Horizontal Axis Wind Turbine Free Wake Model for AeroDyn

Prof. Hugh Currin, P.E., PhD
Professor, Mechanical Engineering, Oregon Institute of Technology

Prof. James Long
Associate Professor, Computer Systems Engineering Technology, Oregon Institute of Technology

August 27, 2009

Introduction

The objective of this research project was to develop a free vortex wake model for aerodynamic analysis of horizontal axis wind turbines. This to be built into the NREL aerodynamic performance code AeroDyn.

AeroDyn provides the aerodynamic input to the structural dynamic code FAST. [1, 2] Together they form an aeroelastic model for analysis of horizontal axis wind turbines. [3] AeroDyn is the predominant aerodynamic code used for design in the United States and has been validated for use in the European Union. [4] The two aerodynamic models in AeroDyn are Blade Element Momentum (BEM) and Generalized Dynamic Wake (GDW). These models generally perform well but are lacking in some important cases. It is likely designers will be searching, in the next 5 years or so, for alternative models for use in particular design conditions.

Free vortex wake models are a likely candidate for advanced modeling. They give a more accurate and detailed description of the aerodynamic wake than BEM or GDW and can be accurately applied in conditions these other models can not. [5] Free wake models are more computationally expensive than the BEM and GDW models but still much more efficient than application of computational fluid dynamic (CFD) codes. [6, 7] The application of CFD models to wind turbines has proven problematic and the run times are much too long for use in engineering design.

Existing Models

As a turbine's rotor extracts energy from the wind, that wind flow is slowed. There is a balance between the energy removed by the turbine and the resulting change in flow. Several wake models have been used to analyze this interaction. One of the simplest, and most widely used, is the Blade Element Momentum (BEM) model. Here a static balance between the blade loads and slowing of the flow is maintained. The balance involves a radial portion of the blade and the flow affected by that blade element. Two problems surface using this approach. First, each blade element, and its flow, has no affect on adjacent elements. This is inadequate for prediction of yawed performance were the turbine wake is skewed. Also the assumed static, or instantaneous, balance between blade forces and flow neglects dynamic wake effects. This method is very good for steady, head on flow but lacking for many yawed and dynamic conditions. The GDW model was developed to address yawed and dynamic flow. This actuator disk model is surprisingly good for its computational simplicity. However, it contains many simplifying assumptions which bring its results into question.

Another way to model the turbine wake is through vortex methods. Here the energy extracted from the flow is modeled as vorticity. The vorticity may be modeled as line vortices shed and

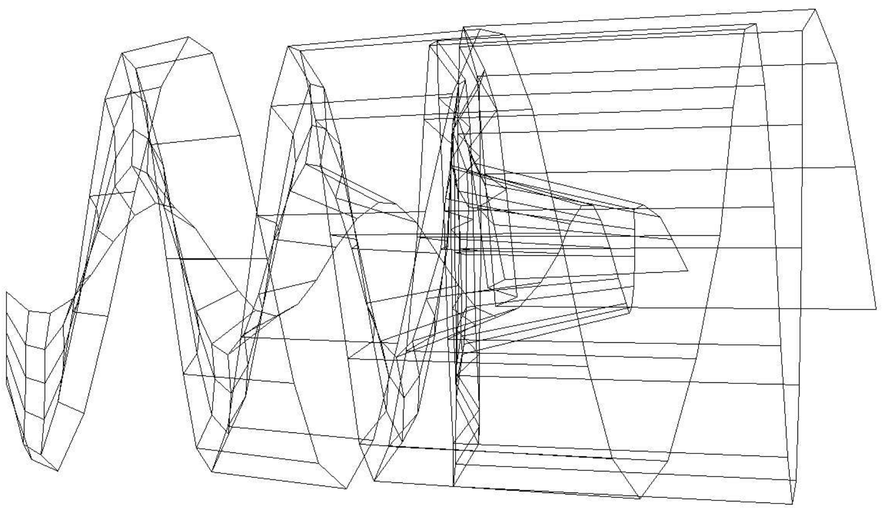

Figure 1: Vortex Wake

trailed into the wake from the blades. This forms a helical vortex lattice representing the turbine wake, similar to that shown in Figure 1. Once free from the blade each wake vortex is free to move under the influence of the other wake vortices, hence the terminology "free vortex wake model." This is a reasonably accurate representation of the wake. Each wake vortex has an effect on all blade elements inherently modeling the interaction between these blade elements. Also dynamic wake effects are inherently modeled since the vortices interact dynamically.

