LAB 14-1043

FWP 14-017566

Institution: **Sandia National Laboratories**

XVis: Visualization for the Extreme-Scale Scientific-Computation Ecosystem

Project PI: **Kenneth Moreland**, Sandia National Laboratories
Christopher Sewell, Los Alamos National Laboratory
Hank Childs, University of Oregon
Kwan-Liu Ma, University of California at Davis
Berk Geveci, Kitware, Inc.
David Pugmire, Oak Ridge National Laboratory

Year-end report, FY16

1 Project Description

The XVis project brings together the key elements of research to enable scientific discovery at extreme scale. Scientific computing will no longer be purely about how fast computations can be performed. Energy constraints, processor changes, and I/O limitations necessitate significant changes in both the software applications used in scientific computation and the ways in which scientists use them. Components for modeling, simulation, analysis, and visualization must work together in a computational ecosystem, rather than working independently as they have in the past. This project provides the necessary research and infrastructure for scientific discovery in this new computational ecosystem by addressing four interlocking challenges: emerging processor technology, in situ integration, usability, and proxy analysis.

Emerging Processor Technology One of the biggest recent changes in high-performance computing is the increasing use of accelerators. Accelerators contain processing cores that independently are inferior to a core in a typical CPU, but these cores are replicated and grouped such that their aggregate execution provides a very high computation rate at a much lower power. Current and future CPU processors also require much more explicit parallelism. Each successive version of the hardware packs more cores into each processor, and technologies like hyperthreading and vector operations require even more parallel processing to leverage each core's full potential.

XVis brings together collaborators from the predominant DOE projects for visualization on accelerators and combines their respective features in a unified visualization library named VTK-m. VTK-m will allow the DOE visualization community, as well as the larger visualization community, a single point to collaborate, contribute, and leverage massively threaded algorithms. The XVis project is providing the infrastructure, research, and basic algorithms for VTK-m, and we are working with the SDAV SciDAC institute to provide integration and collaboration throughout the Office of Science.

***In Situ* Integration** Fundamental physical limitations prevent storage systems from scaling at the same rate as our computation systems. Although large simulations commonly archive their results

before any analysis or visualization is performed, this practice is becoming increasingly impractical. Thus, the scientific community is turning to running visualization *in situ* with simulation. This integration of simulation and visualization removes the bottleneck of the storage system.

Integrating visualization *in situ* with simulation remains technically difficult. XVis leverages existing *in situ* libraries to integrate flyweight techniques and advanced data models to minimize resource overhead. Within our *in situ* visualization tools, XVis integrates existing visualization algorithms and those incorporating emerging processor technology. XVis also studies the latest techniques for new domain challenges and for post hoc interaction that reconstructs exploratory interaction with reduced data.

Usability A significant disadvantage of using a workflow that integrates simulation with visualization is that a great deal of exploratory interaction is lost. Post hoc techniques can recover some interaction but with a limited scope or precision. Little is known about how these limitations affect usability or a scientist's ability to form insight. XVis performs usability studies to determine the consequences of *in situ* visualization and proposes best practices to improve usability.

Unlike a scalability study, which is always quantitative, XVis' usability studies are mostly qualitative. Our goal is not to measure user performance; rather, we want to learn about the limitations and benefits of incorporating *in situ* methods in scientists' workflows. These studies reveal how the simulation, hardware, and users respond to a particular design and setting.

Proxy Analysis The extreme-scale scientific-computation ecosystem is a much more complicated world than the largely homogeneous systems of the past. There is significantly greater variance in the design of the accelerator architecture than is typical of the classic x86 CPU. *In situ* visualization also yields complicated interactions between the simulation and visualization that are difficult to predict. Thus, the behavior observed in one workflow might not be indicative of another.

To better study the behavior of visualization in numerous workflows on numerous systems, XVis builds proxy applications that characterize the behavior before the full system is run. We start with the design of mini-applications for prototypical visualization operations and then combine these with other mini-applications to build application proxies that characterize the behavior of larger systems. The proxy analysis and emerging processor technology work are symbiotic. The mini-applications are derived from the VTK-m implementations, and the VTK-m design is guided by the analysis of the mini-applications.

2 Progress Report

The XVis research plan specified in the proposal is divided into a set of milestones spread over the 3-year period of the project, divided among the projects research areas, and distributed among the participating institutions. Our report is similarly organized by giving progress on each of these milestones. Our report is abbreviated to include only those milestones with relevant work in the time period of this report.

2.1 Emerging Processors

Milestone 1.a, Initial VTK-m Design (Year 1–SNL, Kitware, ORNL, LANL) Provide the research and design for VTK-m functional operation and, in conjunction with SDAV, develop an initial implementation.

Expected Completion: FY15, Q4 Status: Complete

Milestone 1.a was completed in FY15 Q4. However, development of VTK-m continues throughout XVis, and we continue to report on its progress.

Recent work for VTK-m includes the initial implementation of the "filter" interface, which is the high-level API to run algorithms in VTK-m. We have also implemented a basic rendering library to enable in situ visualization by directly integrating VTK-m.

During FY16 Q4 significant effort was placed in removing the Boost dependency from VTK-m and requiring C++11 compiler. The goals of these changes are twofold; first it has allowed for a reduction in complexity inside of VTK-m, secondly it has aided adoption of the library, specifically by the VTK community, which classically resisted a dependency on Boost.

Much of FY16 Q1 and Q2 was focused on creating a stable and user-friendly release of VTK-m. The release of VTK-m 1.0 was in FY16 Q3.

Additionally, we have implemented a demo application for VTK-m that reads a specified VTK file or generates a uniform structured grid set and then uses VTK-m's Marching Cubes filter to compute the isosurface. After that, the demo uses VTK-m's rendering engine to generate image files using OS Mesa. This demonstration code is used as part of our outreach for teaching VTK-m.

