Visualization and Data Analytics Challenges of Large-Scale High-Fidelity Numerical Simulations of Wind Energy Applications

Andrew C. Kirby*, Zhi Yang†, Dimitri J. Mavriplis‡

Department of Mechanical Engineering, University of Wyoming, Laramie, WY 82071, USA

Earl P. N. Duque§, Brad J. Whitlock¶

Intelligent Light, Rutherford, NJ 07070, USA

Visualization and data analysis techniques are explored to alleviate big-data problems found in simulations regarding wind energy applications including full wind farm simulations with blade-resolved geometries for wind turbines. Techniques for streamlining workflows for large-scale simulations are investigated and instrumented in the $\mathbf{W^2A^2KE3D}$[1] software framework. *In-situ* analysis through Libsim is instrumented and used to export data of high-fidelity wind turbine simulations that is post-processed using FieldView and VisIt.

I. Introduction

Visualization and data analytics for large-scale simulations are essential for unlocking the secrets held within data generated from these simulations. The ever evolving parallel computing capabilities continue enabling higher fidelity simulations with the caveat of increased data generation and processing. The rate of compute performance at exascale will greatly outpace storage and I/O performance. The petascale approach of storing complete data sets for later analysis becomes intractable and impossible due to these bottlenecks given current technology trends into exascale; this approach will lead to lost science. Further, writing large quantities of data to disk requires large amounts of time and energy.

In the exascale era, data analysis and visualization must be co-designed and tightly coupled with application development. Modernization techniques such as *in-situ* visualization and analysis will avoid costly data movement and results will be generated as part of the simulation. In-situ processing consists of creating images, performing analysis on the data, or creating data extracts. Creating data extracts can reduce data amounts by orders of magnitude. In-situ analysis reduces I/O costs and allows for output of analyzed results, which are small in size, at higher temporal frequencies.

This work highlights the challenges and insights involved in visualization and analysis of large-scale high-fidelity simulations oriented towards wind energy applications. Efforts to couple the simulation framework with visualization and analysis through in-situ libraries are demonstrated. The implementation of the in-situ library through an abstraction interface allows for locally lossless, high-order data and visualization extraction which is essential for accurate representation of the numerical simulation while providing several orders of magnitude data reduction.

This paper is organized as follows: In Section II, a discussion of current in situ technologies is presented with example software frameworks. Section III introduces the application background and motivation of wind energy application simulations. Details regarding in-situ implementation and workflow are outlined in Section IV, followed by results of single and double wind turbine simulations, and concluded with a full wind farm simulation containing 48 wind turbines in Section V. Lastly, conclusions and future work are discussed in Section VI.

*Corresponding author: Doctoral Student, AIAA student member; akirby@uwyo.edu

†Research Scientist, AIAA member, zyang@uwyo.edu

‡Professor, Max Castagne Professorship, AIAA Associate Fellow; mavripl@uwyo.edu

§Manager of Applied Research, AIAA Associate Fellow; epd@iight.com

¶Post-Processing and Visualization Engineer, Applied Research Group; bjw@ilight.com

II. Related Large-Scale Data Analysis Works

In-situ processing, analysis and visualization techniques have evolved since the 1990s beginning with one of the earliest attempts of in-situ visualization by Haimes[2] to visualize large unsteady computational fluid dynamics simulations. The CATALYST[3] in-situ library was introduced into ParaView by Fabien *et al.*, and similarly Libsim[4] was developed for VisIt by Whitlock *et al.*. Lofstead *et al.* implemented ADIOS[5] as an in-situ analysis framework allowing for *in-transit* analysis, which is the ability to perform analysis when data is staged on a separate compute resource. GLEAN[6] was proposed by Vishwanath *et al.* for simulation-time data analysis and I/O acceleration. Recent efforts to abstract a generic data interface for multiple in-situ libraries has been developed through the SENSEI[7] API. The SENSEI generic data interface allows for several possibilities of in-situ, in-transit, in-flight, and hybrid analysis with capabilities to map data to future high-performance computing architectures.

III. Application Background and Motivation

Wind energy is becoming an important clean energy alternative to traditional energy which is essential for the energy security of the United States and the world abroad. Engineering advancements have made wind a positive option as turbine designs have increased in size, enabling greater energy extraction and bringing down operational cost.[8] To fully optimize wind turbine energy extraction, multiphysics and multiscale simulations are essential for understanding the complex phenomenon. This starts at the small scale with wind turbine blade aerodynamics and structural analysis, increasing to turbine-turbine wake interactions, and finally arriving at atmospheric inflow wind plant interactions, which encompass regional weather effects. The large scale atmospheric coupling effects introduce large-scale flow features that impact wind turbine structural loading and dictate wind turbine wake impingement effects on downstream wind turbines, causing a drop in overall power production.

Data sets generated by high-fidelity numerical simulations of wind energy applications that resolve the blade aerodynamics can span 10-12 orders of magnitude in spatial scales and 4-6 orders of magnitude in temporal scales. Saving frequent volume data-sets can quickly accumulate to hundreds or thousands of terabytes of data. In-situ analysis techniques allow for data extraction and collection at 10x to 1000x data reduction.

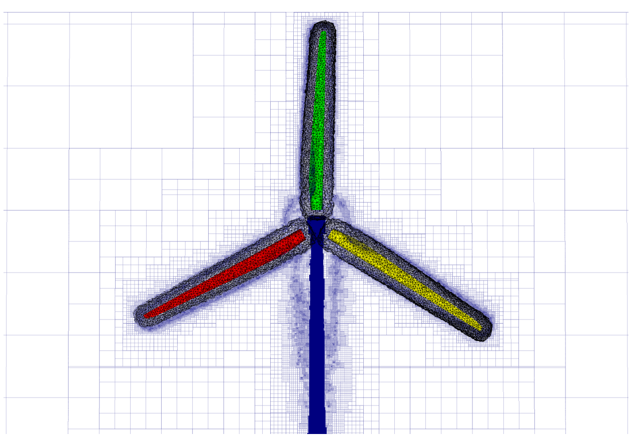


Figure 1. Wind turbine overset mesh system. One turbine blade unstructured mesh is replicated three times, rotated and translated to the initial positions. A fourth unstructured mesh is used to represent the tower and nacelle. The off-body adaptive mesh is visualized in the background.