The researcher's previous work linked a Dynamic Prescribed Vortex Wake model to AeroDyn. [8] A prescribed wake model is similar to a free wake model but the locations of the wake vortices are determined by empirical prescription functions rather than vortex interactions. This prior work provided the basis for the new Free Vortex Wake model. The interface with AeroDyn is the same for the new model and many of the routines and data structures in this prescribed wake model are applicable to a free wake code.

Analysis

A turbine slows air flow as it extracts energy. Optimally, a turbine will slow the flow, at the rotor, by 1/3. This slowing of air flow affects the velocity and angle of attack which the turbine blades see. For accurate performance predictions this slowing of air flow, or turbine wake, must be included in aerodynamic models. A free vortex wake model is one method of depicting this wake, the flow downwind of a turbine.

Basic Theory

A vortex wake model takes, as input, the vortex strengths generated by the turbine blades over time. The wake model determines induced velocities at the blades due to the wake, again as a function of time. This induced velocity, combined with wind speed and blade motions, determines the velocity and angle of attack seen by the blades allowing lift and drag calculation to be done.

These blade loads determine the vortex strengths generated and also feed into the structural code to determine blade motions.

In the current model a lifting line theory is used. This depicts blade lift as a bound vortex along the quarter cord of the blade. If the lift is known radially along the blade the strength of the blade bound vortices can be found using a form of the Kutta-Joukowski theorem [9],

$$dL = \rho V_R \Gamma_b dr \tag{1}$$

where dL - Lift force over blade radial length dr, ρ - air density, V_R - relative wind speed, Γ_b - strength of circulation, and dr - radial length of blade section.

In the current model the blade is separated into radial elements. Discrete vortices are assumed trailed into the wake due to radial changes in blade lift between elements. Likewise discrete vortices are shed from the blade each time step due to changes in lift, and thus bound vorticity over time. The resulting wake is then a ladder of discrete vortices trailing from each blade, as in Figure 1. The "sides" of the ladders are due to changes in bound vorticity from one blade element to the next. The "rungs" are due to changes in vorticity with time, one for each time step.

Once the vortices are generated by the blades they are free to migrate downwind. How they move is determined by the local flow, induced velocity and wind speed. The induced velocity is determined by using the Biot-Savart Law. This states a vortex will induce a velocity at any point removed from that vortex. In Figure 2 the differential vortex element $d\mathbf{S}$ will induce a velocity $d\mathbf{V}$ at point P. With the position vector $\mathbf{R}$ locating point P relative to the vortex element, the Biot-Savart Law [9] gives the induced velocity at P as

$$d\mathbf{V} = \frac{\Gamma d\mathbf{S} \times \mathbf{R}}{4\pi R^3} \tag{2}$$

where Γ is the vortex strength.

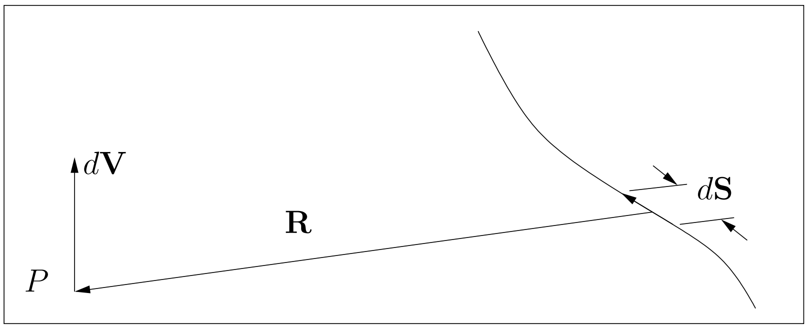

Figure 2: Biot-Savart Law

If a vortex is taken to be along a straight line this equation can be manipulated into a vector equation as. [8]

$$\mathbf{V} = \frac{\Gamma}{4\pi} \frac{\mathbf{r_a} \times \mathbf{r_b}}{|\mathbf{r_a} \times \mathbf{r_b}|^2} \left[\frac{\mathbf{r_b} \cdot \mathbf{\Delta r}}{r_b} - \frac{\mathbf{r_a} \cdot \mathbf{\Delta r}}{r_a} \right] \tag{3}$$

This gives the induced velocity at a point P generated by a straight line vortex between points a and b. The vectors involved are shown in Figure 3.