We have also implemented new features to in VTK-m including parallel versions of cell-to-point, simple stream line integration, histogram computation, atomic arrays, improved rendering.

We have also added a rendering infrastructure that supports things such as camera, windows, and annotations such as axes and colorbars. This infrastructure supports several different renders: rasterization, volume rendering, and ray casting. We have provide testing examples to illustrate how these concepts work in practice.

Milestone 1.b Array Characterization (Year 2–SNL, Kitware) Automatically characterize how arrays are used and leverage that information to optimize memory hierarchy usage.
Expected Completion: FY16, Q4 Status: Complete

In VTK-m random global memory access are very common occurrence inside algorithms that requires two different types of topological information, for example cells, and points or faces and edges. Random global memory access on NVIDIA GPU's cause uncoalesced memory accesses, as global memory reads must be accessed in 32, 64, or 128 byte transactions. When a warp executes an instruction that uses global memory, the fetches are coalesced into the minimum number of transactions possible. With random global memory reads the warps well not be well coalesced resulting in multiple transactions occurring reducing the total throughput of the GPU.

During a design review with NVIDIA engineers they recommended using texture memory when accessing global arrays. Texture memory is global memory backed by the L1 texture cache, allowing for higher throughput when there is 2D locality of the memory fetches. VTK-m now automatically uses custom iterators for CUDA that uses texture memory reads for all global memory reads. This is accomplished by using the ldg CUDA intrinsic that provides texture memory reads without explicit construction of texture objects. This has resulted in a ~10% performance increase when executing Cell based algorithm that require Point based global memory reads.

We have also studied the performance on Intel-based architecture with respect to Array-of-Structures (AoS) vs. Structure-of-Arrays (SoA). A typical array of vectors in VTK-m is stored as an array with each component containing a Vec structure (an AoS in this context). We found that this AoS could sometimes inhibit the engagement of vector operations on x86 processors. However, we also found that when coupled with loop tiling (investigated during Milestone 1.e),

we could efficiently copy blocks of the AoS to an SoA efficiently. The improved caching and vectorization hide the overhead of the copy.

We consider the research for this milestone complete. We plan to have some follow up implementation of the work in VTK-m.

Milestone 1.c Hybrid Parallel (Year 2–LANL) Compare alternative models for the interaction of shared-memory and distributed-memory parallelism within VTK-m.
Expected Completion: FY16, Q4 Status: Nearing Completion

In collaboration with Hamish Carr from the University of Leeds, we explored the interaction between shared-memory data-parallelism and inter-node distributed-memory parallelism in the context of an algorithm for computing contour trees (Reeb graphs). Contour trees encode the topological changes that occur to the contour as the isovalue ranges between its minimum and maximum values. They can be used to identify the most "important" isovalues in a data set according to various metrics (e.g., persistence). Although topological analysis tools such as the contour tree and Morse-Smale complex are now well established, there is still a shortage of efficient parallel algorithms for their computation, in particular for massively data-parallel computation on a SIMD model. We developed a novel data-parallel algorithm for computing the fully augmented contour tree using a quantized computation model. We then extended this to provide a hybrid data-parallel / distributed algorithm, allowing scaling beyond a single GPU or CPU, and tested its scaling using Earth elevation data from GTOPO30 across 16 nodes. Our implementation uses the portable data-parallel primitives provided by Nvidia's Thrust library, as well as MPI for inter-node communication. In July, a paper describing this algorithm, "Hybrid Data-Parallel Contour Tree Computation", was accepted for publication at the Computer Graphics and Visual Computing Conference (CGVC) in September.

We have also initiated a collaboration with the project "A Unified Data-Driven Approach for Programming In Situ Analysis and Visualization", led by Pat McCormick, to evaluate the performance of the prototype integration of VTK-m with Legion that they have developed. Legion is a task-parallel runtime that can schedule tasks using a task graph based on the dependencies between the tasks. This is an alternative to the traditional bulk-synchronous MPI model, as used by VTK. We have successfully compiled the code produced by McCormick's project on the Moonlight supercomputer at Los Alamos, using GASNet, OSMesa, VTK-m, and Legion. We have run their isosurface example across four nodes, and plan to perform a scaling study across up to several hundred nodes, and compare this to VTK-m used within the VTK MPI-based pipeline. However, issues with global dynamic memory allocation used in virtual mapping for Legion have slowed development in McCormick's project, and so their code is not yet ready for us to run a large scaling study. We anticipate that they will be able to resolve these issues within the next month, and that we will be able to begin to obtain some meaningful performance results in the upcoming quarter.

Milestone 1.d Additional Algorithms (Year 3–LANL) Develop algorithms for additional visualization and analysis filters in order to expand the functionality of the VTK-m toolkit to support less critical but commonly used operators.
Expected Completion: FY16, Q4 Status: Preliminary Work

More recently, in collaboration with Hamish Carr as well as Gunther Weber from LBNL, we have helped develop and implement a new data-parallel contour tree algorithm that does not quantize the contour values, allowing for more precise results and less memory usage. Our shared SMP algorithm for parallel contour tree computation has formal guarantees of O(log(n) log(t)) parallel steps and O(nlog(n)) work. It employs "parallel peak pruning", in which superarcs are created in the join tree by identifying peaks and finding their governing saddles, and

recursively pruning the regions for each peak/saddle pair. We implemented this algorithm natively in OpenMP and with the Thrust library, achieving up to 10x parallel speed up on multi-core CPUs and up to 50x speed up on Nvidia GPUs compared to the serial version. In June, we submitted a paper about this algorithm, "Parallel Peak Pruning for Scalable SMP Contour Tree Computation", to the IEEE Symposium on Large Data Analysis and Visualization, and it was accepted in August and will appear in October. Performance was compared to the serial sweep-and-merge algorithm published by Carr in 2003, which was also used to verify the results. Pat Fasel at Los Alamos has converted the Thrust implementation to use VTK-m, and we intend to release this in the open-source VTK-m repository soon.

