Wind Farm Simulations Using a Full Rotor Model for Wind Turbines

Jay Sitaraman [*] Dimitri Mavriplis [†] Earl P Duque [‡]

Abstract

Flow-field simulation of an entire wind farm is performed using a full rotor model for the wind turbines. The analysis scheme combines several technologies, viz. unstructured mesh solvers, dynamic overset gridding, high-order Cartesian discretizations, adaptive mesh refinement, meso-scale/micro-scale coupling and efficient flow visualization to enable high fidelity simulation of an entire wind plant. The *Helios* software framework, which is the product of the DoD CREATE A/V program is utilized to perform the simulations. Results are shown for a wind plant modeled with the same lay out as the Lillgrund wind farm. This manuscript focuses on demonstration of technology and verification of expected trends in the flow field. Comprehensive validation with measurements will be the focus of the future work.

I. Introduction and Objective

Wind turbine aeromechanics represents a complex problem that spans several disparate spatial and temporal scales. The flow field itself includes the large length scale of the atmospheric boundary layer (order of a kilometer) as well the small length scale of wind turbine blade boundary layer (order of a micro-meter). Improved prediction of wind turbine aeromechanics requires reliable modeling of turbulent eddies in this entire length-scale range. The temporal scales are also quite disparate: wind turbine rotation periods are of the order of seconds, while atmospheric inflow modes have time periods ranging a from few minutes to hours. Structural mechanics further compounds this problem by introducing yet another set of temporal scales. Furthermore, the time periods of significant elastic vibrations of the rotor blade and masts are often one to two orders smaller than the wind turbine rotor period. In this context, a physics based, efficient, massively parallel simulation method would provide the predictive scheme that can analyze the aeromechanics of wind turbine systems efficiently and accurately. The broad objective set by Department of Energy (DOE) for extracting 20 percent of the total national electric power generation energy needs from wind resources[1] requires revolutionary improvement in the efficiency and reliability of wind turbines. The state-of-the-art of modeling wind farm aerodynamics relies on parameterization of the wind turbines into actuator lines[2] or actuator disks.[3] Several contemporary research works [4–6] utilize one of these parameterizations for wind turbines in conjunction with a flow solver capable of performing Large Eddy Simulations (LES). Full rotor models, i.e where the actual geometry of the wind turbine blades and towers are modeled exactly are less common and are in most cases restricted to the simulation of isolated wind turbines.[4, 7–11] Turbine vortex wakes and atmospheric boundary layer flow features require adaptive mesh refinement to provide sufficient resolution for preserving their transport and dynamics. Demonstration of AMR is also currently limited to only isolated rotor scenarios.[12, 13] Modeling entire wind farms using full rotor models was considered intractable in the past because of the sheer size of the problem and resolution requirements. However, the advent of large-scale high performance computing facilities with an eventual path towards truly exascale architectures shows promise of making such computations increasingly feasible. In this context, the objective of this paper is to demonstrate one of the first full rotor simulations that has been performed on a full wind plant level. This work is expected to become a pathfinder for future high-fidelity wind farm simulations as well as large-scale computing. Furthermore, data generated by such higher fidelity analysis can be mined to

[*] Assistant Professor, Dept of Mechanical Engineering, University of Wyoming

[†] Professor, Department of Mechanical Engineering, University of Wyoming

[‡] Director of Applied Research, Intelligent Light Inc

significantly improve the prediction capabilities of lower-fidelity, fast analysis schemes for design needs.

The simulations demonstrated in this paper are performed using the *Helios*[14] software framework, which is a product of the DoD CREATE A/V program. Helios is jointly developed by the U.S Army and the University of Wyoming. *Helios* utilizes a multi-mesh framework, with a dynamic overset grid architecture to provide accurate and efficient prediction capabilities for vortex dominated flows that characterize helicopter and wind turbines. Body-conforming grids are utilized near solid walls, while the solution away from the walls is resolved using block-structured Cartesian grids capable of dynamic mesh adaptation. A Dynamic overset grid assembly technique[15] manages the data transfer between the multiple mesh systems. In general, the Reynolds Averaged form of the Navier Stokes equations (RANS) is utilized for the prediction of flow fields. Detached Eddy Simulation (DES) techniques, which smoothly transitions the turbulence closure equations from RANS to LES away from the solid walls is also implemented and is utilized in the present work. *Helios* has been used in the past by the authors to study single turbine aerodynamics [16] and two turbine interactions.[11] Furthermore, in a previous effort,[4] the software framework was extended to incorporate precursor turbulent inflow. Turbulent inflow can be generated using numerical weather prediction codes such as WRF[27] (Weather Research and Forecasting) or from synthetic turbulence generation techniques.[28] The present work forms an important milestone in this context, where flow simulation over an entire wind farm is demonstrated.

II. Methodology

The fluid dynamic model in *Helios* differs from traditional monolithic CFD code architecture owing to its utilization of a heterogeneous meshing paradigm. The meshing paradigm in *Helios* combines the advantages of unstructured grids (their ability to model complex geometry and resolving boundary layers using anisotropic meshes) and Cartesian adaptive grids (efficient high order algorithms, efficient storage, ease of solution based adaptation). The use of multiple mesh paradigms provides the potential for optimizing the gridding strategy on a local basis. For the wind turbine problems in particular, it is apparent that one should employ body fitted unstructured grids around the solid boundaries and then transition the gridding paradigm to solution adaptive Cartesian grids for wake capturing. The use of a heterogeneous meshing paradigm does involve several implementation challenges. For example, integrating different meshing paradigms into a single large monolithic code is complex and has the disadvantage of forcing at least one of the gridding paradigms to be less optimized and less flexible than the original stand-alone solver. In *Helios*, we overcome this challenge by utilizing a multiple-mesh strategy that is implemented through the use of *multiple* CFD codes, each optimized for a particular mesh type. In addition to supporting heterogeneous grids and solvers, the software was also developed in a manner to minimize the analysis burden on the user. While an interface procedure is required for each specific solver, this interface must only be written once. Once the interface exists, the domain connectivity software retrieves the information it needs directly from the flow solvers and the grids themselves, so the human analyst does not need to provide input specific to each problem. The design paradigm used in *Helios* automates most analysis procedures, thus considerably simplifying its routine usage for production needs. For example, for the entire wind farm, the only grid generation required is for a single blade and a tower (i.e. the user needs to create an unstructured grid that extends about one chord from the blade). The creation, replication and partitioning of the entire wind plant, the generation/adaptation of the off-body Cartesian grid system, and the execution of the overset grid assembly and associated data exchange are completely automated from then on and can proceed seamlessly without any user intervention.

III. Technical Approach

The analysis performed here requires the invocation of most of the key participant modules in *Helios*. These are (a) the parallel NSU3D code[29,30] as the unstructured near-body solver (b) the SAMARC code which combines the SAMRAI framework[17] and the high-order ARC3DC code as the off-body solver (c) the parallel domain connectivity module PUNDIT[20] (d) the flight dynamics and mesh motion interface and (e) the meso-scale/micro-scale interface (MMCI[4]). The coupling of all these components is accomplished through a Python-based infrastructure with emphasis on preserving the modularity of the participating solvers. In addition to the advantages in efficiency and ease in code development, coupling existing mature simulation codes through a common high-level infrastructure provides a natural way to reduce the complexity of the coupling task and to leverage the large amount of verification, validation, and user experience that typically go into the development of each separate model. Brief descriptions of the implementations of all the component modules and the python infrastructure are outlined below. More detailed descriptions and validations of the python-based infrastructure can be found in Refs.[18,19]

III.A. Python Infrastructure

Python-based computational frameworks have been developed previously by several researchers[21] as a means of coupling together existing legacy codes or modules. Such a framework has a number of advantages over a traditional monolithic code structure: (1) it is easier to incorporate well-tested and validated legacy codes rather than to build the capabilities into an entirely new code, (2) there is less code complexity in the infrastructure itself, so maintenance and modification costs are less, and (3) it is easier to test and optimize the performance of each module separately, often yielding better performance for the code as a whole. Essentially, Python enables the legacy solvers to execute independently of one another and reference each other's data without memory copies or file I/O. Further, the Python-wrapped code may be run in parallel by utilizing python modules that wrap the standard message passing interface (MPI) implementation. One such implementation that is utilized widely is *mpi4py*.[31]

III.B. Flow Solver

The flow solution methodology in *Helios* utilizes a heterogeneous meshing paradigm consisting of unstructured near-body grids and Cartesian off-body grids. Three distinct modules operate synchronously to facilitate the flow solution procedure. The modules are termed as near-body-solver, off-body solver and domain connectivity module, respectively. In the following sections, we provide brief descriptions of the meshing paradigm and each participating solver.

III.B.1. Meshing Paradigm

The meshing paradigm consists of separate near-body and off-body grid systems. The near-body grid typically extends a short distance from the body, sufficient to contain the boundary layer. This grid can be an unstructured tetrahedral and/or prismatic grid that has been extracted from a standard unstructured volume grid or generated directly from a surface triangulation using hyperbolic marching. The reason for using unstructured grids in the near-body region is to properly capture the geometry and viscous boundary layer effects, which are difficult or impossible to capture with Cartesian grids alone at the high Reynolds numbers of interest for wind turbine aerodynamics. We further note that either structured or unstructured grids will work equally well in the near-body region from the point of view of our infrastructure. Away from the body, the near-body grid solution is interpolated onto a Cartesian background mesh with the aid of the domain connectivity algorithm (see section III.B.3). This transition normally occurs at a distance from the wall wherein the sizing of the near-body grid cells is approximately commensurate with the sizing of the Cartesian mesh in the off-body region.