For high-fidelity blade-resolved wind turbine simulations, this work employs the **W**yoming **W**ind and **A**erospace **A**pplications **K**omputation **E**nvironment (W^2A^2KE3D)[1] software framework. W^2A^2KE3D is designed to support a dynamic overset system using multiple computational fluid dynamics (CFD) solvers and multiple computational meshes. The mesh system generally consists of a collection of *near-body* and *off-body* meshes. The near-body meshes are inherently unstructured and highly anisotropic to model the complex geometry and resolve boundary layers of wind turbine bodies. The off-body mesh is a dynamically adaptive Cartesian grid system which allows for efficient solvers, efficient storage, and ease of dynamic solution-based mesh adaption. This multiple mesh paradigm allows for effective use of solver and mesh technologies in variable flow conditions. This hypothesis is strongly supported in the context of wind energy applications that require capturing boundary layer phenomenon as well as wake dynamics. Figure 1 demonstrates an overset mesh system of a three-bladed wind turbine with a tower and nacelle. Each blade is represented by a single mesh that is replicated, transitioned, and rotated to the correct starting position. The tower geometry is fitted with an independent unstructured near-body mesh component. Lastly, an adaptive Cartesian background mesh provides the inflow and wake tracking computational domain.

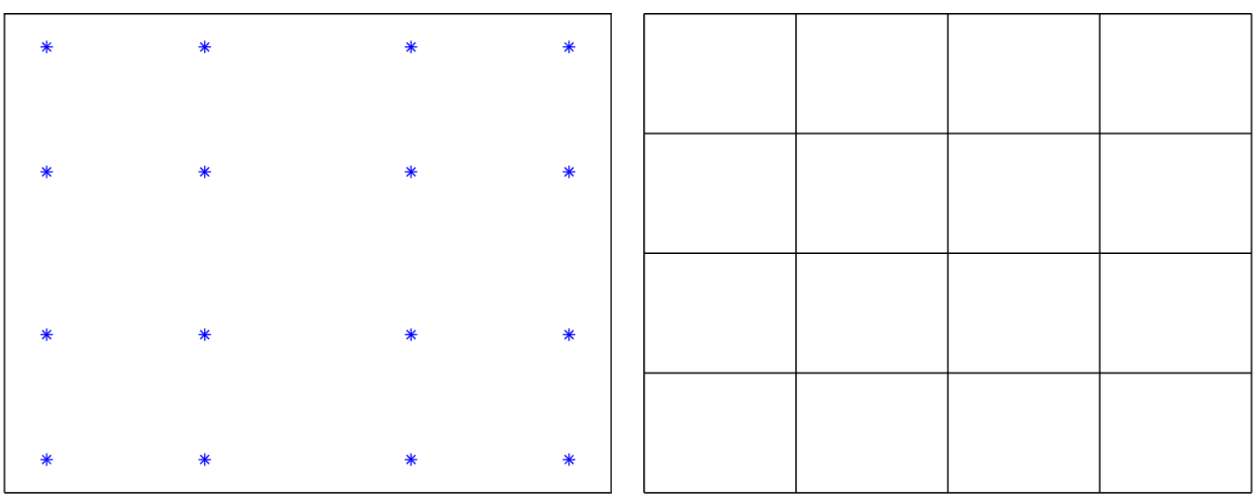
The W^2A^2KE3D framework allows for multiple CFD solvers individually optimized for their respective mesh system in the multiple-mesh paradigm. The use of multiple meshes and multiple flow solvers introduces the requirement of coordination. A driver program choreographs all flow solvers and all mesh systems. This driver allows for solvers to run on disjoint groups of CPU cores, allowing for variable amounts of computational resources to be allocated appropriately where needed. This is particularly important for the off-body solver which uses a dynamically adaptive mesh. During the evolution of a wind turbine simulation, the flow features of interest such as the wake require additional mesh resolution. As the propagation of the wake grows over time, more computational resources are required in the off-body regions. The flow solvers present in the framework can be redistributed to different numbers of cores at the beginning of restarted simulations. This allows for long run-time simulations to be moderately load balanced.

The two flow solvers are NSU3D,[9,10] an unstructured finite-volume solver, and dg4est,[11] a high-order discontinuous Galerkin finite-element solver embedded into the *p4est*[12,13] adaptive mesh refinement framework. The overset mesh assembler used is TIOGA[14,15,16] which is used to exchange solutions between the near-body and off-body meshes. The W^2A^2KE3D framework has been applied to several wind energy applications[17] including a full wind farm simulation[1] containing 48 wind turbines on over 22,000 cores and has been scaled weakly up to 144 wind turbines using over 67,000 cores. Additionally, it has been applied to several aerospace applications.[11,18]

IV. In-Situ Implementation and Workflow

This work utilizes the VisIt Libsim[4] in-situ library, which is implemented directly into the driver program that coordinates all flow solvers within the W^2A^2KE3D framework. This procedure is tightly coupled as the driver program has access to all flow solver solution pointers, which are passed directly through to the Libsim interface. Thus any CFD solver implemented into the framework gains access to in-situ visualization and data analysis capabilities automatically. The Libsim interface is directed by input scripts which are dynamically read allowing for changes during run-time. For example, without stopping and restarting the simulation, the user may change any parameters related to the in-situ visualization such as file output type and frequency, iso-contour values or variables, or cut-plane locations or variables.