Samuel Li, a graduate student working with Chris Sewell at Los Alamos over the summer, has implemented 1D and 2D wavelet transformation worklets in VTK-m. A set of worklets are used to support the wavelet calculation, including boundary handling, decomposition of signals into approximation and detail coefficients, and reconstruction of the original signal from the wavelet coefficients. Coefficients resulting from the wavelet transforms can be used for compression. The simplest strategy is to threshold small coefficients, which has been implemented and achieves fairly good compression. These calculations can be applied recursively to the output to obtain better compression. This implementation of wavelet transforms uses filter banks, which has the advantage of flexibility to support more wavelet filters. Currently it supports four widely used wavelet filters for compression: CDF 9/7, CDF 8/4, CDF 5/3, and HAAR.

He has attempted to minimize the number of data transfers and copies required by the algorithm, and is currently comparing the performance and accuracy of the data-parallel VTK-m algorithm to domain decomposition parallelization strategies as used by the wavelet compression algorithms in VAPOR. By applying the transformation to each element of the input with data-parallelism, it is expected that greater accuracy can be obtained than by using domain decomposition for the same amount of computation, with at least some types of input (such as Gaussians). The VTK-m implementation also allows these same worklets to run on multiple architectures, including GPUs. The 1D wavelet compression worklets have already been merged into VTK-m, while we expect to do the same for the 2D worklets within the next few weeks.

Milestone 1.e Function Characterization (Year 3–Sandia, Kitware) Design methods to characterize how functions behave and leverage this information for heterogeneous architectures.
Expected Completion: FY16, Q4 Status: Preliminary Work

We have investigated how different functional structures effect vectorization on Intel-based architecture. We have found that for a surprising number of function calling parameters, a technique called loop tiling can have dramatic impact on the efficiency of the execution and the effectiveness of vectorization. In loop tiling for loops that iterate an operation over some set of arrays is broken into a nested inner loop of fixed size (say 1024 items). This minor change can have surprising effect on both the compiler and the hardware (caching and vectorization) units. Additionally, in some circumstances we found that while engaging loop tiling we could transform the structure of data (e.g. from AoS to SoA) without incurring a performance penalty.

2.2 In Situ

Milestone 2.b Post Hoc Interaction (Year 1–U Oregon) Implement three algorithms that use extreme-scale features such as non-volatile memory or knowledge of communication efficiencies.
Expected Completion: FY15, Q4 Status: Complete

We have made significant progress on this milestone in FY16. Our effort has focused on deep memory hierarchies, specifically the SSDs appearing increasingly often on leading-edge supercomputers. Following the "in situ reduction+post hoc" paradigm in collaboration with Childs' Early Career award, we wanted to explore the opportunities available from having significantly more memory for storing data. In particular, using that memory to store multiple time slices and then compressing the data to take advantage of temporal coherence. Our experiments specifically focused on wavelet compression. While wavelet compression typically operates on one time slice at a time (3D data), our study also included multiple time slices (4D data). Our findings showed that the 4D approach could take advantage of temporal coherency, and, for all metrics studied, the benefits were approximately a factor of two improvement. We submitted a paper detailing the experiments to the IEEE Visualization conference and it is in review. In terms of our milestone, we believe this study provides evidence that handling for deep memory/SSDs should be added to VTK-m. Further, we are now planning on adding wavelet compression and decompression operators to VTK-m for general usage.

Milestone 2.b explores architectural features currently beyond the expressivity of VTK-m. For this milestone, three architectural features are to be explored and evaluated, to potentially influence the VTK-m design. We have made excellent progress on our first architectural feature, which explores the performance improvements possible when using different types of memory. We performed this milestone within our ray-tracing code (now ported to VTK-m) and found that accessing GPU-specific memory significantly improved performance. This finding was a contributor in the expansion of VTK-m's improved memory. For our second architectural feature, we are exploring the usage of SSD for post-hoc exploration. A study is underway, which we expect to complete in the next three months. Finally, we decided to focus on deep memory hierarchies for the third architectural feature, and also to delay this study until more architectures are available (i.e., NVLink). This delay is consistent with our under spending and should not interfere with the completion of the project.

Milestone 2.c Flyweight In Situ (Year 2–Kitware) Provide flyweight in situ visualization techniques into a feature-rich, general-purpose library.
Expected Completion: FY16, Q4 Status: In Progress

Kitware has been investigating using non-standard memory layouts for arrays and data structures. As a first step towards this goal, we developed the MappedDataArray and MappedDataSet classes, which allow for custom memory layouts. After further evaluation, our conclusion was the overhead introduced by the abstraction used in this approach is too high. We developed and integrated the next generation version of this framework including the classes AOSDataArrayTemplate and SOADataArrayTemplate into VTK. This design depends on template-based polymorphism, and gives us performance that is close to using raw pointers while attaining the objective of allowing tight coupling of VTK in situ with simulations. The updated design allows for things such as constant value arrays, implicit point arrays, and other efficient data model concepts that VTK-m also has.

As our next step in the creation of a lightweight in situ library, we started investigating the use of the Sensei framework. Our first step will be to integrate a stripped and VTK-m enabled VTK into Sensei. This should provide us with a lightweight library that can already be embedded into various simulations codes as well as co-operate with other in situ libraries such as Catalyst and ADIOS.

As part of our collaboration with NVIDIA we developed a prototype that demonstrated VTK-m integration inside VTK and ParaView Catalyst for in situ analysis at Supercomputing 2015. The prototype used the PyFr simulation running on 256 nodes of Titan. The entire operation from

simulation computation, visualization algorithms such as IsoSurface, to rendering happened completely on the GPUs and required no memory copies of PyFr simulation data.