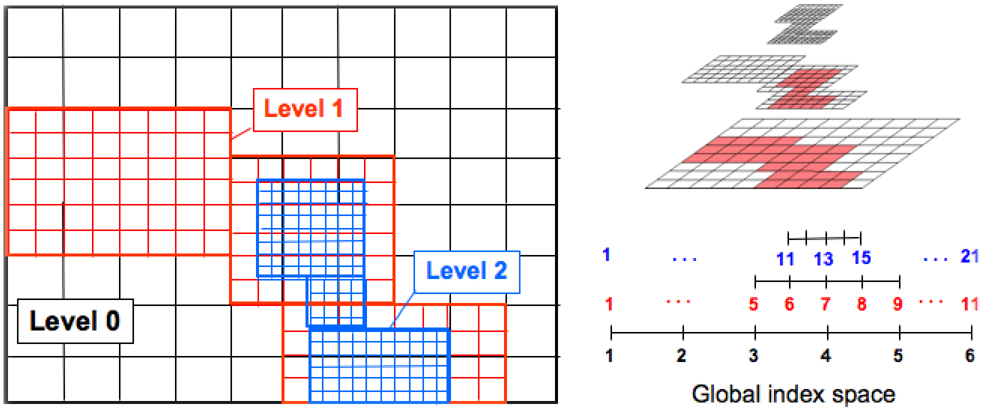


Figure 1. Block structured AMR grid composed of a hierarchy of nested levels of refinement. Each level contains uniformly-spaced logically-rectangular regions, defined over a global index space.

The off-body grid system consists of a hierarchy of nested refinement levels, generated from coarsest to finest. In the Structured Adaptive Mesh Refinement (SAMR) paradigm,[22] the coarsest level defines the physical extents of the computational domain. Each finer level is formed by selecting cells on the coarser level and then clustering the marked cells together to form the regions that will constitute the new finer level. Solution-based refinement progresses as follows: physical quantities and gradients are computed at each Cartesian cell using the latest available solution and those cells that hold values deemed to require refinement are marked. The marked cells are then clustered to form the new set of patches that constitute the next finer level. The process is repeated at each new grid level until the geometry and solution features are adequately resolved. We note that this entire procedure is automated within the software and no user intervention is required. The procedure of adaptive mesh refinement is graphically illustrated in Figure 1.

An example of the overset near-body/off-body meshing strategy is given in Figure 2, which shows the meshes for flow computations over an isolated wind turbine. Here, the mixed-element unstructured near-body grid envelops the blades and tower, while a multi-level Cartesian off-body grid extends from the outer boundary of the near-body grid to the far-field boundary. The two sets of meshes overlap in the so-called fringe region, where data are exchanged between the grids.

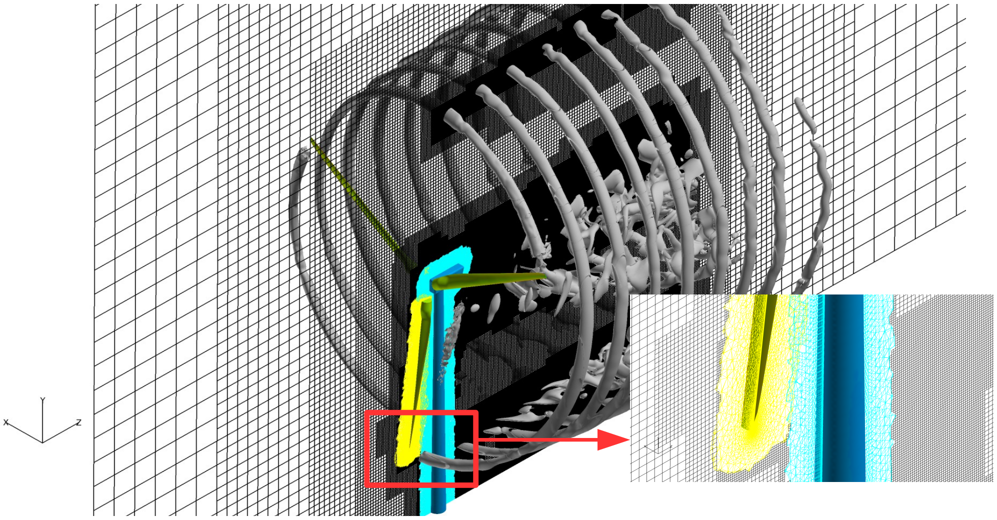


Figure 2. Overset near/off-body gridding. Unstructured grids to capture geometric features and blade/tower boundary layer gradients, block-structured Cartesian grids with Adaptive Mesh Refinement (AMR) to capture far-field flow features.

III.B.2. Near-body solver

For the unstructured mesh solver in the near-body region, we employ the NSU3D[29,30] code, which is an implicit Reynolds-averaged Navier-Stokes (RANS) solver designed for high Reynolds number aerodynamic applications.

The NSU3D discretization employs a vertex-based approach, where the unknown fluid and turbulence variables are stored at the vertices of the mesh, and fluxes are computed on faces delimiting dual control volumes, with each dual face being associated with a mesh edge. This discretization operates on hybrid mixed-element meshes, generally employing prismatic elements in highly stretched boundary layer regions, and tetrahedral elements in isotropic regions of the mesh away from the blade and turbine surfaces. A single edge-based data-structure is used to compute flux balances across all types of elements. The convective terms are discretized as central differences with added matrix dissipation.

The basic solver in NSU3D consists of a local block-Jacobi implicit approach in regions of isotropic grid cells. In boundary layer regions, where the grid is highly stretched, a block line solver is employed to relieve the stiffness associated with high mesh anisotropy.[32] An agglomeration multigrid algorithm is used to further enhance convergence for time-implicit problems.[29] The Jacobi and line preconditioners are used to drive the various levels of the multigrid sequence, resulting in a rapidly converging solution technique. NSU3D incorporates various turbulence models, although in the current work the Spalart-Allmaras single equation model[33] is used exclusively. NSU3D has been benchmarked extensively through regular participation in numerous computational aerodynamic workshops over the last decade,[34–37] and has recently demonstrated good scaling using up to 32,768 cores on the NCAR-Wyoming Supercomputer (NWSC).[38]

In the baseline approach used in Helios, all near-body unstructured component grids are first merged into a single unstructured grid data set, which is then partitioned using a parallel weighted graph algorithm.[39, 40] The partitioning process may lead to potential bottlenecks when scaling to much higher levels of parallelism as anticipated for the purposes of high-resolution large wind farm simulations. Therefore, a component mesh replication strategy has been implemented for large wind farm simulations. In this approach, a single mesh for each unique component is first generated and partitioned individually. Thus, for an entire wind farm of identical turbines, a single blade component mesh and a single tower mesh are generated and partitioned. These component meshes are then replicated internally at run time, for example generating three blade meshes and one tower mesh for each turbine in the wind farm, and each component mesh is assigned to its individual set of processors or cores. Under this scenario, all MPI communication in the NSU3D code occurs only within individual component meshes, with no inter-component communication. Thus, the near-body portion of the wind-farm simulation can be scaled up indefinitely and should achieve near perfect (weak) scaling. This approach places the entire burden of inter-component communication required to enable a viable wind farm simulation on the off-body Cartesian mesh solver, which can be partitioned, load balanced and managed more easily at large scale.

III.B.3. Overset Methodology and domain connectivity module

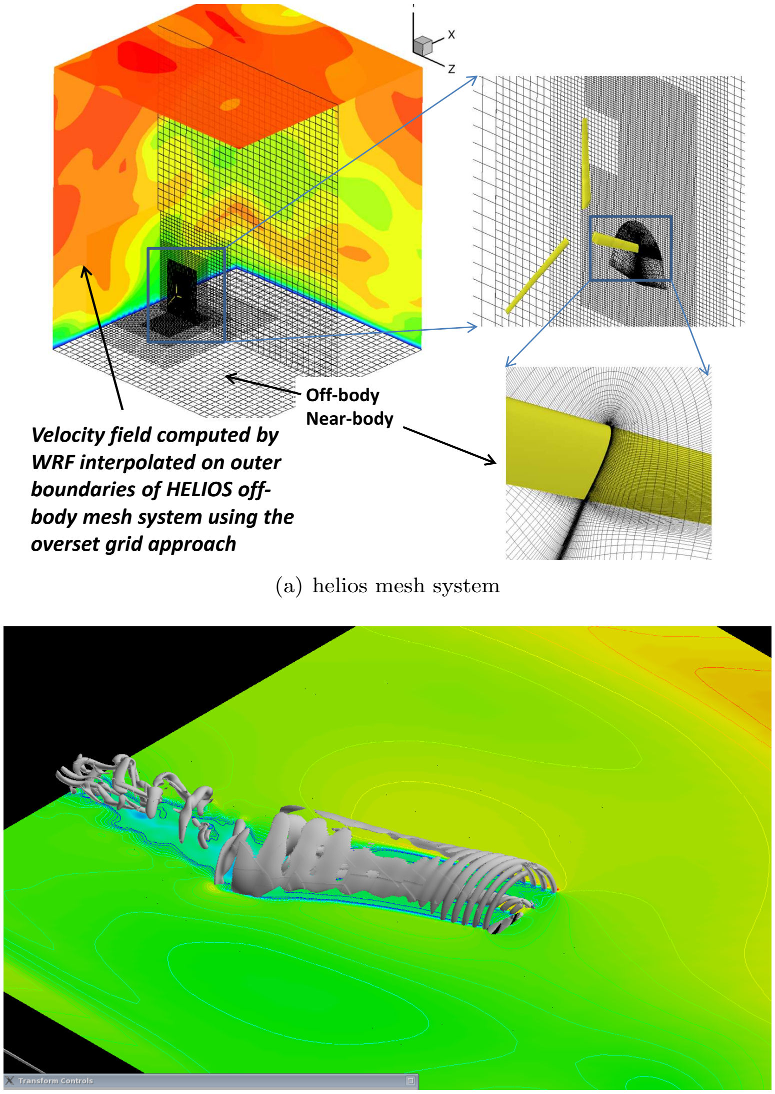
The overall solution procedure for the overset meshes is as follows. At each iteration step, the solutions of the fluid equations in each mesh system are obtained independently with the solution in each fringe region being specified by interpolation from the overlapping “donor” mesh as a Dirichlet boundary condition. At the end of the iteration, the fringe data are exchanged between the solvers so that the evolution of the global solution is faithfully represented in the overset methodology. Further, if one or more of the meshes is moving or changing, the fringe regions and the interpolation weights are recalculated at the beginning of the time step. This procedure is repeated for each iteration step until solution convergence is attained in both the near- and off-body meshes. Note that, since the off-body solver is inviscid in the current implementation, only the mean-flow variables are exchanged between the near and off-body solvers. Non-reflective boundary conditions are utilized for the turbulence closure equations in the near-body solver to mitigate this inconsistency.

