

QBlade v0.95

Guidelines for Lifting Line Free
Vortex Wake Simulations



David Marten

qblade@outlook.com

Copyright 2016, TU Berlin



Notice:

This report was prepared as part of the research focus PAK780 “*Wind Turbine Load Control Under Realistic Turbulent Inflow Conditions*”, funded by the German Science Foundation (DFG).

The software QBlade is being developed and released according to the principles of the General Public License. One important point about the GPL is that this program is distributed without any warranty (neither the warranty of merchantability nor the warranty of fitness for a particular purpose). The resulting software is not intended as a professional product and does not offer any guarantee of robustness or accuracy. It is distributed as a personal use application only.

This software may not be faultless and there will certainly be bugs discovered after the distribution. If you discover a bug please report it on the discussion forum on QBlades sourceforge project page, found at: <http://sourceforge.net/p/qblade/discussion/>. The forum also covers a help section where questions are answered. For other inquiries contact the author directly under qblade@outlook.com.

QBlade is being developed at the Wind Energy Group of the Hermann Föttinger Institute (the Chair of fluid Dynamics) of TU Berlin: <http://fd.tu-berlin.de/>. The project and code are hosted on sourceforge: <http://sourceforge.net/projects/qblade/> and the projects webpage is: <http://q-blade.org/>. If you want to reference QBlade inside a publication please refer to the list of QBlade publications at the end of this document.

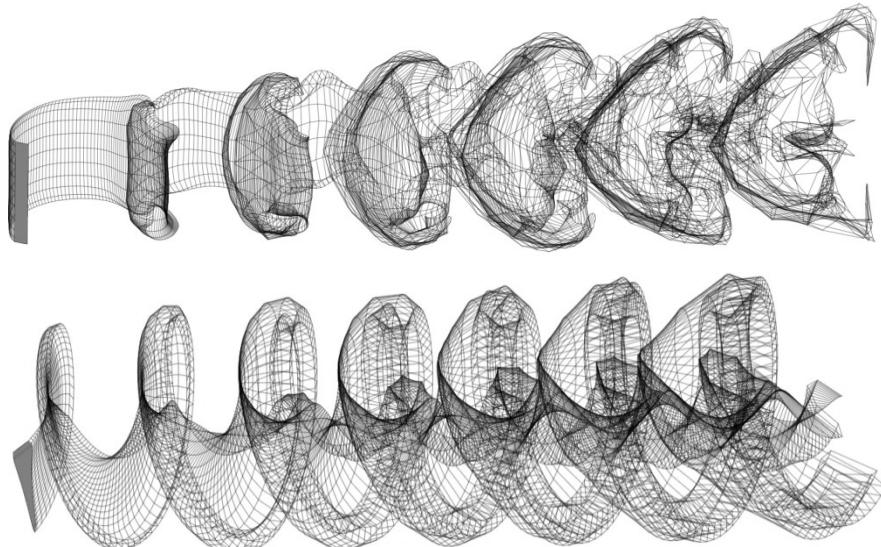


Table of Contents

Introduction.....	1
What's new in this document	1
What's new in QBlade v0.95	2
Theory.....	2
Nonlinear Lifting Line Free Vortex Wake method.....	2
The LLFWV implementation in QBlade	3
Unsteady Aerodynamics Model	5
Attached (potential) flow model.....	5
Dynamic stall model	6
Variable Reynolds number blade definitions	7
Airfoil Self-Noise Evaluation.....	8
Graphical User Interface.....	9
Lifting Line Free Vortex Wake module GL view.....	9
Access to module via main toolbar	9
Simulation control	10
Progress bar.....	10
Velocity planes	11
Turbine visualization	13
Velocity cut plane visualization and export.....	13
Volumetric velocity distribution (ParaView) export.....	15
Lifting Line Free Vortex Wake module graph (2D) view.....	16
Section to plot	17
Styles	17
Time graphs	17
Blade graphs	18
Preprocessor for lift polar decomposition	18
Guidelines for setting up a simulation	20
Name and rotor	21
Blade discretization	22
Simulation type	22
Wind input data.....	24

Rotor angles.....	26
Starting position	28
Operational point.....	28
Length of simulation.....	28
Time discretization.....	29
Modeling parameters.....	29
Wake convection.....	31
Wake modeling.....	32
Vortex modeling	39
Unsteady aerodynamics model.....	41
Special test cases.....	42
Output parameters.....	42
Algorithm parameters	43
Environmental variables.....	43
Simulation output variables:	43
Validation	45
HAWT.....	45
VAWT.....	47
Bibliography.....	48
List of QBlade Publications	50
List of Figures.....	52
List of Tables.....	53

Introduction

This short manual is not intended to give a full overview of the applied simulation methods and their theory. This document is rather meant as a brief guide to enable the users to work with the new version of QBlade and give a short overview of the overall functionality around the “*Lifting Line Free Vortex Wake*” module that was implemented in QBlade v0.9.

This document is an updated version of the “QBlade v0.9 Users Guide”. QBlade v0.95 introduces some new functionality for which additional paragraphs have been added to this document. A dynamic stall model has been implemented which can be used in HAWT & VAWT simulations. Additionally this new version allows assigning multiple polars to blade sections - to take into account the spatial and temporal variation of *Reynolds* numbers during a simulation. Also, for Lifting Line simulations, export functionality to *ParaView* was added, several bugs have been dealt with and overall stability and handling of the code has been improved. Furthermore, through a cooperation with the Poli-USP (Escola Politécnica da Universidade de São Paulo), PNoise – a module to evaluate the generation of airfoil self-noise based on the NASA BPM model, was implemented in the new release.

In QBlade v0.9 a new module for unsteady aerodynamic simulations of wind turbines, based on a nonlinear lifting line free vortex wake algorithm (LLFW) was implemented [1]. As the implemented lifting line method is very general it can be used to simulate both horizontal and vertical axis wind turbines. Especially for the simulation of vertical axis wind turbines the lifting line method presents a large improvement over the double multiple stream-tube method, which was implemented in v0.6. The lifting line does not have any convergence problems and the accuracy is drastically improved. The lifting line method also outperforms the unsteady blade element momentum method, used for horizontal axis wind turbines, which was implemented through the coupling to FAST in v0.8. However, in contrast to FAST simulations the newly implemented lifting line algorithm is currently not coupled to a structural model and thus can only perform aerodynamic simulations with the assumption of rigid rotors. Setting up a lifting line simulation through the user interface in QBlade is a very quick task as reasonable values are already pre-selected for all simulation parameters. Nevertheless, if one wants to optimize the lifting line performance and accuracy to suit the current simulation, a good understanding of all governing parameters and their relevance for the investigated case is needed. The first part of this document gives an overview of the user interface of the Lifting Line module and its functionality, in the second part a short explanation of the underlying theory and the influence of all simulation parameters on the simulation is given.

What's new in this document

- Added section about the unsteady aerodynamics model
- Added information about new blade varying Re number definition
- Added section about new polar decomposition features
- Added section the new PNoise module
- Added section about VAWT wake reduction
- Added new numerical integration schemes (PC2B, Element Centered)
- Added section about velocity field export functionality
- Added section about start-up simulations
- Fixed errors and improved clarity in some sections

What's new in QBlade v0.95

- Added unsteady aerodynamics model with Beddoes-Leishman type dynamic stall model
- All project files have been updated to include decomposed polars for the UA model
- Added multi (Reynolds number) polar blade definition in the blade design module
- Added polar preprocessor for dynamic stall simulations
- Added PNoise module for airfoil self-noise evaluation
- Added floating platform turbine simulations through *.sim files
- Binary windfield files (*.bts) can now be imported through the "Windfield" menu
- Added turbine startup simulations, including adaptive time stepping
- Added ParaView export functionality for velocity fields
- Added vortex centered & higher order velocity integration scheme for improved stability
- Added graph menu to choose number and arrangement of graphs
- Added VAWT blade design: "pitch axis" for blade sections has been added as a design parameter
- Added HAWT blade design: Added Z-Offset parameter to advanced design for pre-bend blades
- Added 3D correction for the Himmelskamp effect to LLFWH HAWT simulations
- Added position of bound vortex and evaluation point for AoA's to LLFWH simulation parameters
- Added estimates for vortex induced velocities to LLFWH simulation dialog
- Added .stl and .txt geometry export functionality for VAWT blades
- Added dialog to change graph arrangements
- Several improvements of overall stability, GUI and numerous bug fixes

Theory

Nonlinear Lifting Line Free Vortex Wake method

The lifting line simulation method belongs to the family of the so called "vortex methods". In Terms of computational cost, complexity and modeling of the physics the vortex methods are situated in between blade element momentum (BEM) methods and computational fluid dynamics (CFD). Using vortex methods, the flow field is modeled as inviscid, incompressible and irrotational; vortex elements are introduced in the form of straight or curved line segments to model both the rotor blades or wings and the wake. The vortex lattice method models rotor blades or wings with a lattice of horseshoe vortices, which are located on the blades mid surface. The panel method models the blades upper and lower surfaces with a mesh of vortex rings. The lifting line method models the blades with a single line of vortices, located on the quarter chord points of the blade. With respect to the wake modeling two main classes of vortex methods can be distinguished. In prescribed wake methods the wake elements are convected on a prescribed path. Free wake methods update the position of the wake end nodes based on the local velocity, which is a combination of inflow velocity and induced velocity from all wake elements in the domain. The advantage of the free wake convection is a much higher accuracy as the wake shape is forming based on the underlying physical principles; however this comes at the price of a significantly higher computational cost compared to prescribed wake methods. The code implemented in QBlade is a "nonlinear lifting line – free vortex wake" algorithm. Nonlinear in this case implies that the circulation, computed for the bound vortices on the lifting line, is obtained from nonlinear lift and drag polars. One advantage of implementing the lifting line method in QBlade is that all existing blade designs that were used in blade element momentum simulations can be used in lifting line simulations without any conversion being necessary.

One large advantage of the vortex methods, compared to BEM methods, is that due to the sound modeling of the macroscopic flow physics, only very few empirical models related to microscopic fluid dynamics, where boundary layer effects play an important role, such as dynamic stall or stall delay need to be added. In many studies [2] [3], vortex methods are identified as suitable to replace BEM codes in the near future to achieve a higher accuracy in turbine design and certification applications. Additionally the vortex methods not only provide simulation results concerning the rotor performance and the blade loads, but also result in the unsteady velocity field around the rotor and wake for every time step.

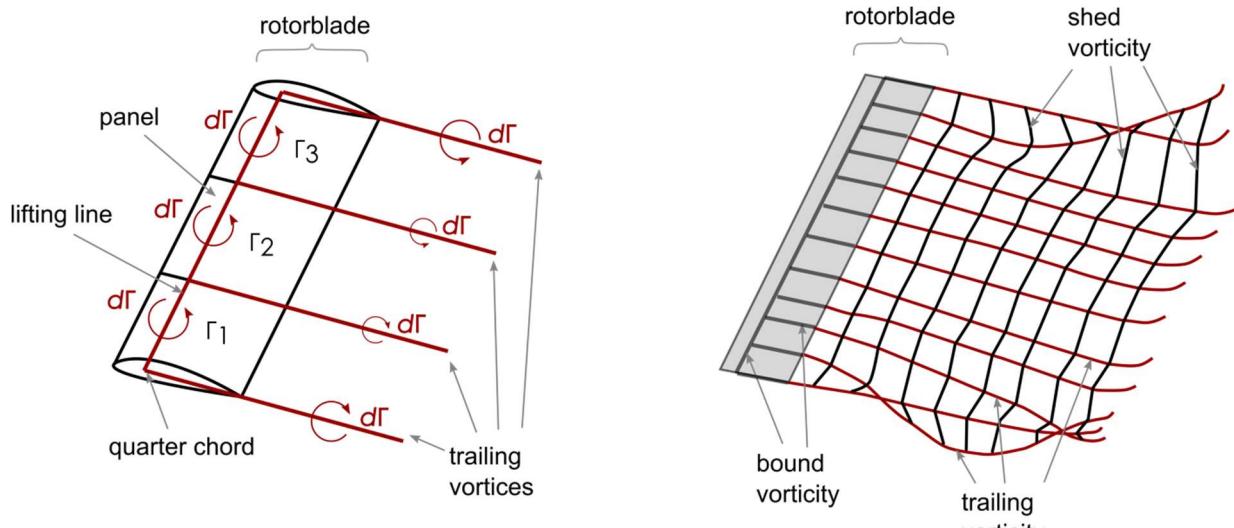


Figure 1. Illustration of blade and wake modeling with the LLFWV algorithm

The LLFWV implementation in QBlade

The lifting line free vortex wake algorithm implemented in QBlade roughly follows the work of van Garrel [4], but includes a range of extra functionality and improvements such as the vortex core modeling, time stepping, iteration loop and provisions for computational efficiency; multi-threading and wake connectivity tracking. Details of the implementation are not given in this document, however in the following section all simulation parameters are explained and short introductions or references to the theory are given if necessary. For completeness an overview over the simulation loop for one time step is given in Figure 2. A large benefit of the implementation in QBlade is the wide range of functionality and the ease with which unsteady LLFWV simulations can be set up and evaluated. A brief overview of the functionality is listed in the following:

- Unsteady aerodynamic simulation of VAWT and HAWT rotors
- Upwind & downwind rotor configurations
- Blade discretization: custom, linear, sinusoidal
- Simulations of skewed inflow, yawed turbine, teeter, pre-cone and blade pitch angles
- Inflow conditions: stationary inflow, turbulent wind field, earth boundary layer, AeroDyn hub height wind file, custom transient QBlade simulation files
- Optional models: ground effects, tower shadow
- Custom wake discretization, vortex core modeling and bound vortex positions
- Velocity integration: Euler forward, predictor corrector and predictor corrector 2 backwards
- Replay of conducted simulations and evaluation of 3D velocity field for every time step
- ParaView export functionality for velocity fields

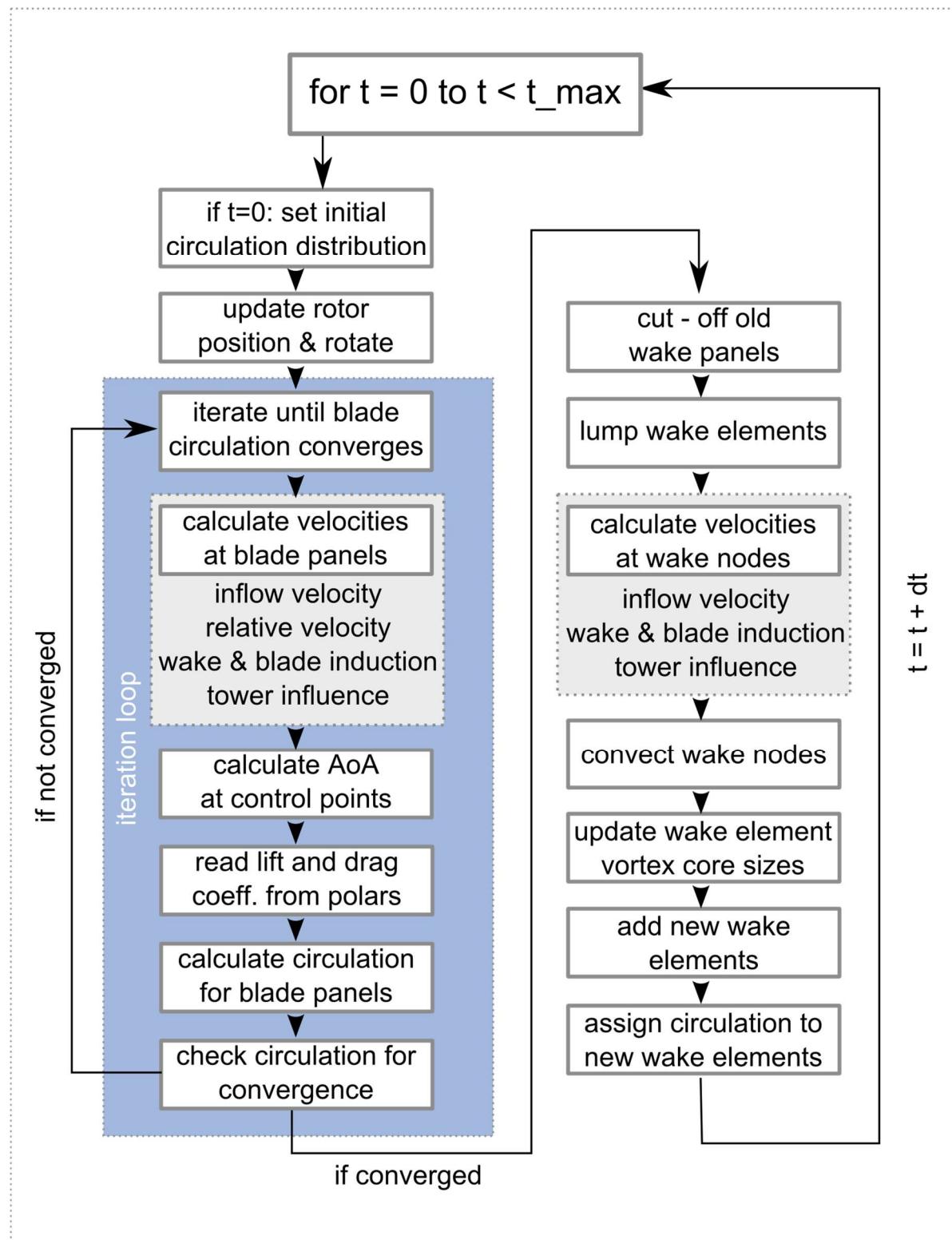


Figure 2. Flowchart of one time step loop of the implemented lifting line algorithm

Unsteady Aerodynamics Model

With the latest release of QBlade v0.95 an unsteady aerodynamics model for 2D airfoil behavior has been implemented and coupled to the LLFW module. The UA model consists of mainly two parts – an attached or potential flow, model as proposed by Gaunaa [5] and the classical Beddoes-Leishman dynamic stall model [6] with a custom formulation for vortex lift.

Please Note:

To use the unsteady aerodynamics model in LLFW simulations the static polars need to be decomposed into their attached and separated lift components. To facilitate this polar decomposition the needed functionality has been implemented into the “360° Polar Extrapolation” module. Existing polars from previous versions of QBlade need to be decomposed in this module in order to use them with the new unsteady aerodynamics model.

Attached (potential) flow model

The potential flow model accounts for non-circulatory (added mass) effects, and circulatory lift which includes wake memory effects that play a role long before the onset of stall. The added mass term models the forces due to the reaction of the fluid to an airfoil motion:

$$C_l^{nc} = \pi \frac{b_{hc}}{U_0} \dot{\alpha}^{str}. \quad (1)$$

The quasi steady lift component is the steady lift that is generated by the airfoil at the current angle of attack computed from the relative airfoil motion but without the influence of shed wake vorticity:

$$C_l^{qs} = C_l^{att}(\alpha_{qs}). \quad (2)$$

Wake memory effects account for the influence of span wise (or shed) vorticity in the wake on the quasi-steady angle of attack. In contrast to the classical formulation for BEM codes, in which the wake memory effect is modeled with an effective angle of attack, computed via step responses, that are described by exponential indicial response functions, the effective angle of attack is directly obtained from the free vortex wake formulation of QBlade.

$$C_l^{circ} = C_l^{att}(\alpha_{eff}). \quad (3)$$

However, as the quasi steady angle of attack, that does not include the effect of shed vorticity, is needed for a later evaluation of the induced drag contribution, the sole influence of shed vorticity on the angles of attack (α_{shed}) is stored separately. α_{shed} is computed via the induction of the total shed vorticity in the vicinity of the blade, up to 8 chord lengths away from the trailing edge. The limitation, with respect to the trailing edge distance, is necessary to exclude the significant influence of shed vorticity (from all previous time steps) on the global flow field from this evaluation that takes place in all VAWT simulations. α_{shed} is then used to synthesize the quasi steady angle of attack from the effective angle of attack.

$$\alpha_{qs} = \alpha_{eff} - \alpha_{shed}. \quad (4)$$

This extra treatment is necessary because the common unsteady aerodynamics models are formulated for BEM codes and are using indicial functions which are replaced by the free vortex wake model in this formulation.

