Top Research Challenges and Opportunities for Near Real-Time Extreme-Scale Visualization of Scientific Data

David Pugmire
Computer Science & Math Division
Oak Ridge National Laboratory
Oak Ridge, TN, USA
pugmire@ornl.gov

Kenneth Moreland
Computer Science & Math Division
Oak Ridge National Laboratory
Oak Ridge, TN, USA
morelandkd@ornl.gov

Tushar M. Athawale
Computer Science & Math Division
Oak Ridge National Laboratory
Oak Ridge, TN, USA
athawaletm@ornl.gov

James Hammer
Electrical Engineering & Computer Science
University of Tennessee, Knoxville
Knoxville, TN, USA
jhammer3@vols.utk.edu

Jian Huang
Electrical Engineering & Computer Science
University of Tennessee, Knoxville
Knoxville, TN, USA
huangj@utk.edu

***Abstract*—The rapid advancement in scientific simulations and experimental facilities has resulted in the generation of vast amounts of data at unprecedented scales. The analysis and visualization of large amounts of data is a challenge in and of itself, but the requirements for timeliness significantly magnify these difficulties. Near real-time visualization is critical to monitor and analyze the data produced by these large facilities, but current production tools are not well-suited to these requirements. In this position paper, we share our perspective on some of the challenges, and thus, opportunities for research that stand in the way of near-real-time visualization of large scientific data.**

***Index Terms*—Visualization, Real-time Systems, Human-computer Interactions**

I. Introduction

The continued growth of sensor and computational technology enables experimental, observational, and computational facilities, enabling the scientific community to probe deeper across a wide range of scientific inquiries. The continued advances in these technologies have resulted in a significant increase in the data produced. These increases include the amount of data, the type and varieties of data, and the velocity at which it is produced. Additionally, there are efforts underway to couple individual facilities together into computing ecosystems to study more complex scientific phenomena [1]. This in turn will magnify the already growing data problem.

Extracting understanding using visualization and analysis of this data will be critical to its success. Further, because of the cost and complexity, knowledge extraction in near real-time (NRT) is critical. In this context, we define NRT as a time constraint on required results. As such, NRT constraints will vary based on the context and scientific needs. NRT constraints could be as small as fractions of a second, or range from minutes to hours, or even days. The wide range of possibilities in NRT makes it particularly challenging as visualization and analysis routines must be adaptable to a wide range of environments and constraints.

As an example, scientists need quick feedback on a simulation running on an exascale computer to ensure that it is converging as expected or if the run should be killed to not waste costly resources. In experimental science, it is critical to ensure that the sensors are properly functioning and gathering the expected data.

Scientists only have a fixed amount of time to use these resources so timely feedback is critical to ensure that the maximum amount of insight can be gained from the time allocations on these systems. Visualization plays a unique role because the visual representations of complex data provide an efficient communication of information.

Large-scale parallel visualization poses a significant challenge in the realm of scientific computing, primarily due to the sheer volume and complexity of data generated by simulation and experimental facilities. These massive datasets can exceed terabytes or petabytes in size making it impossible to visualize on a single resource.

This challenge is compounded by the need for interactive exploration of visualization results to gain insight and understanding of the data. Tools like ParaView [2] and VisIt [3] have been developed that take advantage of parallel processing

This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a non-exclusive, paid up, irrevocable, world-wide license to publish or reproduce the published form of the manuscript, or allow others to do so, for U.S. Government purposes. The DOE will provide public access to these results in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

hardware and the use of client-server software architectures to distribute data processing and rendering tasks across multiple nodes. These tools also leverage VTK-m [4] to take advantage of parallel GPU processing on the nodes. These fully featured production tools support many visualization algorithms for the visualization of scalar and vector fields in data.

However, the production tools of today are not well suited to the demands of NRT. To perform visualization, estimates are made on the required time and amount of resources. NRT adaptation of either is challenging under the rigid constraints imposed by NRT. This results in inefficiency at best and lost opportunities for scientific insight at worst. This path is not sustainable going into the future as systems become more complex, the resources more heterogeneous and the data continue to grow. In this position paper, we discuss the top seven challenges to successful near-real-time visualization of extreme-scale data.

