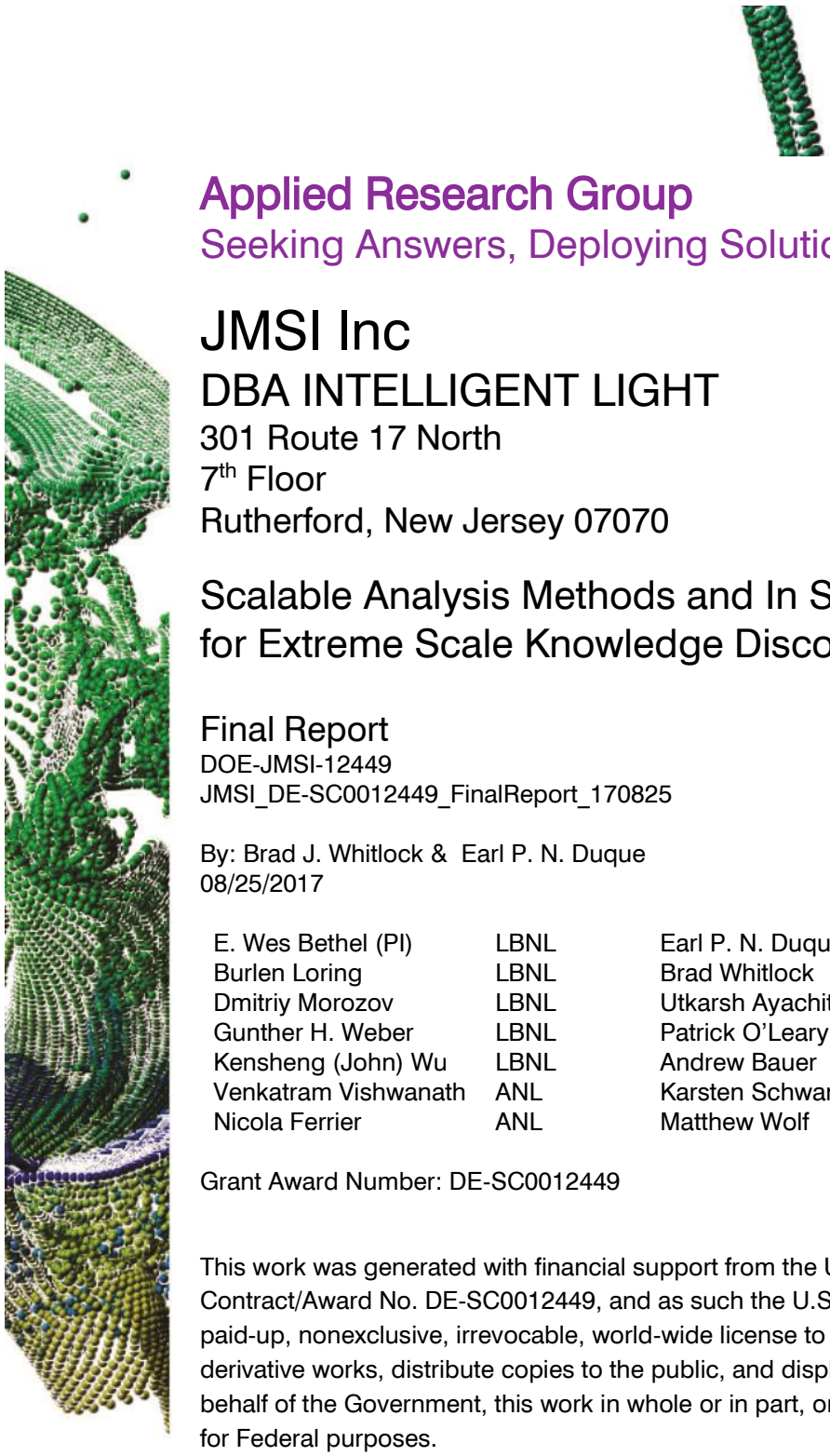

Applied Research Group
Seeking Answers, Deploying Solutions

JMSI Inc
DBA INTELLIGENT LIGHT
301 Route 17 North
7th Floor
Rutherford, New Jersey 07070

Scalable Analysis Methods and In Situ Infrastructure for Extreme Scale Knowledge Discovery

Final Report
DOE-JMSI-12449
JMSI_DE-SC0012449_FinalReport_170825

By: Brad J. Whitlock & Earl P. N. Duque
08/25/2017

E. Wes Bethel (PI)	LBNL	Earl P. N. Duque (co-PI)	Intelligent Light
Burlen Loring	LBNL	Brad Whitlock	Intelligent Light
Dmitriy Morozov	LBNL	Utkarsh Ayachit	Kitware, Inc.
Gunther H. Weber	LBNL	Patrick O'Leary	Kitware, Inc.
Kensheng (John) Wu	LBNL	Andrew Bauer	Kitware, Inc.
Venkatram Vishwanath	ANL	Karsten Schwan *(deceased)*	Georgia Tech
Nicola Ferrier	ANL	Matthew Wolf	ORNL

Grant Award Number: DE-SC0012449

Table of Contents

Executive Summary

High performance computers have for many years been on a trajectory that gives them extraordinary compute power with the addition of more and more compute cores. At the same time, other system parameters such as the amount of memory per core and bandwidth to storage have remained constant or have barely increased. This creates an imbalance in the computer, giving it the ability to compute a lot of data that it cannot reasonably save out due to time and storage constraints. While technologies have been invented to mitigate this problem (burst buffers, etc.), software has been adapting to employ in situ libraries which perform data analysis and visualization on simulation data while it is still resident in memory. This avoids the need to ever have to pay the costs of writing many terabytes of data files. Instead, in situ enables the creation of more concentrated data products such as statistics, plots, and data extracts, which are all far smaller than the full-sized volume data.

With the increasing popularity of in situ, multiple in situ infrastructures have been created, each with its own mechanism for integrating with a simulation. To make it easier to instrument a simulation with multiple in situ infrastructures and include custom analysis algorithms, this project (funded under grant award number: DE-SC0012449) created the SENSEI framework [1]. SENSEI increased portability, making it easier to develop in situ analysis algorithms that are *"write once and run anywhere"* and to identify in situ design patterns. The SENSEI framework consists of a simulation data adaptor, which lets simulations expose their data as VTK [2] data objects which are then passed into SENSEI. From there, the SENSEI framework passes data into simulation infrastructure adaptors which expose the data to other in situ infrastructures such as VisIt's Libsim [3] or Paraview/Catalyst [4]. The SENSEI framework was used to instrument simulation codes such as AVF-LESLIE [5] to perform large simulation runs on DOE HPC systems.