We are holding off marking this milestone as complete until we have demonstrated the integration into Sensei.

Milestone 2.d Data Model Application (Year 2–ORNL) Explore application of new data models to novel architectures appropriate to in situ.
Expected Completion: FY16, Q3 Status: Complete

We have been exploring the integration of VTK-m in situ with several applications running on DOE LCFs. These have included several fusion codes, and a computational seismology code. Efforts have been focused on understanding the scientific workflows being used by these applications. This better allows us to target machine architectures and analysis products for use in situ to help scientists understand their simulations. We have focused these efforts on two SciDAC fusion simulation codes; XGC1 and Xolotl. XGC1, a particle in cell code is used to study plasmas in fusion tokamak devices, particularly in the edge region. Xolotl is a new code that is being used to study the interaction with the tokamak wall.

VTK-m is being used for test and production runs of Xolotl for in situ monitoring and visualization. We are also exploring the use of light-weight visualization services to do analysis and visualization for XGC1 in an in transit setting. We are using the ADIOS middleware system along with the DataSpaces and DIMES transport methods for data movement. The VTK-m services include the calculation of blobby turbulence around the edge and bulk velocity derived from the particle data. In this later example, we are tracking individual particle paths to derive a time varying vector field. This vector field, which is a significantly reduced representation of the particles makes it possible for much more frequent output for post-hoc analysis and visualization.

We mark this milestone as complete although we will have some follow up exploration and implementation as we address application needs.

Milestone 2.e Memory Hierarchy Streaming (Year 2–LANL) Develop streaming out-of-core versions of key visualizations and analysis algorithms to efficiently use deep memory hierarchies within in situ applications.
Expected Completion: FY16, Q4 Status: Complete

We first experimented with the use of the STXXL library from Karlsruhe University for streaming data from disk into main memory and into accelerator memory for isosurface and KD-tree construction algorithms. We then prototyped the combination of such external-memory algorithms with our distributed wrapper for Thrust as a first step towards enabling these algorithms to operate on data that is both distributed across nodes and too large on each node to fit into memory.

This summer, we have implemented basic support for streaming data through the GPU directly in VTK-m. A custom array handle class, ArrayHandleStreaming, wraps a standard array handle and provides an interface of one block at a time. A custom dispatcher, DispatcherStreamingMapField, operates similarly to the standard DispatcherMapField, but uses the function interface infrastructure that is used to iterate through all input array handles to transfer data from the control to the execution environment to also wrap all input and output array handles with an ArrayHandleStreaming class, and then loop through the desired number of blocks by appropriately setting the parameters of the ArrayHandleStreaming class and calling the scheduler. Since all the output data cannot reside in the execution environment memory simultaneously, it also loops through the output array handles at the end of the iteration for each block to synchronize the execution environment with the control environment. These custom array handle

and dispatcher classes allow a VTK-m developer to easily run basic field map worklets when all the data will not fit on the GPU. We have also implemented a StreamingScanInclusive method in the DeviceAdapterAlgorithmGeneral class, which wraps the input and output array handles with an ArrayHandleStreaming and iterates through each block, applying the last result from the previous block to the first element of the next block, and performing the inclusive scan within each block by calling the ScanInclusive method for the active device adapter. A merge request with these streaming classes has been created and we expect it will be integrated into VTK-m in the near future. We plan to implement streaming versions of additional device adapter algorithms which, together with the DispatcherStreamingMapField, should provide VTK-m developers with basic tools with which to construct streaming algorithms.

Milestone 2.f Interface for Post Hoc Interaction (Year 2–U Oregon) Define an abstract VTK-m interface for extreme-scale, implement a prototype for the interface, and exercise the prototype with the three algorithms chosen for Milestone 2.b.
Expected Completion: FY16, Q4 Status: In Progress

We have begun work on this milestone. We have begun with wavelet compression, and have a wavelet compression algorithm for three-dimensional data working in VTK-m. This includes numerous performance evaluations and improvements, and we have found that our code is significantly faster on GPUs than serial code.

We have been underspending on our budget, and the work completed to date is consistent with our underspend. In Year 3, Q1 and Q2, we will implement the remaining two algorithms.

2.3 Usability

Milestone 3.b Prepare Usability Studies (Year 2–UC Davis, U Oregon) Identify participants and meet to discuss goals for the visualization and analysis tasks. Collect user data sets and design each user study according to code and hardware settings.
Expected Completion: FY16, Q2 Status: Complete

University of Oregon Ph.D. student James Kress has embedded with the Oak Ridge visualization team (including mentorship from Dave Pugmire) in an effort to engage the XGC team. Over the last six months, he has conducted over twenty interviews to understand the XGC team's data needs. This study has included physicists, computational scientists, and computer scientists. Information obtained ranges from predicted data sizes to desired visualization and analysis operations. The next phase of this study has used this information to consider in situ feasibility. The desired visualization and analysis operations have been grouped into approximately five equivalence classes, and we are doing performance modeling for these classes, grounded by real-world experiments on Oak Ridge's supercomputers

Since the completion of the milestone, we prepared a manuscript documenting the XGC team's needs and submitted it to the SC16 Workshop on In Situ Analysis and Visualization. It was accepted and will appear in November.

Milestone 3.c Start Usability Studies (Year 2–UC Davis) Conduct pilot studies with algorithms and participants identified in Milestone 3.b.
Expected Completion: FY16, Q4 Status: Nearing Completion

In the area of combustion, we have been working with Dr. Jackie Chen's research group at Sandia National Labs. We have been developing analysis and visualization tools to streamline their current workflow and solve new analysis problems as identified by the domain scientists. We

have completed the development of a scalable histogram-based particle selection scheme which couples both in situ and post processing analyses.