II. Top Seven Challenges

We have identified seven of the most significant challenges to NRT visualization, and describe each in this section. We also discuss the research opportunities associated with each as a way to expand the conversation in the community to research and develop solutions. The order of the challenges does not reflect their relative importance but rather the relative association of challenges with one another.

A. Human in the Loop and AI Assistants

Science projects that need NRT capabilities typically work in teams and depend on collaboration and fast-paced cross-disciplinary decision-making. The complexities of the decision-making scenarios are further exacerbated by the cognitive overload involved in the tasks. In addition, due to the restrictions on wall clock time before each task needs to be completed, there are scarcely any opportunities for scientists to interact with the visualization on their screen, vary the controls of the visualization to navigate the problem space, and iterate towards a deeper understanding of the situation at hand. This hampers the potential of more exact and optimal control of the scientific apparatus in the NRT context.

Due to the recent explosive growth or wider acceptance of AI methods, there is a great opportunity to now explore building AI assistants to help scientists with recurring cognitive tasks, for example, to inspect a sequence of visualizations to determine whether an experiment or simulation is progressing as intended, or whether a feature of interest has occurred. Such AI assistants will have natural explainability because the sequence of images can be easily inspected by a human scientist to confirm the classification made by an AI assistant. In addition, through reinforcement learning, which has become widely adopted in many fields, the human scientist can label the results as a natural step to guide the AI assistant, in ways very similar to how a senior scientist would train or mentor a junior scientist.

Fuguent [5] is an early example that demonstrates the feasibility of creating AI assistants and helping scientists reduce their cognitive overheads and increase the depth of their decision-making. Fugent is trained to inspect Poincaré plots, which is the de facto visualization tool to understand turbulent features in the plasma field of fusion reactors. Making discoveries using Poincaré plots requires significant expertise and experience because Poincaré plots are hard to interpret. In many cases, scientists with significant experience can simply follow their intuition. However, from a knowledge capture perspective, that kind of status quo hampers cross-disciplinary collaboration as well as reproducibility. Fugent used an Asynchronous Advantage Actor-Critic (A3C) framework to learn the sequences of how an expert scientist conducts their work. Fugent also uses Convolution Neural Net (CNN) and Long-Short Temporal Memory (LSTM). After training, Fugent can perform the same type of exploration fully automatically on behalf of the human scientist. The Fugent model takes only about 20MBs to store, thus making this type of assistant easy to share and archive.

With such early evidence of feasibility, there are three opportunities worth further investigation. (1) Creation of a community-shared, community-critiqued set of machinery so that the visualization community can more easily engage domain scientists and help them train tasks-specific AI assistants. (2) Creation of community-shared metrics or processes to evaluate the quality and capability of potentially a large library of task-specific agents. (3) Scaling an AI assistant for use in parallel settings, either symmetrically (i.e. the same agent doing the same task on different data) or heterogeneously (i.e. different agents doing different tasks but collaborating with each other).

B. Progressive Computation

The running time of visualization algorithms is notoriously difficult to predict with any accuracy because the extent of their operations is dependent on the values of the data. For example, rendering time can vary based on how rays get reflected in a scene [6] or how geometry is projected in image space [7]. Contouring time depends on the geometric size of the level sets extracted. The running time of many flow algorithms can depend on the path of the flow such as whether the flow makes loops or how many data partitions it crosses [8].

Consequently, it is difficult to guarantee a visualization algorithm will complete within NRT time constraints. Generally, one has to limit the algorithm to far less than what is possible to ensure these time constraints are met regardless of the form of the data, which provides far less information than what is possible.

A way to capture as much detail as possible while still guaranteeing time constraints is to design progressive visualization algorithms. These algorithms start with a coarse but viable solution and then iteratively refine. In this way, a solution can be guaranteed but surplus time is effectively used to provide more detail.

A visualization-relevant function that is already shown to provide useful iterative refinement is 3D rendering. A general

approach to managing the infinite possible pathways light may travel in a scene is to stochastically sample them [9], [10]. A viable but noisy representation can be provided quickly, and further ray samples may be accumulated to refine the result.

Because a progressive algorithm is used, the result may be coarse or noisy. This may be rectified by further post processing. AI methods in particular are a fruitful approach to clean early iterations. For example, ML models have been shown to accurately denoise images from undersampled raycast images [11].