The project is a benefit to the public because it helps to address the problem of lost science on extreme-scale HPC computer systems. When simulations run at scale, data are extremely large and scientists are forced to make choices that limit the data they can save. This usually means that data are saved too infrequently, increasing the likelihood that simulated phenomena will be incomplete or missing from the saved results. This can lead to lost science. With SENSEI making it easier to instrument simulation codes for in situ, the barrier to using in situ is lowered and simulations can save useful data products and extracts more frequently, producing useful data that are vastly smaller compared to volume data. The creation of the SENSEI framework has also driven improvements in the in situ infrastructures to which SENSEI has been coupled, further reducing the overhead for in situ. For example, to improve sharing simulation data with non-contiguous memory layouts, the VTK library was enhanced with a data array wrapper interface that improves efficiency of memory accesses for non-contiguous data layouts. These improvements were exposed in both Paraview/Catalyst and VisIt's Libsim to make simulation adaptors less likely to make copies of memory. Taken together, the software generated by this project can improve the quality of the scientific data created from large simulation campaigns.

Intelligent Light 301 Route 17 North, 7th Floor, Rutherford, NJ 07070, USA +1 (201) 460-4700

Accomplishments

In addition to JMSI, the DOE-SC0012449 project included participants from Lawrence Berkeley National Laboratory, Argonne National Laboratory, Oak Ridge National Laboratory, Georgia Tech, and Kitware Inc. The overall project goals included the creation of an overarching in situ infrastructure that could be applied to analysis within simulations run at extreme scale on HPC systems. JMSI was primarily active in design and implementation of the SENSEI in situ framework and integrating it with VisIt's Libsim. JMSI also used SENSEI to instrument the AVF-LESLIE combustion simulation code and lead a combustion study. Table 1 contains the subset of project milestones from the original proposal document where JMSI had responsibility.

Table 1 Milestones and associated status/comment

Milestone	Year	Description	Status / Comments
Normalizing Data Access	Y1	Design the common interface, views and iterators for structured data to map between a simulation, all in situ infrastructures, and one relevant analysis pipeline.	**Complete.** A prototype SENSEI interface based on the VTK data model was created and used to compute a histogram in a mini-app.
	Y2	Extend to all select analysis pipelines and relevant simulations, incorporate temporal information. Design the in situ-infrastructure specific translation/bridging for portable deployment.	**Complete.** SENSEI applied to autocorrelation oscillator mini-app. Data access extended to zero-copy of non-contiguous memory in SENSEI and Libsim.
	Y3	Refine and deploy with the select analysis on all three DOE supercomputing architectures.	**Complete.** SENSEI applied to large scale runs on Mira, Titan, and Cori.
Normalizing In Situ Infrastructure Interfaces	Y1	Develop interfaces for analysis pipelines to query resource allocations and constraints from the in situ infrastructures and express resource and execution requirements. Evaluate with one analysis algorithm with all in situ infrastructures.	**Complete**. Resource querying implemented in prototype library and used to convey resource requirements.
	Y2	Refine the interface and evaluate with additional select analysis on all in situ infrastructures.	**Complete**. AVF-LESLIE runs on Titan.
	Y3	Refine the interface and evaluate with additional select analysis on all in situ infrastructures.	**Complete**. AVF-LESLIE runs on Cori / KNL.
Project-wide Milestones	Y1	Whole-team co-design of in situ processing code and normalized interface, data access.	**Complete.** JMSI helped design and create SENSEI library.

Intelligent Light 301 Route 17 North, 7th Floor, Rutherford, NJ 07070, USA +1 (201) 460-4700

Milestone	Year	Description	Status / Comments
	Y2	In Situ processing codes implement normalized data access and interface, operate in resource-constrained environment (e.g., low memory), take advantage of staging and platform-specific features where appropriate.	**Complete**. JMSI used SENSEI to instrument AVF-LESLIE code and ran at 131K cores on ORNL's Titan. Implemented extract and image-based workflow.
	Y3	Project-wide code repos/distribution for in situ analysis algorithms, code for normalized interface and data access implemented in distros for the four in situ infrastructures.	**Complete**. Project code repo exists at https://gitlab.kitware.com/sensei/sensei

Project Activities

JMSI was involved with 2 main areas of the project: the SENSE framework and the AVF-LESLIE combustion studies.

SENSEI Framework

Instrumenting a code to use in situ is hard. A simulation has to be modified to call an in situ infrastructure such as VisIt's Libsim, Paraview/Catalyst, or ADIOS [6] and supply its data to the infrastructure via a data adaptor. Each of these infrastructures has its own ways of exposing and processing simulation data, requiring a different data adaptor for each infrastructure. One of the goals of SENSEI was to make it possible to instrument a simulation once and expose the simulation's data to multiple simulation infrastructures or custom analysis routines. This lowers the barrier to using in situ by imposing a common in situ interface to expose data and to invoke analysis routines. This regularity lets the developer create custom analyses that can plug into the SENSEI framework. SENSEI achieves these goals by using a data model to communicate from the simulation to the SENSEI analysis modules and external in situ infrastructures (see Figure 1). Simulations largely compute on mesh-based data structures which are themselves made of various data arrays. VTK already provided a rich data model that could express many of the mesh data structures we would encounter in simulation codes – and VTK already forms a substrate for tools such as ParaView and VisIt. The SENSEI framework adopted VTK as the data model since it was well-known and generally suitable for this purpose. By introducing code that exposes simulation data arrays as VTK meshes with

Intelligent Light 301 Route 17 North, 7th Floor, Rutherford, NJ 07070, USA +1 (201) 460-4700

data, these high-level constructs can be passed to a large number of existing analysis routines and the data are easy to pass to a variety of in situ infrastructures.

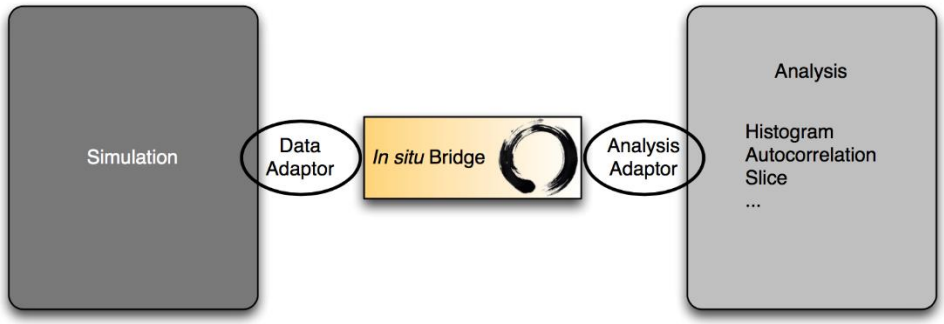

Figure 1 SENSEI Diagram

VTK provides C++ classes that provide a uniform method for storing and accessing memory. The C++ classes are specialized to store certain native data types such as integers, floating-point numbers, double-precision numbers, etc. The data array interface is mostly concerned with storing contiguous arrays of data, though it can be used to access existing memory or even data types that are not contiguous in memory, as one might find in simulations that employ and *array-of-structures* data layout. To handle non-contiguous memory, VTK provides the vtkMappedDataArray class, which provides an interface for derived classes to access custom memory layouts. The vtkMappedDataArray class requires developers to subclass it using C++ inheritance and define a large set of functions that can access data in the custom memory layout. This makes the mapped array difficult to use and inefficient. We designed a more efficient replacement for vtkMappedDataArray that supports *array-of-structures* and *structure-of-array* data layouts.