Dynamic stall model

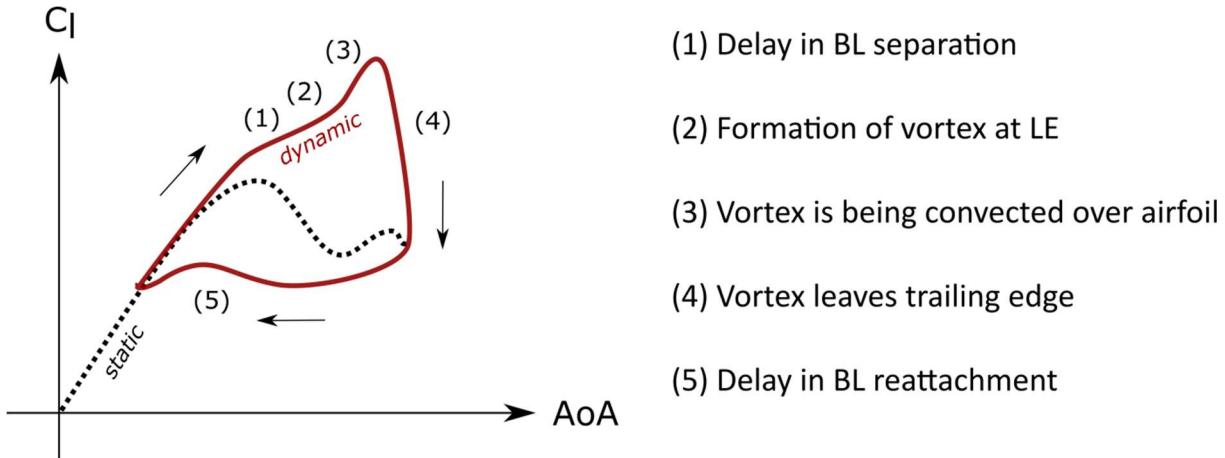


Figure 3. The dynamic stall hysteresis loop

A Beddoes-Leishman dynamic stall model has been implemented in QBlade. The dynamic stall effect is modeled from three contributions. The first contribution is the lagged potential lift (leading edge pressure time lag):

$$\dot{C}_l^{lag} = -\frac{U_0}{b_{hc}} \frac{1}{\tau_p} C_l^{lag} + \frac{U_0}{b_{hc}} \frac{1}{\tau_p} C_l^{pot}. \quad (5)$$

The second contribution models the dynamic circulatory lift and includes the separation point time lag:

$$C_l^{circ,dyn} = C_l^{att}(\alpha_{eff}) f^{dyn} + C_l^{sep}(\alpha_{eff})(1 - f^{dyn}), \quad (6)$$

$$\dot{f}^{dyn} = -\frac{U_0}{b_{hc}} \frac{1}{\tau_f} f^{dyn} + \frac{U_0}{b_{hc}} \frac{1}{\tau_f} f^{st}(\alpha^*), \quad (7)$$

$$\alpha^* = \frac{C_l^{lag}}{\partial C_l / \partial \alpha} + \alpha_0. \quad (8)$$

The vortex lift contribution models the lift overshoot from the leading edge vortex that is formed at the leading edge and convected over the airfoil:

$$\frac{U_0}{b_{hc}} \frac{1}{\tau_v} C_{l,vort} + C_{l,vort} = \dot{C}_v, \quad (9)$$

$$C_v = C_l^{circ,dyn} \left(1 - \frac{(1 + \sqrt{f^{dyn}})^2}{4} \right). \quad (10)$$

The total lift, including attached and separated flow contributions then equals:

$$C_l^{dyn} = C_l^{circ,dyn} + C_l^{nc} + C_{l,vort}. \quad (11)$$

The dynamic drag is evaluated from three contributions. The steady drag at the effective angle of attack:

$$C_d^{eff} = C_d(\alpha_{eff}), \quad (12)$$

the drag induced from shed wake vorticity, using the quasi steady angle of attack that was evaluated in Equation 4:

$$C_{d,ind} = C_l^{circ,dyn} \cdot (\alpha_{qs} - \alpha_{eff}), \quad (13)$$

and the drag change induced through separation delay:

$$C_{d,ind}^f = (C_d^{eff} - C_d(\alpha_0)) \left[\frac{(1-\sqrt{f^{dyn}})^2}{4} - \frac{(1+\sqrt{f^{st}})^2}{4} \right]. \quad (14)$$

The total drag is then computed as the sum of these contributions:

$$C_d = C_d^{eff} + C_{d,ind} + C_{d,ind}^f. \quad (15)$$

More details about the implementation and validation of the unsteady aerodynamics model can be found in [7]. Two exemplary validation graphs from this publication are shown in Figure 4.

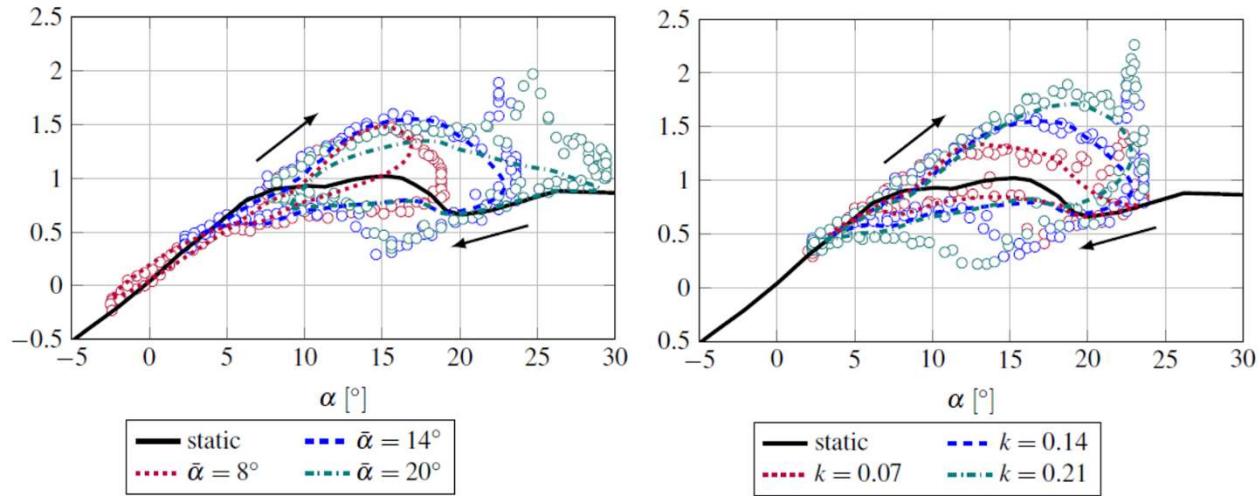


Figure 4. Validation of the UAM with OSU test data (circles) [8]; left: varying mean AoA; right: varying dimensionless frequency

Variable Reynolds number blade definitions

In previous versions of QBlade only one polar could be assigned to each blade section. This leads to inaccuracies - especially during unsteady HAWT or VAWT simulations where the Reynolds numbers at the blade sections can vary heavily. This error even gets pronounced when modeling the startup behavior of VAWT or HAWT where Reynolds numbers are initially very small and then constantly increase as the rotor speeds up.

To more accurately model the behavior of rotors in these situations the blades can be defined using any number of polars at different Reynolds numbers that are associated with the airfoil at the respective blade station.

Moreover, a new unified function has been introduced in QBlade that returns the sectional parameters of a rotor blade. These parameters include lift, drag, attached lift, separated lift, separation function, current Reynolds number, chord, max and min lift and more. This new function is used throughout the code whenever aerodynamic properties are needed for BEM, DMS or LLFWVW simulations. This blade parameter function also interpolates polar data in between different airfoil sections and Reynolds

numbers – this means that the new Dynamic stall model can be used in conjunction with the new variable Reynolds number blade definitions.

The section wise blade parameters are evaluated, and interpolated as a function of radius, AoA and Reynolds number:

$$C_l; C_d; f; C_{l,att}; C_{l,sep}; c; \text{etc} = f(\text{blade}, r, \text{AoA}, \text{Re}). \quad (16)$$

Airfoil Self-Noise Evaluation

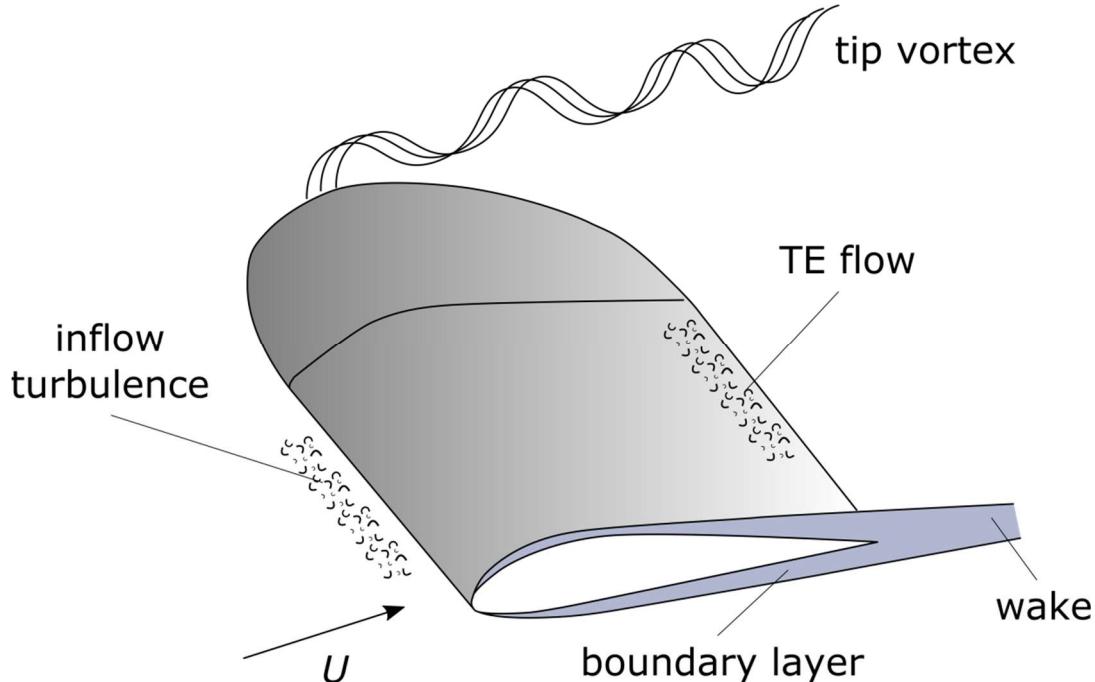


Figure 5. Sources of airfoil self-noise. Image created after [9]

The PNoise module was integrated with the new release of QBlade v0.95. PNoise models the trailing edge (TE) noise, generated by arbitrary airfoil shapes. The TE noise is the airfoil self-noise source that is often considered as the main source of high-frequency noise for modern large wind turbines (Figure 6) operating within a smooth inflow regime. PNoise is based in the semi empirical NASA BPM model [10] and uses XFOIL generated distributions of the turbulent boundary layer displacement thickness as a parameter for the turbulent length scale.

PNoise has been developed [11] in cooperation between the TU Berlin and the Poli-USP and is the first step towards the integration of a full quasi-3D WT noise evaluation module which is currently under development. More information about the PNoise module, setting up simulations, its validity range and validation are published in the document “*Verification and Validation of the Airfoil Trailing-Edge Noise Prediction Module (PNoise) inside QBlade V0.95*” [12], which is supplied with the QBlade software.

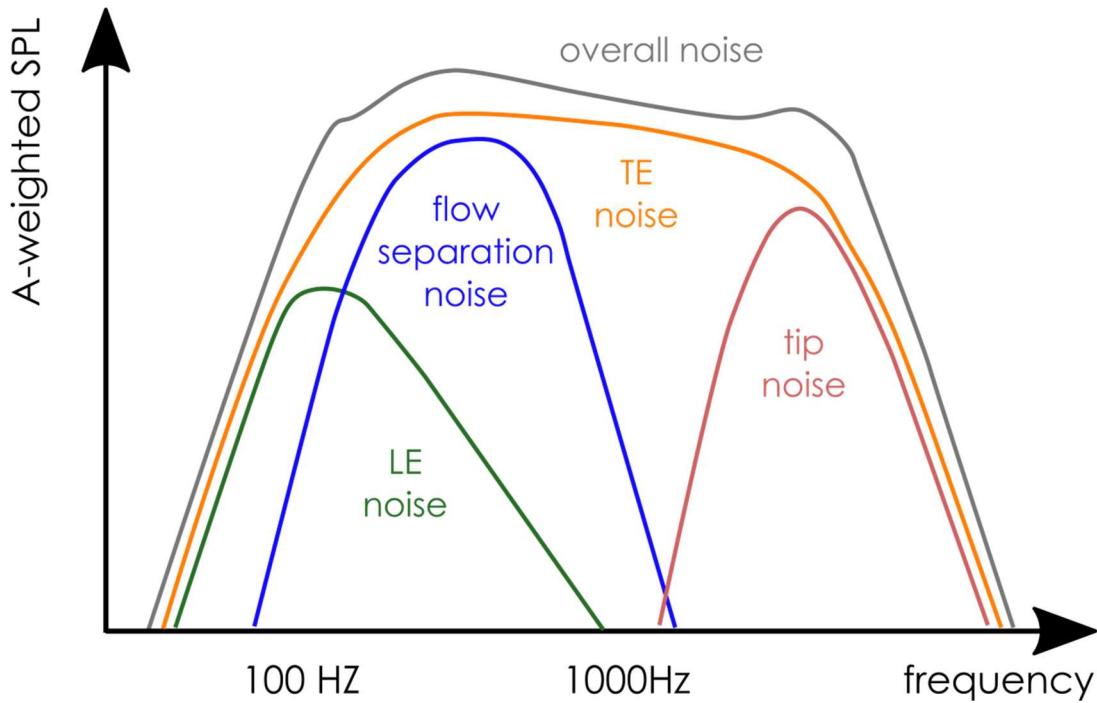


Figure 6. A-weighted 1/3 octave band spectrum from a modern, large WT. Contributions of the relevant flow induced airfoil self-noise mechanisms, reproduced after [13]

Graphical User Interface

Lifting Line Free Vortex Wake module GL view

Access to module via main toolbar

The Lifting Line Module is accessed through QBlades Main Toolbar (Figure 7). To open the Lifting Line Module for HAWT and/or VAWT turbines the HAWT/VAWT switch has to be set accordingly.



Figure 7. QBlade v0.9 Main Toolbar

Once inside the LLFWV Module the Control Bar (Figure 8) will appear below the main Toolbar. In the Control Bar the Graph/3D switch is used to switch between the 3D GL window, that shows a representation of the current Simulation and the graph window, which is used to show and export the various variables from the simulation results. The Simulation Object ComboBox shows the currently selected Lifting Line Simulation and is used to switch between the stored simulation objects. The time slider is used in multiple ways. If in the GL 3D window and the replay of the currently selected simulation was stored the time slider can be used to rewind or forward through the time steps of the simulation. If the replay button is clicked an animation of the simulation is shown by automatically cycling through the time steps. During replay a SpinBox will appear next to the replay button in which the time delay for the replay can be set in [s] to control the replay speed. If in the graph window the time slider is used to set

the time step for which the *Blade Graphs* are plotting the simulation results. A large dot, that marks the currently selected time, will also appear in the *Time Graphs*.



Figure 8. Control Bar of the Lifting Line Module

Simulation control

The Simulation Control part of the DockWindow (Figure 9) is used to setup, edit, rename, delete and start Lifting Line simulations. Lifting Line simulations can be created when a blade object is in the database. A press on the “New” button opens the simulation setup dialog (Figure 21, Figure 22), used to define new LLFWV simulations. When using the Edit/Copy button the simulation setup dialog of the currently selected LLFWV object is opened – this can be used to quickly check the simulation settings or to perform some quick changes of a few settings and create a new object under a different name.

Progress bar

When a simulation is started the progress bar shows its progress and the remaining time steps. A running simulation can be interrupted at any time by pressing the Stop button. The simulation can later be continued by pressing the Continue button. However, when a project is loaded from a project file unfinished simulations cannot be resumed but have to be restarted as not all the needed data is stored due to size constrains.

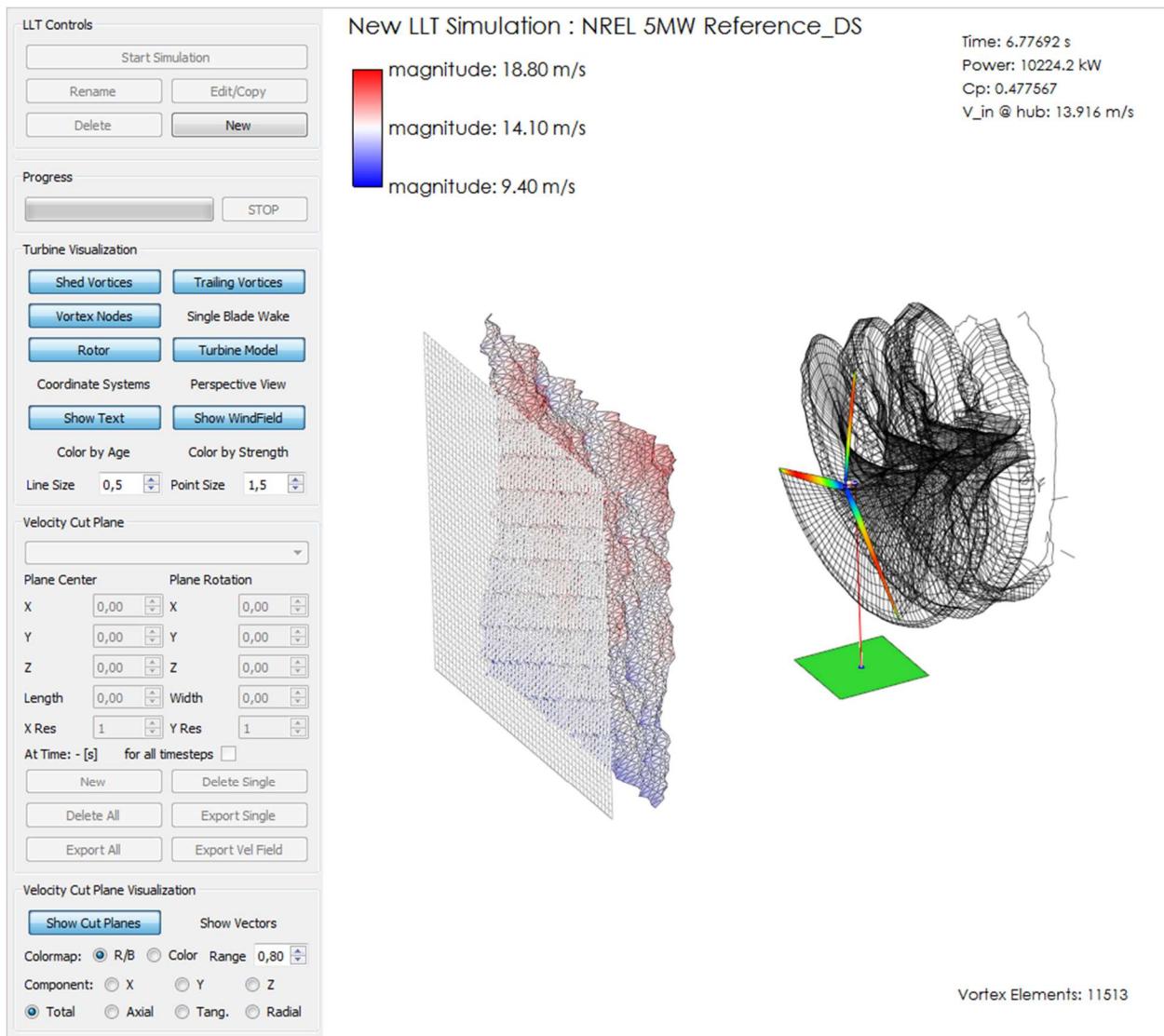


Figure 9. 3D View of the LLFVW Module

Velocity planes

A large benefit of the Lifting Line Theory over the Blade Element Momentum method is that as a result the velocity distribution in the flow field around the rotor can be obtained. The integrated “velocity cut planes” make it possible to access the information of the 3D velocity field in a general way. After a simulation is complete they can be used to visualize or export the velocity distribution. A cut plane is a 2D grid that can be oriented arbitrarily in the simulation domain, for which the velocity vectors are computed at the grid points. If a replay of the simulation has been stored, velocity fields can be reconstructed for any time step that is chosen with the *TimeSlider* (Figure 8). When the replay is not stored the cut planes can only be constructed for the last time step of the simulation. Cut planes are created by pressing the “New” button. The center of the cut plane is defined through the x-y-z coordinates under “Plane Center” in the cut plane dialog (Figure 10). The plane can be rotated around the x-y-z axis under “Plane Rotation”, angles are given in °. The rectangular grids edge lengths are given by the “Length” and “Width” parameters; “X Res” and “Y Res” define the grids resolution. The cut planes position and orientation are shown as a preview in the GL-window (Figure 9). If the “for all time steps”

checkbox is enabled, and a replay has been stored for the simulation, the cut plane is computed for all time steps. This allows creating animations where velocity distributions are included.