Spatially distributed histograms (probability distribution functions) are generated in situ using a set of modules developed at UC Davis. These probability distributions are generated using the full resolution simulation data and can be leveraged in post processing to aid certain analysis tasks. We have been working on investigating how we can use these modules to best interface with the S3D combustion simulation in an unobtrusive manner. Not only must these modules match the scalability of the simulation, scientists need to be able to easily make modifications to the code based on their most recent needs. We have completed scalability tests in large scale production runs.

On the post processing side of things, we have developed a comprehensive visualization software that can utilize the in situ generated histograms to make real time particle selections from large scale datasets. Users can select spatial regions based on desired distributions in the histograms which in turn extracts particle subsets (which were spatially sorted in situ) for later analysis. We have been making detailed design decisions based on a close collaboration with the domain scientists and their analysis needs. We have designed this software (including UI, data management, etc.) so that it can be easily be used by the scientists with little to no modification to their usual workflow.

Our paper describing the recent completed stages of this work has been accepted by the IEEE LDAV 2016 symposium and will be presented in October.

In the area of cosmology, we are working with Dr. Salman Habib's research group at the Argonne National Laboratory to develop and deploy an interactive visualization tool that can effectively use the parallel GPU clusters available to them. We have developed GPU accelerated, parallel rendering methods for interactive visualization of both the particle data and merger tree data. We have also incorporated a few quantitative analysis functionalities. An early version of this tool has been reported in a paper presented at PacificVis 2016.

In this period, we have improved the scalability of halo tag processing. Halo tags are critical for interactive analysis of halo structure, but we must first associate halo tags with their particles in the 3D spatial domain. We have also improved the handling of temporal data in order to facilitate both command line (server-side) and GUI (client-side) use. We are presently working on integrating a phase-space mesh rendering method. The first deployment of this interactive visualization facility at Argonne is expected to be done by December 2016, and subsequently usability studies will be carried out with full participation of the scientists.

We mark the status of this milestone as “nearing completion” rather than “complete” because there is a small amount of implementation for the cosmology application to finish, which we will complete at the beginning of our work for Milestone 3.d.

Milestone 3.d Continue Usability Studies (Year 3–UC Davis, U Oregon) Continue studies of Milestone 3.c.

Expected Completion: FY17, Q4 Status: Preliminary Results

In supporting the combustion simulations, we are currently in the process of deploying the in situ modules for use. While our own tests have been successful in showing the scalability of our tools. More long term evaluation is needed to test the ease at which the domain scientists can invoke or modify our routines as well as their ability to handle the large variety of chemical cases that the S3D simulation can produce.

We are also deploying the post processing visualization software. As the scientists continue to use the tool, we will conduct a thorough usability study to evaluate the ease at which they can use the

software and how successful it is in identifying desired trends in the data. Feedback as well as any new insights into the data will be carefully documented.

University of Oregon Ph.D. student James Kress has been implementing the functionality necessary to meet the needs of the XGC simulation team (as identified in Milestone 3.b). He is complementing this activity with running performance measurements. Ultimately, he will establish and validate a performance model on the feasibility of in situ processing for this team. He is midway through this task, and appears on track for meeting the milestone on time.

Milestone 3.e Apply Usability Studies (Year 3–UC Davis, U Oregon) Conduct a review session with each scientist team on usability study results from Milestone 3.d. Refine the design and implementation of techniques and workflow according to lessons learned.
Expected Completion: FY17, Q4 Status: Preliminary Results

Throughout the user study, we will make necessary modifications to improve the tools and document important lessons learned throughout this process. This will continue as an iterative process by deploying newer versions of our tools. The scientists will be able to continue exploring their data regularly while providing new feedback as changes are being made.

We will also address new analysis needs as identified by the scientists. This will be done by expanding the current functionality of our tools. For example, the scientists now desire a sophisticated system that can be used to analyze the time varying properties of the extracted particle subsets. We plan on integrating this functionality directly into the current visualization tool to ensure immediate feedback on the time varying analysis once the particle subsets have been extracted. Additional usability tests and redesigns will also take place for any new functionality we develop.

2.4 Proxy Analysis

Milestone 4.a Initial Mini-App Implementation (Year 1–SNL, ORNL) An initial implementation of mini-applications based on visualization and in situ workloads.
Expected Completion: FY15, Q4 Status: In progress, delayed

Although milestone 4.a was scheduled to be started at the beginning of the project, the majority of the work has been postponed in lieu of providing a VTK-m prototype (milestone 1.a), which is on the critical path.

The initial prototype for the Marching Cubes mini-app was implemented in FY15, Q4, but further development was stalled due to staffing. This delay is consistent with our under spending and should not interfere with the completion of the project. We have rearranged staffing in FY17 to insure this gets completed. The existing implementation will be hardened and contributed to Mantevo in early FY17. This will give us an implementation to start working on subsequent milestones. A mini-driver for rendering and a mini-app for particle advection will follow soon after.

Milestone 4.b Validate Mini-App Characteristics (Year 2-ORNL) Validate behavior and resource usage of mini-applications against that of real applications and generate performance/resource models.
Expected Completion: FY16, Q4 Status: In Progress

In preparation for proxy analysis, we have been familiarizing ourselves with the Oxbow suite of application characterization tools and have performed some initial within-node characterization of contouring algorithms in existing tools. For example, an early comparison of a sequential

contouring algorithm in VisIt and a data-parallel contouring algorithm in EAVL, one of the predecessor projects to VTK-m, we noticed that while there is no thread-level parallelism in VisIt, it was able to make use of integer SIMD arithmetic, while the highly-parallel EAVL algorithm was not.