In order to provide support for multiple in situ infrastructures and custom data analyses, SENSEI needed to provide a uniform way for calling analysis functions. This uniformity was achieved by creating analysis adaptor classes that interface to specific in situ infrastructures. In the current implementation, analyses are selected using an XML file that also contains any parameters that are to be passed along to an analysis adaptor. The SENSEI framework iterates over the selected analyses, passing the VTK representation of the simulation data to each analysis. Analysis adaptors have been created for histogram creation and for general in situ infrastructures such as ADIOS, Catalyst, and Libsim (see Figure 2).

Intelligent Light 301 Route 17 North, 7th Floor, Rutherford, NJ 07070, USA +1 (201) 460-4700

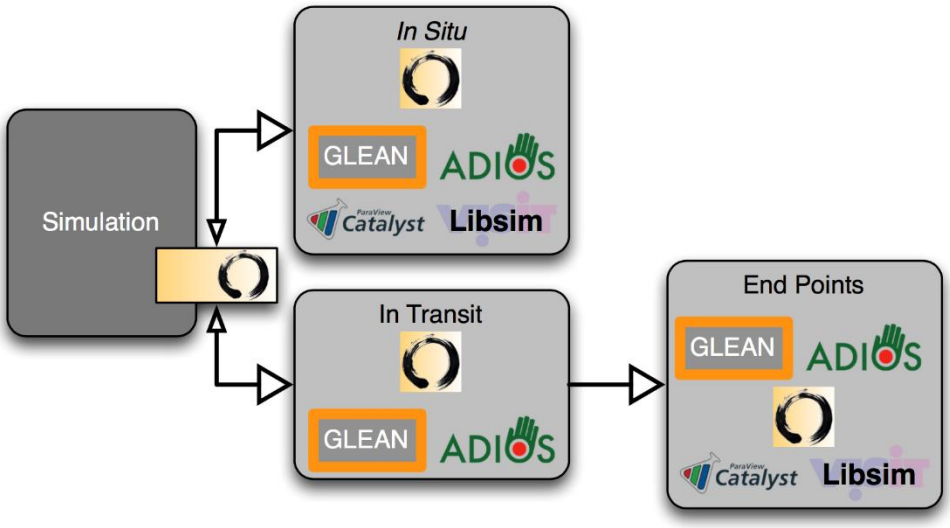

Figure 2 SENSEI as an interface to multiple in situ infrastructures

Combustion

The desire to scale simulation environments to be able to overlap with real-world experimental environments—in terms of physical size and temporal scale— has been a key component in the progression towards exascale. For many systems, however, one or both of these remains a problem. In the case of the intersection between computational and experimental evaluation of industrial-scale combustion systems, it has recently become possible to begin to directly overlap the two. A key difficulty is in determining whether the flame dynamics in the simulation accurately represent the dynamics of an experiment. In addition to the inherently chaotic nature of turbulent combustion, which guarantees that no point-by-point comparison will be possible, there are also issues related to uncertainty in the underlying chemical models and parameters that may cause the simulation and experiment to diverge.

The LESLIE3D code is used in design studies for gas turbine combustors and rocket engines. The simulations solve large eddy simulations of complete real geometry with chemistry and Lagrangian droplets for liquid to gas phase changes and combustion.

The investments from the SENSEI project allow us to impact a wide array of application science stakeholders from a spectrum of industries, labs, and universities including the following: Georgia Tech, United Technologies Research Center, Rocketdyne, Aerojet, Wright-Patterson AFB, Edwards AFB, NASA Glenn, Pratt-Whitney and Solar Turbines.

AVF-LESLIE is written in FORTRAN90 and instrumented for in situ using VisIt/Libsim. An adaptor layer was created to interface Libsim calls with AVF-LESLIE. The adaptor layer contains routines to read an input database that specifies all of the control parameters for Libsim as well as the desired list of data extracts. The adaptor layer hides many of the details of interfacing with Libsim, exposing only a few high-level co-processing functions. Internally, the adaptor passes simulation geometry and solution data to Libsim, which generates the extract surfaces and

Intelligent Light 301 Route 17 North, 7th Floor, Rutherford, NJ 07070, USA +1 (201) 460-4700

saves them to XDB files for secondary post-processing with FieldView. AVF-LESLIE was modified to call the high-level adaptor functions and timers were added to measure the in situ data extraction performance.

Later, when SENSEI started to mature, we changed AVF-LESLIE again to instrument it with a SENSEI data adaptor that calculates vorticity magnitude and exposes data array slices (to remove ghost cells). The SENSEI analysis adaptor for Libsim was provided a VisIt session file to set up the visualization. The visualization consists of 3 isosurfaces and 3 slice planes of vorticity magnitude. Its purpose is to give a visual reference to the evolution of the turbulent flow features from initial mixing through homogenous turbulence. Each time the analysis was called, the VisIt plots were generated using the updated simulation data and images were saved.

We conducted benchmarks on Titan at Oak Ridge Leadership Class Compute Facility. The scaling studies were performed on a Cartesian grid size of 1025^3 and physical non-dimensional domain sizes of $4\pi \times 4\pi \times 2\pi$. The study used between 8192 and 131,072 cores, using all 16 cores per compute node. The benchmarks study the strong-scaling characteristics of AVF-LESLIE, and the memory and computational overhead associated with the in situ methods and infrastructure.

Each run represents AVF-LESLIE running for 100 time steps, with SENSEI being called at each time step and Libsim analysis every 5 time steps. The study simulates unsteady dynamics of a temporally evolving planar mixing layer (TML). This type of fundamental flow mimics the dynamics encountered when two fluid layers slide past one another and is found in atmospheric and ocean fluid dynamics as well as combustion and chemical processing. The two sliding fluid layers are subject to inviscid instabilities and can evolve from largely 2-d laminar flow into fully developed, 3-d homogenous turbulent flow. Figure 3 presents visualizations of the TML flowfield at 10,000, 50,000,

Intelligent Light 301 Route 17 North, 7th Floor, Rutherford, NJ 07070, USA +1 (201) 460-4700

100,000 and 200,000 time steps where the flow evolves from the initial flow field, vortex braids begin to form, wrap and then the flow breaks down leading to homogenous turbulence, respectively.