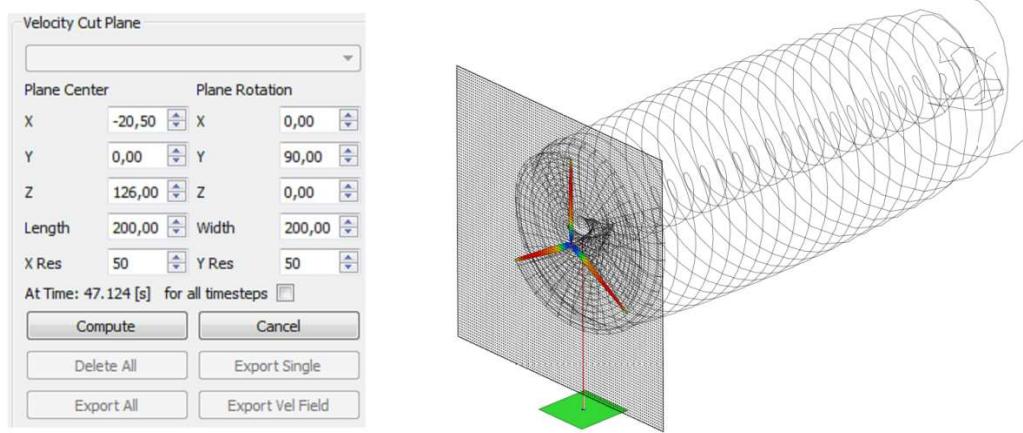


Figure 10. Cut plane dialog and newly created cut plane, upstream of the rotor parallel to the rotor plane

When the “Compute” button is pressed the velocities at the grid points are computed. Depending on the number of grid points and the number of wake elements the computation can require a significant amount of CPU time and can use up a large amount of memory, especially when the option “for all time steps” is enabled. When a cut plane has been computed it can be viewed in the GL-window. Its appearance can be changed in the “Cut Plane Visualization” dialog. More importantly, the cut plane data can be exported to a plain text file, which has the format shown in Table 1. The first three columns are the position vector component of the grid points in [m]. The last three columns are the vector components of the total velocity at this point in [m/s]. The export functionality allows to easily produce velocity data such as wake velocity profiles or axial velocity profiles through the rotor plane to compare simulation results to experimental data.

Table 1. QBlade velocity cut plane export format

Export File Created with QBlade v0.9 on 08.07.2015 at 16:09:36					
Position Vector			Velocity Vector		
X	Y	Z	X	Y	Z
-15	-150	276	12.0274	-0.0970743	0.0992186
-15	-146.97	276	12.0274	-0.0984592	0.102661
-15	-143.939	276	12.0274	-0.0998126	0.106215
-15	-140.909	276	12.0273	-0.101129	0.109884
-15	-137.879	276	12.0273	-0.102404	0.113669
-15	-134.848	276	12.0272	-0.103631	0.117571
-15	-131.818	276	12.0271	-0.104803	0.121592
-15	-128.788	276	12.027	-0.105915	0.125731
-15	-125.758	276	12.0269	-0.106959	0.129991
-15	-122.727	276	12.0267	-0.107928	0.134369
-15	-119.697	276	12.0266	-0.108814	0.138867
-15	-116.667	276	12.0264	-0.109609	0.143482
-15	-113.636	276	12.0262	-0.110306	0.148213

Turbine visualization

The buttons under “Turbine Visualization” activate/deactivate the rendering of the respective features, such as vortex lines or nodes or the turbine or rotor geometry. If “Color by Age” is selected the vortex nodes and lines are colored, depending on the time they are released at the trailing edge, by cycling through the color spectrum. If “Color by Strength” is selected the vortex node and line elements are colored according to their vortex strength.

Velocity cut plane visualization and export

The velocity cut plane visualization dialog is used to change the appearance and the velocity component that is visualized in the cut planes. The velocity vectors at the grid points can be plotted and one of two color maps can be selected that maps the velocity magnitude in the plane. Additionally the velocity component that is to be plotted can be selected in the dialog. The options “Total”, “X”, “Y” and “Z” plot the magnitude or respective component of the velocity in the global coordinate system, while “Axial”, “Tang.” and “Radial” plot the respective component in the rotor coordinate system (see examples in Figure 11, Figure 13 and Figure 12). The SpinBox “range” is used to adjust the colormap range (values outside the range are clipped). This is a practical tool to highlight flow features and was used for the Y- and Z-component in Figure 12.

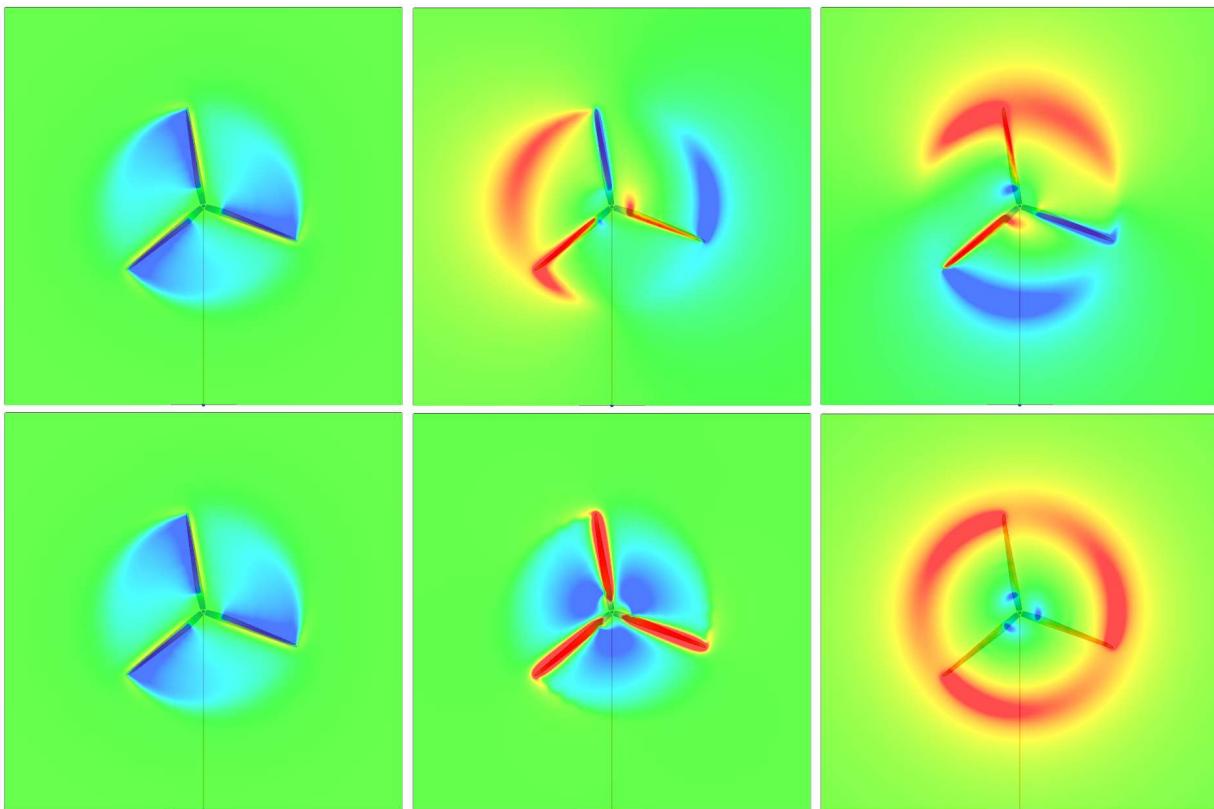


Figure 11. Velocity distribution in the rotor plane; Top row: Magnitude of X, Y and Z component; Bottom row: Magnitude of axial, tangential and radial component

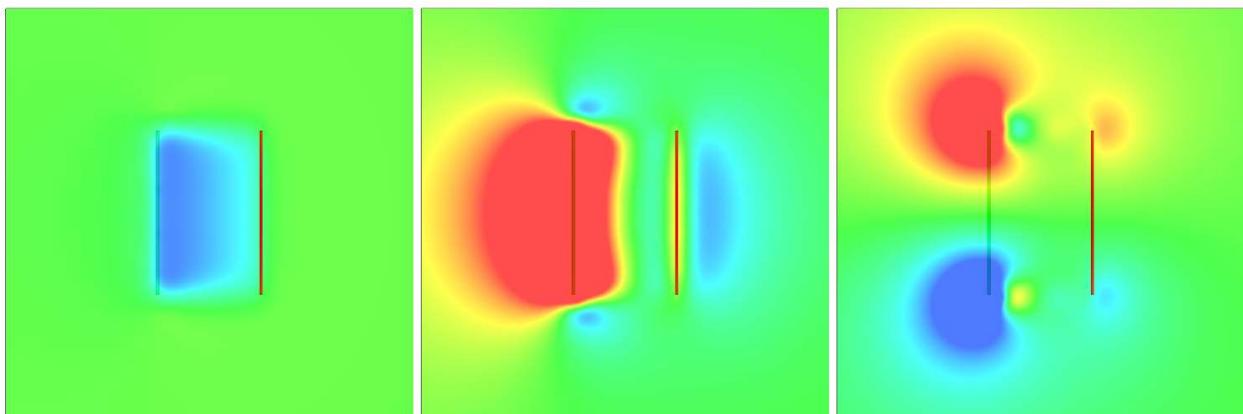


Figure 12. Exemplary velocity distribution of a VAWT H-rotor in the rotor mid plane, from left to right: X, Y and Z component

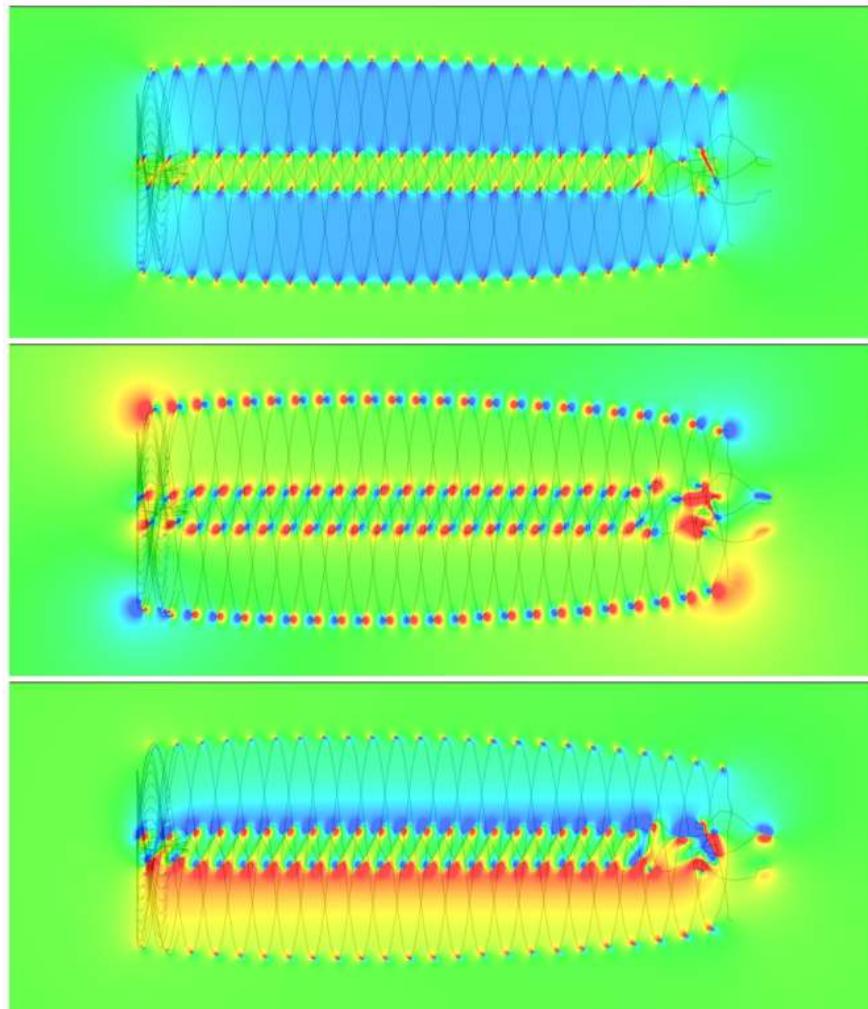


Figure 13. Exemplary velocity distribution in horizontal mid-plane; from top to bottom: X, Y, and Z component

Volumetric velocity distribution (ParaView) export

In the version 0.95 of QBlade, export functionality for the volumetric velocity distribution within a cuboid in the simulation domain has been added. Similar to the velocity cut plane a velocity volume can be computed after a simulation has been completed. If the replay of a simulation has been stored volumetric velocity distributions can be reconstructed and exported for every time step. In contrast to the velocity cut planes, the volumetric velocity distribution cannot be visualized in QBlade. However, it can be exported either as a text file or in the VTK XML syntax format [14], which can be interpreted by the open source visualization software ParaView [15].

The volumetric velocity distribution can be computed by clicking the “Export Vel Field” button on the LLFWV module’s dock widget (Figure 14, left). In the export dialog (Figure 14, right) the center of the cuboid, its dimensions and the resolution in all three directions need to be specified. If the button “All Timesteps” is clicked a velocity plane is exported for every time step of the simulation to facilitate the creation of animations. Note that the total number N of evaluation points is

$$N = X_{res} \cdot Y_{res} \cdot Z_{res} \cdot n_{timesteps} . \quad (17)$$

Thus a high resolution can quickly lead to excessive evaluation times and large file sizes. Once the computation has been completed the velocity distribution can be exported as either a *.txt; *.csv or *.vtu file. The *.txt and *.csv formats are similar to the ones shown in Table 1.

If the velocity is exported in the VTK format two VTK unstructured grid (*.vtu) files, one containing the grid, connectivity and velocity information and the other the blade geometry, are created per time step. Additionally, a collection file (*.pvda) is created to assign the grid files with tags for the time step and the part number.

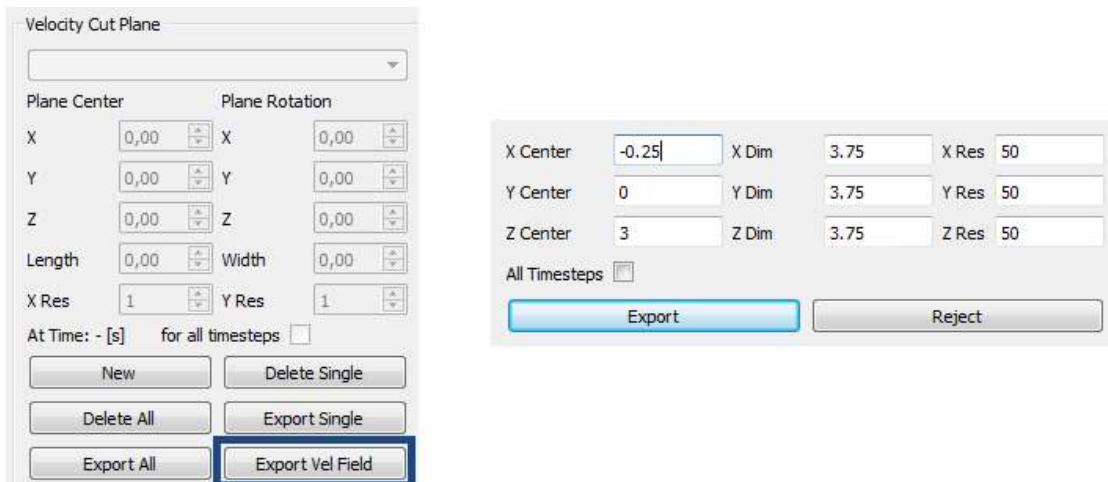


Figure 14. The “Export Vel Field” button and the export dialog

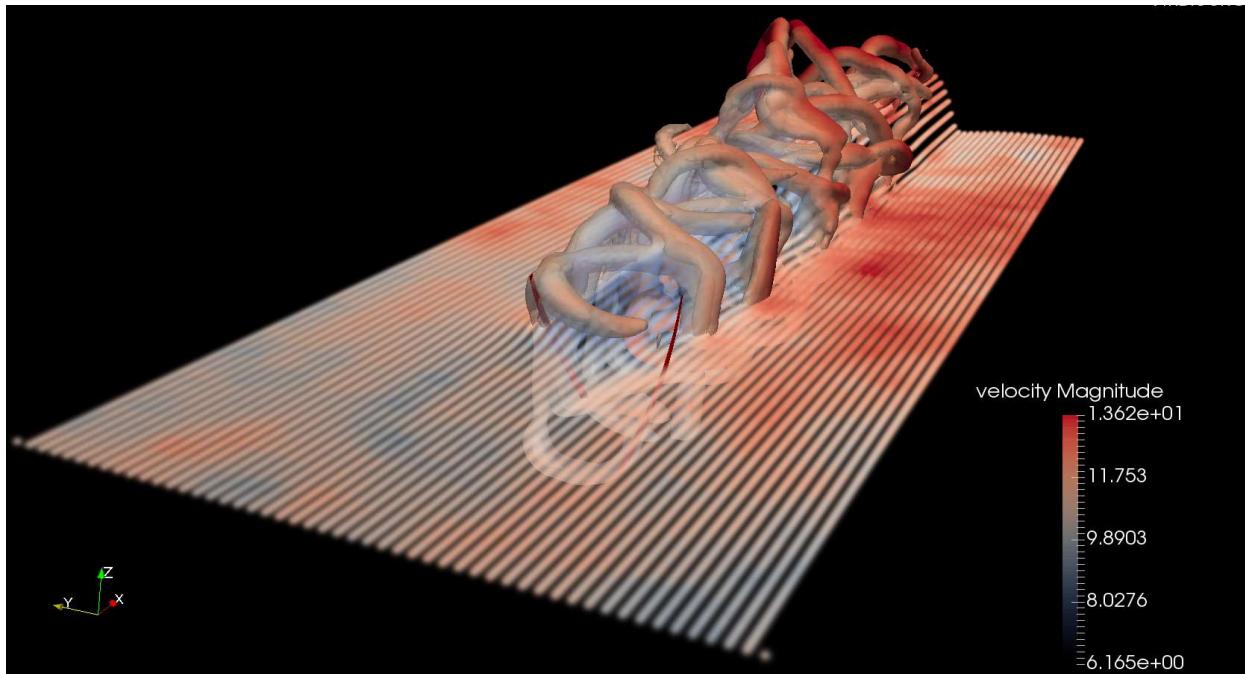


Figure 15. ParaView visualization of streamlines and the vorticity isosurface of a VAWT wake, colored by the velocity magnitude; helix blades of the rotor shown in red

Lifting Line Free Vortex Wake module graph (2D) view

In the graph view the simulation results are plotted. When a simulation is running the graphs are automatically updated for every time step, allowing the user to monitor the performance of the rotor in live. When in the graph window while a simulation is running the simulation runtime is slightly faster, as the GL visualization, that requires a small amount of CPU time during each time step, is deactivated.

The dynamic graphs in QBlade are fully controllable with the mouse. Zooming in and out of a graph is realized with the mouse wheel. Pressing the y or x keys during zooming allows only zooming the respective axis. The graph context menu (Figure 16) can be opened by pressing the right mouse button while on a graph. In the context menu the automatic resetting of the graph scales can be activated or deactivated and the graph type can be set (Time Graph or Blade Graph). A double click on a graph opens the “Graph Settings” menu (Figure 16). In the options menu the variables for the x- and y-axis are set and the graph style can be changed.