A data adapter is developed to convert simulation data to and from the Libsim data model. This data adapter is ubiquitous with finite-volume and finite-element method data; the data model assumes a point-wise data structure with linear elements between solution points. In order to achieve full resolution visualization for the high-order finite elements, subdivision of the finite-element cell solution is performed to generate multiple linear sub-cells. This subdivision is flexible in the choice of the number of linear sub-cells. Practically, we subdivide a p-degree high-order element into $p + 1$ uniform sub-cells in each coordinate direction as shown in Figure 2. This technique enables output of higher-order iso-surfaces and cut-planes at run-time which provides lossless data reduction by avoiding output of large CFD volume data.



(a) High-order element solution points. (b) Subdivided element into linear sub-elements.

Figure 2. High-order element subdivision for higher-order plotting.

Traditional CFD workflows write large volume data sets to disk for later post analysis. Figure 3 demonstrates the traditional post-processing workflow starting with CFD data being read into a visualization software where objects are created, such as cutting planes, iso-surfaces, and stream-lines, followed by actions performed on the newly formed objects, including visualization, animation, and image exportation. This procedure may be costly and time-consuming for object construction, which at the completion of the post-processing session, is discarded. Technical debt may incur for this workflow if the analysis is incomplete and post-processing needs to be repeated.

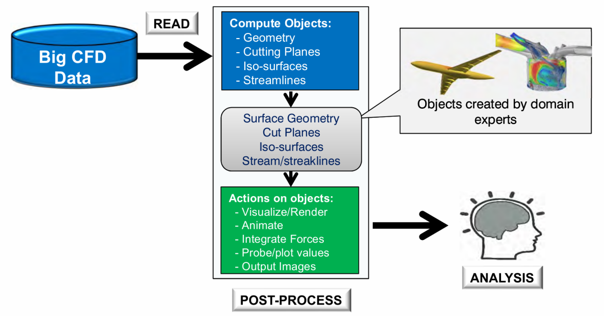
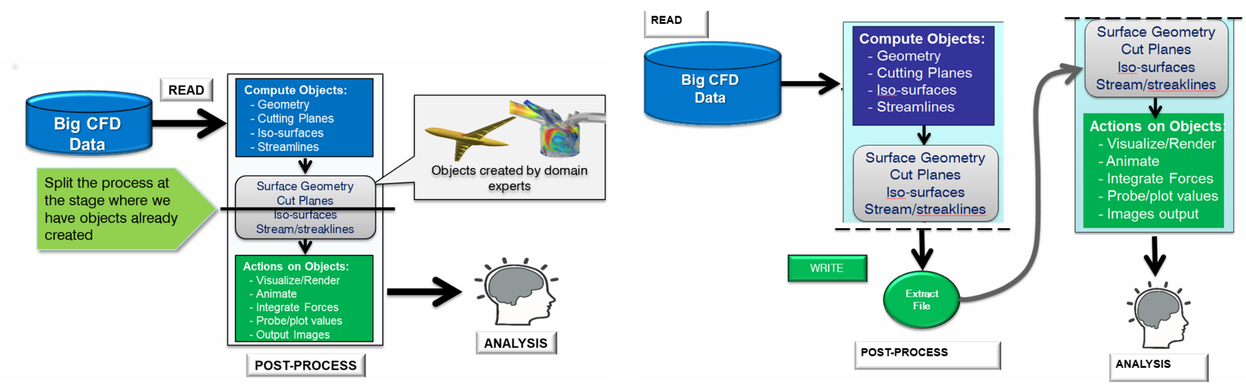


Figure 3. Traditional post-processing workflows used for scientific analysis.

To combat this issue and increase productivity, the post-processing workflow can be split into multiple stages as shown in Figure 4(a). After the data objects are created in the scientific post-processing software, they are extracted and stored for repeated use, as demonstrated in Figure 4(b), and, further, new data objects can be created from the data extracts. One such extraction-based workflow is the FieldView eXtract DataBase (XDB) approach. In an XDB workflow, shown in Figure 5, the post-processing objects are created and saved in FieldView as an XDB file, which can be used repeatedly.



(a) Splitting the post-processing workflow

(b) Extract-based workflow

Figure 4. Extract-based workflows allow for continued and repeated analysis on data extracts.

Further, an in-situ workflow can be adopted where the data objects that are usually extracted during post-processing are extracted during run-time instead. This increases productivity substantially by avoiding costly data writing and transferring of large volume datasets. Figure 6 shows a combined in-situ XDB workflow which starts with data extraction using VisIt Libsim, then directly exports XDB files.

We adopt a hybrid in-situ workflow using VisIt Libsim for the run-time data extraction of data objects. The data can be written as XDB files or Silo[19] files, which can be visualized in parallel via remote-hosting with FieldView or VisIt, respectively. Using the remote-host capability, the extracted data objects remain on the computing host, thereby removing data movement and, thus, increasing productivity. Using VisIt, imported data can be exported as a different file type; VisIt currently supports 12 export formats[4] including FieldView XDB.

Figure 5. FieldView eXtract DataBase (XDB) workflow.

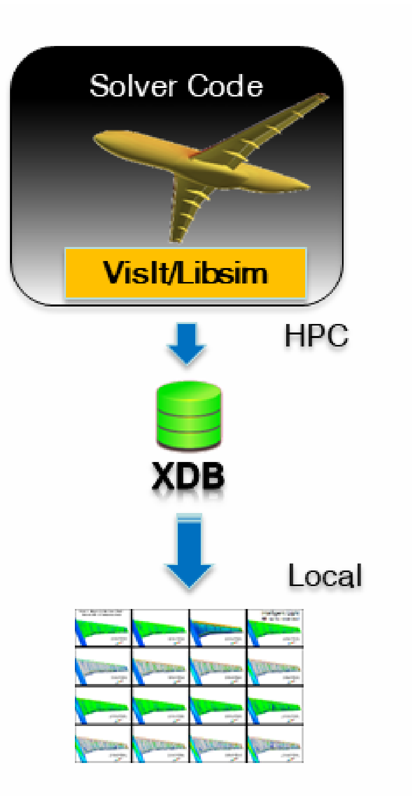


Figure 6. In-situ XDB workflow: VisIt Libsim can directly output XDB formatted files.