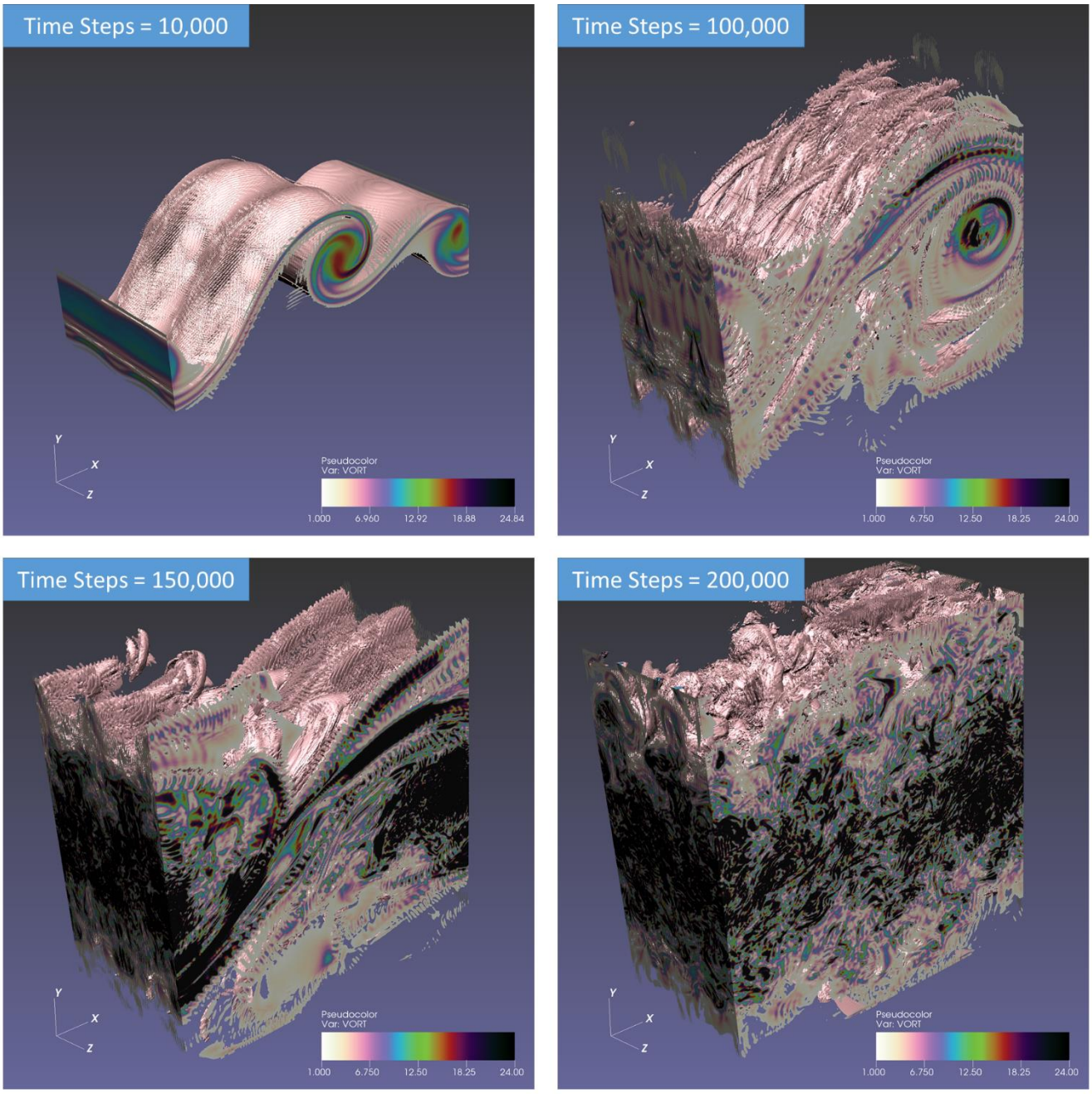

Figure 3 The Evolution of Temporal Mixing Layer from Initial to Vortex Breakdown

Intelligent Light 301 Route 17 North, 7th Floor, Rutherford, NJ 07070, USA +1 (201) 460-4700

Before in situ processing was implemented, AVF-LESLIE scaled well up to 16k cores, but efficiency degrades at higher core counts. After SENSEI/Libsim in situ rendering was added, the per-iteration time for AVF-LESLIE increased due to the time taken for in situ processing. The time taken to initialize SENSEI increases with processor count, largely due to one-time Libsim initialization costs.

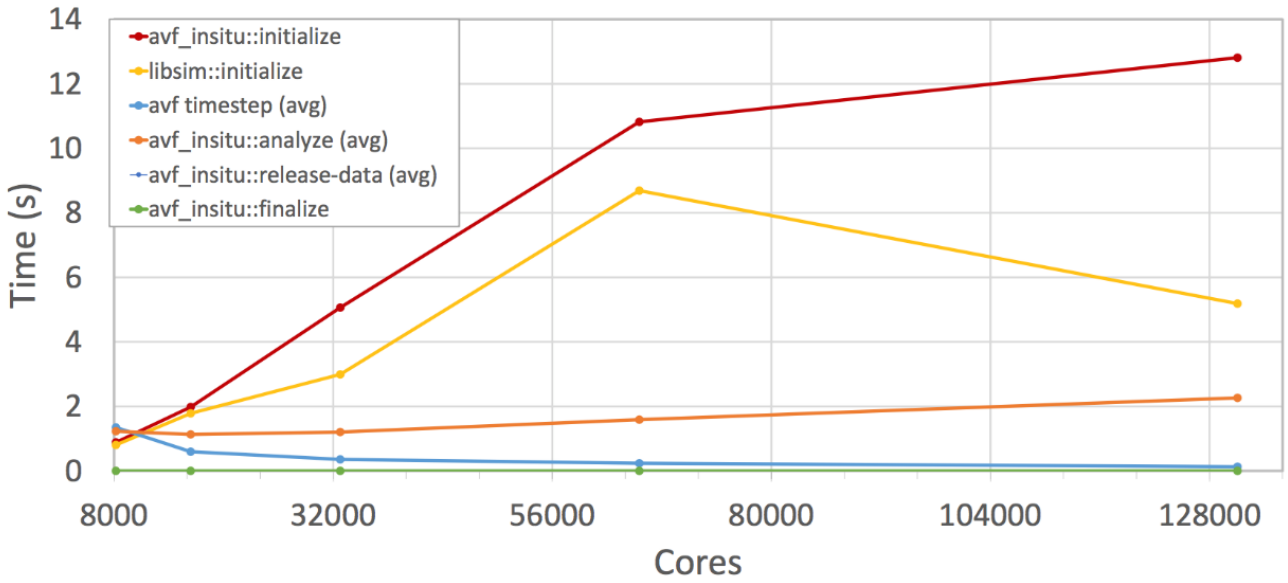

Figure 4 AVF-LESLIE Performance with SENSEI/Libsim In Situ Processing

The analysis time which includes the time to expose data to SENSEI, read a session file, set up plots, perform data extraction, render geometry, create the composited image, and save the image, quickly exceeded the time spent in the solver due to the complexity of the visualization, as shown in Figure 4. In Situ analysis time is highly dependent on the complexity of the analysis, or in this case, the nature of visualizations being produced. Over the 100 time step run, the costs of calling Libsim to produce rendered images added an average of 1-1.5 seconds per time step to the solver runs over the numbers of cores tested.