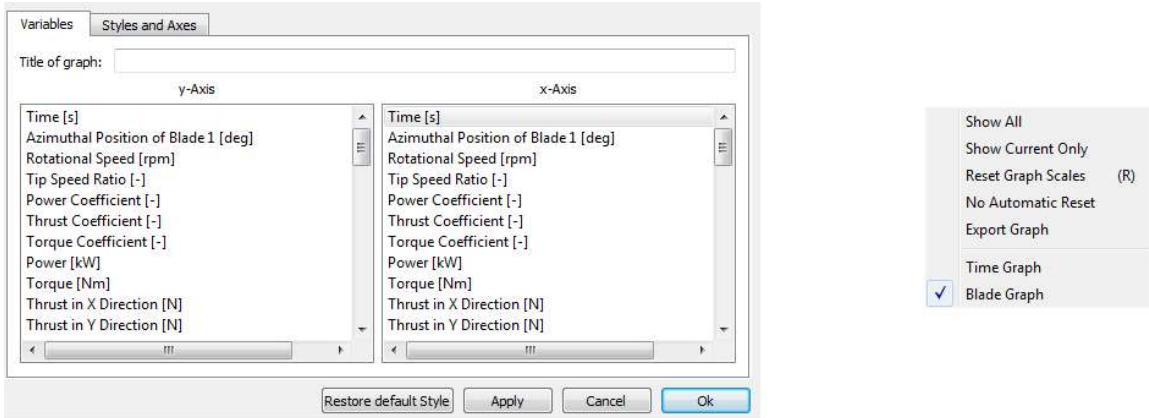


Figure 16. Graph Options(left) and Graph context menu (right)

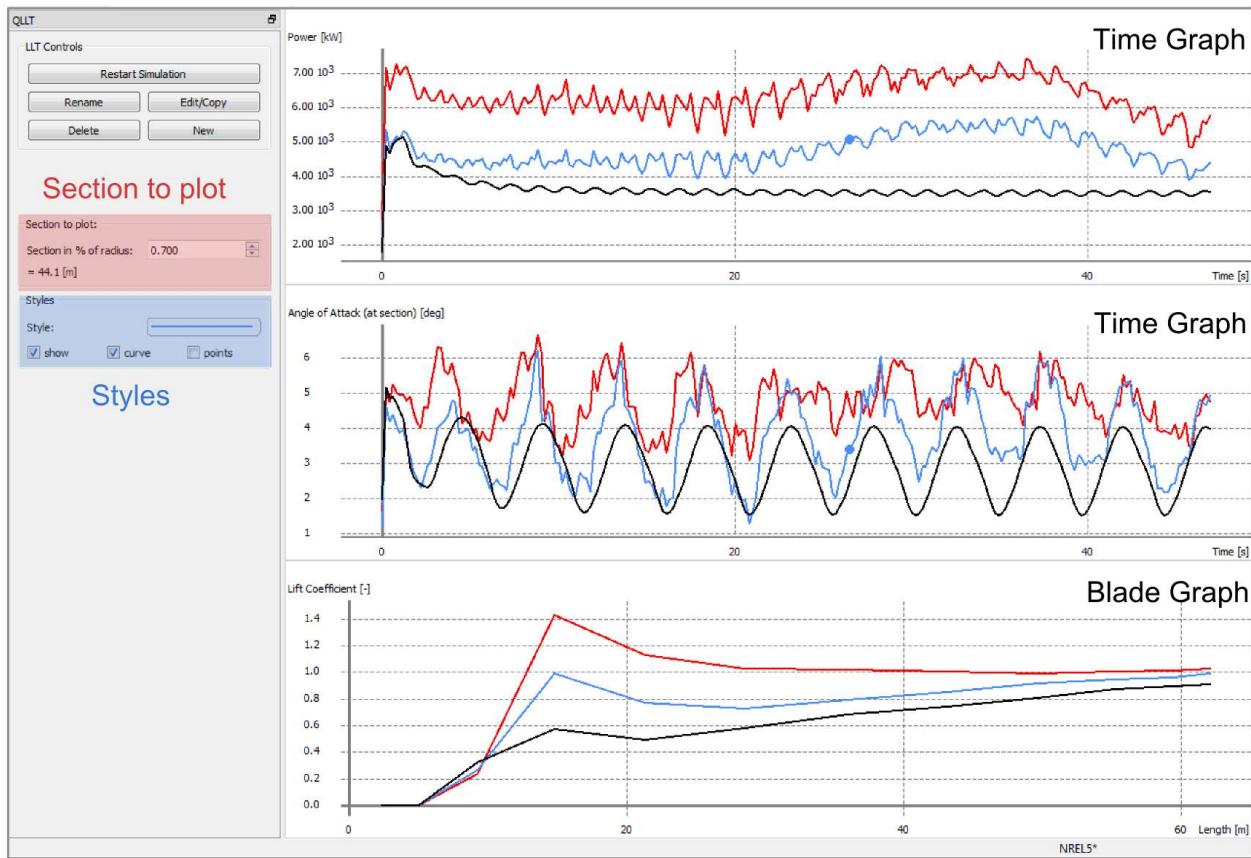


Figure 17. Graph View of the LLFWV Module

Section to plot

The SpinBox in “section to plot” is used to select a radial position of the rotor blade for which the local blade variables are plotted in the *time graphs*. The radial position is defined in % of blade radius. This also enables the comparison of local blade variables between rotors of different size, as the radial position is made dimensionless.

Styles

The styles dialog enables to set the appearance of the graph curves. Each simulation object can have its own style. If *points* is selected the computed points are highlighted. The *curve* switch enables the plotting of the point’s connections and *show* shows or hides the curve. All curves can be hidden or shown using “Show All/Hide All” in the graph context menu (Figure 16).

Time graphs

The time graphs show the global variables for rotor performance over the simulation time, such as the power- or thrust coefficient. However it is also possible to plot local blade variables in the time graphs. The blade variables are marked with “(at station)”. The blade variables are plotted for the radial position that is selected in the “section to plot” SpinBox (Figure 17).

Blade graphs

The blade graphs show the distribution of local blade variables over the rotor blade (if length/height is chosen for the x-axis), such as the Lift coefficient or the local inflow velocity. The variables are always plotted for blade 1 only. The variables are plotted for the time step that is currently selected at the TimeSlider (Figure 8).

Preprocessor for lift polar decomposition

To use the unsteady aerodynamics model in a LLFW simulation the polars associated to the simulated rotor need to be decomposed into their attached and separated lift counterparts (Figure 18).

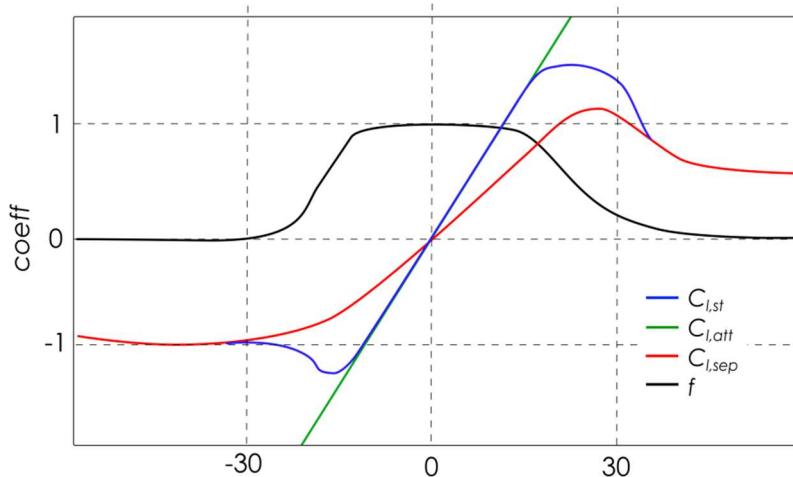


Figure 18. Visualization of the different parts of the polar decomposition

The relationship between the lift components and the lift from the static polar is:

$$Cl_{st} = f \cdot Cl_{att} + (1 - f) \cdot Cl_{sep}. \quad (18)$$

The decomposition functionality is implemented in the “360° Polar Extrapolation Module” and is automatically performed whenever a polar is extrapolated. In case of using imported polars, or polars from a project file that was created with a QBlade version below v0.95 the “Decompose” button performs the lift and drag decomposition without changing the extrapolation. To view the distribution of attached and separated lift and the f function, these parameters have been added to the 360Polar class and can be shown in graphs through the “Graph Settings” menu (Figure 19).

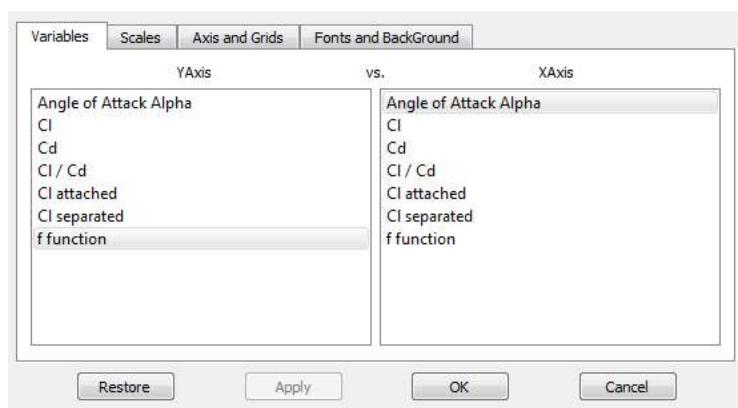


Figure 19. The “Graph Settings” menu

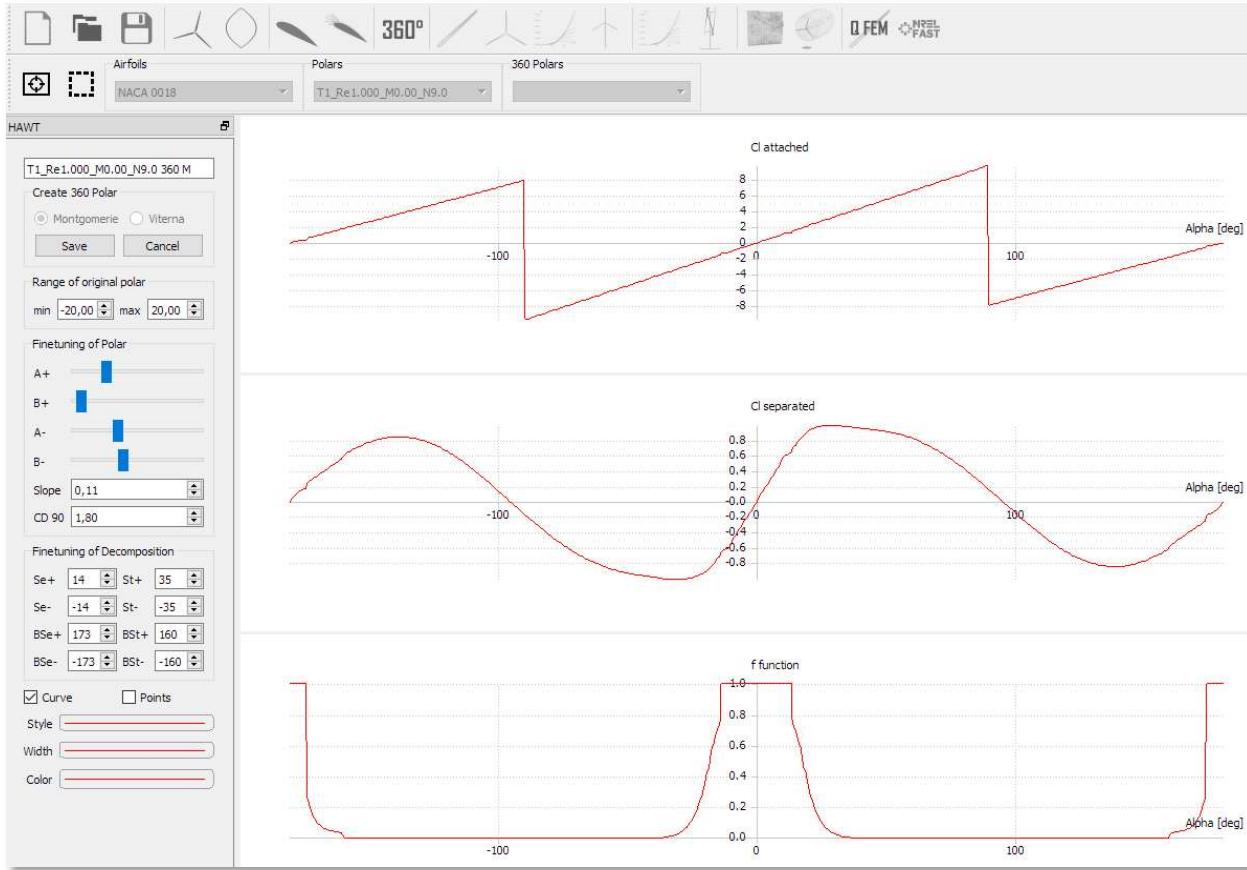


Figure 20. The “360 Polar Extrapolation Module” GUI in QBlade showing attached and separated lift and the separation function f

The algorithm for the decomposition is based on the ATEFlap report published at DTU [5]. The lift decomposition requires some user input to ensure smooth results due to the high variability of polar data. The user has to specify the angle of attack at the onset of separation (Se) and in deep stall (St) for positive and negative angles of attack in forward and backward “flight”. For a good performance and convergence of the dynamic stall model it is highly important that the decomposed curves are smooth (as in Figure 20) and don’t show any kinks or discontinuities. Smooth curves can be obtained by carefully choosing the correct angles of attack for the parameters in Table 2 and observing their influence on the attached lift, separated lift and the f function in the graph view (Figure 20).

Table 2. Definition of polar decomposition parameters

Parameter	Definition	Airfoil Orientation
Se+	Onset of separation for positive AoA	forward
St+	Deep stall point for positive AoA	forward
Se-	Onset of separation for negative AoA	forward
St-	Deep stall point for negative AoA	forward
BSe+	Onset of separation for positive AoA	backward
BSt+	Deep stall point for positive AoA	backward
BSe-	Onset of separation for negative AoA	backward
BSt-	Deep stall point for negative AoA	backward

Guidelines for setting up a simulation

A LLFWV simulation can be defined through a simple setup dialog (Figure 21; Figure 22). In the following section all simulation parameters and their role in a simulation are described. The section is ordered into groups, in the same order as they appear in the setup dialog.

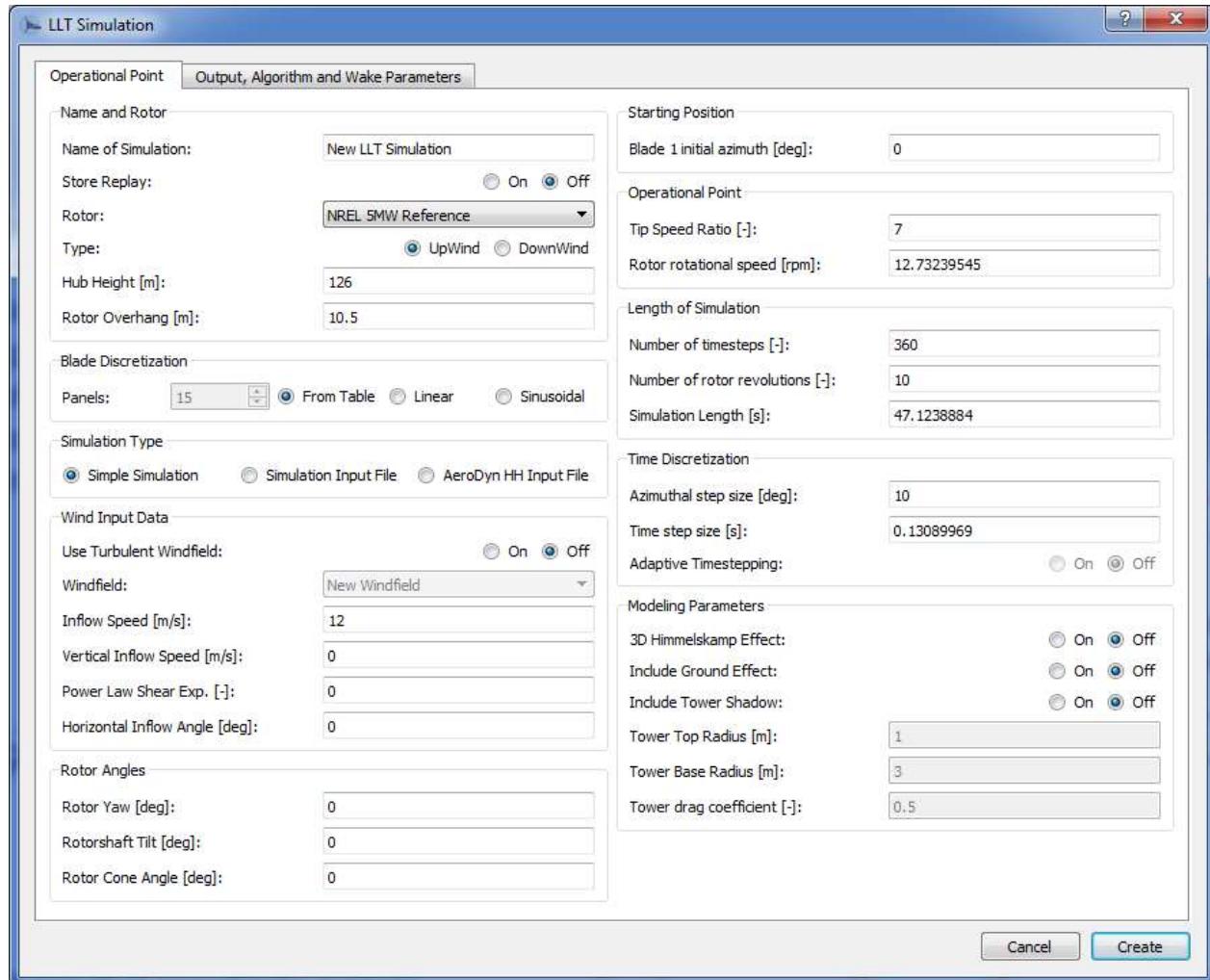


Figure 21. 1st tab of LLFWV simulation setup dialog

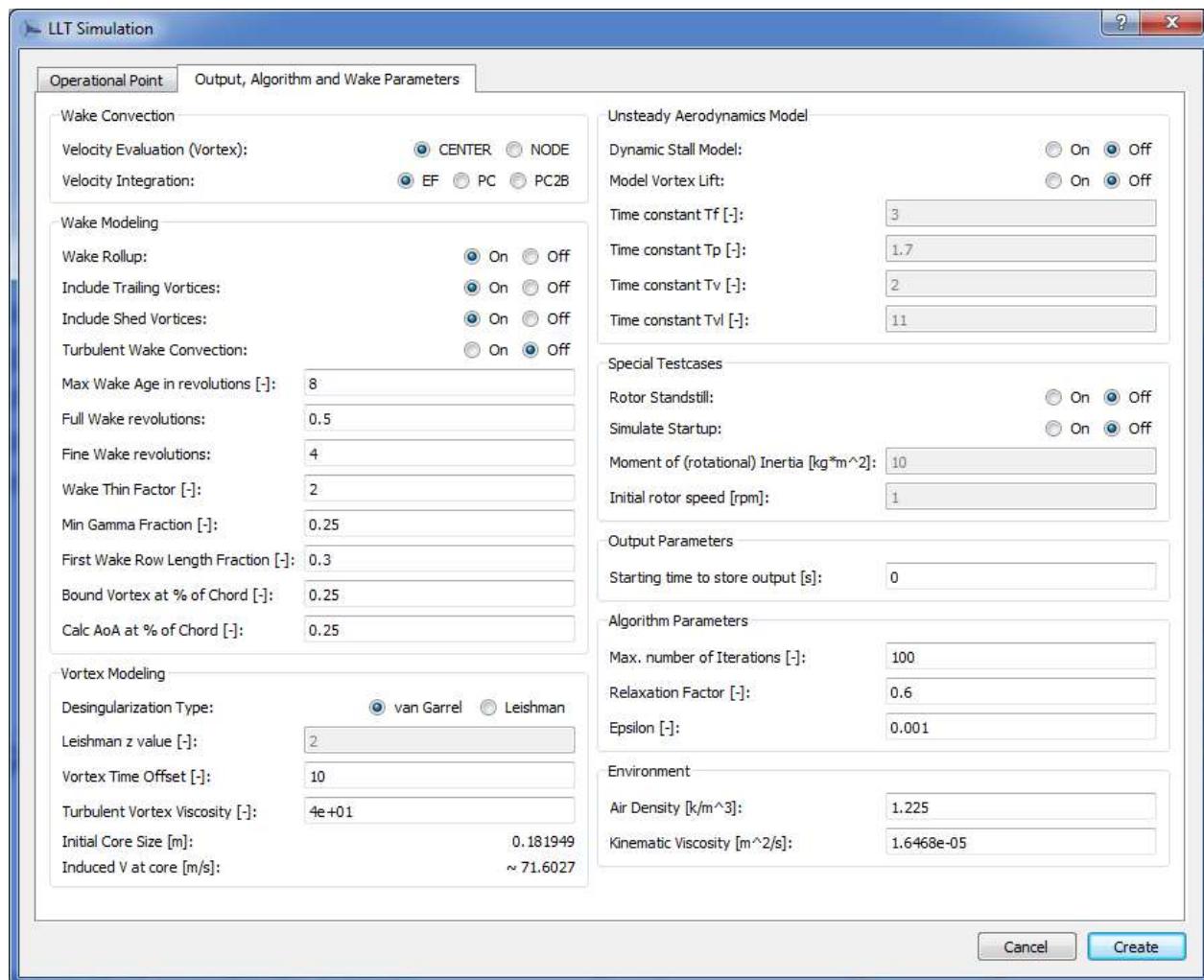


Figure 22. 2nd tab of LLFVW simulation setup dialog

Name and rotor

Name of simulation:

To assign a unique name to the simulation under which it will be stored in the database.

Store replay:

If the button is set to “On” all wake node positions are stored for every time step of the simulation. If the replay is stored, an animation of the simulation can be viewed after the simulation is complete. Storing of the wake nodes also enables to recreate the velocity field at any time step during post-processing. The store replay option can require, depending on the wake discretization and the number of time steps, a large amount of memory space. If you are using the 32bit version of QBlade, a crash will occur if more than 2GB of memory are requested. It is recommended to use the 64Bit version with at least 4GB of RAM if replays are stored.

Rotor:

The rotor for which the simulation is defined can be selected from this drop-down menu.

Type:

The selection defines the turbine as upwind or downwind. This option is only available for HAWT simulations.

Hub height / Rotor overhang / Height of rotor centre

These values define the wind turbine geometry and are important when calculating the position of the rotor with respect to the wind field (when turbulent wind field or earth boundary layer is used); tower (when tower model is turned on); ground (when ground effects are turned on).

Blade discretization

The user can choose from linear or sinusoidal blade discretization and define the total number of blade panels. If “From Table” is selected the discretization is directly obtained from the blade table that was created in the “Blade Design Module”.

Simulation type

Simple simulation:

A simple simulation does not require an input file. Fixed values for the rotational speed of the rotor, the wind inflow angle and rotor yaw have to be defined once and are then used throughout the whole simulation.