In this period, we have begun exploring the characteristics of the Marching Cubes mini-app (developed as part of Milestone 4.a) with the Oxbow suite. These early investigations show similarities and differences with the EAVL version of the algorithm. For example, in terms of instruction mix both have a high proportion of integer operations (about 40%) and memory operations (about 25%), but algorithmic choices resulted in different tradeoffs between branch operations and memory movement operations. In memory bandwidth behavior, both had similar read bandwidths, but a noticeable difference in write bandwidths. This preliminary data is shown in the following table, and continuing studies will explore further characteristics and compare the proxy app behavior with the recently developed VTK-m algorithm.

	Instruction Mix (%)				Memory Bandwidth			
Application	BrOps	IntOps	MemOps	Moves	Read B/Cycle	Read MB/s	Write B/Cycle	Write MB/s
Mini-app	20.35	40.06	27.60	11.01	0.07	240.76	0.02	63.18
EAVL	8.76	38.99	24.62	27.12	0.10	283.92	0.04	109.62

Progress on this milestone got delayed due to some shifting of staff. We are reengaging folks from the Oxbow project to continue work on this milestone.

3 Other Activities

3.1 Publications

"Visualization and Analysis Requirements for In Situ Processing for a Large-Scale Fusion Simulation Code." James Kress, David Pugmire, Scott Klasky, and Hank Childs. In *SC16 Workshop on In Situ Analysis and Visualization (ISAV)*, November 2016.

"Optimizing Multi-Image Sort-Last Parallel Rendering." Matthew Larsen, Kenneth Moreland, Chris Johnson, and Hank Childs. In *Proceedings of the IEEE Symposium on Large Data Analysis and Visualization (LDAV)*, October 2016.

"Parallel Peak Pruning for Scalable SMP Contour Tree Computation." Hamish Carr, Gunther Weber, Christopher Sewell, and James Ahrens. In *Proceedings of the IEEE Symposium on Large Data Analysis and Visualization (LDAV)*, October 2016.

"In Situ Generated Probability Distribution Functions for Interactive Post Hoc Visualization and Analysis." Yucong Ye, Tyson Neuroth, Franz Sauer, Kwan-Liu Ma, Giulio Borghesi, Aditya Konduri, Hemanth Kolla, and Jacqueline Chen. In *Proceedings of the IEEE Symposium on Large Data Analysis and Visualization (LDAV)*, October 2016.

"Parallel Distributed, GPU-Accelerated, Advanced Lighting Calculations for Large-Scale Volume Visualization." Min Shih, Silvio Rizzi, Joseph Insley, Thomas Uram, Venkatram Vishwanath, Mark Hereld, Michael E. Papka, and Kwan-Liu Ma. In *Proceedings of the IEEE Symposium on Large Data Analysis and Visualization (LDAV)*, October 2016.

"Hybrid Data-Parallel Contour Tree Computation." Hamish Carr, Christopher Sewell, Li-Ta Lo, and James Ahrens. In *Proceedings of the Computer Graphics and Visual Computing Conference*, September 2016.

"External Facelist Calculation with Data-Parallel Primitives." Brenton Lessley, Roba Binyahib, Robert Maynard, and Hank Childs. In *Eurographics Symposium on Parallel Graphics and Visualization (EGPGV)*, June 2016.

"VTK-m: Accelerating the Visualization Toolkit for Massively Threaded Architectures." Kenneth Moreland, Christopher Sewell, William Usher, Li-ta Lo, Jeremy Meredith, David Pugmire, James Kress, Hendrik Schroots, Kwan-Liu Ma, Hank Childs, Matthew Larsen, Chun-Ming Chen, Robert Maynard, and Berk Geveci. *IEEE Computer Graphics and Applications*, 36(3), May/June 2016.

"An Integrated Visualization System for Interactive Analysis of Large, Heterogeneous Cosmology Data," Annie Preston, Ramyar Ghods, Jinrong Xie, Franz Sauer, Nick Leaf, Kwan-Liu Ma, Esteban Rangel, Eve Kovacs, Katrin Heitmann, Salman Habib. In *Proceedings of IEEE PacificVis*, April 2016.

"The Tensions of In Situ Visualization." Kenneth Moreland. *IEEE Computer Graphics and Applications*, 36(2), March/April, 2016.

"Visualization Techniques for Studying Large-Scale Flow Fields from Fusion Simulations." Franz Sauer, Yubo Zhang, Weixing Wang, Stéphane Ethier, Kwan-Liu Ma, *Computing in Science and Engineering*, 18(2), 68–77, March/April 2016.

"Visualization for Exascale: Portable Performance is Critical." Kenneth Moreland, Matthew Larsen, and Hank Childs. *Supercomputing Frontiers and Innovations*, 2(3), 2015.

"ParaView Catalyst: Enabling In Situ Data Analysis and Visualization." Utkarsh Ayachit, Andrew Bauer, Berk Geveci, Patrick O'Leary, Kenneth Moreland, Nathan Fabian, Jeffrey Mauldin. In *Proceedings of the First Workshop on In Situ Infrastructures for Enabling Extreme-Scale Analysis and Visualization (ISAV 2015)*, November 2015.

"Integrated explorer for cosmological evolution." Annie Preston, Franz Sauer, Ramyar Ghods, Nick Leaf, Jinrong Xie, Kwan-Liu Ma, In *IEEE Scientific Visualization (SciVis)*, 107–114, October 2015.

"Volume rendering with data parallel visualization frameworks for emerging high performance computing architectures." Hendrik A. Schroots, Kwan-Liu Ma, In *SIGGRAPH Asia Visualization in High Performance Computing*, 3:1–3:4, November 2015.

"High performance heterogeneous computing for collaborative visual analysis." Jianping Li, Jia-Kai Chou, Kwan-Liu Ma, In *SIGGRAPH Asia Visualization in High Performance Computing*, 12:1–12:4, November 2015.