Since the Libsim visualization was complex, it was executed one out of every 5 times in which SENSEI was invoked by the solver. This means that 4/5 times, the SENSEI analysis time was low and the 1/5 times that Libsim analysis was invoked, the time was high. To see the actual costs of calling Libsim to produce the visualizations, AVF-LESLIE reported the time spent in SENSEI analysis for each time step. Figure 5 shows for the 65K run that the cost of generating the images via Libsim is in the range of 7-8 seconds while the normal SENSEI overhead for the data adaptor is less than 0.5 seconds, showing that SENSEI overhead is low and analysis overhead can be arbitrarily high, depending on the requested operations.

It is important to contrast the in situ overhead to the traditional post hoc workflow. At 1025^3 and 65K core, AVF requires approximately 24 seconds to save a time step of volume data. Therefore, based upon the current overhead numbers, one can afford 3-4 times greater temporal resolution for visualization and analysis in comparison to writing out volume data for post hoc processing. This capability is very important for transient data

Intelligent Light 301 Route 17 North, 7th Floor, Rutherford, NJ 07070, USA +1 (201) 460-4700

and resolving turbulent flow characteristics.

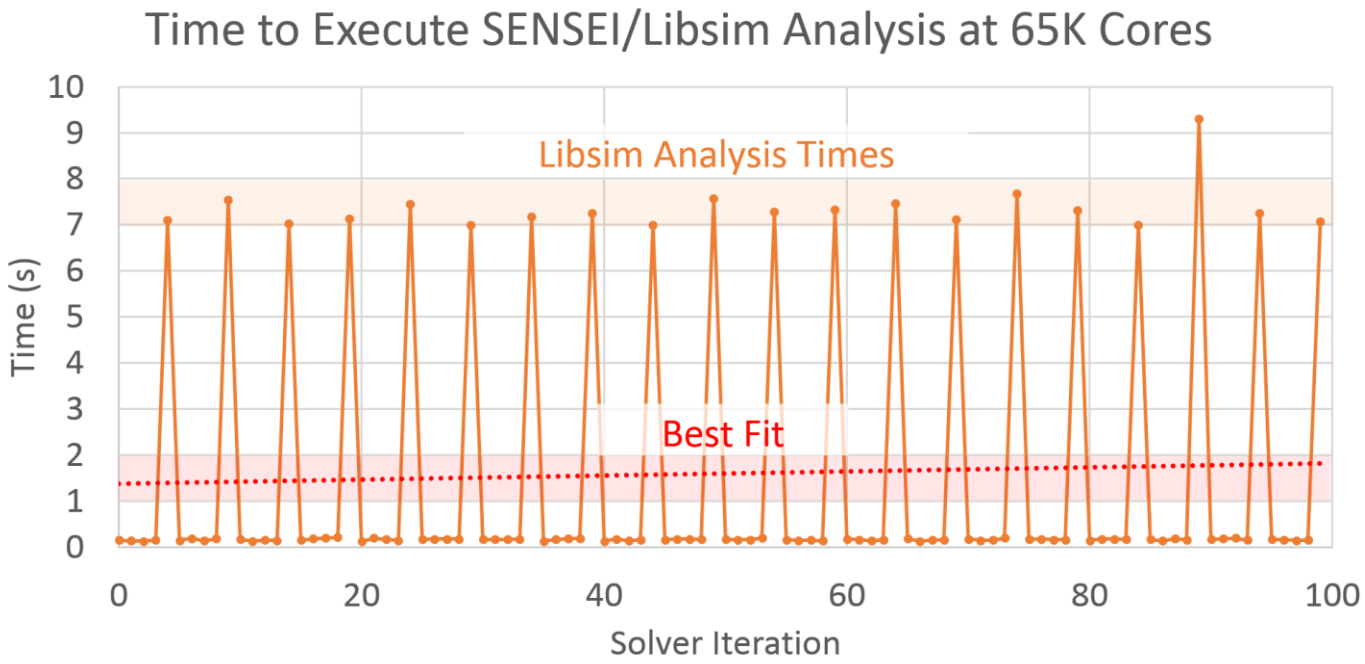

Figure 5 Per Iteration cost for calling SENSEI at 65K when Libsim analysis pipeline is invoked every 5 time steps.

VisIt Static Builds

The extreme scale runs for AVF-LESLIE took place on the Titan and Cori computers, which are both Cray machines. Early runs using VisIt and Libsim built in its typical shared library configuration yielded very poor start up times due to stress on the file system as Libsim runtime libraries and VisIt plugins were loaded dynamically at runtime. The overhead for loading the libraries on Cori could be as much as 40 minutes, quickly exhausting the time the job had for actual work. To combat these problems, the VisIt build system was given some updates. The VisIt build system already had some support for building VisIt as a set of static executables on Linux, yet more attention was needed for Cray to ensure the creation of statically-built binaries and Libsim libraries. The VisIt build is normally handled by a script called *build_visit*, which builds all of the VisIt 3rd party dependencies and then VisIt itself. It turned out that changes were needed in a few areas. First, a newer version of Mesa3D [7] was needed to get VisIt's offscreen rendering to work. Rather than dynamically swapping in Mesa3D in place of OpenGL, build_visit on Cray was modified to use Mesa3D as the OpenGL for both VTK and for VisIt to prevent runtime rendering failures. Second, build_visit was enhanced so it can reliably detect and pass on MPI settings to the VisIt build. A number of other smaller enhancements were made to build_visit to make sure that it can produce a working static build of VisIt and the Libsim runtime library. When used at scale, the statically linked executables using these libraries take a matter of seconds to load and initialize.

Intelligent Light 301 Route 17 North, 7th Floor, Rutherford, NJ 07070, USA +1 (201) 460-4700

Libsim Zero-copy

The team took the concepts explored for data access via the VTK vtkMappedDataArray class and wrote a derived class for Libsim that implements the vtkDataArray interface. The new class enables tuple-level access to array data as usual, yet the data for each tuple can come from different, scattered memory locations. The new class also permits components of multi-component data such as vectors to be assembled from different scalar arrays. A tuple-level view is provided dynamically without making problem-sized copies of the data. These features enable the new memory class to provide views of data that look contiguous but may in fact come from data stored in various forms: structure of array data, array of structure data, and coordinate fields which come from separate x,y,z data arrays.

Libsim was enhanced with new functions that extend its VariableData object so that data can be provided to it with a stride and offset, enabling support for array of structure data. VariableData objects with multiple components can now be formed from multiple sub-arrays, which can each have a different memory layout. These Libsim improvements enable simulations to provide data and memory traversal hints to VisIt so VisIt can operate on non-contiguous data without first having to copy the data into VTK's preferred contiguous formats. These features appeared in VisIt 2.10.