C. Interruptable Algorithms

The previous challenge (II-B) argued the need for the dynamic adjustment in the amount of detail or precision a visualization algorithm provides. Implicit in this discussion is the need to start and stop iterations of the algorithm as the time it takes is monitored. For this to work, it must be possible to interrupt the algorithm as its runtime approaches the time constraint.

Whenever the running time of an algorithm cannot be bounded, there is a chance the algorithm is still running once a given time constraint is hit. At this point, any result provided by the algorithm is of no value. Any compute cycles spent after the deadline are clearly wasted. Worse, these cycles are likely taking valuable resources that are needed for a subsequent time constraint as the system progresses.

To make the most of resource utilization in this dynamic scheduling system, it is imperative that running computations be stopped once their result serves no purpose. Halting a visualization algorithm, particularly one that operates on parallel devices, can be non-trivial. Fast interruption can be achieved with frequent polling and communication, but these add overhead and impede the results under normal operation. Designing an efficient stopping mechanism can be a difficult operation in its own right.

D. Pervasive In Situ Processing

A major direction for current and future scientific inquiry is the coupling of multiple facilities to answer scientific questions. These facilities include high-performance computing (HPC), experimental and observational. These coupled systems will provide the ability to answer many new scientific questions. At the same time, these systems, which are complex, distributed, and heterogeneous, will be a significant challenge for the visualization tools of today. Further, to be responsive to the constraints of NRT, the visualization tools must be agile and capable of executing in several different modes. This includes disk-based post-processing modes and the large variety of in situ processing modes [12].

Post-hoc visualization tools that use bulk synchronous parallel processing are well established and can largely be considered a solved problem [13], [14]. Tools like ParaView [2] and VisIt [3] are widely used in full production systems. In situ processing paradigms have been an active area of research and resulted in many tools and systems [15]–[20]. Despite these efforts, production tools that are efficient and easily deployed across such complex environments are lacking. At present, most in situ visualizations tend to be problem-specific, difficult to generalize, and require collaborations between application and computer scientists and some bespoke software components. One of the primary findings identified in recent DOE workshop reports [21] is the need to overcome challenges that block in situ processing from being pervasive.

In a prior work [22], we described three properties needed by in situ visualization tools to become pervasive. Two of these, agility, and elasticity, are particularly relevant to the support of NRT analysis and visualization and are summarized below. *Agility* is the ability of visualizations to be easily used across a wide range of use cases. This includes multiple scientific domains, both post-hoc and in situ processing modes, and multiple interaction modes (e.g., notebooks, web, desktop tools, etc). *Elasticity* is a concept first developed by the cloud computing community where the efficient use of shared resources is critical. In a NRT setting, visualization tasks must be able to scale up and scale down to efficiently use available resources. This includes elasticity over a given resource as well as elasticity over a set of connected resources.

E. Time and Cost Models for Visualization Algorithms

The difficulty of determining the running time of a given algorithm is discussed in section II-B. This challenge is amplified when considering it from a system-wide perspective. To meet the time constraints in a NRT setting there is a cost. This cost can be measured in different ways, including node-hours, watt-hours, etc. For visualization, the challenge is driven by the cost differences between algorithm selection, in situ placement strategy, data complexity, and heterogeneous architectures (from HPC to edge). The choices made can have dramatic differences in the resulting cost.

The challenge is right-sizing the cost of a visualization with the NRT constraints. Meeting NRT constraints can involve many different decisions. These include the type of algorithm used, the amount and type of resource allocated, the amount and fidelity of the data and the quality and/or amount of resulting visualizations. If the cost to meet NRT constraints is too high, the necessity of the visualization, or the imposed time constraints are too aggressive, changes must be made.

To address these challenges, predictive cost models are needed. These predictive cost models take as input a description of the data (amount, velocity, format, accuracy, etc), amount and type of resource, execution strategy (e.g., post-hoc, in situ, in transit, etc) and algorithm, and give a predicted time for execution. Initial work has been done investigating the predictive models for different visualization methods. This includes in situ rendering algorithms [23], costs, in terms of node-hours for in situ and in transit processing paradigms [24], [25], and time to solution for both in situ and in transit processing [26].