V. Results

A single NREL WindPACT-1.5MW wind turbine simulation is performed using a in-situ visualization techniques. Additionally, a two turbine Siemens SWT-2.3-93 wind turbine simulation is performed and visualized with VisIt Libsim. Lastly, a complete wind farm with 48 wind turbines is performed highlighting remote-host parallel visualization.

V.A. NREL WindPACT-1.5MW

Simulation of a NREL WindPACT-1.5MW wind turbine is performed and analyzed using in-situ data reduction techniques. To perform statistical analysis, high temporal frequency output of every 2° of rotor revolution resolution is required. This simulation is performed for 50 revolutions which aggregates to 9,000 time instances of data required. Figure 7 demonstrates in-situ extracts visualized in VisIt showing the wake evolution through seven cut-planes across the wake as a function of rotor diameter D downstream of the wind turbine. At each output time instance, 7 cut-planes of resolution 400 by 400 are output containing five flow variables and three coordinates. Additionally, a center plane of resolution 2000 by 400 with the same variables is extracted and is shown in Figure 8. An iso-contour of velocity magnitude is shown Figure 9 demonstrating the complex tip vortex phenomenon evolving down stream of the wind turbine.

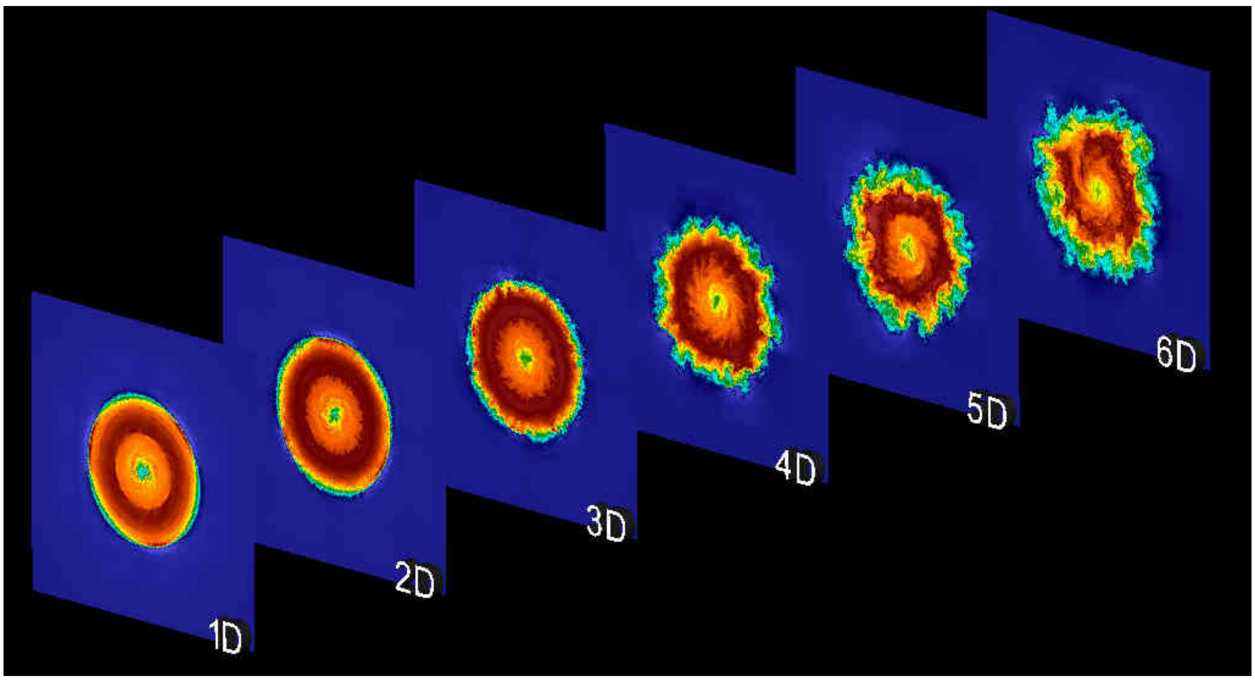


Figure 7. Downstream cross flow cut-planes at various positions of the NREL WindPACT-1.5MW wid turbine.

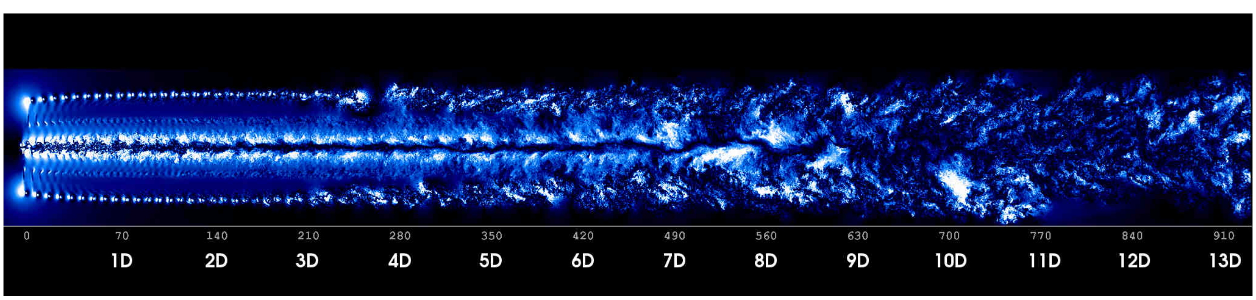


Figure 8. Instantaneous flow visualization of the the NREL WindPACT-1.5MW wind turbine. The absolute tangential flow velocity is visualized demonstrating the wake propagation downstream annotated by rotor diameter lengths.