Libsim Rendering

During preliminary runs on Oak Ridge's Titan computer, we identified rendering bottlenecks in VisIt's scalable rendering code as we executed a SENSEI/Libsim analysis that would generate images from isosurface plots of vorticity. The bottlenecks stemmed mainly from code within Libsim's runtime library that broadcast an image to all MPI ranks after successful compositing. The composting stage of scalable rendering assembles partial images from different MPI ranks into a single image with contributions from other MPI ranks. VisIt uses the IceT [8] library from Sandia to implement one of its image compositing pathways. VisIt's use of IceT has been shown previously to exhibit scaling problems around 10K processors using parallel Libsim simulations that output rendered images. We ported VisIt to IceT 2.1 first in hopes that would increase efficiency. During the port, we located the cause of some significant scaling problems in how VisIt was sending the composited back out to all MPI ranks. This is sometimes

Intelligent Light 301 Route 17 North, 7th Floor, Rutherford, NJ 07070, USA +1 (201) 460-4700

necessary when rendering images that have shadow maps so we were able to resolve part of the scaling problem by only broadcasting images when shadows are needed.

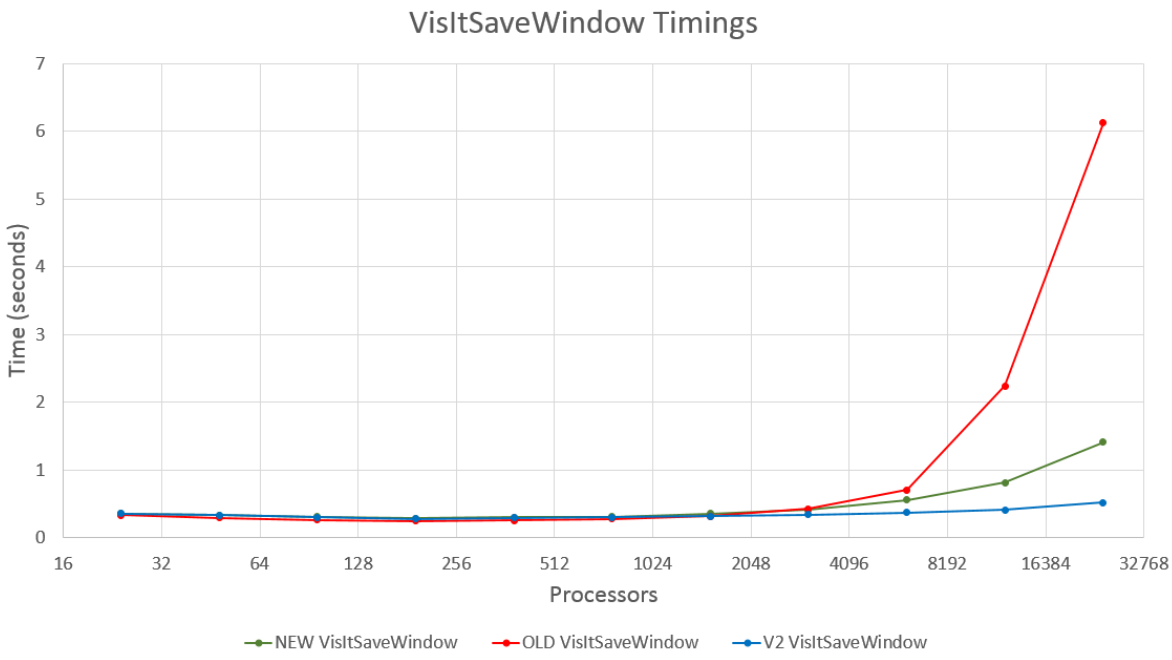

Figure 6 VisItSaveWindow Timings

We ran a weak scaling study using the scalable rendering benchmark script on the Edison machine at NERSC. Edison is a Cray with 2, 12-core Intel "Ivy Bridge" processors at 2.4 GHz and 64 GB of memory per node. The core count was varied from 24 processors to 24K processors. The timings for the benchmark's calls to the Libsim function *VisItSaveWindow()* were gathered. This call includes rendering, compositing, and time spent saving the image to PNG format. The scaling study indicated that the bulk of the scaling problem was solved by not broadcasting the image for shadow rendering. However, there was still a lesser scaling problem. Analysis of the timing logs showed the new problem happened in several places inside VisIt where the offscreen rendering window was resized several times during each render, causing large releases and reallocations of image buffer memory. These problems were corrected and the scaling study was run again to show that VisIt's rendering/compositing is now far more scalable (see Figure 6). At 24K processors, the improved code is over 11x faster.

Write Groups for XDB Export

The combustion simulation experiments with AVF-LESLIE and in situ extract generation had revealed a performance bottleneck above 10,000 cores due to geometry aggregation to rank 0 in the VisIt XDB exporter. I/O

Intelligent Light 301 Route 17 North, 7th Floor, Rutherford, NJ 07070, USA +1 (201) 460-4700

experiments on Edison confirmed that writing separate files is the fastest method for writing files up to a point. A cross-over is reached where it becomes favorable to have groups of MPI ranks write data instead of either writing all files independently. VisIt was extended with the notion of write groups, which enable it to partition its set of MPI ranks into smaller groups, which each aggregate geometry among their group members and then write separate XDB files (see Figure 7). Subsequent tests of in situ extracts with AVF-LESLIE showed the method to be scalable out to 62K cores, with the time spent to extract and write XDB files remaining nearly flat across a range of scales. These features appeared in VisIt 2.10.