Simulation input file:

A simulation input file is used to define a transient simulation. Values for rotor speed, horizontal and vertical wind speed, wind direction and yaw and pitch angle of each individual rotor blade can be defined for different points in time in the input file (Table 3). When the simulation requests values in between the defined points in time a linear interpolation is employed. If the current simulation time is lower or larger than the minimum or maximum point in time defined in the input file then values at the lowest, respectively largest, point in time are used for the current time step. It is only necessary to define the first three columns of the input file; all other values (vertical wind speed, wind direction, yaw and pitch angle) are set to zero in this case. The file type has to be *.sim to be recognized for import in QBlade.

In the new version of QBlade v0.95 additional values for prescribed floating platform movement can be specified using the *.sim file format. To do this, additional columns need to be added after the pitch angle column. The platform movement is described by platform roll, pitch, yaw and x-, y- and z-translation (see Figure 23 for reference).

Table 3. Sample QBlade Simulation file, to be copy/pasted in a text file

SIM	ROT	H WND	V WND	WND	YAW	PLAT	PLAT	PLAT	PLAT	PLAT	BLA 1	BLA 2	BLA 3	
TIME	SPEED	SPEED	SPEED	DIR	ANG	ROLL	PITCH	YAW	T_X	T_Y	T_Z	PITCH	PITCH	PITCH
0.00	1.00	11.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
5.00	2.00	11.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
10.00	4.00	11.00	0.00	5.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
15.00	7.00	11.00	0.00	10.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	3.00	0.00	0.00
20.00	11.00	11.00	0.00	17.00	0.00	0.00	0.00	0.00	0.00	0.00	6.00	0.00	0.00	0.00
25.00	12.00	11.00	0.00	27.00	0.00	0.00	0.00	0.00	0.00	0.00	9.00	0.00	0.00	0.00
30.00	13.00	11.00	0.00	40.00	10.00	0.00	0.00	0.00	0.00	0.00	9.00	0.00	0.00	0.00
35.00	12.00	11.00	0.00	40.00	20.00	0.00	0.00	0.00	0.00	0.00	9.00	0.00	0.00	0.00
40.00	11.00	11.00	0.00	40.00	30.00	0.00	0.00	0.00	0.00	0.00	5.00	0.00	0.00	0.00
45.00	11.00	11.00	0.00	40.00	40.00	0.00	0.00	0.00	0.00	0.00	2.00	0.00	0.00	0.00
50.00	11.00	11.00	0.00	40.00	40.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
55.00	11.00	11.00	0.00	40.00	40.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
60.00	11.00	11.00	0.00	40.00	40.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
65.00	11.00	11.00	0.00	40.00	40.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
70.00	11.00	11.00	0.00	40.00	40.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
75.00	11.00	11.00	0.00	40.00	40.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
80.00	11.00	11.00	0.00	40.00	40.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
85.00	10.00	11.00	0.00	40.00	40.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
90.00	7.00	11.00	0.00	40.00	40.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
95.00	3.00	11.00	0.00	40.00	40.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

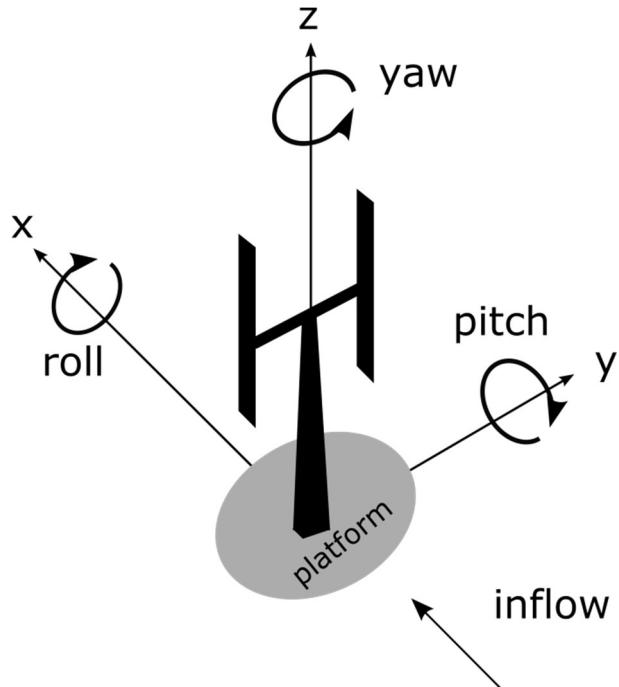


Figure 23. Definition of platform angles and coordinate system in QBlade

AeroDyn HH input file:

AeroDyn hub height input files (Table 4) can be used inside a simulation. The file should be of the *.wnd type to be imported. For a full description of AeroDyn hub height input files see the AeroDyn user's guide on pages 13-17 [16].

Table 4. Sample AeroDyn type hub-height wind file, to be copy/pasted in a text file

Time	Wind Speed	Wind Dir	Vert. Speed	Horiz. Shear	Vert. Shear	LinV Shear	Gust Speed
0.000	15.000	5.000	-1.000	0.020	0.140	0.000	0.000
0.100	16.545	4.755	-0.900	0.022	0.140	0.000	0.000
0.200	17.939	4.045	-0.800	0.024	0.140	0.000	0.000
0.300	19.045	2.939	-0.700	0.027	0.140	0.000	0.000
0.400	19.755	1.545	-0.600	0.030	0.140	0.000	0.000
0.500	20.000	0.000	-0.500	0.033	0.140	0.000	0.000
0.600	19.755	-1.545	-0.400	0.036	0.140	0.000	0.000
0.700	19.045	-2.939	-0.300	0.040	0.140	0.000	0.000
0.800	17.939	-4.045	-0.200	0.045	0.140	0.000	0.000
0.900	16.545	-4.755	-0.100	0.049	0.140	0.000	0.000
1.000	15.000	-5.000	0.000	0.054	0.140	0.000	0.000
1.100	13.455	-4.755	0.100	0.060	0.140	0.000	0.000
1.200	12.061	-4.045	0.200	0.066	0.140	0.000	0.000
1.300	10.955	-2.939	0.300	0.073	0.140	0.000	0.000
1.400	10.245	-1.545	0.400	0.081	0.140	0.000	0.000
1.500	10.000	0.000	0.500	0.090	0.140	0.000	0.000
1.600	10.245	1.545	0.600	0.099	0.140	0.000	0.000
1.700	10.955	2.939	0.700	0.109	0.140	0.000	0.000
1.800	12.061	4.045	0.800	0.121	0.140	0.000	0.000
1.900	13.455	4.755	0.900	0.134	0.140	0.000	0.000
2.000	15.000	5.000	1.000	0.148	0.140	0.000	0.000

Wind input data

Use turbulent windfield:

If the switch is set to “On” a turbulent windfield from the database of QBlade can be used in a simulation. In case a turbulent wind field is used in conjunction with a *Simulation Input File* or *AeroDyn input file* the column with wind speeds is overridden and replaced with wind data from the wind field. To use a windfield in a simulation, its dimensions need to be large enough for the rotor that is simulated. It is also important to check that the height of the wind field center and the hub height of the turbine match, such that the rotor is fully submerged in the windfield. In case of a wind speed being requested for a location outside of the spatial or temporal boundaries of the wind field (if a part of the rotor sticks out of the wind field or the LLFW simulation time is larger than that of the wind field) the mean velocity at hub height of the selected wind field is used. Using Taylor’s frozen turbulence hypothesis, the turbulent wind field is “marched” through the simulation domain along the mean inflow direction and with the mean inflow velocity (Figure 24). To make sure that the rotor is fully submerged in the quasi 3 dimensional wind field at the beginning of a simulation (such as when the simulation starts with a yawed rotor) the wind field is already marched by its width along the mean flow direction before the first time step. This approach is consistent with the treatment in AeroDyn and FAST, and thus ensures comparability of simulation results between these codes and QBlade.

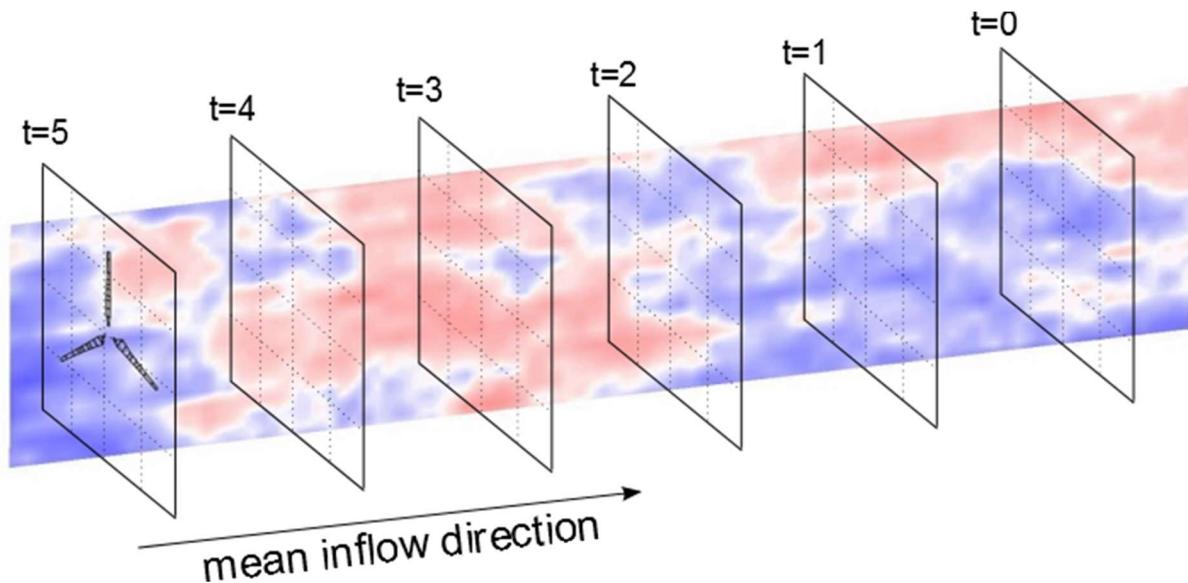


Figure 24. Turbulent wind field marched through domain under Taylor's frozen turbulence hypothesis

Windfield:

The windfield, to be used for the current simulation, can be selected from the drop down menu.

Inflow speed [m/s]:

This value defines the horizontal inflow speed component for the current simulation. It is defined in the x-direction of the global coordinate system.

Vertical inflow speed [m/s]:

This value defines the vertical inflow speed component for the current simulation. A positive value corresponds to an upwards velocity.

Power law shear exponent [-]:

A simple atmospheric boundary layer can be used through the definition of the power law shear exponent:

$$u(x) = u_{hub} \left(\frac{z}{z_{hub}} \right)^\alpha, \quad (19)$$

Where α is the Hellmann exponent, u_{hub} the velocity that is defined as inflow speed and z_{hub} the value defined as hub height. An example for values of the Hellmann exponent is given in (Table 5) [17].

Table 5: Exemplary values of the Hellmann exponent

Location	Hellmann exponent α
Unstable air above open water surface:	0.06
Neutral air above open water surface:	0.10
Unstable air above flat open coast:	0.11
Neutral air above flat open coast:	0.16
Stable air above open water surface:	0.27
Unstable air above human inhabited areas:	0.27
Neutral air above human inhabited areas:	0.34
Stable air above flat open coast:	0.40
Stable air above human inhabited areas:	0.60

Horizontal inflow angle [°]:

This value defines the horizontal inflow angle for the current simulation. It is defined in mathematically positive direction in a top down view (see Figure 25).

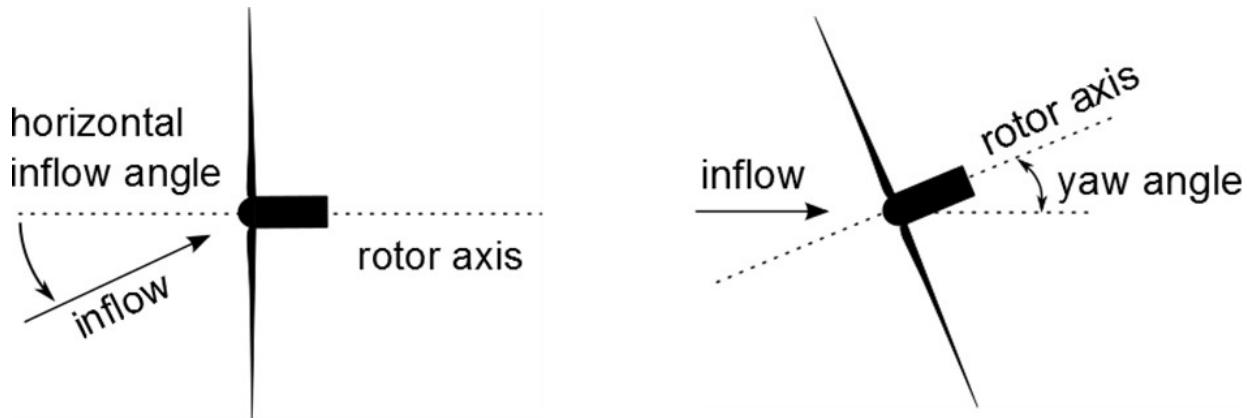


Figure 25. Definition of the horizontal inflow and yaw angle; top down view

Rotor angles

Rotor yaw (HAWT only) [°]:

This value defines the rotor yaw angle for the current simulation. It is defined in mathematically positive direction in a top down view (see Figure 25).

Rotor teeter (HAWT only) [°]:

This value defines the rotor teeter angle for the current simulation. Its definition is depicted in Figure 26.

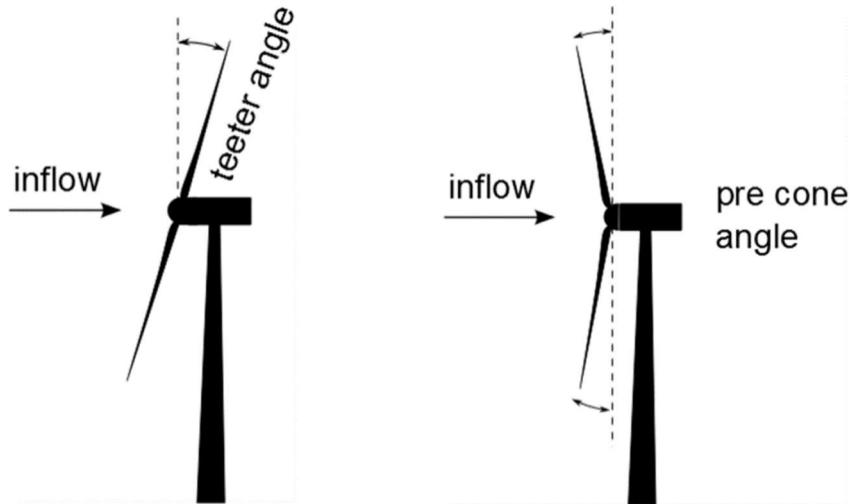


Figure 26. Definition of the rotor teeter and pre-cone angle; side view

Rotor pre cone angle (HAWT only) [°]:

This value defines the rotor pre-cone angle for the simulation. Its definition is depicted in Figure 26.

Y-roll / X-roll angle (VAWT only) [°]:

The Y- and X-roll angles define an inclination of the VAWT turbine around the X- and Y axis of the VAWT turbines tower bottom coordinate system (Figure 27).

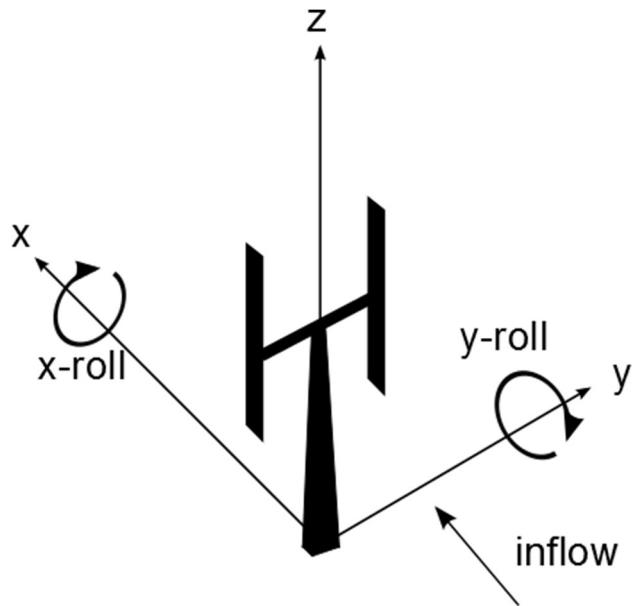


Figure 27. Definition of X- and Y- roll angles for VAWT

Starting position

Blade 1 initial azimuth [deg]:

This value sets the initial azimuthal of blade 1 for the start of the simulation.

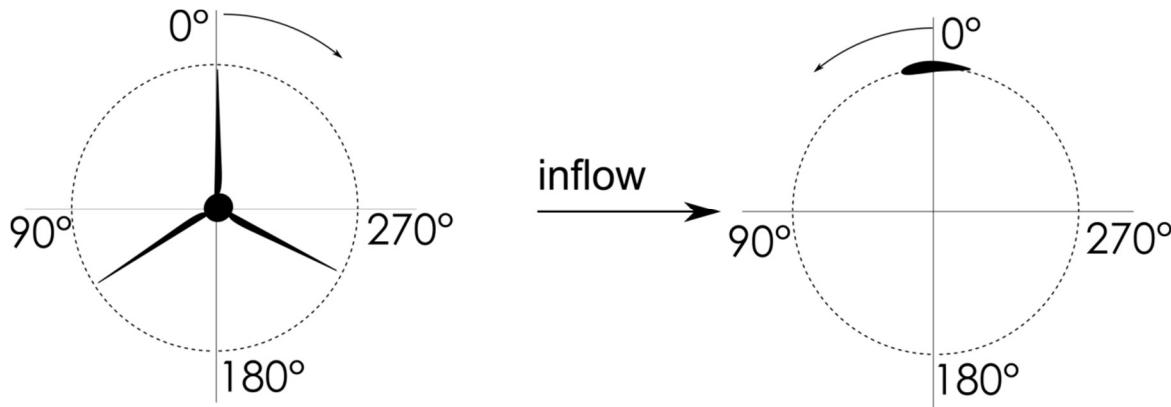


Figure 28. Definition of azimuthal angles for HAWT (left) and VAWT (right) rotors

Operational point

Tip speed ratio [-]:

This value defines the tip speed ratio for which the current simulation will be carried out. If the tip speed ratio is changed the rotational speed is updated to result in the chosen tip speed. Because the tip speed ratio depends on the wind inflow speed it can only be set as constant for a “Simple Simulation”.

Rotor rotational speed [rpm]:

This value defines the rotational speed of the rotor for which the current simulation will be carried out. If the rotational speed is changed the tip speed ratio is updated to result in the chosen rotational speed.

Length of simulation

Number of time steps [-]:

This value defines the total number of time steps for which the current simulation will be carried out. If the number of time steps is changed the number of rotor revolutions and the total simulated time are updated to result in the chosen number of time steps.

Number of rotor revolutions [-]:

This value defines the number of full rotor revolutions for which the current simulation will be carried out. If the number of rotor revolutions is changed the number of time steps and the total simulated time are updated to result in the chosen number of rotor revolutions. The total number of rotor revolutions cannot be chosen when a “Simulation Input File” is used because the rotational rotor speed is not constant in this case.

Simulation length [s]:

This value defines the length of the current simulation. If the simulation length is changed the number of rotor revolutions and the total number of time steps are updated to result in the simulation length.

Time discretization

Azimuthal step size [°]:

This value defines the time step size in terms of azimuthal discretization. If the azimuthal step size is changed the time step size is updated to result in the azimuthal discretization. An azimuthal discretization step size cannot be defined when a “Simulation Input File” is used because the rotational rotor speed is not constant in this case and the implemented LLFW method is currently only capable of constant time stepping.

Time step size [s]:

This value defines the size of a time step. If the time step size is changed the azimuthal step size is updated to result in the chosen time step size.

Adaptive time stepping:

If this is enabled the time step size dT will be adapted every time step to result in the chosen azimuthal step size (based on the current rotational speed). This option is thought for the efficient simulation of rotor start-up cases where the rotational speed varies significantly.

Modeling parameters

3D Himmelskamp effect:

This activates Snel’s correction for the Himmelskamp effect, which increases lift at inner blade stations due to rotational effects. The following formula is used and explained in more detail in [18]:

$$C_{L,3D} = C_{L,2D} + \frac{3.1\lambda^2}{1+\lambda^2} g \left(\frac{c}{r}\right)^2 \left(\frac{dc_l}{d\alpha} |_{linear} (\alpha - \alpha_0) - C_{L,2D} \right) \quad (20)$$

Include ground effects:

If this switch is set to “On” ground effects are included in the simulation. The ground effects are modelled by mirroring the blade and wake vortices at the ground plane (Figure 29). As the mirroring doubles the number of vortices that are needed to be evaluated during the free wake convection step the computational time is effectively doubled as well.