This space needs additional exploration to develop predictive cost models that cover a much wider range of algorithm types, execution modes, and amounts and types of data. Such models could be used in a constrained optimization approach

to determine the data, resource, and execution strategy requirements to meet a given time budget. If for example, the resource requirements are too high, or the data requirements are too aggressive to meet the time constraints, decisions can be made to adjust the priority, schedule appropriate resources, or adjust expectations. In the context of a complex workflow which are increasingly used to control scientific campaigns, these models will be critical in balancing the required resources across a wide variety of tasks that must be completed.

F. Data Refactoring

When NRT systems involve large volumes of data, data refactoring becomes a key concern. Data refactoring transforms data and often allows information to be represented with less memory. This can be a necessary port of capturing data in a NRT system.

Common general forms of data refactoring are the compression techniques used to reduce the size of the data. Data from physical systems generally require "lossy" techniques to get a reasonable compression [27], which means that the error introduced by the compression must be managed. This can be done by expressing a maximum tolerance for the error where a trade-off between error and compression ratio [28]. However, if the loss can be expressed in terms of added uncertainty, the techniques suggested in Section II-G can be applied. This would allow the tolerance of the compression to be relaxed and potentially increase the compression ratio achieved.

Other less general forms of refactoring are also possible. The data collected in a NRT system may be in a more verbose form than is strictly needed. In this case, analysis might be done in parts where early parts will refactor the data to a manageable format and other parts, perhaps offline, will complete the desired visualization. A similar division of operations is also a characteristic of workflows that combine different forms of in situ visualization [29], [30].

G. Data Uncertainity

Vast amounts of data from simulations and experiments are accompanied by uncertainty from data sources, complex HPC workflows, and algorithms. This uncertainty is nontrivial to analyze but cannot be ignored. Communication of uncertainty is crucial to improving decision-making and increasing trust in results [31]–[33]. Visualization of uncertainty in NRT is particularly challenging, considering additional memory and computational costs of analyzing uncertainty. For example, representing uncertainty with the mean and width (or standard deviation) doubles the amount of data. Uncertainty representation with more complex models, e.g., histograms, can further increase memory and computational bottlenecks [34], [35]. Further, the existing brute-force Monte Carlo algorithms for uncertainty visualization [36]–[38] cannot be accommodated in NRT systems because of their high processing costs, slow convergence, and poor scaling with increase in size and dimensionality of data.

Thus, there are two opportunities relevant to uncertainty visualization in the context of NRT that must be investigated. (1) Finding efficient ways uncertainty can be visualized to meet time constraints of NRT visualization systems. (2) Determining how uncertainty could be used as a decision-making mechanism to achieve NRT extreme-scale visualization. A few recent developments replaced expensive Monte Carlo sampling with closed-form solutions [35], [39], [40] and combined them with GPU acceleration [41] and AI [42] approaches for NRT uncertainty rendering. The research in closed-form solutions, hardware acceleration, and AI models for uncertainty visualization, however, is in its very early stages. Further, even though a few studies investigated how different probability models (e.g. uniform, Gaussian, histogram) affect cost and accuracy of visualization, little research has focused on how models can be adaptively chosen to achieve NRT results with maximal accuracy. Thus, research in development of efficient uncertainty analysis algorithms and their adaptive use must be pushed for trustworthy NRT visualization of large data.

III. Discussion

In this paper, we have identified and discussed the top 7 challenges associated with near-real-time visualization of large scientific data. These challenges include a wide array of issues. While there are some similarities to the general challenges associated with the visualization of large data, the requirements for NRT interaction result in some very particular challenges. Identifying and understanding these challenges is critical in identifying areas where additional research is needed.

The challenges presented highlight the multifaceted nature of near-real-time visualization. As such, collaboration among a range of sub-disciplines and co-design of solutions will be critical to providing the necessary functionality for next-generation science.

The intent of this paper is to offer our considered opinions of the work that is required and provide a roadmap for researchers and practitioners aiming to provide visualization solutions for large data in time- and resource-constrained environments. We also hope this paper serves as a starting point for increased conversation on NRT visualization and to foster collaboration among researchers to provide solutions for the scientific community.

References

[1] William L. Miller, Deborah Bard, Amber Boehnlein, Kjiersten Fagnan, Chin Guok, Eric Lançon, Sreeranjani Ramprakash, Mallikarjun Shankar, Nicholas Schwarz, and Benjamin L. Brown. Integrated research infrastructure architecture blueprint activity (final report 2023).