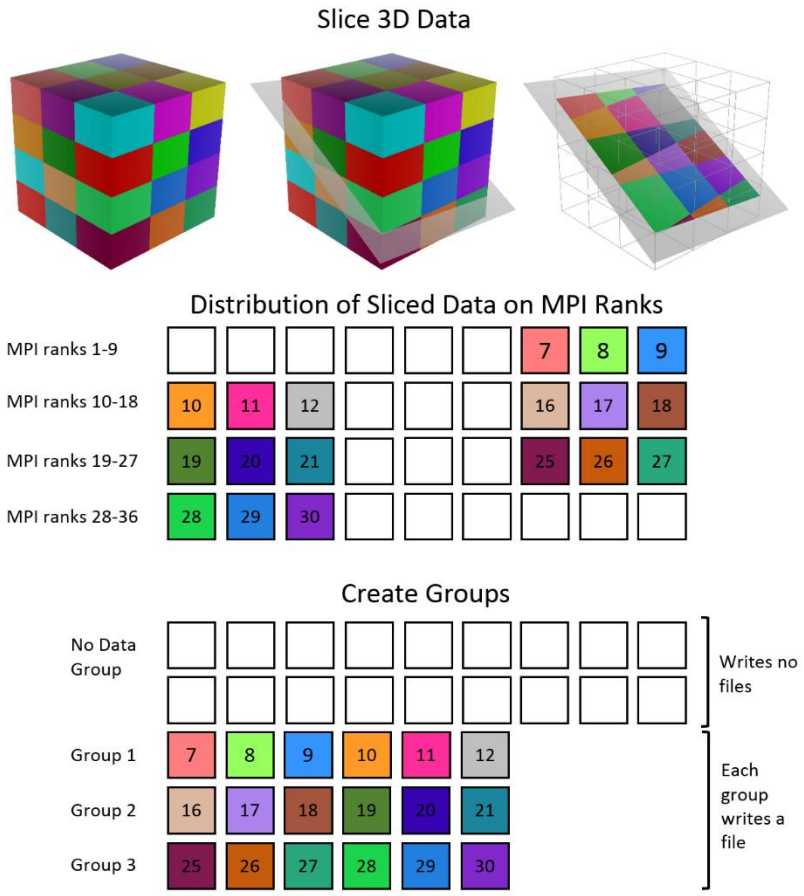

Figure 7 Approach to writing data using write groups

AVF reader for VisIt

The AVF-LESLIE simulation can write files in Xdmf format [9], in which each MPI rank in the parallel job writes its own HDF5 file with an associated XML file that describes the contents of the HDF5 file in terms of visualization objects. Then there is an overarching XML file that contains pointers to the XML files for each MPI rank. VisIt contains a file reader plugin that can read the Xdmf format and reconstruct meshes and fields from the stored data so they may be visualized. Using the VisIt Xdmf plugin on a large dataset consisting of tens of thousands of files proved to be too slow due to the Xdmf library reading and parsing all of the XML description files. We needed a faster way to read the AVF-LESLIE data so we created a new *"AVF"* reader for VisIt that reads the first XML file and then makes an assumption that the rest will be similar, avoiding the need to read them all. The new reader can then directly process the HDF5 data files without having to use the XML schema files. Since this results in far fewer file accesses, the AVF reader greatly speeds up the reading process, enabling VisIt to efficiently process data from AVF-LESLIE datasets split among tens of thousands of files. The AVF reader also provides extra hints to VisIt so it can automatically construct ghost data across domain boundaries, which are

Intelligent Light 301 Route 17 North, 7th Floor, Rutherford, NJ 07070, USA +1 (201) 460-4700

needed to eliminate artifacts at the domain boundaries (see Figure 8).

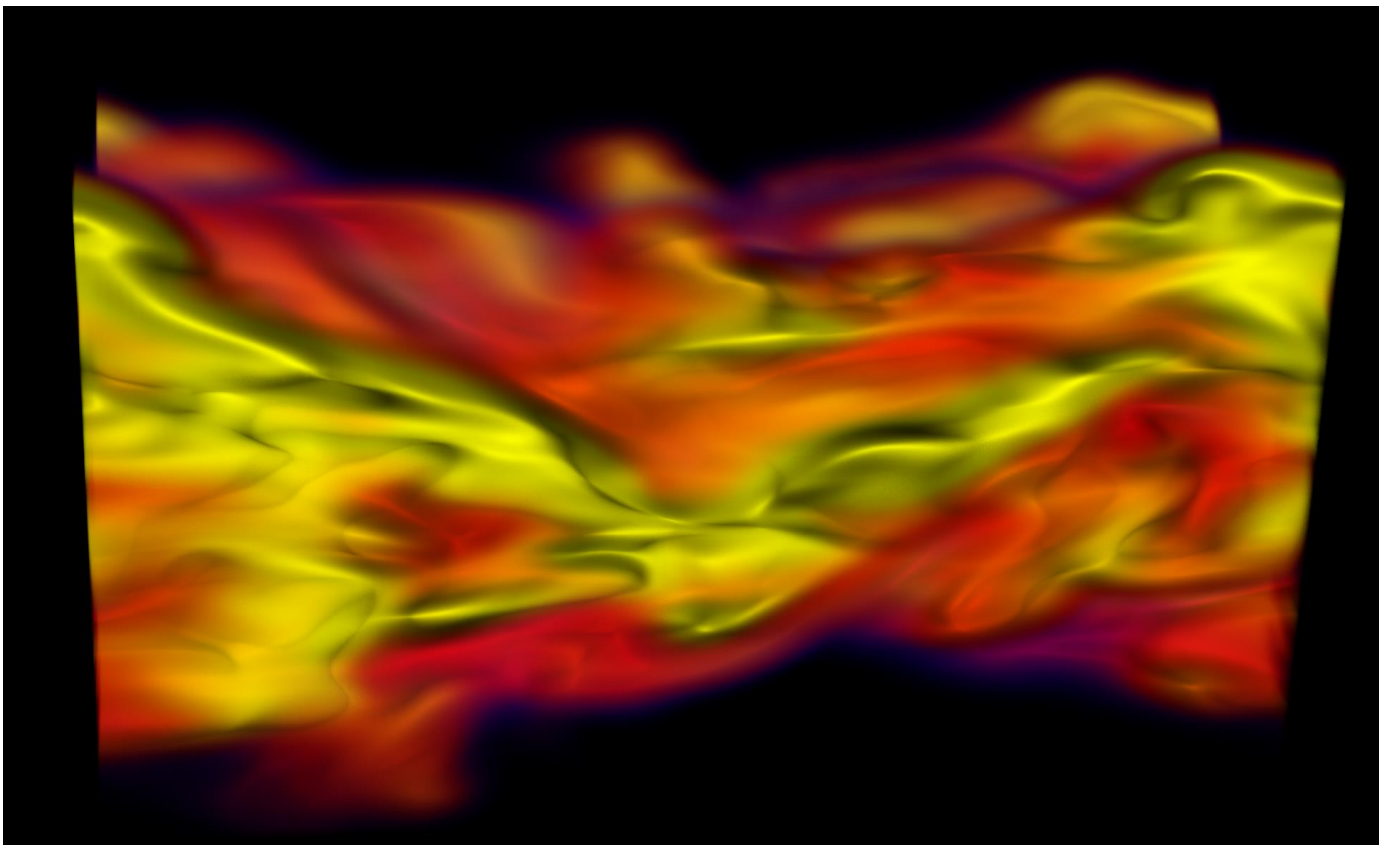

Figure 8 VisIt volume rendering from large AVF-LESLIE combustion dataset (no artifacts).

Products

Publications

The following publications feature JMSI's Brad Whitlock and Earl P. N. Duque.

Peer-reviewed Journal Articles

Brad J. Whitlock, Earl P. N. Duque, *"In Situ Visualization and Production of Extract Databases",* Supercomputing Innovations and Frontiers, vol. 3, no. 4 (2016), pp. 19-29, DOI: 10.14529/jsfi160402

Peer-reviewed Conference Papers

Utkarsh Ayachit, Brad Whitlock, Matthew Wolf, Burlen Loring, Berk Geveci, David Lonie, and E. Wes Bethel. The SENSEI Generic in Situ Interface. In Proceedings of the 2nd Workshop on In Situ Infrastructures for Enabling Extreme-scale Analysis and Visualization, ISAV '16, pages 40–44, Piscataway, NJ, USA, November 2016. IEEE Press.