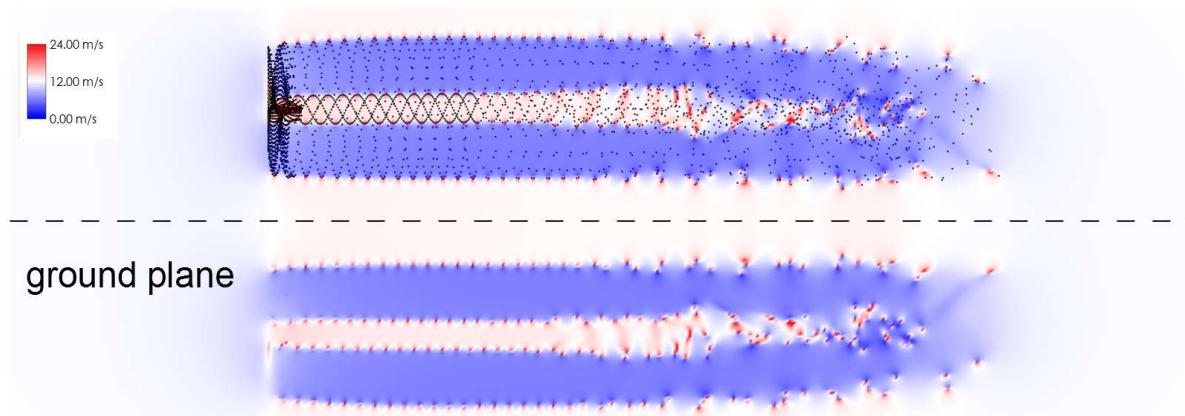


Figure 29. Vortex system mirrored at the ground plane; showing velocity distribution in the rotor mid-plane from side view

Include tower shadow:

When this switch is set to “On” a model for tower shadow effects is included in the simulation. The model implemented in QBlade is an adaption of the tower shadow model employed by AeroDyn and which is based on the work of Bak, et al [19]. This model is based on a superposition of the analytical solution for potential flow around a cylinder and a downwind wake model using a tower drag coefficient. The tower shadow model only affects velocity components in the x-y plane of the tower base coordinate system; the z-component of the velocity (perpendicular to the tower cross sections) remains unaffected. When the tower influence at an arbitrary point in space is calculated, the potential flow field of the tower is rotated such that its x-axis aligns with the respective velocity vector (free stream and induced velocity at the point) at the evaluation point (Figure 30). It is then assumed that the velocity at the evaluation point is the uniform inflow velocity for the potential flow solution. For this solution we know the velocity distribution at all points in the tower model coordinate system. We then calculate the x- and y- velocity component at the evaluation point and transform this vector back into the global coordinate system. The tower model is only used when the z-component of the evaluation points position vector is smaller or equal to the tower height, this however introduces a discontinuity in the 3D velocity field of the simulation domain at the tower top (but its effect on simulation results usually is quite small).

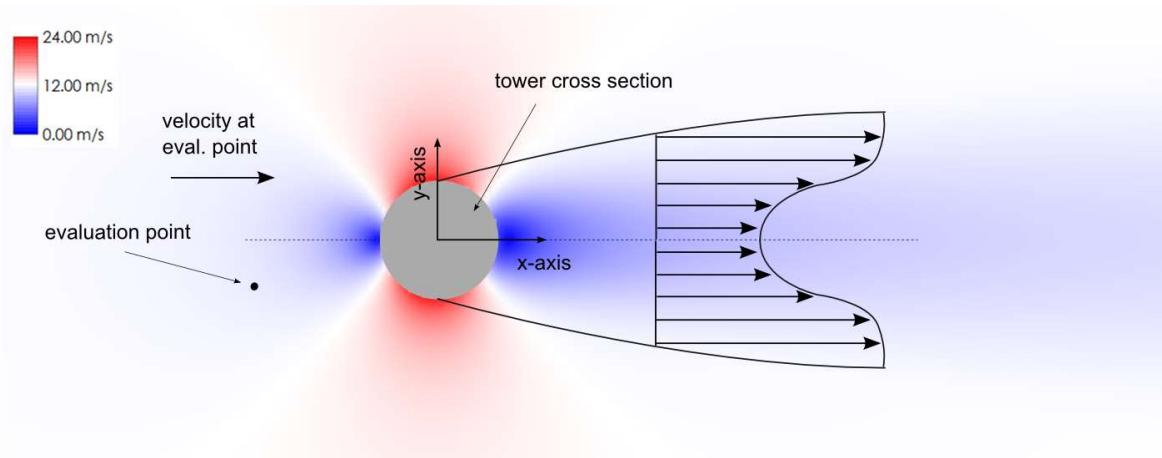


Figure 30. Schematic view of the tower shadow model, overlaid with velocity distribution from QBlade calculation, drawing reproduced from [5]

Tower height (VAWT only) [m]:

For VAWT turbines this value defines the height of the tower. Contrary to a HAWT the tower height is not always the same as the hub (or rotor center) height. This means for VAWT that in one case the tower has an influence over the whole rotor height, in a different case only over its lower half (Figure 31).

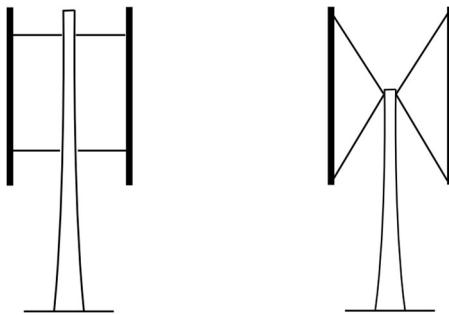


Figure 31. Different tower heights for the same VAWT rotor

Tower top radius/tower base radius [m]:

The tower top radius and tower base radius define the tower radii at the respective position. Depending on the z-coordinate of the evaluation point a linear interpolation is used to obtain the tower radius for the tower shadow model.

Tower drag coefficient [-]:

This value defines the tower drag coefficient which is used to compute the wake deficit in the tower shadow model (see [5]). This value is more important for downwind or vertical rotor configurations.

Wake convection

Velocity evaluation (Vortex):

This option selects where the velocity will be evaluated during the wake convection step. If *NODE* is selected the velocities are evaluated at the nodes (or end points) of the vortex line elements (see Figure 32). If *CENTER* is selected the velocities are evaluated at the central points of the vortex lines. The resulting velocity field is then re-projected onto the node points. This process leads to a slightly smoothed velocity distribution on the wake nodes and to a more stable wake structure and is the recommended option in most cases.

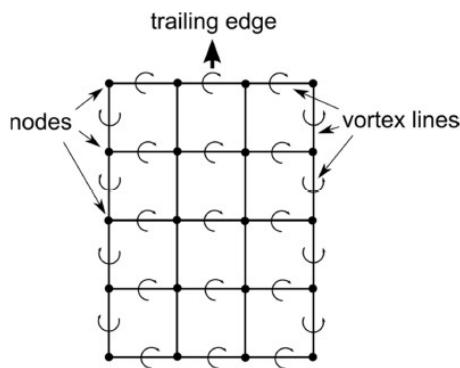


Figure 32. Simplified wake structure

Velocity integration:

One of three types of velocity integration can be selected for the free convection step of the wake.

The 1st order Euler forward integration scheme (*EF*):

$$\vec{x}_{t+1,EF} = \vec{x}_t + (V_\infty + V_{ind}(\vec{x}_t))\Delta t. \quad (21)$$

The 2nd order predictor-corrector scheme (*PC*):

$$\vec{x}_{t+1,PC} = \vec{x}_t + (2V_\infty + V_{ind}(\vec{x}_t) + V_{ind}(\vec{x}_{t+1}))\frac{\Delta t}{2}. \quad (22)$$

The 2nd order predictor-corrector backwards scheme (*PC2B*) proposed by Bhagwat and Leishman [20]:

$$\vec{x}_{t+1,PC2B} = \vec{x}_{t,PC2B} + (3\vec{x}_{t+1,PC} - \vec{x}_{t,PC2B} - 3\vec{x}_{t-1,PC2B} + \vec{x}_{t-2,PC2B})\frac{1}{4}. \quad (23)$$

Wake modeling

The modeling of the wake is highly important in a lifting line simulation. It affects both; the computational cost and the accuracy of a simulation. It is often possible to speed up the simulation by a factor of 10 or more without greatly affecting the accuracy. On the contrary in some situations (such as high induction cases at a high TSR) the wake discretization has a very large effect on the computed performance of the rotor. As the computational cost for the evaluation of one time step is roughly proportional to N^2 , where N is the number of vortex elements, it is important to limit the maximum number of vortex elements in a simulation. In QBlade the wake is separated into three zones; the near field zone 1, where shed and trailing vorticity (as shed from the blades) exist; zone 2, from which the shed vorticity is removed and in which the trailing vorticity is concentrated into lumped vortices and zone 3, which has a coarser spatial discretization than zone 2. This zoning strategy is employed to enable a high accuracy of the near field wake and the rotor performance, while keeping the computational cost as low as possible. The parameters, governing the extensions of the three zones are: *max wake age*, *full wake length*, *fine wake length* and *wake thin factor* and are discussed in the following. The process that is employed to concentrate the trailing vorticity into lumped vortices is explained under the point “*Min. Gamma Fraction*” in this document.

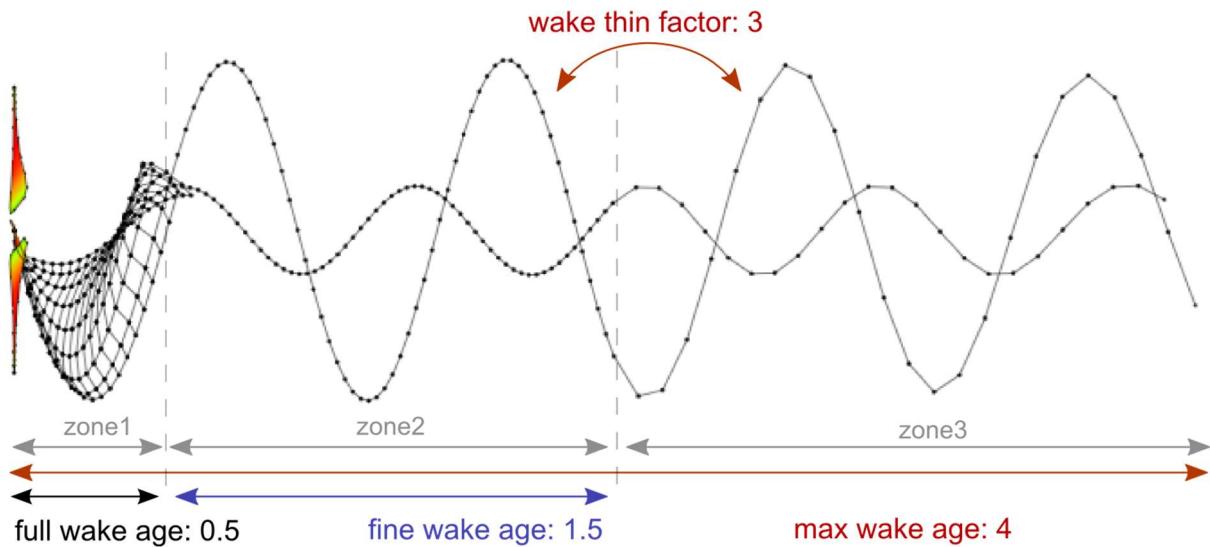


Figure 33: Example of wake zones and parameters with exemplary values given

Wake rollup:

If this switch is set to “On” the wake elements are convected with the local inflow velocity and the induction computed from all other wake elements. If set to “Off” free wake simulation is disabled and the wake elements are only convected by the inflow velocity.

Include trailing vortices:

If this switch is set to “On” trailing vorticity (oriented in stream wise direction) is shed from the trailing edge and thereby included in the simulation.

Include shed vortices:

If this switch is set to “On” shed vorticity (oriented in span wise direction) is shed from the trailing edges and included in the simulation.

Turbulent wake convection:

If this switch is set to “On” the wake elements are convected with the local turbulent inflow velocity (in case a turbulent wind field is used). In this case the convection of the wake elements is also influenced by the tower shadow model. If the turbulent wake convection is switched “Off”, the wake elements are only convected with the mean inflow velocity.

Prescribed wake:

If this switch is set to “On” all wake elements move on a straight, prescribed path that is parallel to the rotor axis and originates from their release point at the trailing edge. This also means that in this case the induction from other wake elements only has an effect in the direction of the prescribed path. This feature is intended to be used in case the wake is breaking down very early in a highly turbulent wind field.

Max wake age in revolutions / in time steps [-]:

This value defines the maximum allowable age of a wake vortex element (see Figure 33). The wake age is represented as the dimensionless number rotor revolutions, starting from the time the vortex element is shed from the trailing edge of the blade. After a vortex element reaches the maximum age it is removed from the simulation. This parameter is highly important to limit the maximum number of vortex elements in the simulation which largely defines the computational time needed for a simulation. At the same time the accuracy of the computed rotor parameters and performance is affected by this value. By how much the accuracy is affected is a function of the TSR that is currently being simulated. The graphs in Figure 35 & Figure 36 show that for high TSR’s a larger number of rotor revolutions is needed. In the example shown in Figure 35 a wake age of 14 rotor revolutions is needed at a TSR of 10 to reach a relative error of 1%. The three TSR’s shown in Figure 35 & Figure 36 represent an above rated power TSR case (black), a rated power TSR case (red) and a below rated power TSR case (blue).

When a simulation input file (*.sim) is used during a simulation, or a startup case is simulated, the maximum wake age cannot be defined in rotor revolutions (as the azimuthal discretization is not constant) and is defined as the maximum number of time steps a wake element is allowed to exist. Make sure to adjust the “*Max Wake Age*” value in this case, as the default value of 8 is far too small when “*Max Wake Age*” is measured in time steps.

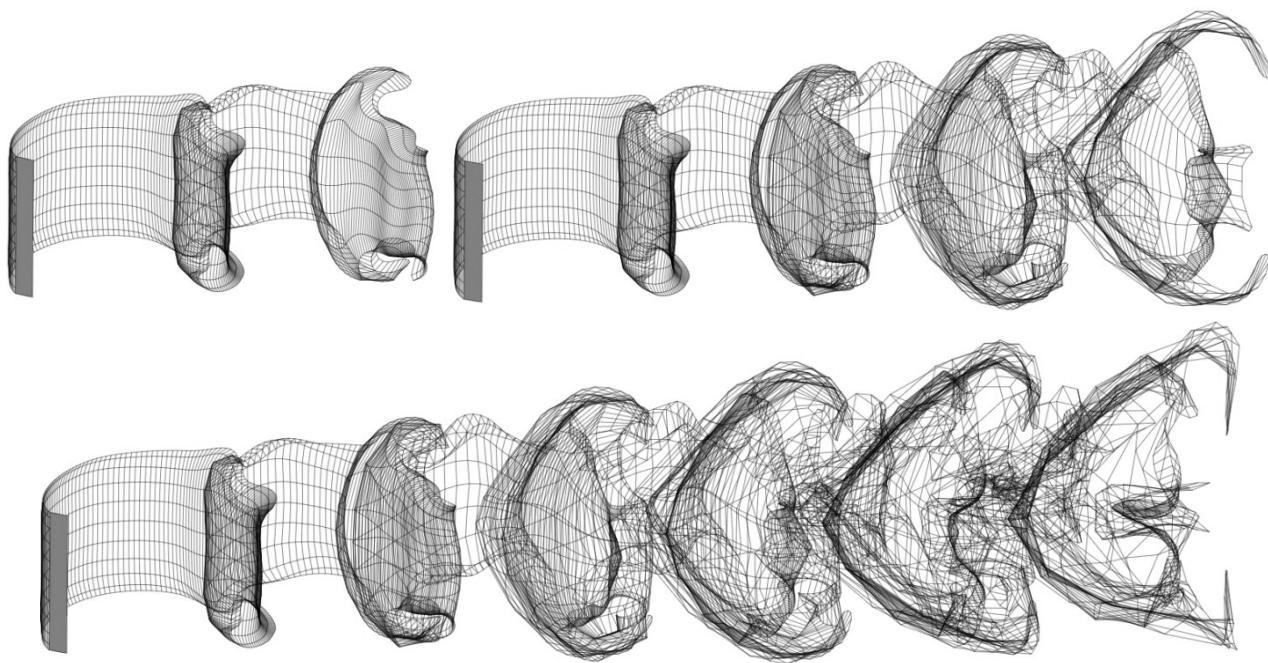


Figure 34. Examples for different “Max Wake Age” values for a VAWT; from top left: 2 revs, 4 revs, 6 revs

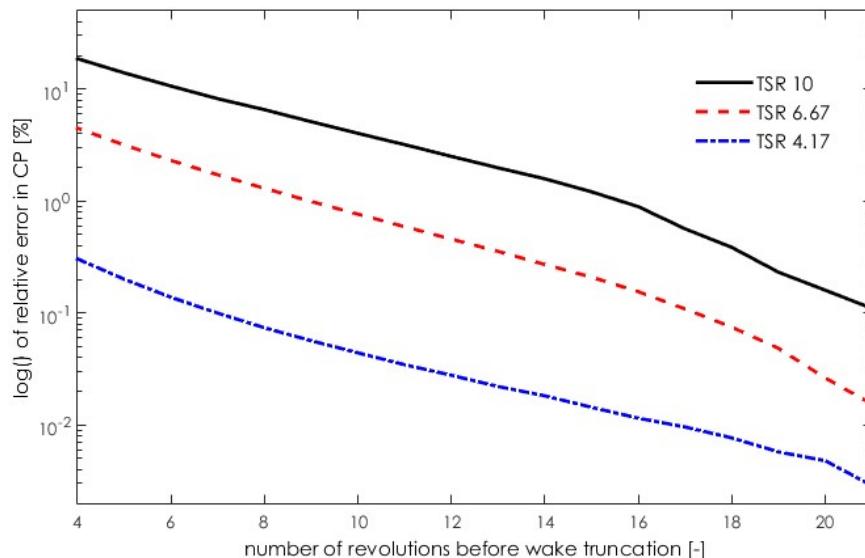


Figure 35. HAWT Case: The effect of maximum wake age on accuracy (Cp Betz value was used as integral value to compare accuracy)

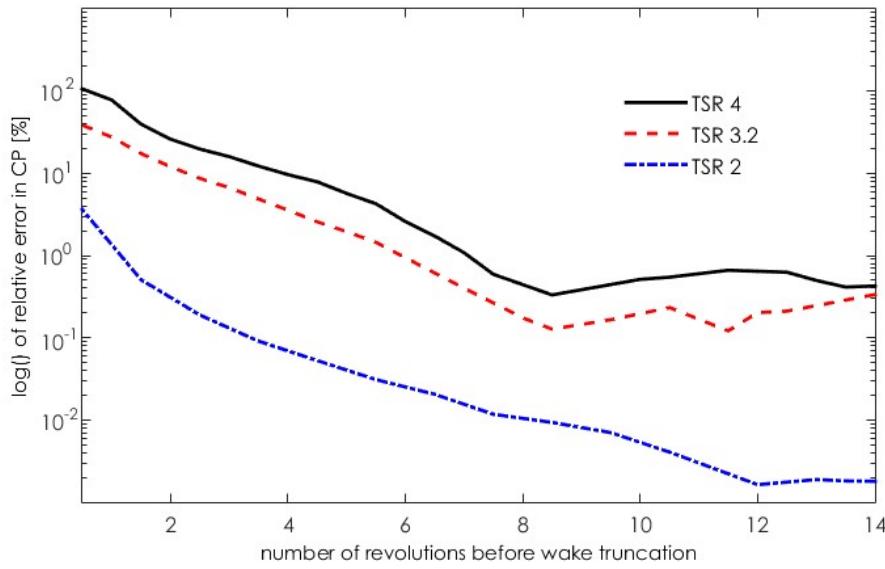


Figure 36. VAWT case: The effect of maximum wake age on the accuracy of the power coefficient

Full wake age in revolutions / in timesteps (HAWT only) [-]:

The parameter “full wake age” defines after how many rotor revolutions the shed vorticity is removed from the wake and the trailing vortices are concentrated into lumped vortices (see Figure 33). In highly unsteady simulations a “longer” full wake increases the accuracy, whereas in steady state simulations (such as Cp – TSR predictions) where the shed vorticity tends to zero a relatively short “full wake” length is sufficient. Similar to the parameter “max wake age” the “full wake age” influence on the accuracy of the result is also a function of the TSR (see Figure 37). The full wake age is usually defined in rotor revolutions, however when a *.sim file is used the age is defined in “number of time steps”.