[2] James Paul Ahrens, Berk Geveci, and C. Charles Law. Paraview: An end-user tool for large-data visualization. In *The Visualization Handbook*, 2005.

[3] Hank Childs, Eric Brugger, Brad Whitlock, Jeremy Meredith, Sean Ahern, Kathleen Bonnell, Mark Miller, Gunther Weber, Cyrus Harrison, David Pugmire, Thomas Fogal, Christoph Garth, Allen Sanderson, E. Wes Bethel, M. Durant, David Camp, Jean Favre, O. Rübel, Paul Navratil, and F. Vivodtzev. VisIt: An end-user tool for visualizing and analyzing very large data. *SciDAC*, pages 1–16, 01 2011.

[4] 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. VTK-m: Accelerating the visualization toolkit for massively threaded architectures. *IEEE Computer Graphics and Applications*, 36(3):48–58, May/June 2016.

[5] James Hammer, Tanner Hobson, David Pugmire, Scott Klasky, Kenneth Moreland, and Jian Huang. A personalized AI assistant for intuition-driven visual explorations. In *IEEE Intl Conference on eScience*, Sept. 2024.

[6] Peter Shirley. *Realistic Ray Tracing*. CRC Press, second edition, 2003.

[7] Kenneth Moreland, Wesley Kendall, Tom Peterka, and Jian Huang. An image compositing solution at scale. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis (SC '11)*, November 2011.

[8] Robert Sisneros and David Pugmire. Tuned to terrible: A study of parallel particle advection state of the practice. In *2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 1058–1067, May 2016.

[9] Marc Levoy. Volume rendering by adaptive refinement. *The Visual Computer*, 6(1):2–7, January 1990.

[10] Naohisa Sakamoto, Takuma Kawamura, Hiroshi Kuwano, and Koji Koyamada. Sorting-free pre-integrated projected tetrahedra. In *Proceedings of the 2009 Workshop on Ultrascale Visualization*, November 2009.

[11] Attila T. Áfra. Intel® Open Image Denoise, 2024. https://www.openimagedenoise.org.

[12] Hank Childs, Sean D. Ahern, James Ahrens, Andrew C. Bauer, Janine Bennett, E. Wes Bethel, Peer-Timo Bremer, Eric Brugger, Joseph Cottam, Matthieu Dorier, Soumya Dutta, Jean M. Favre, Thomas Fogal, Steffen Frey, Christoph Garth, Berk Geveci, William F. Godoy, Charles D. Hansen, Cyrus Harrison, Bernd Hentschel, Joseph Insley, Chris R. Johnson, Scott Klasky, Aaron Knoll, James Kress, Matthew Larsen, Jay Lofstead, Kwan-Liu Ma, Preeti Malakar, Jeremy Meredith, Kenneth Moreland, Paul Navrátil, Patrick O'Leary, Manish Parashar, Valerio Pascucci, John Patchett, Tom Peterka, Steve Petruzza, Norbert Podhorszki, David Pugmire, Michel Rasquin, Silvio Rizzi, David H. Rogers, Sudhanshu Sane, Franz Sauer, Robert Sisneros, Han-Wei Shen, Will Usher, Rhonda Vickery, Venkatram Vishwanath, Ingo Wald, Ruonan Wang, Gunther H. Weber, Brad Whitlock, Matthew Wolf, Hongfeng Yu, and Sean B. Ziegeler. A terminology for in situ visualization and analysis systems. *The International Journal of High Performance Computing Applications*, 34(6):676–691, 2020.

[13] Hank Childs, Dave Pugmire, Sean Ahern, B. Whitlock, Mark Howison, NFN Prabhat, Gunther H Weber, and E Wes Bethel. Visualization at extreme-scale concurrency. 1 2012.

[14] Hank Childs, David Pugmire, Sean Ahern, Brad Whitlock, Mark Howison, Prabhat, Gunther H. Weber, and E. Wes Bethel. Extreme scaling of production visualization software on diverse architectures. *IEEE Computer Graphics and Applications*, 30(3):22–31, 2010.