Intelligent Light 301 Route 17 North, 7th Floor, Rutherford, NJ 07070, USA +1 (201) 460-4700

Utkarsh Ayachit, Andrew Bauer, Earl P. N. Duque, Greg Eisenhauer, Nicola Ferrier, Junmin Gu, Kenneth E. Jansen, Burlen Loring, Zarija Lukĭ c, Suresh Menon, Dmitriy Morozov, Patrick O'Leary, Reetesh Ranjan, Michel Rasquin, Christopher P. Stone, Venkat Vishwanath, Gun- ther H. Weber, Brad Whitlock, Matthew Wolf, K. John Wu, and E. Wes Bethel. Performance analysis, design considerations, and applications of extreme-scale in situ infrastructures. In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC '16, pages 79:1–79:12, Piscataway, NJ, USA, 2016. IEEE Press.

Andrew C. Bauer, Hasan Abbasi, James Ahrens, Hank Childs, Berk Geveci, Scott Klasky, Kenneth Moreland, Patrick O'Leary, Venkatram Vishwanath, Brad Whitlock, and E. Wes Bethel. In Situ Methods, Infrastructures, and Applications on High Performance Computing Platforms, a State-of-the-art (STAR) Report. In Computer Graphics Forum, Proceedings of Eurovis 2016, Groningen, Netherlands, June 2016. Wiley-Blackwell.

Brad J. Whitlock, Steve M. Legensky, and Jim Forsythe. *"In Situ Infrastructure Enhancements for Data Extract Generation"*, 54th AIAA Aerospace Sciences Meeting, AIAA SciTech, (AIAA 2016-1928).

Earl P.N Duque, Brad J. Whitlock, Steve M. Legensky, Christopher P. Stone, Reetesh Ranjan, and Suresh Menon. The impact of in situ data processing and analytics upon scaling of CFD solvers and workflows. In Proceedings of the 27th International Conference on Parallel CFD (ParCFD 2015), 2015.

Web Sites

The project produced the following web sites:

- *http://www.sensei-insitu.org/*
- *https://gitlab.kitware.com/sensei/sensei*

Networks / Collaborations

JMSI worked closely with the project organizations to develop the SENSEI software library and execute large scale runs on several HPC systems such as ORNL's Titan, NERSC's Cori, and ANL's Mira. JMSI is a frequent contributor to the open source VisIt project. Several of the efficiency improvements that reduce bottlenecks in VisIt's Libsim in situ infrastructure were contributed back to the open source project at LLNL. Other JMSI collaborations are listed in Table 2.

Table 2 Networks and Collaborations

Organization	Category	Description
LBNL	collaboration	JMSI worked closely with Wes Bethel and Burlen Loring from LBNL when developing the SENSEI software.

Intelligent Light 301 Route 17 North, 7th Floor, Rutherford, NJ 07070, USA +1 (201) 460-4700

Kitware	collaboration	JMSI worked closely with Kitware's Utkarsh Ayachit on designing the SENSEI software's library interface.
ORNL	collaboration	JMSI worked with Matthew Wolf from ORNL to execute some science runs on ORNL's Titan computer using the AVF-LESLIE software.
Georgia Tech	collaboration	JMSI worked with Suresh Menon's group at Georgia Tech on the AVF-LESLIE combustion code. JMSI instrumented AVF-LESLIE with Libsim and SENSEI and executed several large scale runs.
US Navy NAVAIR	In situ outreach	JMSI has supported Jim Forsythe from US Navy NAVAIR as he integrated VisIt/Libsim to add in situ XDB extract generation into CREATE-AV Kestrel, a solver for fixed-wing and rotor-based aircraft.
JAXA	In situ outreach	JMSI has been providing support to Dr. Seiji Tsutsumi (JAXA) who is looking to instrument UPACS with Libsim,
RIKEN	In situ outreach	JMSI has been corresponding with Dr. Kenji Ono (RIKEN Japan). Dr. Ono is responsible for visualization on K Computer and is interested in combining VisIt/Libsim with his HIVE system.
NASA	In situ outreach	JMSI collaborated with Dr. Pieter Buning (NASA Langley Research Center) to integrate Libsim into OVERFLOW2.
SC16	In situ outreach	JMSI coordinated a Libsim birds of a feather session at the SC16 conference, bringing together in situ enthusiast from around the world.

Technologies / Techniques

JMSI helped to design and develop the SENSEI framework, which enables a simulation to create a single set of adaptor functions that can couple it to multiple in situ infrastructures.

Inventions / Patent Applications, Licensing Agreements

Not applicable.

Other Products

The SENSEI framework was developed and it enables simulations to create a single data adaptor that can couple to several analysis adaptors. As a result of many extreme scale runs on HPC systems, the Libsim in situ library for the open source VisIt project was enhanced so it can be built more easily, share data in a zero-copy manner with the simulation, and can export data more efficiently in parallel.

Intelligent Light 301 Route 17 North, 7th Floor, Rutherford, NJ 07070, USA +1 (201) 460-4700

Acknowledgement

JMSI wishes to thank the U.S. Department of Energy for their generous support for in situ research under grant DE-SC0012449. The support has enabled JMSI to improve in situ processing and extract generation in the VisIt software package. The work accomplished herein is an important part of our company's long-term strategy for handling the largest datasets and it is already generating substantial interest among our customers.

References

[1] W. Bethel and e. al., "Performance Analysis, Design Considerations, and Applications of Extreme-scale In Situ Infrastructures," in *SC16*, Salt Lake City, 2016.

[2] Kitware, "VTK - The Visualization Toolkit," [Online]. Available: http://vtk.org/VTK/project/about.html. [Accessed 29 May 2015].

[3] B. Whitlock, J. M. Favre and J. S. Meredith, "Parallel In Situ Coupling of a Simulation with a Fully Featured Visualization System," in *Eurographics Symposium on Parallel Graphics and Visualization*, Llandudno, Wales, UK, 2011.

[4] U. Ayachit, A. Bauer, B. Geveci, P. O'Leary, K. Moreland, N. Fabian and J. Mauldin, "ParaView Catalyst: Enabling In Situ Data Analysis and Visualization," in *ISAV 2015*, Austin, TX, 2015.

[5] E. P. Duque, B. J. Whitlock, S. M. Legensky, C. P. Stone, R. Ranjan and S. Menon, "The Impact of In Situ Data Processing and Analytics Upon Scaling of CFD Solvers and Workflows," in *27th International Conference on Parallel Computational Fluid Dynamics*, Montreal, 2015.

[6] "ADIOS," Oak Ridge National Laboratory, [Online]. Available: https://www.olcf.ornl.gov/center-projects/adios/. [Accessed 25 August 2017].

[7] "The Mesa 3D Graphics Library," [Online]. Available: http://mesa3d.org/. [Accessed 6 Dec 2014].

[8] "IceT," [Online]. Available: http://icet.sandia.gov/. [Accessed 6 Dec 2014].

[9] "XDMF," [Online]. Available: http://www.xdmf.org/index.php/Main_Page. [Accessed 25 August 2017].

Intelligent Light 301 Route 17 North, 7th Floor, Rutherford, NJ 07070, USA +1 (201) 460-4700