The Biot-Savart Law predicts infinite velocity at the center of, or on the line of, a vortex. This is not realistic so a core model is used to reduce velocities inside a set distance from the vortex.

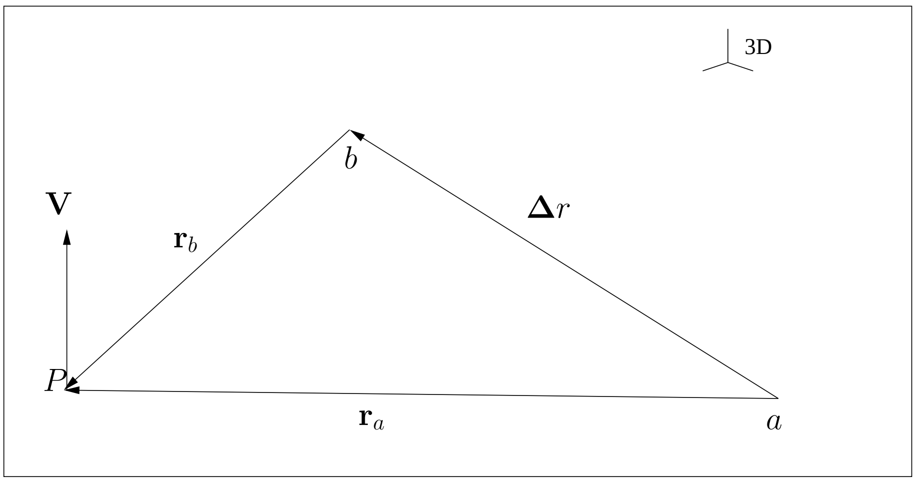

Figure 3: Biot-Savart Vectors

With this form of the Biot-Savart Law we can determine the induced velocity, generated by a particular straight vortex segment, anywhere in the flow field. To find the overall induced velocity from the wake the individual contributions from all vortex segments are summed. Thus the Biot-Savart Law is repeatedly applied to find the induced velocity at each blade element, the local flow at the blade is determined.

The vortices in the wake will move over time. A free wake model determines the motion of each vortex, or vortex end, through Biot-Savart calculations. The induced velocity at each wake point, vortex end, is determined by summing contributions from each wake vortex. This is done for each wake point each time step. This calculation of induced velocity at each wake point is the computationally expensive portion of a free vortex wake model.

The motion of each wake point is then a first order ordinary differential equation in time. Actually there are three equations, X/Y/Z at each wake point. A predictor-corrector numerical procedure is used to track each wake point through time.

Free Wake Model

The current Free Wake Model uses a Prescribed Vortex Wake Model to initiate the wake structure. [8] The code could start with a helical wake but the prescribed wake model is a better starting condition.

Once a starting wake is found a time stepping solution is initiated. This is driven by the structural code which calls the aerodynamic code once each time step. The Free Wake Model procedure can be described, in broad brush strokes, as follows.

- Induced velocities at each wake point are known for the previous time step from the prior iteration. (If not know on the first pass they are calculated)
- An Ordinary Differential Equation (ODE) predictor-corrector solution is used. Here the predictor is used to predict the position of each wake point at this new, current, time. (In the current code a simple Euler predictor-corrector is used.)
- With an estimate of the current wake, induced velocities at the blade elements are found.
- With induced velocities at the blade estimated lift forces and bound vorticity are found.

- Having estimates of bound vorticity the trailed and shed vorticity are estimated.
- A numerical ODE corrector is applied using new estimates of wake positions and trailed & shed vorticity. This will give the position of each wake point at the new, current, time. (This is computationally expensive portion of a free wake model involving many Biot-Savart calculations.)
- Knowing the current wake positions a final iteration is done to determine bound vorticity, as well as shed & trailed vorticity. The following steps are repeated until convergence is obtained.
  - Determine induced velocity at each blade element.
  - Using lift and drag tables find blade loads and bound vorticity.
  - With bound vorticity update trailed and shed vorticity.
- Pass loads back to structural code and wait for next iteration.

Additional numerical method corrector steps could be included in the iteration above. However this would significantly slow the calculations.

The current Free Wake Model is very simple. The aim of this project was to develop the core code without refinements. Downwind the wake is simply truncated, cut off after a certain number of cycles. Some vortex models bunch wake vorticity into one tip vortex after a small number of rotor cycles. It would be possible to also bunch vorticity into one tip and one root vortex. Both methods would significantly reduce the number of vortex elements thus speeding execution. Such combining of vorticity has not been done in the current model.