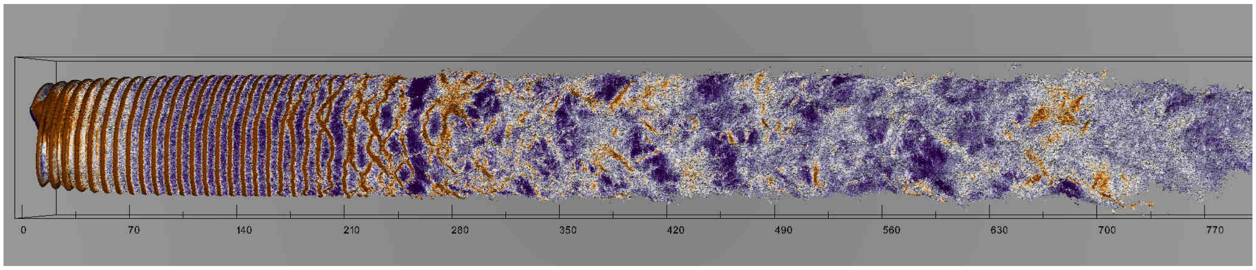


Figure 9. An iso-contour of the velocity magnitude demonstrating the vortex structure evolution of the NREL WindPACT-1.5WM wind turbine. The data extract were creating using in-situ in Libsim, and the data was post-processed using VisIt.

For this case, the off-body volume data grew linearly over the simulation starting at a size of 629 MB and ending at 51.8 GB. If the volume data had to be output at the same 2° rotor revolution frequency over 50 rotor revolutions, a total of 235,955.4 GB (230.4 TB) of data would have been written to disk. Alternatively, using in-situ analysis outputting seven cut-planes at 0.0095 GB per plane and one center cut-plane at 0.04 GB per plane resulted in a total of 960 GB of data. This results in a data reduction factor of 246. Table 1 summarizes the results of the in-situ data reduction.

Table 1. Single wind turbine data reductions obtained by in-situ workflow. A total of 72,008 cut-planes were written over 50 rotor revolutions in place of 9001 volume data files.

Volume Files	Volume Size (GB)	Cut-Planes	Cut-Plane Size (GB)	Data Reduction
9001	235,955.4	72,008	960	246.2x

V.B. Two Siemens SWT-2.3-93 Wind Turbines

Simulation of two generic Siemens SWT-2.3-93 wind turbines using specifications from the IAE Wind Task 31-Wakebench[20] is performed. The wind turbine contains three blades and a tower with a nacelle. Each near-body blade mesh contains 2,219,940 nodes and a tower with nacelle mesh containing 504,960 nodes. The off-body was simulated using a $p = 2$, third-order accurate, discontinuous Galerkin discretization.

The simulation was performed on the NWSC-Wyoming Cheyenne[21] supercomputer using 5,328 cores decomposed into 720 cores for the near-body unstructured meshes and 4,608 cores for the off-body adaptive Cartesian mesh. Figure 10 shows two iso-contours of Q-criterion of the turbine wakes. Figure 11 demonstrates the adaptive off-body mesh tracking the turbulent wake features being shed from the wind turbines. Table 2 presents an example of the data reduction that was attained by using the in-situ workflow. A reduction factor of 11.6 is achieved by directly extracting multiple iso-surfaces and cut-planes at run-time. Further, the time required to perform in-situ visualization of 22 cut-planes and 5 iso-surfaces totaling 10.7GB was 60.2 seconds. Data was extracted every 30 time steps with the flow solver taking approximately 40 seconds per time step, thus giving an overhead of approximately 5% for the in-situ extraction.

Table 2. Two wind turbine simulation data reductions obtained by in-situ workflow.

Grid Points	Grid File & Volume File Size (GB)	Iso-Surfaces & Cut-Planes (GB)	Data Reduction
508,077,274	40.5	3.5	11.6x