The domain connectivity module (PUNDIT[15, 41]) is responsible for performing all the overset grid assembly operations. These include determination of donor-recipient relationships and appropriate interpolation schedules between multiple overlapping meshes. For static meshes, these operations are done once, at the beginning of the computation, while, for the more general case of moving or adapting meshes, the determinations of donors and weights has to be repeated within the time-stepping or iteration loop.

All the overset grid assembly operations are fully automated and do not require any geometry input from the user. In addition, PUNDIT performs all the domain connectivity operations on the partitioned grid system in a fully parallel environment. PUNDIT adopts the implicit hole cutting procedure followed by NAVAIR-IHC.[25] The core idea of this approach is to retain the grids with the finest resolution at any location in space as part of the computational domain and interpolate data at all coarser grids in this region from the solution obtained on the fine grid. This results in the automatic generation of optimal holes without any user specification as in the case of explicit hole-cutting. Moreover, the implicit hole-cutting procedure produces an arbitrary number of fringe points based on mesh density compared to traditional methods which use a fixed fringe width (usually single or double). The critical parameter that quantifies the quality of a grid cell or node is termed as “resolution capacity”. PUNDIT nominally uses the cell volume as the resolution capacity for a grid cell and the average of cell volumes of all associated grid cells as resolution capacity for a grid node. The near-body to near-body and near-body to off-body domain connectivity procedures are separated in PUNDIT to facilitate automatic off-body grid generation and to improve efficiency (Meakin et al.[26]).

III.B.4. Mesoscale Microscale Coupling Interface (MMCI)

This module handles the interpolation of data between the mesoscale and microscale codes. MMCI currently can only handle one-way coupling between precursor turbulence generators and micro-scale wind turbine aerodynamic prediction. MMCI is general and has been used with a variety of CFD codes including *Helios* and actuator line LES based CFD simulations.[13] In this coupling approach, the mesoscale simulations


x
z
Off-body
Near-body
Velocity field computed by WRF interpolated on outer boundaries of HELIOS off-body mesh system using the overset grid approach
Transform Controls

(a) helios mesh system

(b) Wake response to turbulent inflow

Figure 3. (a) *Helios* mesh system consisting of near- and off-body meshes. Near-body meshes are body-conforming and provide and effective method for resolving the viscous sub-layer near the moving wind turbine walls (wall spacing = $1\mu m$). Off-body meshes provide nesting and adaptation capabilities and effective scaling of the resolution to match grid resolution of WRF data (outer grid spacing = $10m$). (b) Iso-vorticity contours overlaid with velocity field after 50000 steps of unsteady simulation: velocity fields on plane parallel to the ground and passing through the turbine hub are shown.

are performed first. The data from these precursor simulations are given to MMCI, which post processes and generates the boundary flow field information, and initial condition required for the CFD codes. The entire framework as setup uses a high level object oriented python based computational platform which is compatible with the architecture of *Helios*.

One of the important requirements in the performance of the coupled LES solver is the interpolation of the SGS quantities between the different codes. This is necessary since the SGS quantities have much larger values on the coarse atmospheric boundary layer resolving grids when compared with the fine microscale grids. This makes it necessary to introduce a compatibility condition between the mesoscale and microscale codes for the efficient interpolation of the SGS variables. As *Helios* is an Eulerian CFD code, the compatibility condition required for both these codes is the same.

In the current study, the approach described in reference [42] is used to interpolate the viscosity and turbulent kinetic energy at the overset boundaries. This approach is based on a heuristic balance of the budget of turbulent kinetic energy. The TKE budget can be symbolically expressed as:

$$\text{Unsteady} + \text{Convection}=\text{Production}+\text{Dissipation}+\text{Diffusion} \tag{1}$$

In the above system, the convection, unsteady and diffusion terms can be neglected at the overset regions and a balance between turbulent kinetic energy (TKE) production and dissipation is considered. This results in

$$(2\mu_t S_{ij}^d - \frac{2}{3}\rho k \delta_{ij})\frac{\partial U_i}{\partial x_j} = C_E \rho \frac{k^3/2}{\Delta} \tag{2}$$

where $\mu_t = C_k \rho \Delta \sqrt{k}$ is the dynamic viscosity, C_k is a model constant, Δ is the filter width, ρ is the fluid density, k is the SGS TKE, $S_{ij}^d = S_{ij} - S_{nn}/3$ is the deviatoric part of the strain tensor $S_{ij} = 1/2(\partial U_i/\partial x_j + \partial U_j/\partial x_i)$, and C_E is a model constant. If we substitute the expression for the dynamic viscosity in Eq. (2) and neglect the second-term on the left-hand side (negligible for low-speed flows), we obtain

$$k = 2\frac{C_k}{C_E}\Delta^2 S_{ij}^d \frac{\partial U_i}{\partial x_j} \tag{3}$$

If the effect of buoyancy (for convective boundary layers) or a non-linear model for SGS stress tensor is considered, an explicit solution still exists for the TKE. Once the value of TKE is calculated at the overset boundaries from the interpolated velocities, the dynamic viscosity can be obtained from it. The above condition is implicitly satisfied when the Smagorinsky model is used while it has to be explicitly satisfied for the other models.

The interpolation of the mesoscale velocity and turbulence variables on to the outer boundary of the microscale wind-turbine off-body aerodynamic mesh (the largest, or lowest level, Cartesian mesh system in the case of Helios) is accomplished by invoking the same overset grid connectivity algorithm that is implemented in Helios. MMCI generates a mesh system that envelopes the outer boundaries of the Cartesian grid system where the turbulent inflow solution is injected. MMCI is treated in a similar fashion as a regular flow solver component by the *Helios* framework, such that a time varying solution is expected to evolve in the associated mesh system. The overset grid assembler is agnostic about the exact nature of the solution and performs its modular task of identifying data exchange patterns and interpolating the associated data. This approach makes the process of interpolation of turbulent inflow to outer boundaries very similar to that of interpolation on internal overset boundaries and hence maintains the same level of efficiency and parallelism.

Figure 3(a) shows an example of *Helios* simulation of an isolated wind turbine using the MMCI interfaces for precursor turbulence. The outer boundaries of *Helios* meshes are setup such that they are overlapped by the boundary representation created by MMCI. Consequently, weighted interpolation is performed through the overset grid assembler to project velocity field from precursor turbulence on to the *Helios* outer boundaries. Figure 3(b) shows flow visualization obtained for an unsteady simulations. The velocity field from meso-scale simulations can be observed to induce unsteady perturbations and meandering to the vortex wake emanating from the turbine.

III.C. Flow Visualization and post-processing

The wind farm solutions is inherantly unsteady resulting in terabytes worth of unsteady grid and solution files. Post-processing and managing this large set of data requires automation to quickly create images that compare the solutions to each other, perform further analysis like wake integrations and to compare against experimental data. However, the large size and remote location prevents moving the data off the large storage systems. Figure 4 illustrates a typical Volume Data file-based post-processing workflow. In this workflow, the solver has written files to disk of the complete grid and solution volume. The post-processor reads the volume data and then computes the various graphics based objects such a the geometric surface, coordinate cutting plans, iso-surfaces of arbitrary scalar functions, streamlines, etc.. After the Post-processing Objects are created, they are then further processed to render a graphical image, integrate scalar functions on the surface to yield integral functions like force and moments, and plot values on the surface such as Pressure Coefficient Distributions. Much of the computational costs and wall time is taken up by the the post-proessing tool reading in the volume dataset and creating the Post-Procssing Objects. In a typical workflow, the Post-Processing Objects are thrown away at the end of the session. As part of an XDB workflow (Figure 5), the Post-Processing Objects are saved as an XDB file for future and repeated use.

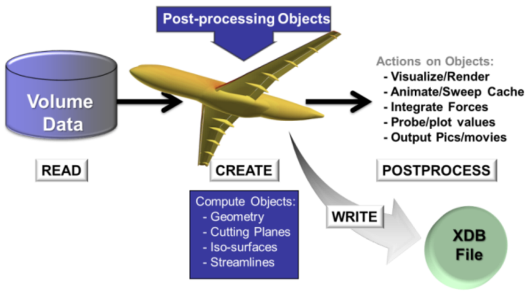


Figure 4. Traditional Volume Data post-processing

Furthermore, the XDB file can be created by executing FieldView in batch on a large HPC resource or they can be created interactively and saved. In either method, the XDB file can then be read into FieldView where all the other Post-Processing actions can be performed on the XDB extracts. Since the extracts are at the fidelity of the original data, there is no loss in accuracy yet several orders of magnitude of data redcution can be obtained. As mentioned, an XDB workflow utilizes small data files; XDBs are typically on the order of 10-100x smaller than the volume data. The data can be read back into FieldView very quickly and allows for a much more efficient workflow; particularly when working with data that has been created on remote servers and on relatively slow networks

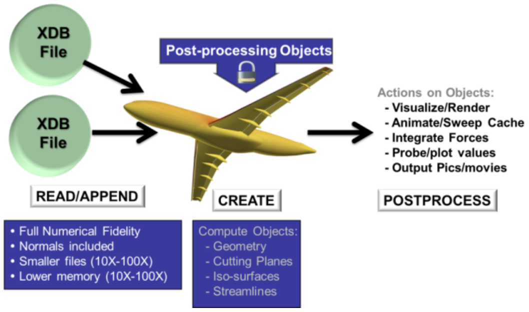


Figure 5. eXtract DataBase (XDB) Post-Processing

III.C.1. Using FieldView on NCAR Wyoming Supercomputing Center for batch and interactive post-processing

The FieldView XDB surfaces of the wind farm was created consisting of x-y-z Coordinate Cut Planes and iso-surfaces of vorticity using parallel processing with up to 16 shared memory processors directly on Caldera. FieldView was ran either in Client-Server mode for interactive sessions or in batch mode. The XDB files are then created using FieldView programing language (FVX) resulting in coordinate and iso-surfaces. The surfaces are then further post-processed using Interactive sessions to make images and movies.