Little work has yet been spent refining the predictor-corrector numerical method used. Currently a simple Euler predictor-corrector is implemented.

Multi-core Processing

Biot-Savart calculations take the vast majority of the codes execution time. These calculations can be done independently which is tailor made for a multi-core processor. The open source library, OpenMP, was chosen to introduce multiple threaded computation into the algorithm because of the non-intrusive nature of the macro language. Using simple OpenMP compiler directives, the code base was modified to perform multi-threaded calculations without impacting the code structure or requiring the introduction of specialized threading libraries. Compilers that do not support the OpenMP compile time directive simply treat the macros as comments.

To target candidate areas in the algorithm where parallel execution would have the largest impact, the Intel suite of multi-core programming tools was used. To gain insight into areas of the code base where a multiple threads would have largest impact, the tool vTune Analyzer was used. The run was configured with 2 blades and 10 blade elements. Initial code runs and analysis were performed on an Intel Dual Core 2.4 Ghz cpu running Windows Vista with 4 G-bytes RAM.

Initial analysis showed:

- 88.6 percent of CPU cycles were spent in routine biotsav.
- 10.0 percent of CPU cycles were spent in routine pwindvel.

Routine pwindvel contains a sequence of nested do loops. At the core of these loops is a call to the biotsav routine. This was a key place to introduce the OpenMP directives:

- OMP Parallel Do
- OMP Private
- OMP Reduction

This placement allows multiple threads to increase the overall simulation run-time performance.

Results

Very few runs have been done to validate the new Free Vortex Wake Model. The goal of the work was to develop the basic code. Runs which have been done model the UAE turbine. This was a highly instrumented wind tunnel test of a thirty foot turbine. [10] Computer runs modeled operation at 8 m/s using four available models, BEM, GDW, Prescribed Vortex Wake and the new Free Vortex Wake. This model assumed a rigid structure and steady head on wind. The results from each model are shown in Table 1. Each run used the same input data.

Table 1: Power Calculations

Power Calculations		
Model	Power (kW)	Rotor Torque (Nm)
Blade Element Momentum	8.06	1070
Generalized Dynamic Wake	8.55	1140
Dynamic Prescribed Wake	8.08	1070
Free Vortex Wake	8.21	1090

For this same model axial inductions were plotted at each blade element. This is a good way to determine steady state characteristics giving an indication of performance radially along the blade. Figure 4 shows this plot for the four models available.

The Dynamic Inflow, Generalized Dynamic Wake, model is not expected to predict axial induction well. It shows a trend through the center of data but details aren't useful. The Blade Element Momentum model follows the other models away from the blade tip and root. At the tip and root it predicts higher axial induction than the other models. A Prandtl root and tip loss model was applied. The Dynamic Prescribed Wake and Free Vortex Wake models follow nearly identical trends. The Free Wake predicts a slightly lower axial induction at all points than the Prescribed Wake model. This steady state correlation shouldn't be a surprise, the prescription functions used in that model were derived from a free wake model.

The data in Figure 4 does indicate the new Free Wake Model is functioning as expected.

The run times are also of interest. Table 2 below shows the run times for each model. This was using 10 blade elements, 15^o azimuth steps while retaining 5 wake cycles.

The Free Wake model is considerably slower than the other models. It is some 35 times slower than the prescribed wake model and 400 times slower than the standard BEM model.

It would be advantageous to speed up the free wake solution times. As stated above, the Biot-Savart calculations, which take most of the time in a free wake model, are very parallel in nature. Results of applying multi-core parallel processing to the model was undertaken. Sample runs were done on an idle machine with .5 percent operating system overhead. All runs with the exception of the single thread run, had 100 percent CPU usage. Optimization was set to speed over code size. Initial results for the Dual Core 2.4 Ghz CPU are shown in Table 3.