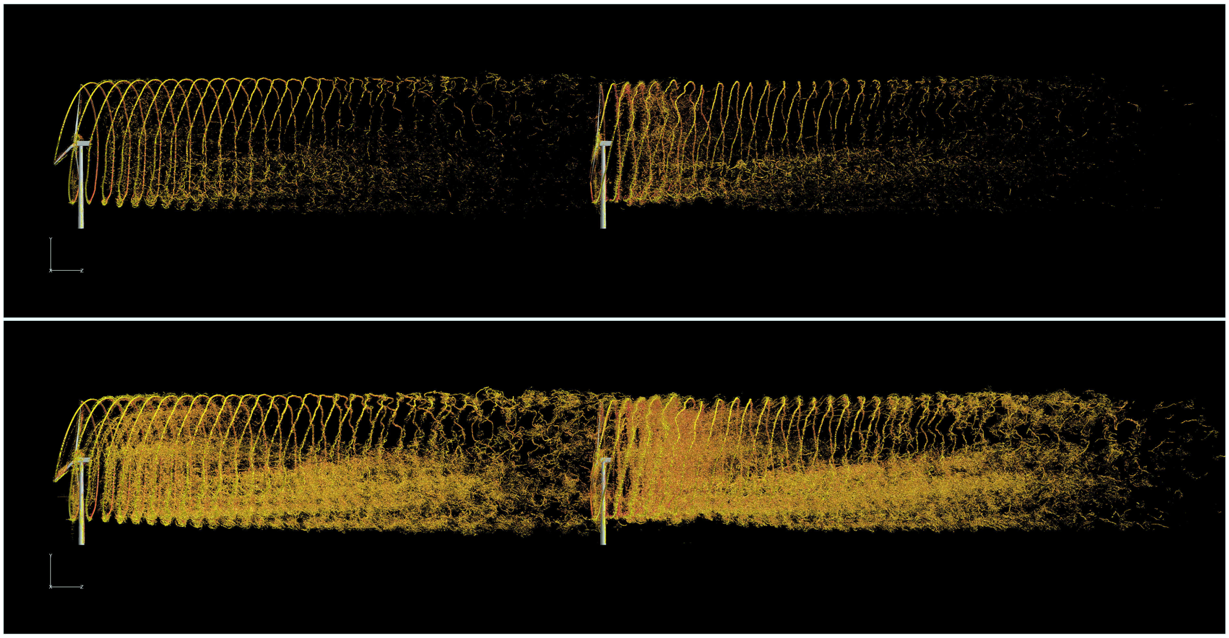
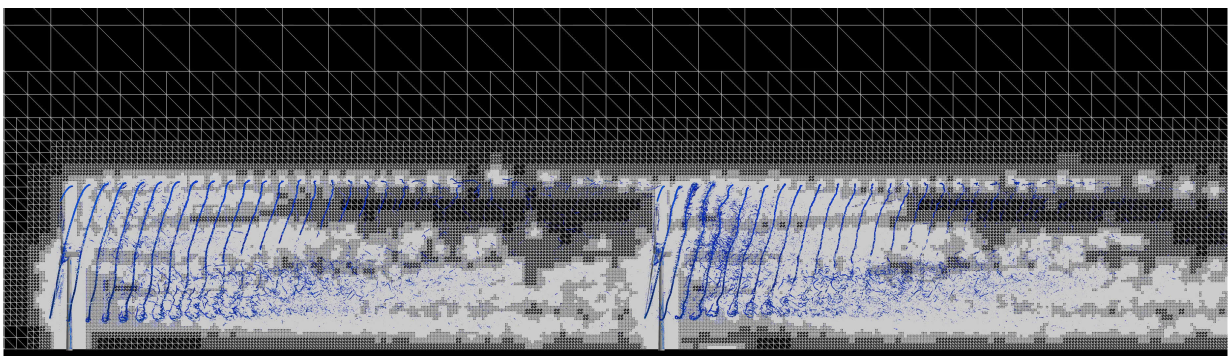
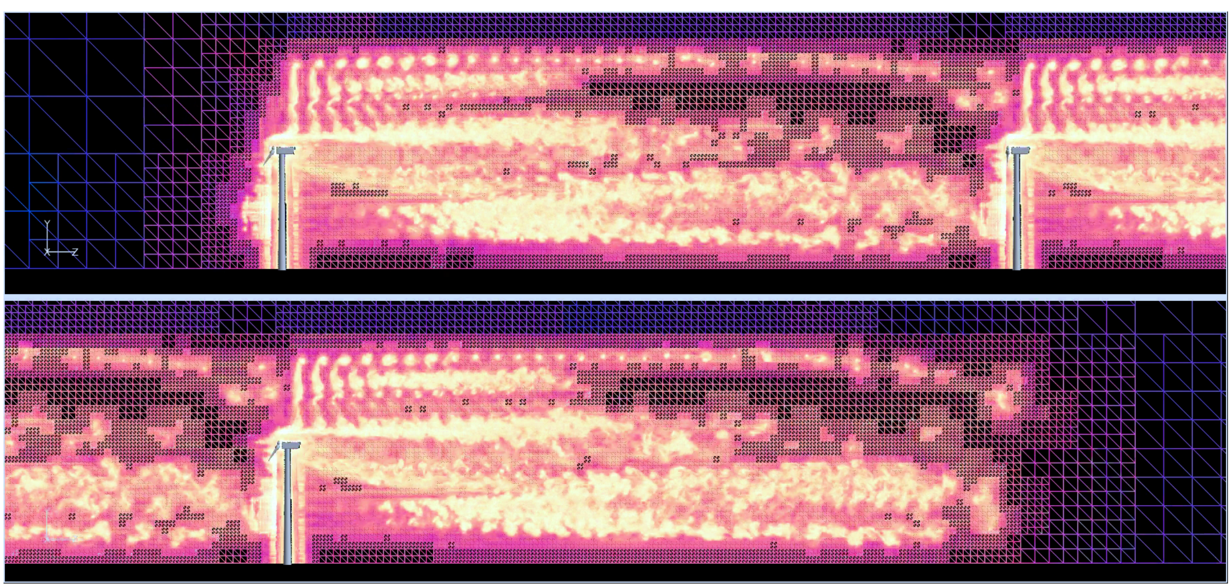


Figure 10. Two iso-contours of Q-criterion of the wake formed from two generic Siemens SWT-2.3-93 wind turbines.



(a) Adaptive mesh visualized with Q-criterion.



(b) Zoomed view of each wind turbine wake.

Figure 11. Higher-order element subdivision for higher-order plotting of two wind turbine wakes.

V.C. Lillgrund Wind Farm

A simulation using the Lillgrund wind farm which uses the Siemens SWT-2.3-93 wind turbine was performed. The Lillgrund wind farm contains 48 wind turbines in an arrangement with downstream spacing of 4.3 rotor diameters and side spacing of 3.3 diameters. The simulation used over 22,000 cores and off-body mesh grew to more than 7.5 billion variable unknowns.

Figure 12 shows the wind plant configuration with iso-contours of velocity magnitude. The visualization was rendered from volume data using remote-host rendering on 360 cores. The volume data generated for the visualization did not use the higher-order element subdivision but rather only plotted the eight corner points of a cell. Thus the high-order solution representation was degraded but still aggregated 23 GB of data per frame. Further, a hybrid remote-hosting workflow was adopted; the original data, in VTK format, was converted to FieldView XDB format through VisIt's export database capability. Figure 13 demonstrates iso-contours visualized in FieldView using XDB files generated through VisIt's export database.