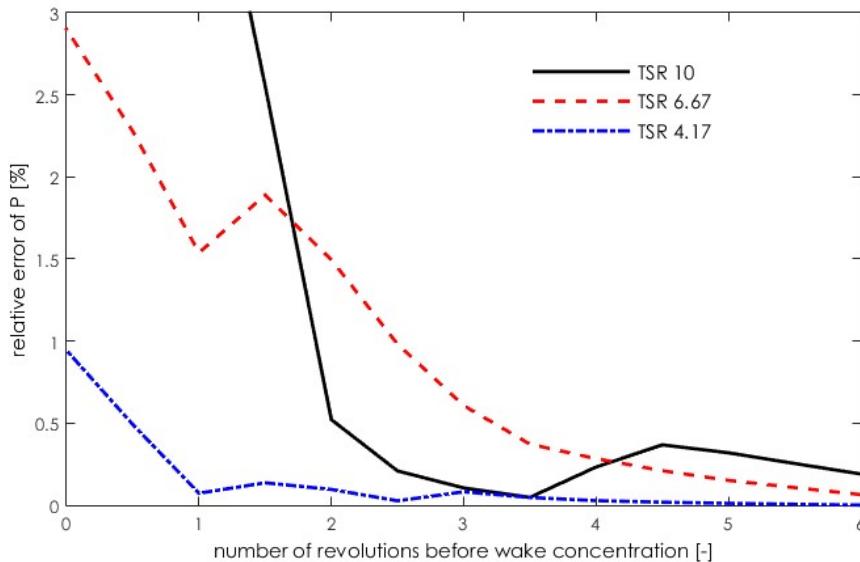


Figure 37. HAWT case: The effect of full wake age on the accuracy of the power coefficient

Fine wake age in revolutions / in timesteps (HAWT only) [-]:

The parameter “fine wake age” defines the length of zone 2 in Figure 33. The length of zone 3 (where the spatial discretization of the lumped trailing vorticity is coarser than in zone 2) is defined by subtracting the *full wake* and the *fine wake* length from the *total wake* length.

Wake thin factor (HAWT only) [-]:

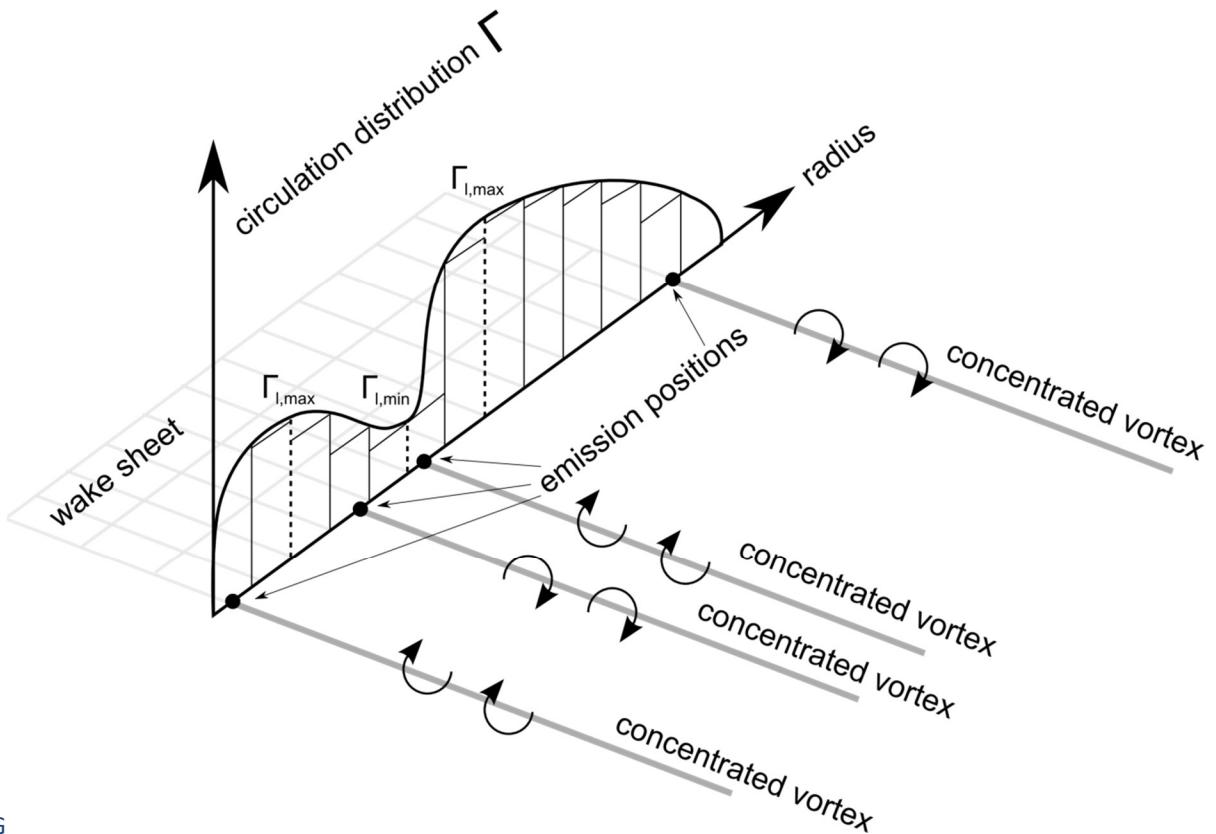
The “wake thin factor” defines by how much the wake is thinned out in zone 3, resulting in a coarser spatial resolution of the wake and a smaller number of vortex elements in the far wake region. A factor of 4 means that the number of wake nodes in zone 3 is by a factor of 4 smaller than in zone 2 (or 4 vortex elements are combined into 1 between zone 2 and zone 3).

Min gamma fraction (HAWT) [-]:

To limit the number of free vortex elements in the wake, the trailing vorticity is concentrated into lumped vortices between zone 1 and zone 2. The strength of the lumped vortices is obtained by integrating the circulation of the wake sheet from local minima to local maxima, as shown in Figure 38.

$$\Gamma_{con} = \int_{x_{min}}^{x_{max}} \Gamma_{trail}(x) dx . \quad (24)$$

In the illustrated case (double hump circulation distribution) this approach leads to four concentrated vortices. Their emission points are obtained by weighting the positions of the converted vortices (in the full wake sheet) by their contributing vortex strength. The circulation distribution of the wake sheet at the end of zone 1 can have very general shapes. If the distribution is only slightly jittering many local minima and maxima would be detected by the algorithm and no significant reduction of number of trailing vortex elements could be obtained. To prevent this, the parameter “min gamma fraction” defines the smallest strength that a concentrated trailing vortex is allowed to have. If the strength of the concentrated vortex is below this limit the vortex is lumped with the nearest concentrated vortex in its vicinity. The minimum strength that is allowed is defined as a fraction of the largest strength in the vortex sheet (usually the tip vortex), so a value of “min gamma fraction” of 0.1 would mean that the vortex strength of the weakest concentrated vortex cannot be smaller than 10% of the strongest vortexes strength. If “min gamma fraction” is continuously increased only the tip and the hub vortex will remain eventually, if its value is 0 no concentration of trailing vorticity will happen at all. Overall “min gamma fraction” can be seen to control the spatial resolution of the trailing vorticity in zone2.



G

Figure 38. Illustration of the implemented approach for the concentration of trailing vorticity

Reduce wake by x % (VAWT) [-]:

In case a VAWT rotor is simulated the far wake cannot be easily concentrated or simplified as both shed and trailing vorticity have an important influence on the flow field, even in a steady state simulation. The strategy that is employed in this case is to simply reduce the newly created vortex elements with the lowest circulation at each time step. The % of wake elements that shall be removed in this way is the user input. As an example: A user input of 0.1 would remove the 10% of the newly created wake elements with the lowest circulation (see also the example in Figure 39). Figure 40 shows the effect of the VAWT wake reduction on the accuracy for three representative TSR's (above, at and below rated TSR). It can be seen that up to 50% of wake elements can be removed in this way without affecting the averaged C_p value by more than 1%, compared to the full wake case. A wake reduction of 50% speeds up the simulation roughly by a factor of 4.

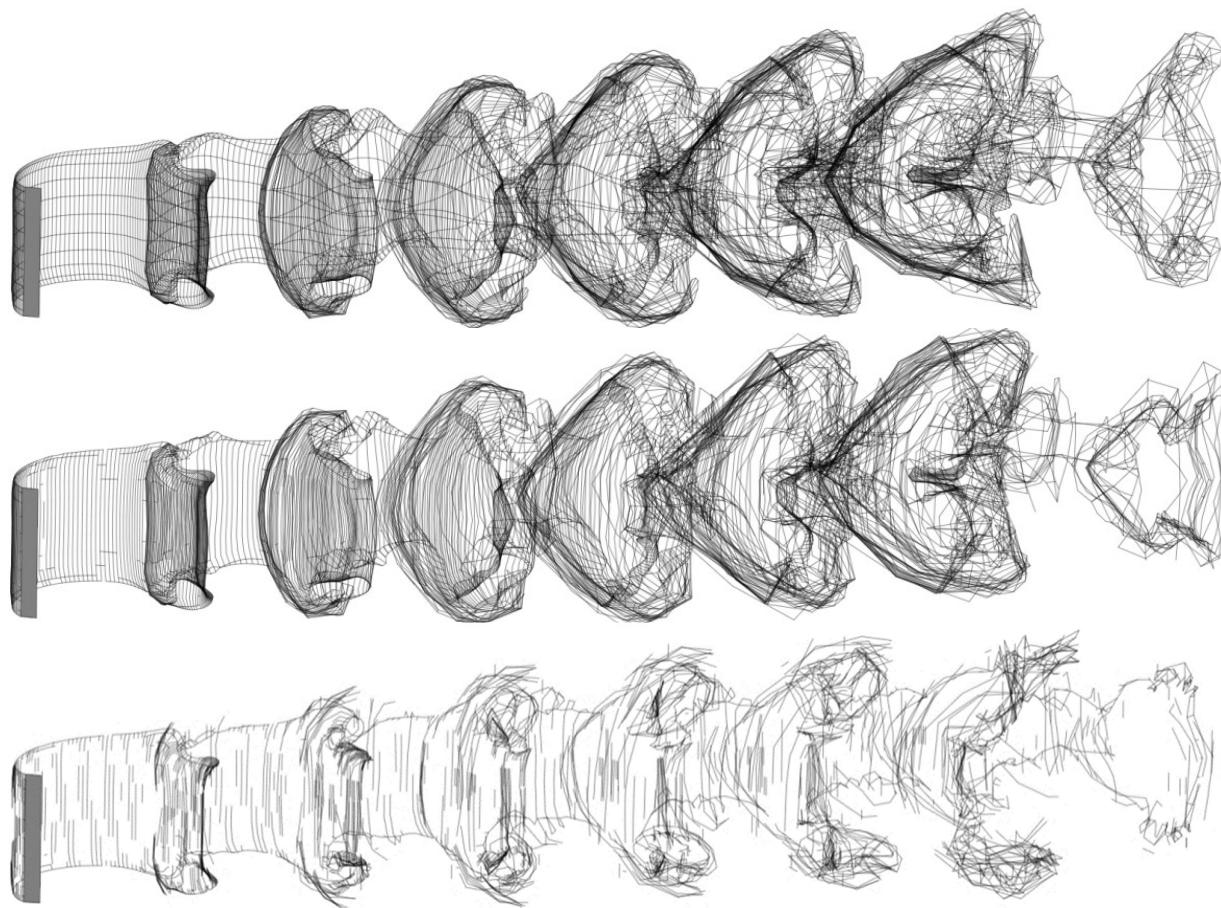


Figure 39. An example for the VAWT wake reduction - top: full wake; middle: 35% reduced; bottom: 70% reduced

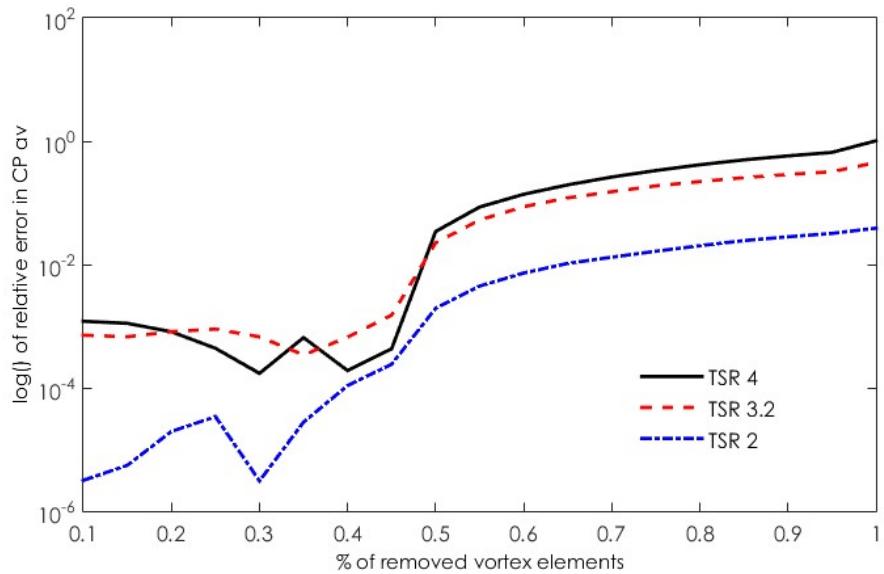


Figure 40. The effect of wake reduction on accuracy (Mean Cp was used as integral value to compare accuracy)

First wake row length fraction [-]:

This sets the length of the first wake row in stream wise direction. A value of 0.1 would mean that the length of the first wake row is 10% of its normal length. The distance of the first wake row has an effect on the wake – blade induction. A value of 0.3 is usually proposed in the literature [ref KATZ PLOTKIN].

Vortex modeling

The Biot-Savart-Equation exhibits a singularity (see Figure 41) at the core where $\vec{r} = 0$:

$$\vec{u}_\Gamma(x) = \frac{\Gamma}{4\pi r_1 r_2 (r_1 r_2 + \vec{r}_1 \cdot \vec{r}_2)} \cdot \frac{(r_1 + r_2)(\vec{r}_1 \times \vec{r}_2)}{(\vec{r}_1 \times \vec{r}_2)}. \quad (25)$$

To prevent this singularity from affecting the simulation, and also to model the viscous core of the bound and free vortices more accurately, a model for a viscous vortex core was implemented in QBlade. Many different models to describe the tangential velocity distribution around the core exist, such as the Ramasay and Leishman model:

$$\vec{u}_\Gamma(x) = K_v \frac{\Gamma}{4\pi r_1 r_2 (r_1 r_2 + \vec{r}_1 \cdot \vec{r}_2)} \cdot \frac{(r_1 + r_2)(\vec{r}_1 \times \vec{r}_2)}{(\vec{r}_1 \times \vec{r}_2)}. \quad (26)$$

The viscous parameter K_v is defined as follows:

$$K_v = \frac{h^2}{(r_c^{2z} + h^{2z})^{1/z}}; \quad \text{with } h = \frac{|r_1 \times r_2|}{|r_2 - r_1|}. \quad (27)$$

Van Garrel suggests in his report [4] to use a cut-off radius, that is simply added to the denominator of Eq.5 in the form of $(\delta l_0)^2$, where the length l_0 is the length of the vortex element and δ a fixed number between 0 and 0.1 . This ensures that the induced velocity smoothly approaches zero in the vicinity of the core. This is a very elegant and computationally efficient implementation, as the viscous core modeling is directly implemented in the calculation of the induced velocity. In QBlade the growth of the viscous vortex core is modeled and the core size r_c is used directly instead of a fixed length fraction δl_0 :

$$\vec{u}_\Gamma(x) = \frac{\Gamma}{4\pi r_1 r_2 (r_1 r_2 + \vec{r}_1 \cdot \vec{r}_2) + r_c^2} \cdot \frac{(r_1 + r_2)(\vec{r}_1 \times \vec{r}_2)}{(\vec{r}_1 \times \vec{r}_2)}. \quad (28)$$

For other vortex models a viscous parameter (such as K_v) needs to be reevaluated whenever the Biot-Savart Equation is computed which is considerably increasing the computational cost. To ensure comparability to other codes both approaches are implemented in QBlade and the user has to choose one in the simulation setup. However due to considerable advantages in terms of computational efficiency it is suggested to use Van Garrel's approach.

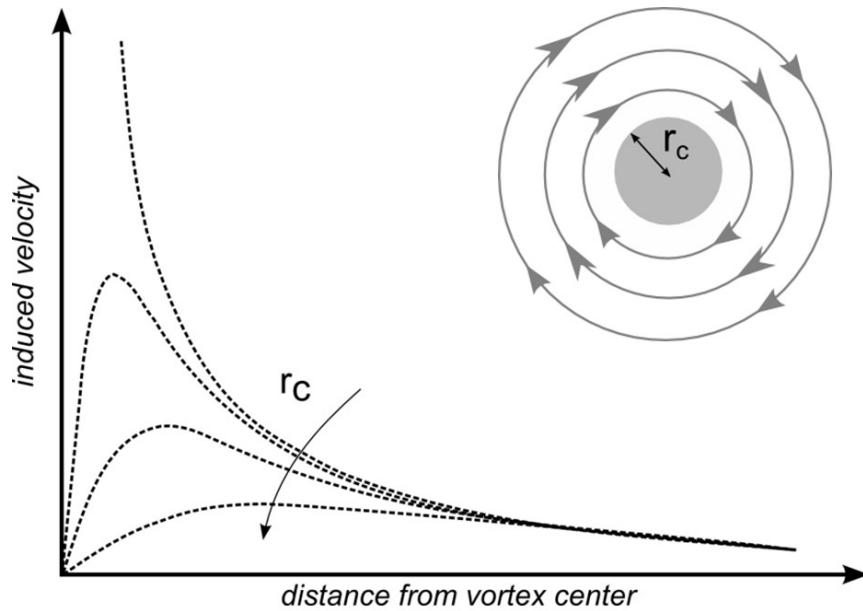


Figure 41. Schematic view of the influence of the core size on the tangential velocity induced by a vortex element

Both implementations require the radius of the viscous vortex core that is computed with the Lamb-Oseen vortex model, including filament straining, where Leishman added an additional parameter δ_v to model turbulent dissipation:

$$r_c = \left[\frac{4\alpha \delta_v v (t_v + S_c)}{1 + \epsilon} \right]. \quad (29)$$

The two free parameters, the time offset parameter S_c and the turbulent viscosity δ_v are used to adjust the initial core size and its growth rate in time (see Figure 42).

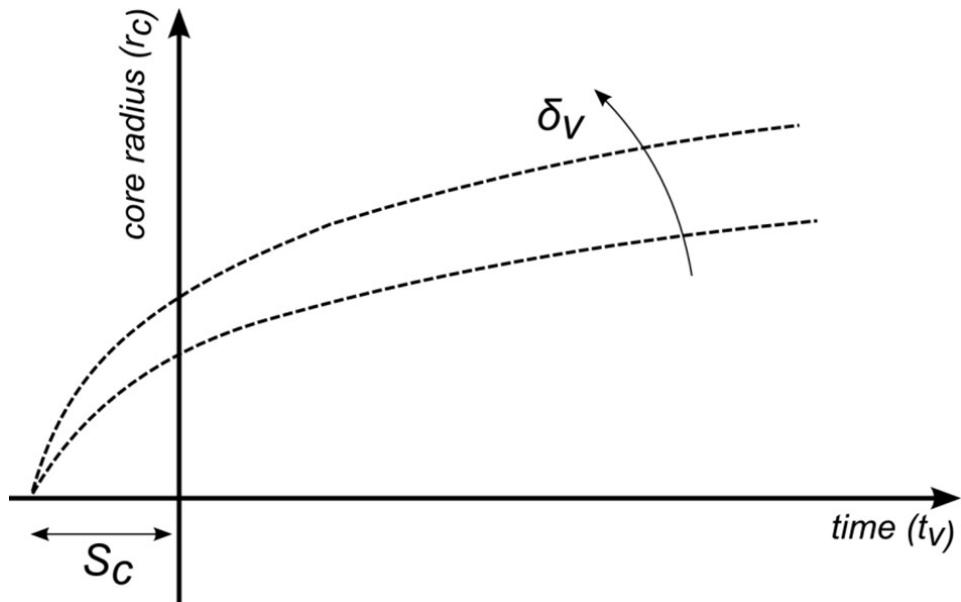


Figure 42. Schematic view of the influence of S_c and δ_v on the vortex core size

No empirical relationships to obtain values for these parameters exist to date. Sant [21] reports that experimental investigations suggest using a δ_v value of roughly 10 for small scale rotors and a value between 100-1000 for full scale rotors. The parameter S_c is needed to prevent vortices from having a zero core radius when being released from the trailing edge. These two parameters can have a significant effect on the computed rotor performance and choosing too large values for them reduces the induction in the near wake and leads to over predictions of rotor power output.

Desingularization type:

Here the user can choose to either use Van Garrel's or Leishman's model.

Leishman z exponent [-]:

This defines the z-exponent of Eq.7. $z=1$ corresponds to the Scully model [22] and $z=2$ approximates the Lamb-Oseen vortex model [23].

Vortex time offset [s]:

This defines the S_c parameter in Eq.9.

Turbulent vortex viscosity [-]:

This defines the δ_v parameter in Eq.9.

Maximum vortex stretching factor [-]:

During the simulation, the vortex line elements undergo dilation and contraction when the vortex nodes are convected away from - or closer to each other. The resulting strain modifies the vortex core size (Equation 11). Under some circumstances, in areas with large velocity gradients, the vortex stretching and strain can become unphysically large and can lead to a distortion of the whole wake structure during the convection step. To prevent unphysically large vortex stretching the "Maximum Vortex Stretching Factor" parameter limits the stretching of each vortex to a maximum. All vortices, whose lengths exceed their initial lengths (when first shed from the trailing edge) multiplied by this parameter are removed from the wake to prevent wake distortion and unphysical behavior.