III.C.2. XDB Data reduction

Table III.C.2 presents an example of the data reduction that was attained by using an XDB workflow. For the coordinate cut planes across the turbine rows (Z-direction), with 2 scalars stored on those planes we see 38 fold data reduction. For the Iso-Surfaces with a ground plane (Y-coordinate cut) and the Velocity Vector field, the data reuction is 9.7.

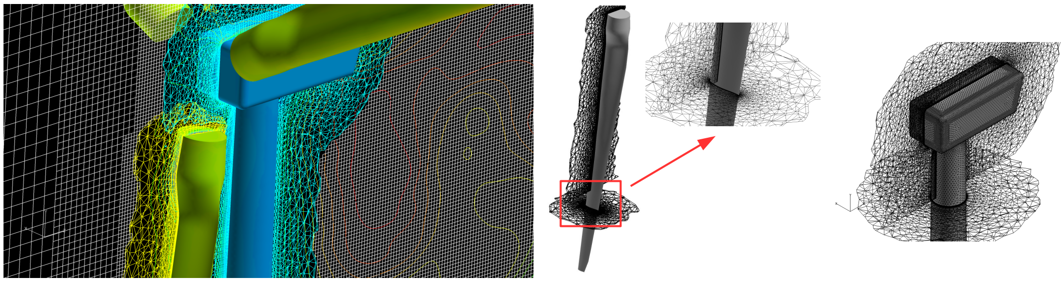
Grid Points	# Overset grids	Grid File + Solution File Size (GB)	Z-Coordinate Cut Planes with Vorticity and Velocity scalars (GB)	Iso Surface + Ground Plane with Velocity Vector (GB)
165,861,067	18271	5.49	0.144	0.566

IV. Results: Single turbine simulation

IV.A. Mesh system and boundary conditions

Figure 6 shows the mesh system utilized to model a single large off-shore type (NREL 5MW) wind turbine where multiple overlapping unstructured meshes are used for the blade and tower. The entire system was constructed using just a single blade mesh and tower mesh and assembling them at run-time. The blade mesh and tower mesh are shown separately in Figure 6(b). The blade surface was generated by a lofting technique, using the available sectional airfoil data,[43] twist and taper distributions. The geometry of the tower is notional and was developed from a generic CAD model. Volume mesh generation was performed using the NASA supported VGRID[44] software. Anisotropic grids composed of mixtures of prismatic layers and tetrahedral elements are used near the solid walls to resolve the blade and tower boundary layers. These grids have enough resolution to provide a $y+$ of 1 near the wall. A shaft tilt angle of 5^o and precone angle of -2.5^o was included while constructing the rotor system. The off-body mesh system is auto-generated and is set such that the resolution at the finest level is comparable to the resolution of the outer boundaries of the unstructured mesh system.

The axis system used is as follows: z is the wind direction, x is span wise and y is the up vector (along the tower). For uniform inflow conditions, the off-body mesh system has inflow and outflow boundaries specified at the front ($z = z_{min}$) and back ($z = z_{max}$) boundaries respectively. The span wise ($x = x_{min}$ and $x = x_{max}$)and top boundaries ($y = y_{max}$) use free stream boundary conditions, which are enforced appropriately using the nature of characteristic flow variables (Riemann invariants). The lower boundary uses ($y = y_{min}$) uses a slip wall.



(a) Mesh system zoomed near the hub

(b) Blade and Tower unstructured grids

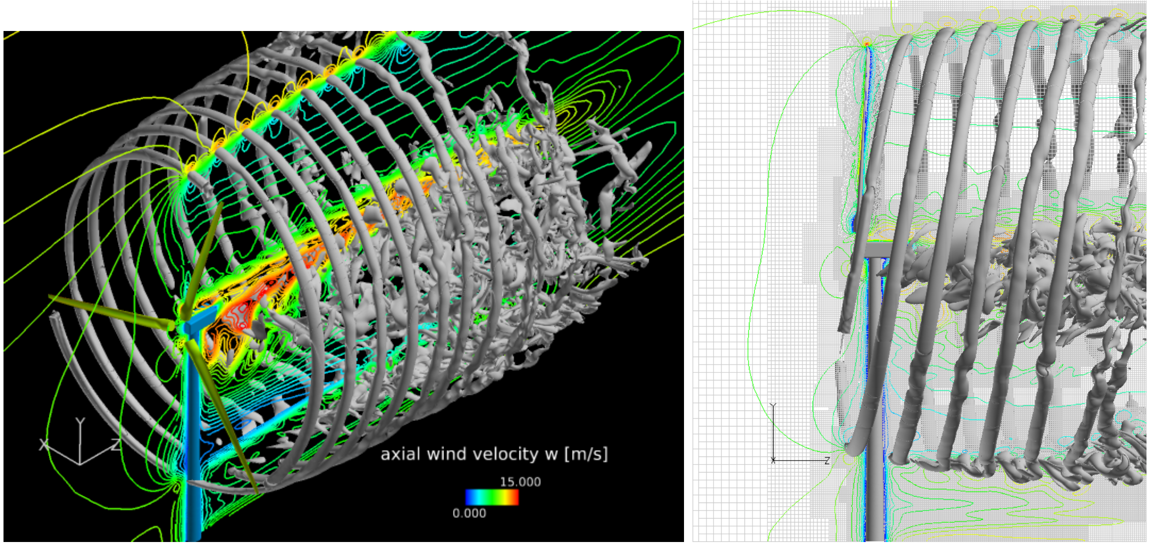
Figure 6. Mesh system used in Helios to model the NREL 5MW off-shore wind turbine

IV.B. Flow visualization in uniform inflow

Figure 7(a) shows the vorticity structures and velocity field behind an isolated turbine operating in steady inflow conditions. Adaptive mesh refinement illustrated in Figure 7(b) is critical for maintaining coherent vortex structures without undue numerical dissipation. Note that the vorticity structures show different patterns on the lower side of the stream tube where they experience interference from the tower. In addition, the large unsteadiness shed from the root section and nacelle can be observed. Further, it can be observed that the wake system shows instability and the beginning of the break down process after about 4 revolutions ($1.5D$) behind the turbine.

IV.C. Performance prediction

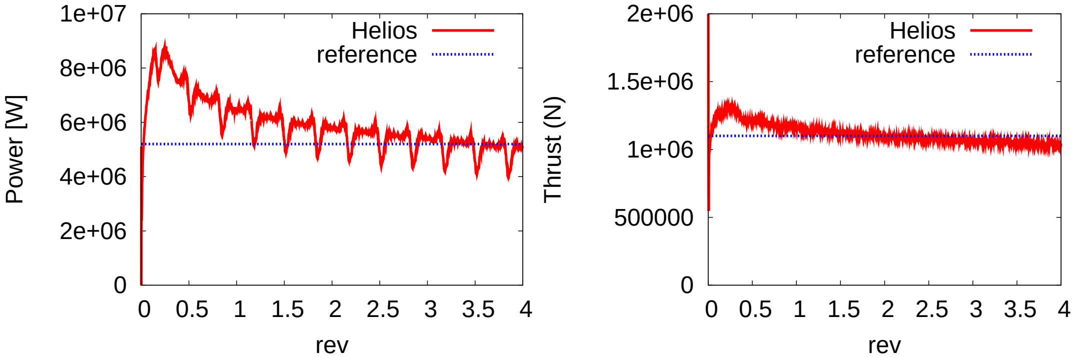
Figure 8 shows the power and thrust predicted for the NREL 5MW turbine by Helios. The reference values are those reported in Ref,[43] and are computed using a blade element theory based analysis.[45] The high frequency content in the power and thrust are attributed to aerodynamic interactions between the rotor system and the tower/nacelle. In addition, the shaft tilt angle induces a 3/revolution component owing to the in-plane component of the free stream velocity field.



(a) Vorticity contours

(b) Adaptive Mesh Refinement to track vortex structures

Figure 7. Vorticity and velocity field around a single turbine with adaptive mesh refinement that targets vorticity.