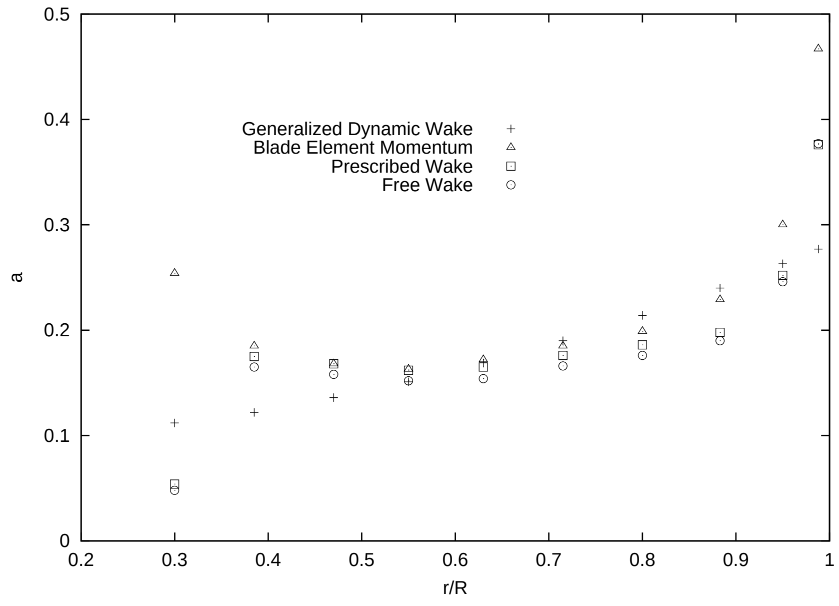

Figure 4: Axial Induction

Table 2: Program Run Times

Run Times	
Model	Time (s)
Blade Element Momentum	2.624
Generalized Dynamic Wake	2.620
Dynamic Prescribed Wake	27.97
Free Vortex Wake	1027.5

Note that, as expected, if more threads are requested than cores available run times start to increase.

Runs of one and two processing threads show a significant decrease in processing time as the thread count increases. When the thread count exceeds the CPU available cores, thread creation and scheduling overhead cause the run time to increase; however, the run time remains significantly better than the single threaded processing model. The initial results indicate increasing number of processing cores has the potential to continuously decrease run time.

The simulation program was then taken to a machine configured with two quad core, 2.6 Ghz CPUs, also running Windows Vista. The runs were made on simulations starting with a single thread increasing consecutively to 8 threads. The results of the simulation runs on the quad core CPUs are shown in Table 4. Figure 5 shows the multi-core results graphically.

Parallelization of the Biot-Savart algorithm proves to be very beneficial in decreasing overall simulation run time. The most significant reduction in overall run time is with the addition of a second thread, reducing time by approximately 45%. As processing threads are added, run time continues to decrease, showing less benefit; however, with the 8 cores in the system running at full capacity, the trend continued downward without hitting a plateau. This indicates adding beyond 8 processing cores would continue to decrease simulation run times.

Table 3: Dual Core Run Times

Run Times	
Threading Mode	Time (s)
1 Thread	420.774
2 Threads	250.226
3 Threads	356.241
4 Threads	353.227
5 Threads	401.778
6 Threads	418.205
7 Threads	476.424
8 Threads	526.703

Table 4: Eight Core Speedup

Run Times		
Threading Mode	Time (s)	Speedup %
1 Thread	571.234	0.0
2 Threads	312.468	45.3
3 Threads	232.700	59.3
4 Threads	184.125	67.8
5 Threads	165.790	71.0
6 Threads	144.546	75.0
7 Threads	131.472	77.0
8 Threads	128.367	77.5

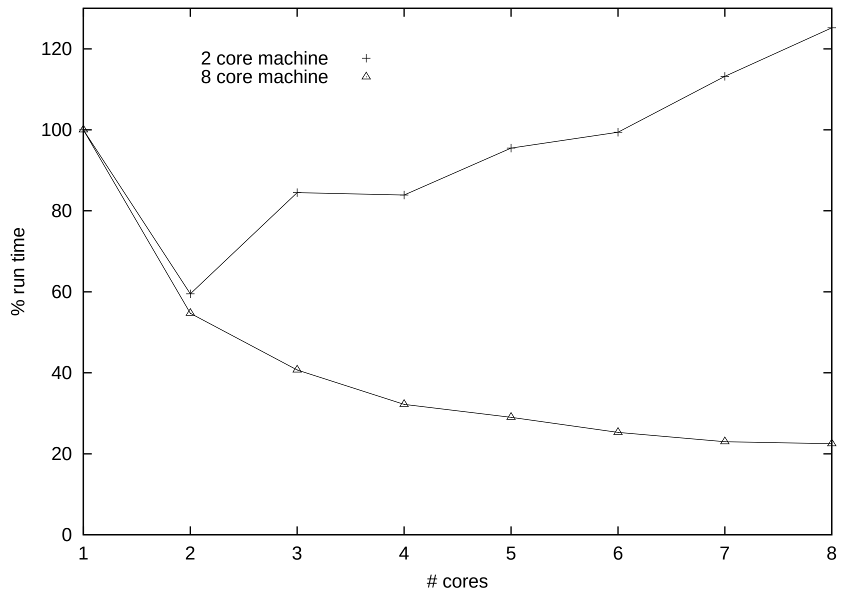

Figure 5: Mulit-core Run Times

Conclusions

The two research goals were to develop the core of a Free Vortex Wake model and to show the applicability of multi-core processing to such a model.