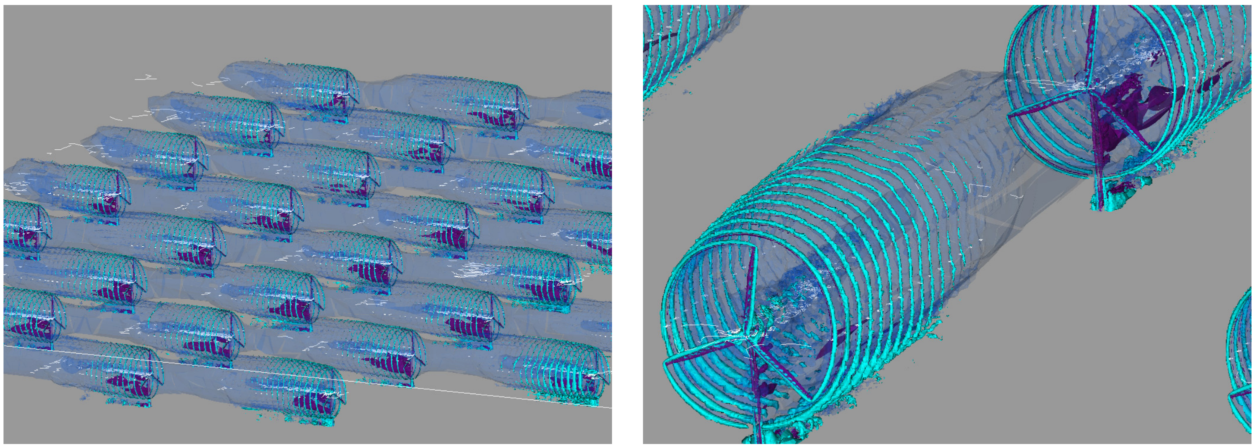
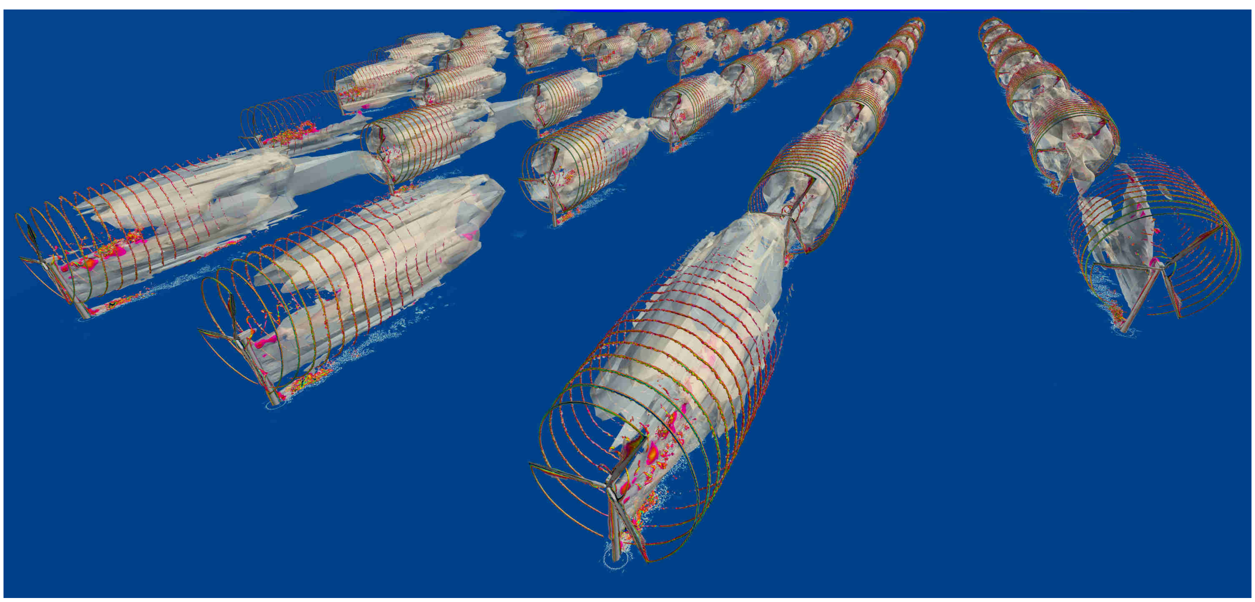


Figure 12. Lillgrund wind farm visualization.



(a) Velocity magnitude and q-criterion iso-surfaces.

Figure 13. Lillgrund wind farm wake structures composed of 48 Siemens SWT-2.3-93 wind turbines visualized in FieldView with XDB workflow.

VI. Conclusions and Future Work

In-situ processing is paramount as computing capabilities increase in power and computing capacity. In-situ workflows further provide opportunities for increased productivity as the data extraction and visualization is co-produced with simulation results at run-time. Libsim was instrumented into the W^2A^2KE3D framework enhancing visualization capabilities by utilizing compute resources in parallel to perform data extraction, remote-host visualization, and parallel rendering abilities. Workflow productivity increased as data size was reduced by an order of magnitude and no movement of data was required. For statistical analysis of wind turbine physics, high temporal frequency data is required which can lead to significant amounts of data storage using volume data. In-situ data techniques lead to 250x reductions in data size, enabling high-resolution temporal data to be generated and analyzed easily.

High-fidelity simulation of full wind farms containing 100 wind turbines will require approximately 100 billion degrees of freedom. This equates to approximately 500 GB for mesh and solution storage of a single time instance; 2,000 time instances of volume data result in one petabyte of data. Visualization of full wind farms using volume data with locally lossless element subdivision is intractable. Data reductions via in-situ techniques will enable visualization and analysis feasibility.

Future work will involve instrumenting the SENSEI API into W^2A^2KE3D for in-situ abstraction. This will enable end-users the ability to choose between Paraview Catalyst or VisIt Libsim for in-situ visualization, and have the ability to perform analysis *in-transit* using ADIOS, allowing visualization and analysis to be performed on separate compute resources during run-time allowing the simulation to continue uninterrupted.

VII. Acknowledgements

Computer time was provided by the NCAR-Wyoming Supercomputer Center (NWSC), the University of Wyoming Advanced Research Computing Center (ARCC), and the NSF Blue Waters sustained-petascale computing project. This research was conducted with an Accelerated Science Discovery (ASD) project providing early access to the NWSC-2 Cheyenne supercomputer.

This work is supported in part by ONR Grants N00014-14-1-0045 and N00014-16-1-2737 and by the U.S. Department of Energy, Office of Science, Basic Energy Sciences, under Award DE-SC0012671. The first author was supported in part by the NSF Blue Waters Graduate Fellowship as part of the Blue Waters sustained-petascale computing project, which is supported by the National Science Foundation (awards OCI-0725070 and ACI-1238993). The last two authors were supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research under Award Number DE-SC0012449.