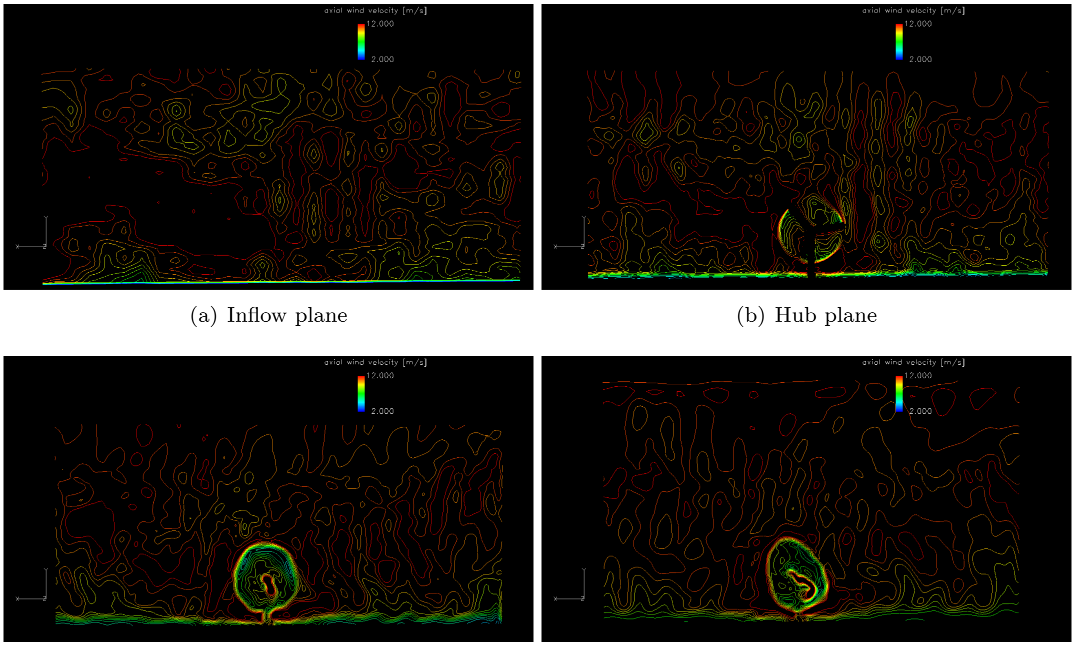
(a) Power vs rotor revolution

(b) Thrust vs rotor revolution

Figure 8. Power and thrust of the NREL 5MW turbine predicted by *Helios* at $V_{inf} = 11.4m/s$.

IV.D. Turbulent inflow and flow visualization

Figure 9 shows velocity field at axial slices located at the inflow plane, hub plane of the turbine, one radius behind the turbine and two radii behind the turbine respectively. It can be observed that the turbine wake is distorted from the circular shape by the inflow turbulence leading to meandering and faster break down.



(a) Inflow plane

(b) Hub plane

(c) z=1R plane

(d) z=1R plane

Figure 9. Velocity contours illustrating turbulent inflow and its influence on an isolated turbine

IV.E. Results: Wind farm simulations

The layout of the Lillgrund[46] off shore wind farm is chosen for this study. The turbines used in Lillgrund wind farm are 2.3 MW Siemens turbines. The exact geometry of these turbines is unavailable and hence does not permit generation of an accurate full rotor model. Therefore, the NREL 5MW turbine mesh system shown in the isolated rotor study is chosen as the rotor model in this case as well. Note that the primary purpose of this study is the demonstration of technology which mitigates the need for having the exact wind turbine geometry (which is often proprietary). The entire wind farm is created by replicating the mesh system several times. The Lillgrund wind farm has 48 turbines which have a lay out shown in Figure 10. The turbine mesh system used for the isolated turbine is replicated and re-positioned to form an entire wind farm. Note that the only meshes needed to be generated for this process are for a single blade and tower, i.e. same as those required for the isolated turbine simulation. The blade meshes are then replicated 144 times to form 48 3-bladed turbines. In similar fashion, the tower meshes are also replicated 48 times to form the towers of each of the 48 turbines. The total number of near-body grid points for the entire mesh system is close to 100 million. The off-body mesh system grew from an initial number of 50 million nodes to 180 million nodes in 60 adaptation cycles. A total of 30 rotor revolutions (equivalent to 150 seconds of physical time) is simulated. The entire simulation incurred 500,000 CPU hours and was executed on 3840 CPU cores. Note that the Cartesian mesh system utilized for the wind farm simulation is not as high a resolution as the isolated turbine (to make the computations tractable). The highest resolution in the Cartesian mesh system in this case is $2m$ ($0.5\times$ mean chord) compared to $0.5m$ ($0.125\times$ mean chord) used in the isolated rotor simulation. In general, the coarse resolution will cause dissipation of small scale structures in the wake, but will provide adequate resolution for modeling larger scale wake dynamics. Precursor turbulent inflow was generated using the Numerical Weather Prediction software WRF.[27] The turbulent wind flow has a mean wind velocity of 10 m/s and a turbulent intensity of 12% at the hub height location of the turbines. Turbulent inflow is injected at the inflow planes using the technique outlined in section III.B.4.

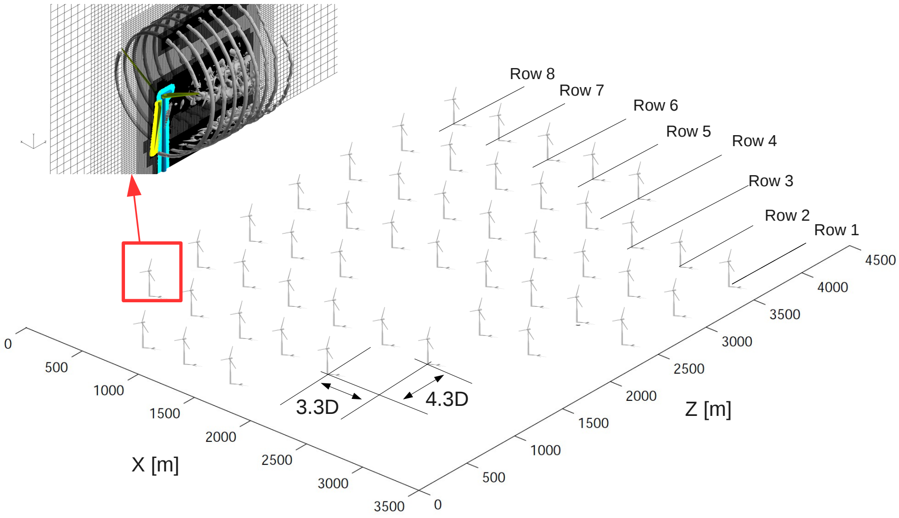


Figure 10. Wind farm lay out similar to that in the Lillgrund wind farm (located 10 km off the coast of southern Sweden). Full geometry of each turbine is modeled using unstructured meshes with an overlapping off-body mesh system for the far field.

IV.E.1. Power prediction in uniform and turbulent inflow

Figure 11 shows the power variation in each of the rows of the wind farm. Velocity deficits caused in the wake of the turbine will lead to power loss in downwind turbines. As expected, the power difference is the largest between the first and second turbines in each row. Since the second turbine is producing much less power compared to the first turbine, it does not add a significant contribution to the overall velocity deficit. Furthermore, the interaction of the vortex wake with upwind turbines makes the entire wake system more turbulent, thus increasing meandering and accelerating breakdown, leading to similar power predictions for all following turbines. In rows 4 and 5, there is larger spacing between the second and third turbines. This layout in the Lillgrund wind farm is because of a shallow water topography found in these locations, which made it difficult to place turbines. Because of the larger spacing, the third turbine in rows 4 and 5 shows lower power deficit when compared to the leading turbine. Although uniform inflow is not realistic, Helios simulations appear to capture the trends in power variation similar to those observed in measurements.[46] Introduction of turbulent inflow, as expected, shows a consistent increase in power for all of the rear turbines. One of the mechanisms of increased power generation is similar to that described above. Turbulent inflow causes increased meandering and faster break down for the wake of the leading turbine itself causing decreased wind velocity deficits and increased power generation. In addition, turbulent inflow causes transport of momentum flux from upper layers of the atmospheric boundary layer in to the turbine layer, which further improves the power generation of the rear turbines.

IV.E.2. Flow visualization

Figure 12 shows the vorticity field overlaid with velocity contours after 14 revolutions of wake evolution and 30 revolution of wake evolution respectively. Before mutual interaction (at 14 revolutions), the wake structure of all turbines look very similar with a initial helical vortex structure that merges into a vortex tube. After the wake interaction (at 30 revolutions), the wake structure of the rear turbine become completely turbulent. This is an expected trend and is consistent with the observations in the two turbine interaction studies performed in the past using Helios.[11] Once the wake system becomes turbulent, there is increased meandering and faster break down. Both of these mechanisms improve the power generation of rear turbines by decreasing the deficit in wind velocity. In Figure 13, the vorticity field is shown overlaid with velocity contours for a turbulent inflow scenario. It can be observed that, when compared with the uniform inflow condition, the turbulent inflow causes faster meandering and break down of the wake vorticity from the leading turbine. The end result can be observed in the power plots shown in Figure 11, where increased power generation can be noted when compared with uniform inflow. Note that the simulations have only progressed up to 30 revolutions (or 150 seconds of physical time). Therefore, the turbulent inflow introduced at the inflow