A basic Free Vortex Wake model has been developed and integrated into AeroDyn. The model is rudimentary but will form a good basis for ongoing research.

The introduction of multi-threading into the algorithm was done with the OpenMP library. This allowed multi-threading to be added to the simulation with no modification of the algorithm itself. The results of better than 75% program speed-up are very promising, showing that the Free Vortex Wake calculation can be done in reasonable processing time with the new generation of multi-core CPUs.

The researchers are pleased to report the goals of the project have been met.

Future Development

A Free Vortex Wake model linked to AeroDyn is a powerful tool on which to base future research. The first carry on work will be to better validate the code, both statically and dynamically. The code, as well as AeroDyn and its structural code FAST are open source. A validated code will be an attractive vehicle for ourselves and other researchers to investigate turbine operation. Two areas of such research are anticipated to be yawed flow and dynamic flow. The new model should give insight into these regimes of operation.

The main hurdle to using such free wake models is the computational expense they entail. The multi-core work done is anticipated to make this, and similar codes, more attractive. There is a research opportunity to improve execution times through model refinement as well as multi-core processing. Both are a good basis for ongoing research. Techniques such as tip and root vortex roll up are examples. Research here would be to implement the techniques followed by validation of the new models.

Opportunities for ongoing research are now being investigated.

Expenses

The project budget was for faculty time only, no equipment funds. A multi-core processor is now on loan from Intel.

The times expended on the project by Hugh Currin and Jim Long exceeded those proposed though this was anticipated from the start. The expenses were as follows:

Faculty Time and Cost		
Prof. Currin	169.5 hours @ \$49/hr	\$ 8306.
Prof. Long	162.5 hours @ \$40/hr	\$ 6500.
Indirect Costs		
OIT federally approved indirect rate of 58.5% on \$14,806 in salaries		\$ 8622.
Total Project Cost		\$ 23,468.

Dr. Jason Jonkman from the National Wind Technology Center at the National Renewable Energy Lab assisted on the project. However his time was paid directly by NREL.

Prof. Coton consulted on the project but was not able to contribute the 17 hr anticipated. These funds were distributed to OIT faculty for their additional hours.

References

[1] Laino, D. J., and Hansen, A. C., 2002. User's guide to the aerodynamics computer software aerodyn. Tech. rep., NREL, Golden CO.

[2] Moriarty, P. J., and Hansen, A. C., 2005. Aerodyn theory manual. Tech. rep., NREL, Golden CO.

[3] Jonkman, J. M., and Buhl Jr., M. L., 2005. Fast user's guide. Tech. Rep. NREL/EL-500-38230 (previously NREL/EL-500-29798), National Renewable Energy Laboratory, Golden CO, August.

[4] Buhl, M. L., and Manjock, A., 2006. "A comparison of wind turbine aeroelastic codes used for certification". In 44th AIAA Aerospace Sciences Meeting and Exhibit.

[5] Gupta, S., and Leishman, J. G., 2005. "Comparison of momentum and vortex methods for the aerodynamic analysis of wind turbines". In Proceedings of the 24th ASME Wind Energy Symposium, Reno, NV, Jan. 10-13.

[6] Leishman, J. G., 2002. "Challenges in modeling the unsteady aerodynamics of wind turbines". *Wind Energy,* **5**, pp. 85–132.

[7] Hansen, M. O. L., and et. al., 2006. "State of the art in wind turbine aerodynamics and aeroelasticity". *Progress of Aerospace Science,* **42**, pp. 285–230.

[8] Currin, H. D., Coton, F. N., and Wood, B., 2007. "Dynamic prescribed vortex wake model for aerodyn". In 45th AIAA Aerospace Sciences Meeting and Exhibit.

[9] Leishman, J. G., 2000. *Principles of Helicopter Aerodynamics.* Cambridge University Press.

[10] Shreck, S., 2002. "The nrel full-scale wind tunnel experiment". *Wind Energy,* **5**, pp. 77–84.