[15] William F. Godoy, Norbert Podhorszki, Ruonan Wang, Chuck Atkins, Greg Eisenhauer, Junmin Gu, Philip Davis, Jong Choi, Kai Germaschewski, Kevin Huck, Axel Huebl, Mark Kim, James Kress, Tahsin Kurc, Qing Liu, Jeremy Logan, Kshitij Mehta, George Ostrouchov, Manish Parashar, Franz Poeschel, David Pugmire, Eric Suchyta, Keichi Takahashi, Nick Thompson, Seiji Tsutsumi, Lipeng Wan, Matthew Wolf, Kesheng Wu, and Scott Klasky. Adios 2: The adaptable input output system. a framework for high-performance data management. *SoftwareX*, 12:100561, 2020.

[16] Matthew Larsen, Eric Brugger, Hank Childs, and Cyrus Harrison. Ascent: A Flyweight In Situ Library for Exascale Simulations. In *In Situ Visualization For Computational Science*, pages 255 – 279. Mathematics and Visualization book series from Springer Publishing, Cham, Switzerland, May 2022.

[17] Utkarsh Ayachit, Andrew Bauer, Berk Geveci, Patrick O'Leary, Kenneth Moreland, Nathan Fabian, and Jeffrey Mauldin. Paraview catalyst: Enabling in situ data analysis and visualization. In *Proceedings of the First Workshop on In Situ Infrastructures for Enabling Extreme-Scale Analysis and Visualization (ISAV 2015)*, pages 25–29, November 2015.

[18] Utkarsh Ayachit, Andrew C. Bauer, Ben Boeckel, Berk Geveci, Kenneth Moreland, Patrick O'Leary, and Tom Osika. Catalyst revised: Rethinking the paraview in situ analysis and visualization API. In *High Performance Computing*, pages 484–494, June 2021.

[19] Brad Whitlock, Jean M. Favre, and Jeremy S. Meredith. Parallel In Situ Coupling of Simulation with a Fully Featured Visualization System. In Torsten Kuhlen, Renato Pajarola, and Kun Zhou, editors, *Eurographics Symposium on Parallel Graphics and Visualization*. The Eurographics Association, 2011.

[20] E. Wes Bethel, Burlen Loring, Utkarsh Ayachit, David Camp, Earl P. N. Duque, Nicola Ferrier, Joseph Insley, Junmin Gu, James Kress, Patrick O'Leary, David Pugmire, Silvio Rizzi, David Thompson, Gunther H. Weber, Brad Whitlock, Matthew Wolf, and Kesheng Wu. The sensei generic in situ interface: Tool and processing portability at scale. In Hank Childs, Janine C. Bennett, and Christoph Garth, editors, *In Situ Visualization for Computational Science*, pages 281–306, Cham, 2022. Springer International Publishing.

[21] Tom Peterka and et al. ASCR workshop on in situ data management: Enabling scientific discovery from diverse data sources. Technical report, U.S. DOE ASCR, 2 2019.

[22] David Pugmire, Jian Huang, Kenneth Moreland, and Scott Klasky. The need for pervasive in situ analysis and visualization (p-isav). In Hartwig Anzt, Amanda Bienz, Piotr Luszczek, and Marc Baboulin, editors, *High Performance Computing. ISC High Performance 2022 International Workshops*, pages 306–316, Cham, 2022. Springer International Publishing.

[23] Matthew Larsen et al. Performance modeling of in situ rendering. In *SC'16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 276–287. IEEE, 2016.

[24] James Kress et al. Comparing the efficiency of in situ visualization paradigms at scale. In *International Conference on High Performance Computing*, pages 99–117. Springer, 2019.

[25] James Kress et al. Opportunities for cost savings with in-transit visualization. In *ISC High Performance 2020*. ISC, 2020.

[26] James Kress, Matthew Larsen, Jong Choi, Mark Kim, Matthew Wolf, Norbert Podhorszki, Scott Klasky, Hank Childs, and David Pugmire. Comparing time-to-solution for in situ visualization paradigms at scale. In *2020 IEEE 10th Symposium on Large Data Analysis and Visualization (LDAV)*, pages 22–26, 2020.

[27] Mark Ainsworth, Scott Klasky, and Ben Whitney. Compression using lossless decimation: Analysis and application. *SIAM Journal on Scientific Computing*, 39(4):B732–B757, 2017.