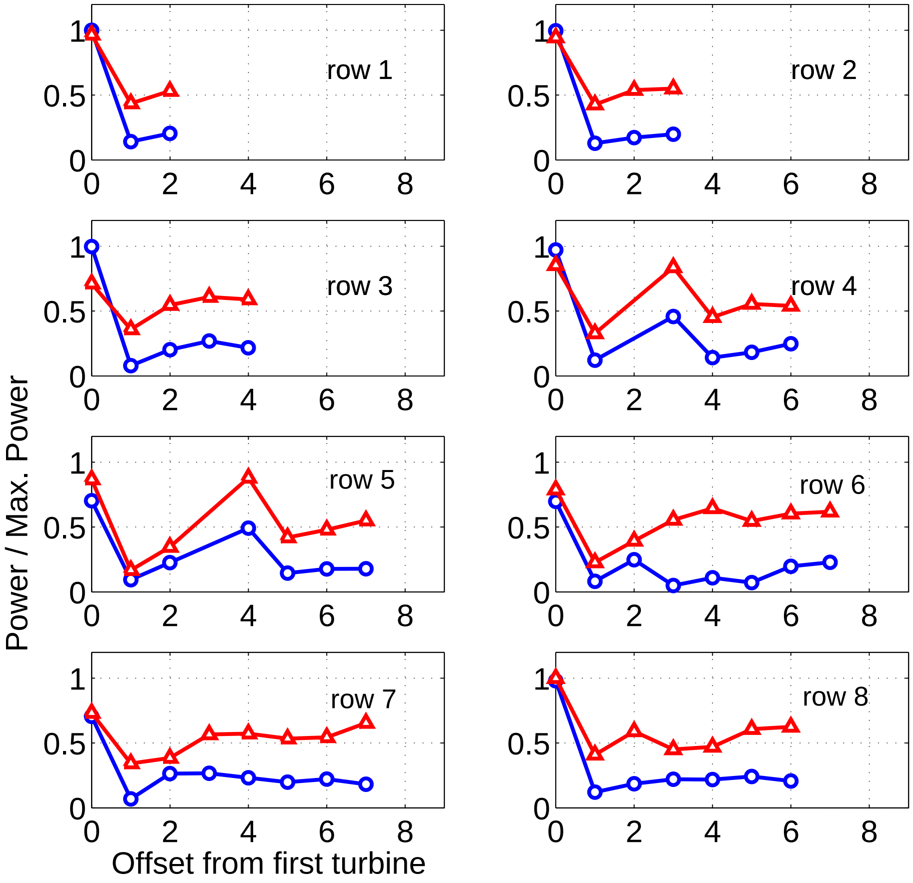
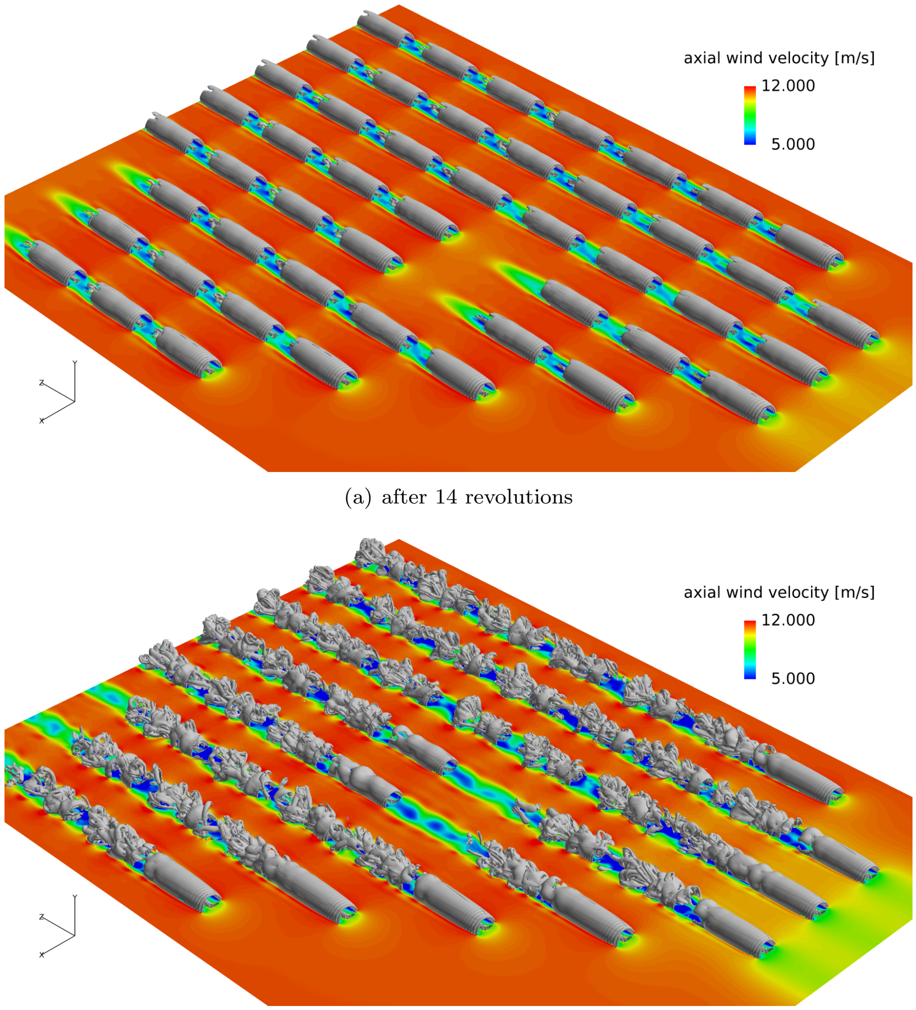


Figure 11. Power variation in each row of the wind farm (Blue circles = uniform inflow, Red triangles = turbulent inflow)

plane has only advected approximately 1500 meters, which is one-third of the farm length. Computations are ongoing and are expected to continue until a physical simulation time of 10 minutes is reached. Figure 14 shows the vorticity distribution at different axial slices along the wind farm. The locations are chosen such that they are at one, two and three rotor diameters behind each turbine column. From Fig 14(a) it can be observed that the vortex wake remains near circular up to three diameters downstream of the turbine with minimal amount of spreading. From the second column results in Fig 14(b), it can be observed that the turbine-wake interaction has caused the middle three turbines to have much larger fluctuations of the vortex wake. The outward turbines in this column are in a leading spot and hence show similar patterns as the leading turbine in column 1. Column 5 has two turbines that have larger spacing compared to the preceding turbine (in rows 4 and 5). The amount of fluctuations are smaller for these turbines compared to their counterparts, which show high levels of fluctuations owing to the closer turbine-wake interactions. Figure 15 shows the vorticity distribution and the corresponding velocity distributions at different axial slices along the wind farm. The observations further confirm the recurring theme in this discussion, i.e. the turbine-wake interactions occurring at the rear turbines introduce a high degree of fluctuations that cause break down of the tubular vortex wake. Vortex break down leads to decreased wind deficits and as a consequence increased power generation.

V. Conclusions

Flow-field simulations of an entire wind plant are conducted using the Helios software framework with full geometric modeling of all individual wind turbines. Computations are performed using dynamic overset grids with adaptive mesh refinement that targets vortex wakes. Results presented use relatively coarse grids to mitigate computational costs and make the simulations tractable. Analysis of predicted performance and



(a) after 14 revolutions

(b) after 30 revolutions

Figure 12. Flow Visualization of the wind plant wake system at two time instances of wake evolution with uniform inflow. Vorticity iso-surfaces ($\omega = 0.34s^{-1}$) are shown overlaid with axial wind velocity contours

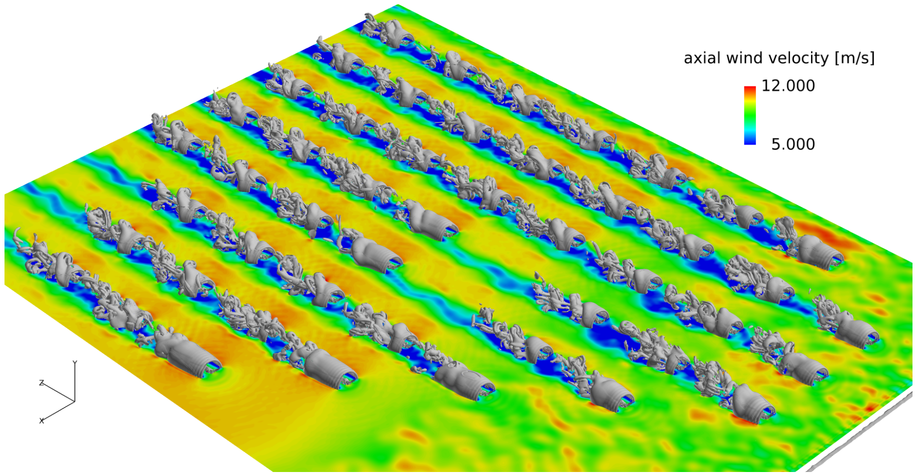
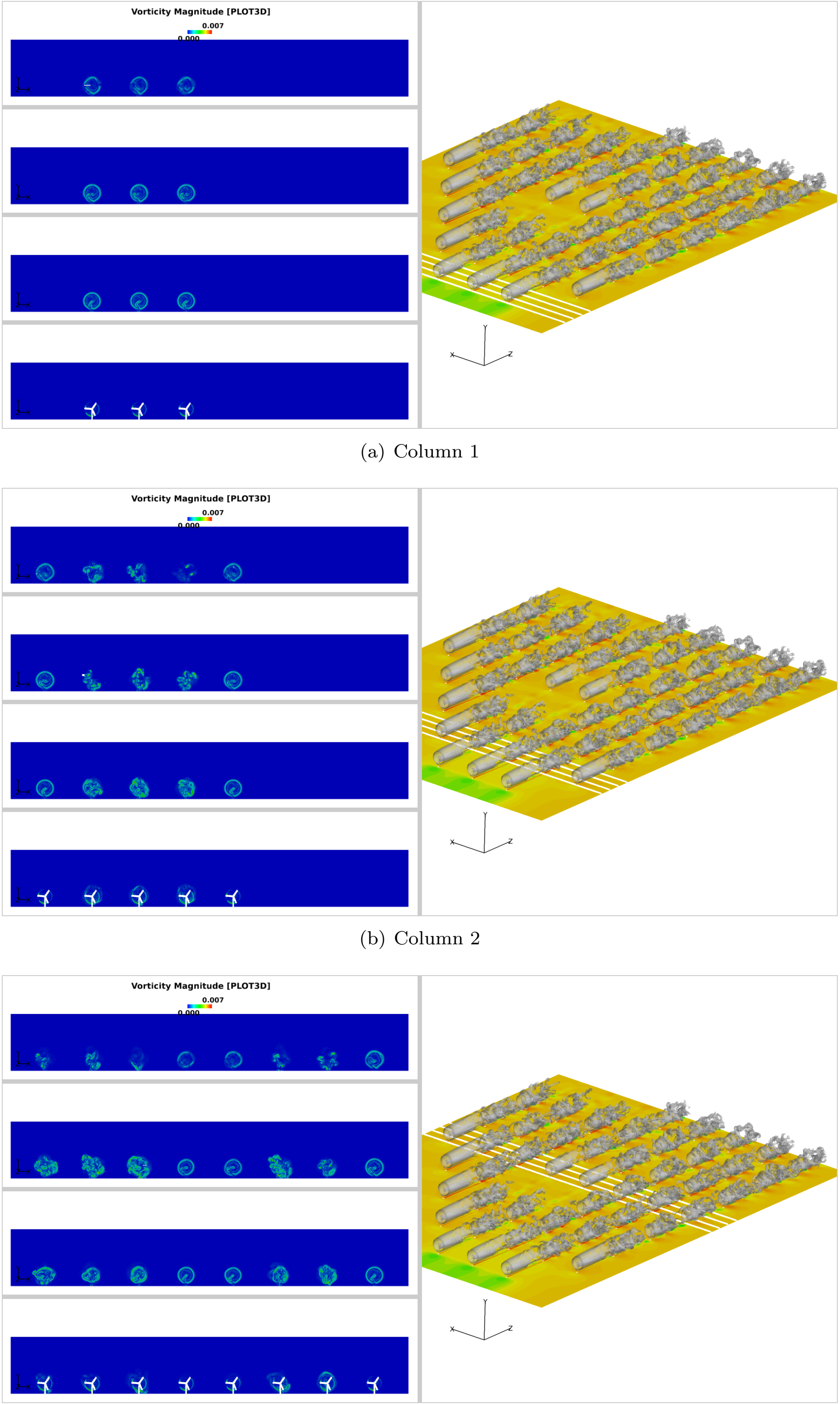


