resulting in direct tangible impact for enabling simulation science and discovery.