[28] Jieyang Chen, David Pugmire, Matthew Wolf, Nicholas Thompson, Jeremy Logan, Kshitij Mehta, Lipeng Wan, Jong Youl Choi, Ben Whitney, and Scott Klasky. Understanding performance-quality trade-offs in scientific visualization workflows with lossy compression. In *5th International Workshop on Data Analysis and Reduction for Big Scientific Data (DRBSD-5)*, November 2019.

[29] Janine C. Bennett, Hasan Abbasi, Peer-Timo Bremer, Ray Grout, Attila Gyulassy, Tong Jin, Scott Klasky, Hemanth Kolla, Manish Parashar, Valerio Pascucci, Philippe Pebay, David Thompson, Hongfeng Yu, Fan Zhang, and Jacqueline Chen. Combining in-situ and in-transit processing to enable extreme-scale scientific analysis. In *Proceedings of the Conference on High Performance Computing, Networking, Storage and Analysis (SC '12)*, number 49, 2012.

[30] Valentin Bruder, Matthew Larsen, Thomas Ertl, Hank Childs, and Steffen Frey. A hybrid in situ approach for cost efficient image database generation. *IEEE Transactions on Visualization and Computer Graphics*, 29(9):3788–3798, September 2023.

[31] Ken Brodlie, Rodolfo Allendes Osorio, and Adriano Lopes. A review of uncertainty in data visualization. In John Dill, Rae Earnshaw, David Kasik, John Vince, and Pak Chung Wong, editors, *Expanding the Frontiers of Visual Analytics and Visualization*, pages 81–109. Springer Verlag London, 2012.

[32] Kristin Potter, Paul Rosen, and Chris R. Johnson. From quantification to visualization: A taxonomy of uncertainty visualization approaches. In Andrew M. Dienstfrey and Ronald F. Boisvert, editors, *Uncertainty Quantification in Scientific Computing*, pages 226–249. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[33] Aasim Kamal, Parashar Dhakal, Ahmad Y. Javaid, Vijay K. Devabhaktuni, Devinder Kaur, Jack Zaientz, and Robert Marinier. Recent advances and challenges in uncertainty visualization: a survey. *Journal of Visualization*, 24(5):861–890, May 2021.

[34] Kai Pöthkow and Hans-Christian Hege. Nonparametric models for uncertainty visualization. *Computer Graphics Forum*, 32(3pt2):131–140, July 2013.

[35] Tushar M. Athawale, Bo Ma, Elham Sakhaee, Chris R. Johnson, and Alireza Entezari. Direct volume rendering with nonparametric models of uncertainty. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):1797–1807, Feb. 2021.

[36] Kai Pöthkow, Britta Weber, and Hans-Christian Hege. Probabilistic marching cubes. *Computer Graphics Forum*, 30(3):931–940, June 2011.

[37] Christoph Petz, Kai Pöthkow, and Hans-Christian Hege. Probabilistic local features in uncertain vector fields with spatial correlation. *Computer Graphics Forum*, 31(3pt2):1045–1054, 2012.

[38] Shusen Liu, Joshua A. Levine, Peer-Timo Bremer, and Valerio Pascucci. Gaussian mixture model based volume visualization. In *IEEE Symposium on Large Data Analysis and Visualization (LDAV)*, pages 73–77, Oct. 2012.

[39] Tushar M. Athawale, Sudhanshu Sane, and Chris R. Johnson. Uncertainty visualization of the marching squares and marching cubes topology cases. In *2021 IEEE Visualization Conference (VIS)*, pages 106–110, 2021.

[40] Tushar M. Athawale, Chris R. Johnson, Sudhanshu Sane, and David Pugmire. Fiber uncertainty visualization for bivariate data with parametric and nonparametric noise models. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):613–623, 2023.

[41] Zhe Wang, Tushar M. Athawale, Kenneth Moreland, Jieyang Chen, Chris R. Johnson, and David Pugmire. $FunMC^2$: A Filter for Uncertainty Visualization of Marching Cubes on Multi-Core Devices. In Roxana Bujack, David Pugmire, and Guido Reina, editors, *Eurographics Symposium on Parallel Graphics and Visualization*. The Eurographics Association, 2023.

[42] Mengjiao Han, Tushar M. Athawale, David Pugmire, and Chris R. Johnson. Accelerated probabilistic marching cubes by deep learning for time-varying scalar ensembles. In *IEEE Visualization Conference (VIS)*, pages 155–159, 2022.