Figure 13. Flow Visualization of the wind plant wake system with turbulent inflow. Vorticity iso-surfaces ($\omega = 0.34s^{-1}$) are shown overlaid with axial wind velocity contours

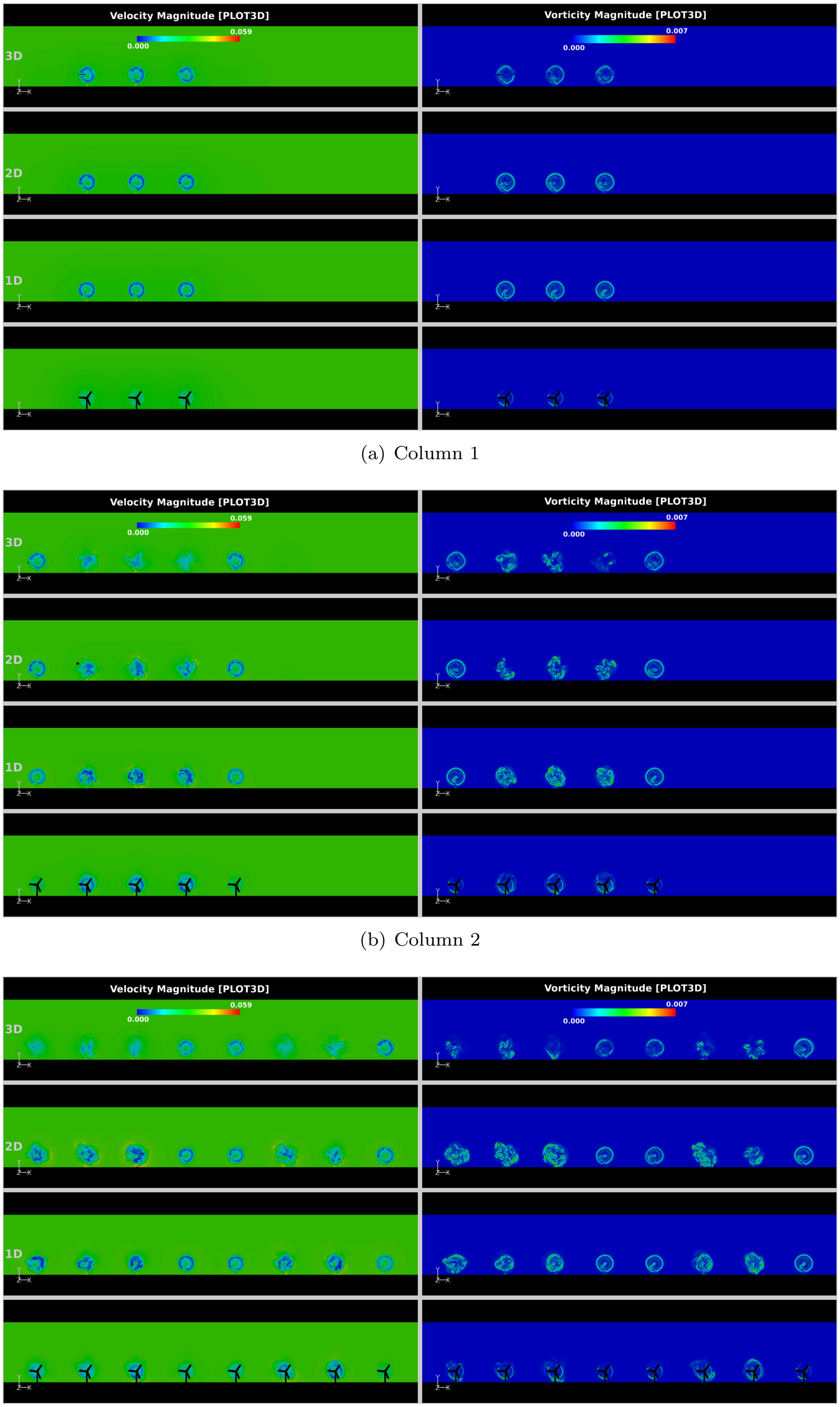

Vorticity Magnitude [PLOT3D]
0.007
0.000
x
y
z
Vorticity Magnitude [PLOT3D]
0.007
0.000
x
y
z
Vorticity Magnitude [PLOT3D]
0.007
0.000
x
y
z

(a) Column 1

(b) Column 2

(c) Column 5

Figure 14. Wake vorticity distribution at twelve different axial slices of the wind farm. Results use uniform inflow. Figures show vorticity contours at locations, $z = 0D$, $z = 1D$, $z = 2D$ and $z = 3D$ on three different column locations. Column numbers are based on turbine numbers in row 6 and 7, which are the complete rows. First column line passes through the leading turbine in row 6, second column line passes through the second turbine in each rows 6 and fifth column line passes through the fifth turbine in row 6 respectively.



(a) Column 1

(b) Column 2

(c) Column 5

Figure 15. Wake vorticity distribution at twelve different axial slices of the wind farm. Results use uniform inflow. Figures show vorticity contours at locations, $z = 0D$, $z = 1D$, $z = 2D$ and $z = 3D$ on three different column locations. Column numbers are based on turbine numbers in row 6 and 7, which are the complete rows. First column line passes through the leading turbine in row 6, second column line passes through the second turbine in each rows 6 and fifth column line passes through the fifth turbine in row 6 respectively.

wind deficits show expected trends–(1) the second turbine in each row has the largest performance deficit (2) power generation is enhanced by turbulent inflow and (3) turbine/wake interaction mitigates velocity deficits in the wake. The focus of this work is to demonstrate technology and verify expected trends in wind farms. Future work will address rigorous validation, where the simulation of realistic wind farms will be validated with measured data. Furthermore, full rotor simulations provide rich data sets in both near and far-field regions. The near-field data can be mined to understand surface pressure distributions, separation patterns and other blade boundary layer dynamics. The far field-data can be mined (shown to a limited extend in this paper) to understand wake velocity deficits and vortex break down patterns. Extensive analysis of simulation data is also intended as part of future work. This work is the ambitious first step towards performing high resolution simulations using full rotor models for entire wind farm installations. Such an effort is unprecedented in the community and is expected to be a pathfinder for future wind energy as well as high performance computing research.

VI. Acknowledgements

The first author acknowledges support from Lawerence Livermore National Labs with Dr. Wayne Miller as the technical monitor. The authors would also like to thank Dr. Bob Meakin who directs the DoD CREATE/AV program and Dr. Roger Strawn at the Army Aeroflightdynamics directorate at NASA Ames for providing access to the use of HELIOS software framework. We are also grateful for the computational time provided by the National Center for Atmospheric Research (NCAR) at the NCAR Wyoming Supercomputing center.

References

[1] of Energy, U. S. D., "20% Wind Energy by 2030: Increasing Wind Energy Contribution to U. S. Electricity Supply," July 2008.

[2] Troldborg, N., Sørensen, J. N., and Mikkelsen, R. F., *Actuator line modeling of wind turbine wakes*, Ph.D. thesis, Technical University of DenmarkDanmarks Tekniske Universitet, Department of Energy EngineeringInstitut for Energiteknik, 2008.

[3] Mikkelsen, R., *Actuator disc methods applied to wind turbines*, Ph.D. thesis, Technical University of Denmark, 2003.

[4] Gopalan, H., Gundling, C., J., S., W., M., and Mirocha, J., "Coupled Mesoscale Microscale Model for Wind Resource Estimation and Turbine Aerodynamics Using an Overset Grid Approach," *51st AIAA Aerospace Sciences Meeting and Exhibit*, Dallas, TX, January 2013, pp. AIAA Paper 2013–1209.

[5] Churchfield, M. and Lee, S., "NWTC Design Codes (SOWFA by Matt Churchfield and Sang Lee)," *http://wind.nrel.gov/designcodes/simulators/SOWFA/*, Last modified 14-March-2012; accessed 14-March-2012.

[6] Calaf, M., Meneveau, C., and Meyers, J., "Large eddy simulation study of fully developed wind-turbine array boundary layers," *Physics of Fluids*, Vol. 22, 2010, pp. 015110.