Images were created using FieldView as provided by Intelligent Light through its University Partners Program.

VIII. Disclaimer

References

[1]Kirby, A. C., Brazell, M. J., Yang, Z., Roy, R., Ahrabi, B. R., Mavriplis, D. J., Sitaraman, J., and Stoellinger, M. K., "Wind Farm Simulations Using an Overset *hp*-Adaptive Approach with Blade-Resolved Turbine Models," AIAA Paper 2017-3958, 23rd AIAA Computational Fluid Dynamics Conference, Denver, CO., June 2017.

[2]Haimes, R., "pV3: A Distributed System for Large-Scale Unsteady CFD Visualization," AIAA Paper 1994-0321, 32nd Aerospace Sciences Meeting, Reno, NV., January 1994.

[3]Fabian, N., Moreland, K., Thompson, D., Bauer, A. C., Marion, P., Gevecik, B., Rasquin, M., and Jansen, K. E., "The paraview coprocessing library: A scalable, general purpose in situ visualization library," *Large Data Analysis and Visualization (LDAV), 2011 IEEE Symposium on*, IEEE, 2011, pp. 89–96.

[4]Whitlock, B., Favre, J., and Meredith, J., "Parallel In Situ Coupling of Simulation with a Fully Featured Visualization System," Eurographics Symposium on Parallel Graphics and Visualization 2011.

[5]Lofstead, J. F., Klasky, S., Schwan, K., Podhorszki, N., and Jin, C., "Flexible io and integration for scientific codes through the adaptable io system (adios)," *Proceedings of the 6th international workshop on Challenges of large applications in distributed environments*, ACM, 2008, pp. 15–24.

[6]Vishwanath, V., Hereld, M., and Papka, M. E., "Toward simulation-time data analysis and i/o acceleration on leadership-class systems," *Large Data Analysis and Visualization (LDAV), 2011 IEEE Symposium on*, IEEE, 2011, pp. 9–14.

[7]Ayachit, U., Bauer, A., Duque, E. P., Whitlock, B. J., Eisenhauer, G., Ferrier, N., Gu, J., Jansen, K. E., Loring, B., Lukic, Z., Menon, S., et al., "Performance analysis, design considerations, and applications of extreme-scale in situ infrastructures," *High Performance Computing, Networking, Storage and Analysis, SC16: International Conference for*, IEEE, 2016, pp. 921–932.

[8]"Wind Vision: A New Era for Wind Power in the United States," Tech. rep., Technical report, US Department of Energy, Washington, DC, 2015.

[9]Mavriplis, D. J., "Grid resolution study of a drag prediction workshop configuration using the NSU3D unstructured mesh solver," AIAA Paper 2005-729, 23rd AIAA Applied Aerodynamics Conference, Toronto, Ontario Canada, June 2005.

[10]Mavriplis, D. and Long, M., "NSU3D Results for the Fourth AIAA Drag Prediction Workshop," *Journal of Aircraft*, Vol. 51, No. 4, 2014, pp. 1161–1171.

[11]Brazell, M. J., Kirby, A. C., and Mavriplis, D. J., "A high-order discontinuous-Galerkin octree-based AMR solver for overset simulations," AIAA Paper 2017-3944, 23rd AIAA Computational Fluid Dynamics Conference, Denver, CO., June 2017.

[12]Burstedde, C., Ghattas, O., Gurnis, M., Isaac, T., Stadler, G., Warburton, T., and Wilcox, L., "Extreme-scale AMR," *Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*, IEEE Computer Society, 2010, pp. 1–12.

[13]Burstedde, C., Wilcox, L. C., and Ghattas, O., "p4est: Scalable algorithms for parallel adaptive mesh refinement on forests of octrees," *SIAM Journal on Scientific Computing*, Vol. 33, No. 3, 2011, pp. 1103–1133.

[14]Roget, B. and Sitaraman, J., "Robust and efficient overset grid assembly for partitioned unstructured meshes," *Journal of Computational Physics*, Vol. 260, 2014, pp. 1–24.

[15]Brazell, M. J., Sitaraman, J., and Mavriplis, D. J., "An overset mesh approach for 3D mixed element high-order discretizations," *Journal of Computational Physics*, Vol. 322, 2016, pp. 33–51.

[16]Crabill, J. A., Sitaraman, J., and Jameson, A., "A High-Order Overset Method on Moving and Deforming Grids," *AIAA Modeling and Simulation Technologies Conference*, 2016, p. 3225.

[17]Kirby, A. C., Brazell, M. J., Mavriplis, D. J., and Sitaraman, J., "An Overset Adaptive High-Order Approach for Blade-Resolved Wind Energy Applications," AHS Forum 72, West Palm Beach, FL., May 2016.

[18]Brazell, M. J., Kirby, A. C., Sitaraman, J., and Mavriplis, D. J., "A multi-solver overset mesh Approach for 3D mixed element variable order discretizations," AIAA Paper 2016-2053, 54th AIAA Aerospace Sciences Meeting, San Diego, CA., June 2016.

[19]Tech. rep., Silo Documentation Version 4.0, Lawrence Livermore National Laboratory, Livermore, CA.

[20]Moriarty, P., Rodrigo, J. S., Gancarski, P., Chuchfield, M., Naughton, J. W., Hansen, K. S., Machefaux, E., Maguire, E., Castellani, F., Terzi, L., et al., "IEA-Task 31 WAKEBENCH: Towards a protocol for wind farm flow model evaluation. Part 2: Wind farm wake models," *Journal of Physics: Conference Series*, Vol. 524, IOP Publishing, 2014, p. 012185.

[21]"NSF NWSC-2 Cheyenne," 2016, Computational and Information Systems Laboratory. 2017. Cheyenne: SGI ICE XA System (Climate Simulation Laboratory). Boulder, CO: National Center for Atmospheric Research. doi:10.5065/D6RX99HX.