"Scalable Parallel Distance Field Construction for Large-Scale Applications." Hongfeng Yu, Jinrong Xie, Kwan-Liu Ma, Hemanth Kolla, Jacqueline H. Chen, *IEEE Transactions on Visualization and Computer Graphics*, 21(10), 1187–1200, October 2015.

"In situ depth maps based feature extraction and tracking." Yucong Chris Ye, Yang Wang, Robert Miller, Kwan-Liu Ma, Kenji Ono, In *Large Scale Data Analysis and Visualization (LDAV)*, 1–8, October 2015.

"Fast uncertainty-driven large-scale volume feature extraction on desktop PCs." Jinrong Xie, Franz Sauer, Kwan-Liu Ma, In *Large Scale Data Analysis and Visualization (LDAV)*, 17–24, October 2015.

"Scalable visualization of discrete velocity decompositions using spatially organized histograms." Tyson Neuroth, Franz Sauer, Weixing Wang, Stéphane Ethier, Kwan-Liu Ma, In *Large Scale Data Analysis and Visualization (LDAV)*, 65–72, October 2015.

3.2 Chairs

Symposium Co-Chair, *Large Scale Data Analysis and Visualization (LDAV)*, Hank Childs, October 23, 2016.

Papers Co-Chair, *IEEE Information Visualization (InfoVis)*, Kwan-Liu Ma, October 23–28, 2016.

Papers Co-Chair, *Large Scale Data Analysis and Visualization (LDAV)*, Kenneth Moreland, October 23, 2016.

Papers Co-Chair, *EuroVis*, Kwan-Liu Ma, June 6-10, 2016.

Workshop Co-Chair, *The 10th Workshop on Ultrascale Visualization*, Kwan-Liu Ma, SC15, November 16, 2015.

Papers Co-Chair, *Large Scale Data Analysis and Visualization (LDAV)*, Hank Childs, October 25–26.

3.3 Committees

Program Committee, *SPIE Visualization and Data Analysis (VDA)*, Hank Childs, January 29–February 2, 2017.

Program Committee, *12th International Symposium on Visual Computing (ISVC)*, Kenneth Moreland, December 12–14, 2016.

Program Committee, *ACM SIGGRAPH ASIA 2016 Symposium on Visualization (SA16VIS)*, Kwan-Liu Ma, December 5–8, 2016.

Program Committee, *In Situ Infrastructures for Enabling Extreme-scale Analysis and Visualization (ISAV)*, Kenneth Moreland, November 13, 2016.

Program Committee, *In Situ Infrastructures for Enabling Extreme-scale Analysis and Visualization (ISAV)*, Hank Childs, November 13, 2016.

Program Committee, *Cooperative Design, Visualization, and Engineering (CDVE)*, Kwan-Liu Ma, October 24–27.

Program Committee, *IEEE Scientific Visualization (SciVis)*, Kenneth Moreland, October 23–28, 2016.

Program Committee, *IEEE Symposium on Visualization for Cybersecurity (VizSec)*, Kwan-Liu Ma, October 24, 2016.

Program Committee, *IEEE Large Data Analysis and Visualization (LDAV)*, Christopher Sewell, October 23, 2016.

Program Committee, *IEEE Large Data Analysis and Visualization (LDAV)*, Kwan-Liu Ma, October 23, 2016.

Program Committee, *Graph Drawing & Network Visualization*, Kwan-Liu Ma, September 19–21, 2016.

Program Committee, *IEEE Cluster*, Kenneth Moreland, September 12–16, 2016.

Program Committee, *EuroVis*, Kenneth Moreland, June 6–10, 2016.

Program Committee, *Eurographics Symposium on Parallel Graphics and Visualization (EGPGV)*, Kenneth Moreland, June 6–7, 2016.

Program Committee, *Eurographics Symposium on Parallel Graphics and Visualization (EGPGV)*, Hank Childs, June 6–7, 2016.

Program Committee, *IEEE Pacific Visualization (PacificVis)*, Hank Childs, April 20–23, 2016.

Program Committee, *IEEE Conference on Multimedia and Big Data (BigMM)*, Kwan-Liu Ma, April 20–22, 2016.

Program Committee, *IEEE BigDataService*, Kwan-Liu Ma, March 29–April 1, 2016.

Program Committee, *Workshop on Emotion and Visualization (EmoVis)*, Kwan-Liu Ma, March 10, 2016.

Program Committee, *SPIE Visualization and Data Analysis*, Hank Childs, February 16–18, 2016.

NSF III 2016 Review Panel, Kenneth Moreland.

Program Committee, *ACM/IEEE Supercomputing*, Hank Childs, November 15–20, 2015.

Program Committee, *Visual Performance Analysis* (SC workshop), Hank Childs, November 20, 2015.

Program Committee, *IEEE Scientific Visualization (SciVis)*, Kenneth Moreland, October 25–30, 2015.

Program Committee, *IEEE Scientific Visualization (SciVis)*, Hank Childs, October 25–30, 2015.

Program Committee, *IEEE Cluster*, Kenneth Moreland, September 8–11, 2015.

Program Committee, *Eurographics Symposium on Parallel Graphics and Visualization (EGPGV)*, Kenneth Moreland, May 25–26, 2015.

3.4 Presentations and Other Outreach

"Exascale Visualization and In Situ Processing," Invited lecture, Hank Childs, University of Tennessee, Knoxville, TN, September 2016.

"In Situ Processing: Opportunities, Challenges, and Instantiations," Invited talk, Hank Childs, *Smoky Mountains Computational Sciences and Engineering Conference*, September 1, 2016.

"In Situ Processing: Opportunities, Challenges, and Instantiations," Invited talk, Oak Ridge National Laboratory, Oak Ridge, TN, August 2016.