[7] Johansen, J., Sørensen, N. N., Michelsen, J., and Schreck, S., "Detached-eddy simulation of flow around the NREL Phase VI blade," *Wind Energy*, Vol. 5, No. 2-3, 2002, pp. 185–197.

[8] Duque, E. P., Burklund, M. D., and Johnson, W., "Navier-Stokes and comprehensive analysis performance predictions of the NREL phase VI experiment," *TRANSACTIONS-AMERICAN SOCIETY OF MECHANICAL ENGINEERS JOURNAL OF SOLAR ENERGY ENGINEERING*, Vol. 125, No. 4, 2003, pp. 457–467.

[9] Stone, C., Lynch, C. E., and Smith, M. J., "Hybrid RANS/LES simulations of a horizontal axis wind turbine," *48th AIAA Aerospace Sciences Meeting and Exhibit,(Orlando, FL), AIAA-2010-0459*, 2010.

[10] Xu, G. and Sankar, L. N., "Computational study of horizontal axis wind turbines," *TRANSACTIONS-AMERICAN SOCIETY OF MECHANICAL ENGINEERS JOURNAL OF SOLAR ENERGY ENGINEERING*, Vol. 122, No. 1, 2000, pp. 35–39.

[11] Gundling, C., Roget, B., Sitaraman, J., and Rai, R., "Comparison of Wind Turbine Wakes in Steady and Turbulent Inflow," *50th AIAA Aerospace Sciences Meeting*, Nashville, TN, January 2012, pp. AIAA Paper 2012–0899.

[12] Wissink, A., Potsdam, M., Sankaran, V., Sitaraman, J., Yang, Z., and Mavriplis, D., "A coupled unstructured-adaptive Cartesian CFD approach for hover prediction," *American Helicopter Society 66th Annual Forum*, 2010.

[13] Sitaraman, J., Gundling, C., Roget, B., Masarati, P., and Mavriplis, D., "Computational Study of Wind Turbine Performance and Loading Response to Turbulent Inflow Conditions," *Presented at the 69th Annual Forum of American Helicopter Society, Phoenix, Arizona, May 21-23, 2013*, 2013.

[14] Sitaraman, J., Wisssink, A., Sankaran, V., Jayaraman, B., Datta, A., Yang, Z., Mavriplis, D., Saberi, H., Potsdam, M., O'Brien, D., Cheng, R., Hariharan, N., and Strawn, R., "Application of the HELIOS Computational Platform to Rotorcraft Flowfields," AIAA Paper 2010-1230, Presented at the 48th AIAA Aerospace Sciences Meeting, Orlando FL.

[15] Sitaraman, J., Floros, M., Wissink, A., and Potsdam, M., "Parallel Domain Connectivity Algorithm For Unsteady Flow Computations Using Overlapping and Adaptive Grids," *Journal of Computational Physics*, Vol. 229, No. 12, 2010, pp. –.

[16] Gundling, C., Roget, B., and Sitaraman, J., "Prediction of Wind Turbine Performance And Wake Losses Using Analysis Methods Of Incremental Complexity," *AIAA Aerospace Sciences Meeting and Exhibit*, Orlando, FL, January 2011, pp. AIAA Paper 2011–0458.

[17] R. D. Hornung, A. M. Wissink, S. R. Kohn, Managing complex data and geometry in parallel structured amr applications, Engineers and Computers 22 (3) (2006) 181–195.

[18] A. Wissink, J. Sitaraman, D. Mavriplis, T. Pulliam, V. Sankaran, A python-based infrastructure for overset CFD with adaptive Cartesian grids, in: AIAA 48th Aerospace Sciences Meeting, no. AIAA-2008-0927, AIAA, Washington, DC, 2008.

[19] J. Sitaraman, A. Katz, B. Jayaraman, A. Wissink, V. Sankaran, Evaluation of a multi-solver paradigm for CFD using overset unstructured and structured adaptive Cartesian grids, in: AIAA 48th Aerospace Sciences Meeting, no. AIAA-2008-0660, AIAA, Washington, DC, 2008.

[20] J. Sitaraman, M. Floros, A. Wissink, M. Potsdam,"Parallel Domain Connectivity Algorithm For Unsteady Flow Computations Using Overlapping And Adaptive Grids," Journal of Computational Physics, Volume 229, Issue 12, p. 4703-4723.

[21] J. J. Alonso, P. LeGresley, E. Van Der Weide, pyMDO: A framework for high-fidelity multi-disciplinary optimization, in: AIAA/ISSMO 10th Conference on Multidisciplinary Analysis and Optimization, no. AIAA 2004-4480, AIAA, Washington, DC, 2004.

[22] M. J. Berger, P. Colella, Local adaptive mesh refinement for shock hydrodynamics, JCP 82 (1) (1989) 65–84.

[23] D. Mavriplis and W. ,D. Levy Transonic Drag Prediction Using an Unstructured Multigrid Solver, *Journal of Aircraft*, Vol. 42, No. 4, July-August, 2005.

[24] D. Mavriplis,"DLRF6 WB and WBF Results using NSU3D," Proceedings of the 3rd Drag Prediction Workshop, San Francisco, CA, June 2006.

[25] Y. Lee, J. D. Baeder, Implicit hole cutting - a new approach to overset grid connectivity, in: AIAA 16th Conference on Computational Fluid Dynamics, no. AIAA 2003-4128, AIAA, Washington, DC, 2003.

[26] R. L. Meakin, A. M. Wissink, W. C. Chan, S. A. Pandya, J. Sitaraman, On strand grids for complex flows, in: AIAA 18th Conference on Computational Fluid Dynamics, no. AIAA 2007-3834, AIAA, Washington, DC, 2007.

[27] Skamarock, W. C. and Klemp, J. B., "A time-split nonhydrostatic atmospheric model for weather research and forecasting applications," *J. Comp. Phys.*, Vol. 227, No. 7, 2008, pp. 3465–3485.

[28] Mann, J., "The spatial structure of neutral atmospheric surface-layer turbulence," *Journal of Fluid Mechanics*, Vol. 273, No. 1, 1994, pp. 141–168.

[29] Mavriplis, D. J. and Venkatakrishnan, V., "A Unified Multigrid Solver for the Navier-Stokes Equations on Mixed Element Meshes," *International Journal for Computational Fluid Dynamics*, Vol. 8, 1997, pp. 247–263.

[30] Mavriplis, D. J., "Unstructured Mesh Discretizations and Solvers for Computational Aerodynamics," AIAA Paper 2007-3955.

[31] Dalcín, L., Paz, R., and Storti, M., "MPI for Python," *Journal of Parallel and Distributed Computing*, Vol. 65, No. 9, 2005, pp. 1108–1115.

[32] Mavriplis, D. J., "Multigrid Strategies for Viscous Flow Solvers on Anisotropic Unstructured Meshes," *Journal of Computational Physics*, Vol. 145, No. 1, Sept. 1998, pp. 141–165.

[33] Spalart, P. R. and Allmaras, S. R., "A One-equation Turbulence Model for Aerodynamic Flows," *La Recherche Aérospatiale*, Vol. 1, 1994, pp. 5–21.

[34] Mavriplis, D. J., "Results from the Third Drag Prediction Workshop using the NSU3D Unstructured Mesh Solver," AIAA Paper 2007-0256.

[35] Mavriplis, D. J. and Long, M., "NSU3D Results from the Fourth AIAA CFD Drag Prediction Workshop," AIAA Paper 2010-4364.

[36] Long, M. and Mavriplis, D. J., "NSU3D Results from the First AIAA CFD High-Lift Prediction Workshop," AIAA Paper 2011-863 presented at the 49th AIAA Aerospace Sciences Meeting.

[37] Mavriplis, D. J., Z.Yang, and Long, M., "Results using NSU3D for the First AIAA Aeroelastic Prediction Workshop," AIAA Paper 2013-0786, 51st AIAA Aerospace Sciences Meeting, Grapevine TX.

[38] Mavriplis, D. J. and Mani, K., "Unstructured Mesh Solution Techniques using the NSU3D Solver," AIAA Paper 2014-xxx, AIAA Scitech 2014 Conference, National Harbor, MD.

[39] Karypis, G. and Kumar, V., "A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs," *SIAM Journal on Scientific Computing*, Vol. 20, No. 1, 1999, pp. 359–392.

[40] Chevalier, C. and Pellegrini, F., "PT-SCOTCH: A tool for efficient parallel graph ordering," *Parallel Computing*, Vol. 34, No. 6, 2008, pp. 318–331.

[41] Roget, B. and Sitaraman, J., "Robust and Scalable Overset Grid Assembly for Partitioned Unstructured Meshes," *51st AIAA Aerospace Sciences Meeting and Exhibit*, Dallas, TX, January 2013, pp. AIAA Paper 2013–797.

[42] Moeng, C., Dudhia, J., Klemp, J., and Sullivan, P., "Examining two-way grid nesting for large eddy simulation of the PBL using the WRF model," *Monthly weather review*, Vol. 135, No. 6, 2007, pp. 2295–2311.

[43] Jonkman, J. M., Butterfield, S., Musial, W., and Scott, G., *Definition of a 5-MW reference wind turbine for offshore system development*, National Renewable Energy Laboratory Colorado, 2009.

[44] Pirzadeh, S., "Unstructured viscous grid generation by the advancing-layers method," *AIAA journal*, Vol. 32, No. 8, 1994, pp. 1735–1737.

[45] Jonkman, J. M. and Buhl Jr, M. L., "FAST users guide," *Golden, CO: National Renewable Energy Laboratory*, 2005.

[46] Larsson, A., "Practical experiences gained at Lillgrund offshore wind farm," *7th international workshop on large scale integration of wind power and on transmission networks for offshore wind farms*, 2008, pp. 26–27.