"Visualization: A Tool for Data Exploration and Storytelling," Invited lecture, Kwan-Liu Ma, Hokudai University, Hokkaido, Japan, August 9, 2016.

"Scientific Visualization for the Public," Invited talk, Kwan-Liu Ma, Hanzhou Low Carbon Science & Technology Museum, Hangzhou, China, July 8, 2016.

"Visualization: An Exploratory and Explanatory Tool," Invited talk, Kwan-Liu Ma. *International Symposium on Visual Computing*, Hangzhou, China, July 7, 2016.

"Recent Advances in Visualization Research," Invited seminar, Kwan-Liu Ma, National Chiao Tung University, Hsinchu, Taiwan, June 29, 2016.

"Big-Data Visualization Techniques for Studying Behaviors, Connections, and Evolution," Invited Talk, Kwan-Liu Ma, Institute of Statistical Science, Academia Sinica, Taipei, Taiwan, June 17, 2016.

"Visualization: An Essential Tool for Scientific Discovery and Storytelling," Invited talk, Kwan-Liu Ma, Pacific Science Congress, Academia Sinica, Taipei, Taiwan, June 16, 2016.

"Visualization: A Tool for Exploration and Storytelling," Invited talk, Kwan-Liu Ma, Biophotonics Seminar, UC Davis, June 2, 2016.

"Visualizing Extreme Scale CFD Simulations," Plenary speech, Kwan-Liu Ma, *Parallel CFD 2016 Conference*, May 11, 2016.

"Visualization: A Tool for Data Exploration and Storytelling," Invited talk, Kwan-Liu Ma, Taipei Medical University, Taiwan, April 28, 2016.

"The In Situ Terminology Project," Hank Childs, *Department of Energy Computer Graphics Forum (DOECGF)*, April 28, 2016.

"Recent Advances in Visualization Research," Invited talk, Kwan-Liu Ma, Institute of Sociology, Academia Sinica, Taiwan, April 27, 2016.

"Big Data Visualization," Invited talk, Kwan-Liu Ma, *Summit Forum on Big Data Visualization*, Fudan University, Shanghai, China, April 14, 2016.

"Visualization Toolkit: Improving Rendering and Compute on GPUs," Presentation, Robert Maynard, *GPU Technology Conference*, April 2016.

"Adapting the Visualization Toolkit for Many-Core Processors with the VTK-m Library." Presentation, Christopher Sewell and Robert Maynard, *GPU Technology Conference*, April 2016.

"Exascale Visualization: What Will Change," Invited talk, Hank Childs, National Center for Atmospheric Research, Boulder, CO, March 2016.

"Topics in Visualization," Invited talk, Institute for Visualization and Interactive Systems, Kwan-Liu Ma, University of Stuttgart, Germany, March 11, 2016.

"Exploratory and Explanatory Visualization," Keynote speech, Kwan-Liu Ma, *3rd EMBO Conference on Visualizing Biological Data (VIZBI)*, March 9, 2016.

"Exascale Visualization: What Will Change." Invitied talk, National Center for Atmospheric Research, Boulder, CO, March 2016.

"XVis, VTK-m, and the ECP," Kenneth Moreland, Data/Vis Panel for the Exascale Computing Initiative Project, February 19, 2016.

"Data Visualization," Invited talk, Kwan-Liu Ma, *Medical Health Informatics*, UCDMC, Sacramento, CA, November 25, 2015.

"VTK-m: Building a Visualization Toolkit for Massively Threaded Architectures," Invited presentation, *Ultrascale Visualization Workshop*, November 2015.

"Visualization and High Performance Computing," Kwan-Liu Ma, Keynote speech, Symposium on Visualization in HPC, SIGGRAPH Asia, November 2, 2015.

"Advanced Concepts and Strategies for Visualizing Large-Scale, Complex Simulation Data," Kwan-Liu Ma, Invited Talk, *International Computational Accelerator Physics Conference (ICAP)*, October 14, 2015.

"Exascale Visualization: Get Ready for a Whole New World," Hank Childs, Invited talk, *International Computing for the Atmospheric Sciences Symposium (iCAS2015)*, Annecy, France, September 2015.

"VTK-m," Jeremy Meredith, FASTMath PI Meeting, September 2015.

"New Techniques for Visualizing Large-Scale Scientific Data," Kwan-Liu Ma, Invited talk, Software Center for High Performance Numerical Simulation, Chinese Academy of Engineering Physics, Beijing, China, September 2, 2015.

VTK-m Code Sprint, LLNL, September 1-2, 2015.

“VTK-m Overview,” Kenneth Moreland, VTK-m Code Sprint, September 1, 2015.

“Trends and Advanced Concepts for Scientific Visualization,” Kwan-Liu Ma, Keynote speech, China Scientific Data Conference, August 26, 2015.

“VTK-m: Accelerating the Visualization Toolkit for Multi-core and Many-core Architectures,” Christopher Sewell, et al., SciDAC PI Meeting (poster), July 2015.

“VTK-m,” Kenneth Moreland, DOECGF, April 2015.

“Hands-on Lab: In-Situ Data Analysis and Visualization: ParaView, Catalyst and VTK-m,” Marcus Hanwell and Robert Maynard, GTC Lab, March 2015.

“Visualization Toolkit: Faster, Better, Open Scientific Rendering and Compute,” Robert Maynard and Marcus Hanwell, GTC Presentation, March 2015.

“Roadmap for Many-Core Visualization Software in DOE,” Jeremy Meredith, GTC Presentation, March 2015.

4 Acknowledgement

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, under contract number 14-017566.

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy’s National Nuclear Security Administration under contract DE-AC04-94AL85000.

U.S. DEPARTMENT OF ENERGY Sandia National Laboratories

SAND 2016-9521 R