Unsteady aerodynamics model

Activate UA model:

Activates the unsteady aerodynamics model during the simulation. This only has an effect if the polars used in the blade definition have been decomposed previously (see *Polar Decomposition*).

Activate vortex lift:

Activates the vortex lift component of the Beddoes-Leishman type dynamic stall model.

Please Note:

The vortex lift model, especially the logic behind the formation and convection of the vortex, is still under development and only implemented as an experimental version. Especially for situations of turbulent inflow or very large AoA fluctuations (as in low TSR VAWT simulation) this model does not always give good results. In the current release this feature is deactivated, it will be activated once a suitable formulation – working for a wide range of both HAWT and VAWT cases has been implemented.

Time constant Tf [-]:

Time constant for separation point (boundary layer) lag.

Time constant Tp [-]:

Time constant for LE pressure lag.

Time constant Tv [-]:

Time constant for vortex lift decay.

Time constant Tvl [-]:

Time constant for vortex lift convection.

Special test cases

Rotor standstill:

If this option is switched on the simulation will be carried out at a rotational and a tip speed ratio of 0.

Simulate startup:

If this option is activated a rotor startup simulation will be carried out. The rotor starts at the defined initial rotational speed and accelerated by the generated torque. The following formula, neglecting friction or an opposing generator torque, is used to compute the acceleration:

$$\dot{\omega} = \frac{T}{I}, \quad (30)$$

where I is the moment of rotational inertia.

Moment of (rotational) inertia [kg*m^2]:

Defines the moment of rotational inertia in Equation 12.

Initial rotor speed [rpm]:

Defines the initial rotor speed at the beginning of the startup simulation. The initial rotor position can be defined in the *Starting Position* group.

Output parameters

Starting time to store output [s]:

Defines the starting time from which the output is stored (both visualization and performance parameters). This is to be used when the initial transient should not be present in the results of a simulation.

Algorithm parameters

Maximum number of iterations [-]:

This parameter defines the maximum number of iterations when calculating the bound circulation on the lifting line.

Relaxation factor [-]:

The relaxation factor r is in effect when updating the bound circulation at the blade during the iteration loop:

$$\Gamma_{i+1} = r \Gamma_i + (1 - r) \Gamma_{i-1}. \quad (31)$$

In cases with very fine blade discretization it can be necessary to reduce r if convergence is not achieved.

Epsilon [-]:

This parameter defines the convergence criterion. If the largest difference in bound circulation, between two successive iterations, is smaller than ε the iteration loop ends (if $(\Gamma_{i+1} - \Gamma_i) < \varepsilon$).

Environmental variables

Air density [kg/m³]:

This parameter defines the air density, used to compute forces and torque.

Kinematic viscosity [m²/s]:

This parameter defines the kinematic viscosity, used to compute the Reynolds number.

Simulation output variables:

To provide an overview the simulation output variables are listed here with a short description. The *global* rotor output variables can be plotted in time graphs only whereas the *local* blade variables can be plotted in both time and blade graphs. If a blade variable is plotted in a time graph the value at the blade station that is selected in the “Section to Plot” dialog (Figure 17) is plotted in the graph.

The rotor output variables (Table 6) are values that are obtained through integration of the local blade variables over the blade length or variables describing the temporal variation simulation characteristics such as of inflow speed or azimuthal angle.

Table 6. Rotor Output (Global) Variables

Name	Description	Unit
Time	The simulation time	[s]
Azimuthal Position of Blade X	As defined in Figure 28	[°]
Rotational Speed	The current rotor speed	[rpm]
Tip Speed Ratio	The current tip speed ratio	[‐]
Power Coefficient (HAWT)	The current Cp Betz for HAWT	[‐]
Momentary Power Coefficient (VAWT)	The current Cp Betz for VAWT	[‐]

Momentary Thrust Coefficient (VAWT)	The current Thrust coefficient for VAWT	[-]
Momentary Torque Coefficient (VAWT)	The current Torque coefficient for VAWT	[-]
Momentary Power (VAWT)	The momentary rotor power	[kW]
Momentary Torque (VAWT)	The momentary rotor torque	[Nm]
Averaged Power Coefficient (VAWT)	Cp Betz averaged over one revolution	[-]
Averaged Thrust Coefficient (VAWT)	Thrust Coefficient averaged over one revolution	[-]
Averaged Torque Coefficient (VAWT)	Torque Coefficient averaged over one revolution	[-]
Averaged Power (VAWT)	Rotor power averaged over one revolution	[kW]
Averaged Torque (VAWT)	Rotor torque averaged over one revolution	[Nm]
Thrust Coefficient	The current thrust coefficient	[-]
Torque Coefficient	The current thrust coefficient	[-]
Power	The current rotor power	[kW]
Torque	The current rotor torque	[Nm]
Thrust in X	Thrust force in global X-direction	[N]
Thrust in Y	Thrust force in global Y-direction	[N]
Thrust in Z	Thrust force in global Z-direction	[N]
Wind Speed at Hub Height	Undisturbed inflow velocity w/o induction	[m/s]
Pitch Angle of Blade X	The current pitch angle each blade	[°]
Platform pitch angle	The platform pitch angle around the y axis	[°]
Platform roll angle	The platform roll angle around the x axis	[°]
Platform yaw angle	The platform yaw angle around the z axis	[°]
Platform Translation X	The platform translation in x direction	[m]
Platform Translation Y	The platform translation in y direction	[m]
Platform Translation Z	The platform translation in z direction	[m]
Yaw Angle (HAWT)	The rotor yaw angle	[°]
Horizontal Inflow Angle	The horizontal inflow angle	[°]
Vertical Inflow Angle	The vertical inflow angle	[°]
OOP Root Bending Mom Blade X (HAWT)	The out of plane root bending moment	[Nm]
IP Root Bending Moment Blade X (HAWT)	The in plane root bending moment	[Nm]
Vortex Core Size Evolution from t=0s	Evolution of the core size of a vortex released at t=0s	[m]
Iterations	Iterations needed to compute bound blade circulation	[-]
Elapsed Simulation Time	Real time needed to run the simulation	[s]

The local blade output variables (Table 7) are values obtained at the individual blade stations for every blade, thus they can be plotted over the blades length or height in the *blade* graphs. As stated previously the variables at the currently selected “Section to Plot” can also be plotted in time graphs

Table 7. Blade Output Variables

Name	Description	Unit
Lift Coefficient	Lift coefficient from polar data	[\cdot]
Drag Coefficient	Drag coefficient from polar data	[\cdot]
Lift to Drag Ratio	L/D ratio from polar data	[\cdot]
Circulation	The bound circulation	[m 2 /s]
Angle of Attack	The computed AoA that is used to get polar data	[$^\circ$]
Length / Height	Length (HAWT) or height (VAWT)	[m]
Chord	The chord length	[m]
Blade Twist Angle	The blade twist angle	[$^\circ$]
Normal Force Coefficient	Cn defined in the current direction of the rotor axis	[\cdot]
Tangential Force Coefficient	Ct defined in current tangential direction	[\cdot]
Normal Force	Computed from Cn	[N/m]
Tangential Force	Computed from Ct	[N/m]
Dynamic Pressure	From total velocity	[Pa]
Total Velocity	Total velocity IPS (<u>In Plane of Station</u>)	[m/s]
Total Velocity (without Induction)	Total velocity IPS without induction	[m/s]
Inflow Velocity (without Induction)	Free stream velocity IPS without induction	[m/s]
Relative Velocity from Blade Movement	The velocity IPS from blade movement	[m/s]
Velocity induced from Wake	The wake induced velocity IPS	[m/s]
Velocity induced from Tower	The tower induced velocity IPS	[m/s]
Axial induction	The axial induction factor	[\cdot]
Interference Factor (1-a) (VAWT)	The axial interference factor to compare with the DMST	[\cdot]
Tangential Induction (HAWT)	The tangential induction factor at the station	[\cdot]
Reynolds Number	The Reynolds number at the station	[\cdot]
Delta Re Blade – Re Polar	Delta between the blade Re and the Re of the used polar	[\cdot]

Validation

HAWT

The HAWT part of QBlades Lifting Line module has been extensively validated against numerical and experimental data from the literature. The data used for the validation was obtained from the MexNext-I [24] [25] project, where numerical tools were compared to the MEXICO [26] experiment, and the NREL Phase VI [27] campaign. Overall the implementation in QBlade showed surprisingly accurate results and even outperformed some higher fidelity models. An excerpt of the HAWT validation is shown in the following graphs (Figure 43, Figure 44 Figure 45).

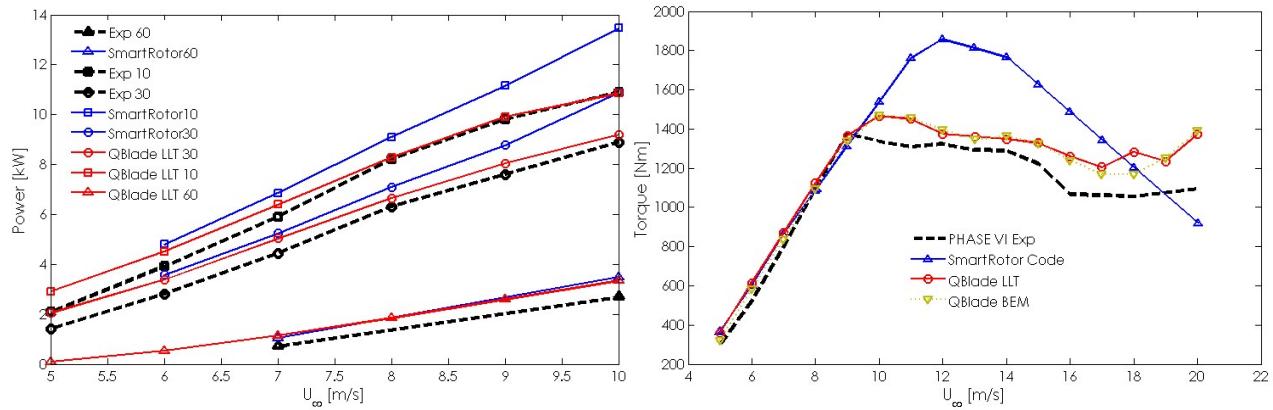


Figure 43. Validation with PHASE VI data and the SmartRotor code from [28]. Left: Power output for different yaw cases; right: power curve

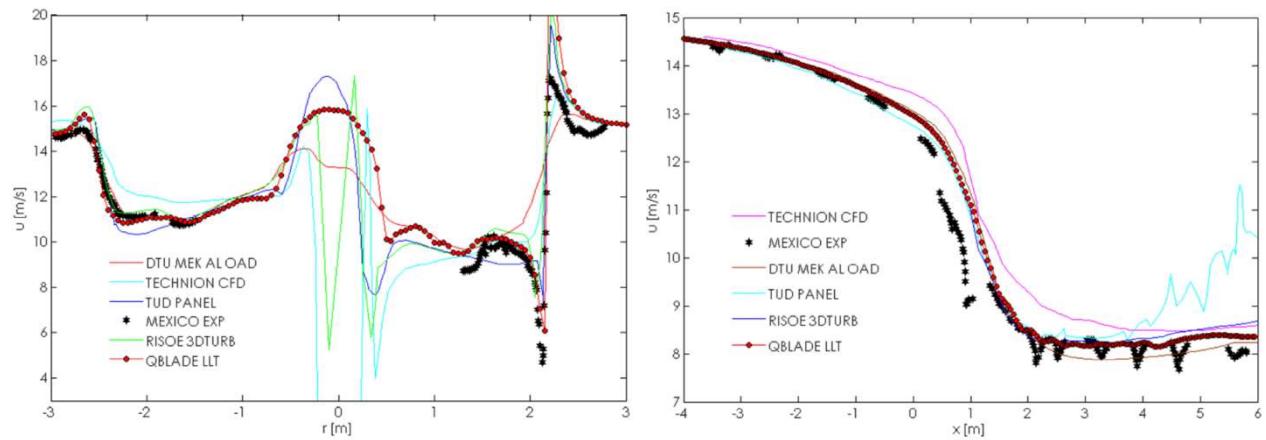


Figure 44. Validation with MexNext-I data from [12]. Left: radial traverse 0.15m behind and parallel to rotor plane showing axial velocity component under 30° yaw; right: axial traverse through rotor plane showing axial-component of velocity under 30° yaw

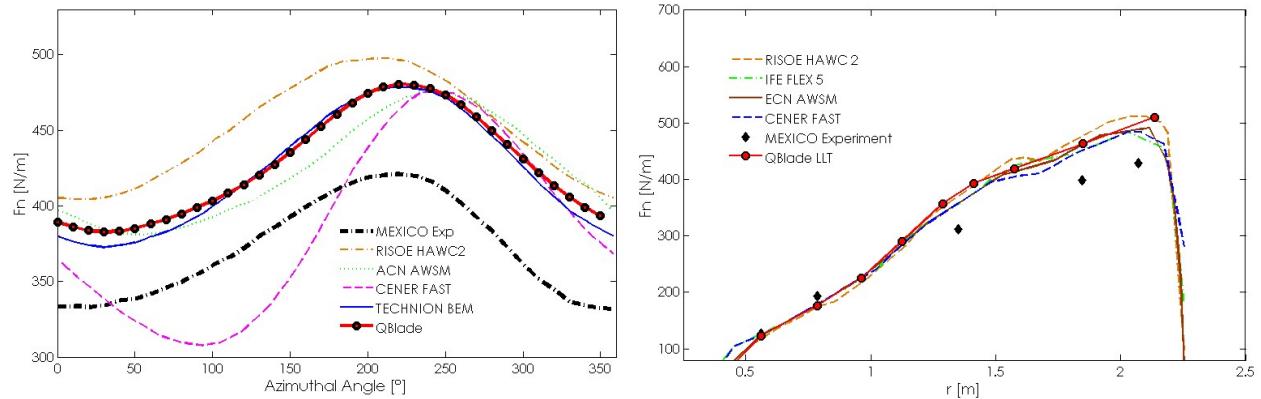


Figure 45. Validation with MexNext-I data from [12]. Left: azimuthal variation of normal force under 30° yaw; right: normal force variation over the blade radius

VAWT

Due to a lack of similar detailed and large scale test cases for VAWT, a thorough validation of the VAWT part of the Lifting Line module has not yet been accomplished. Recently a comparison between QBlade's LLFWV, the double multiple stream tube code *VARDAR* [29] and experimentally validated 2D U-RANS simulations [30] (both performed at the University of Firenze) was carried out and published [31]. The main focus of the investigation was to assess the influence of post stall polar data on VAWT performance predictions, though the results also show surprisingly good agreement between the different methods. What could also be shown in this study is the importance of the unsteady aerodynamics model and the dynamic stall model for VAWT simulations. The comparisons are based on a generic one bladed H-Darrieus rotor; more information is detailed in [31].

When comparing Figure 46 and Figure 47 it can be seen that, by using the unsteady aerodynamics model, the modeled lift overshoot (between 60° and 150° azimuthal angle) leads to a much better matching between the exemplary torque coefficient profile at TSR=2.8, computed with the URANS and the LLFWV simulation. Due to the unsteady aerodynamics model the predicted power coefficient is slightly increased for low TSR and slightly increased for high TSR, leading to an overall better prediction of the power coefficient in the whole TSR range.

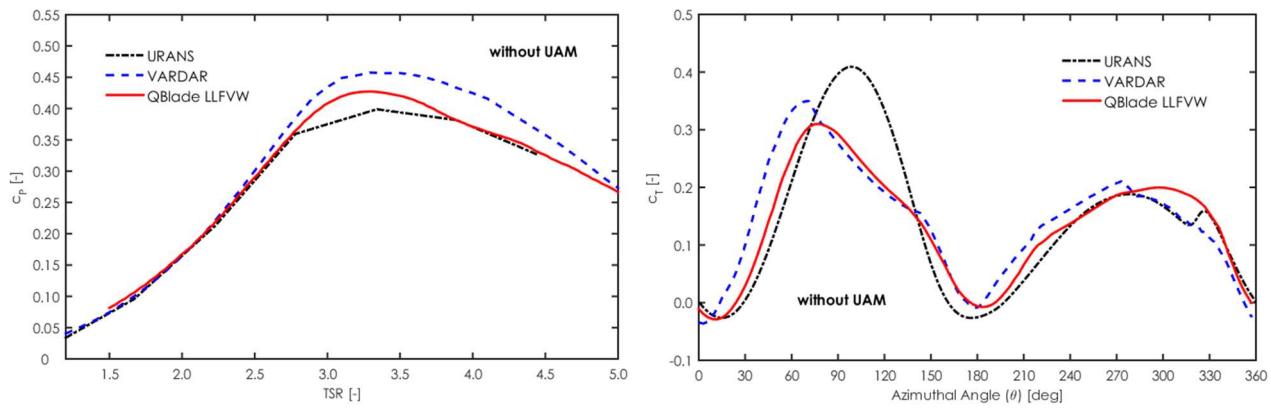


Figure 46. Comparison of averaged power coefficient (left) and torque coefficient profile (right) at TSR=2.8; without unsteady aerodynamics model

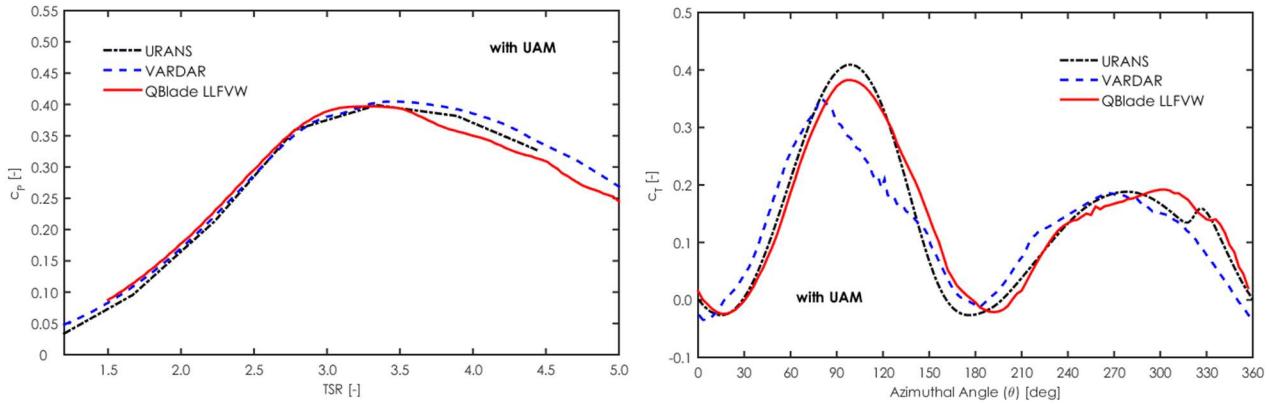


Figure 47. Comparison of averaged power coefficient (left) and torque coefficient profile (right) at TSR=2.8; using unsteady aerodynamics model

Bibliography

- [1] D. Marten, M. Lennie, G. Pechlivanoglou, C. N. Nayeri and C. O. Paschereit, "Implementation, Optimization and Validation of a Nonlinear Lifting Line Free Vortex Wake Module Within the Wind Turbine Simulation Code QBlade," *Journal of Engineering for Gas Turbines and Power*, December 2015.
- [2] J. Schepers, "Engineering models in wind energy aerodynamics," TU Delft, Netherlands, 2012.
- [3] M. Borg, A. Shires and M. Collu, "Offshore floating vertical axis wind turbines, dynamics modeling state of the art. part I: Aerodynamics," *Renewable and Sustainable Energy Reviews*, 2014.
- [4] A. V. Garrel, "Development of a wind turbine aerodynamics simulation module," ECN, Netherlands, 2003.
- [5] L. Bergami and M. Gaunaa, "ATEFlap Aerodynamic Model, a dynamic stall model including the effects of trailing," DTU, Netherlands, 2012.
- [6] M. H. Hansen, M. Gaunaa and H. Aagaard Madsen, "A Beddoes-Leishman type dynamic stall model in state-space and indicial formulations," DTU, Netherlands, 2004.
- [7] J. Wendler, D. Marten, G. Pechlivanoglou, C. N. Nayeri and C. O. Paschereit, "Implementation and Validation of an Unsteady Aerodynamics Model for Horizontal and Vertical Axis Wind Turbines Within the Simulation Tool QBlade," in *ASME Turbo Expo 2016*, Seoul, South Korea, 2016.
- [8] NREL, "OSU Wind-Tunnel Test Data," 2016. [Online]. Available: https://wind.nrel.gov/airfoils/OSU_data/data/. [Accessed 10 May 2016].
- [9] S. Wagner, R. Bareiß and G. Guidati, *Wind Turbine Noise*, Berlin: Springer, 1996.
- [10] T. F. Brooks, S. Pope and M. A. Marcolini, "Airfoil Self-Noise and Prediction," NASA, Langley, 1989.
- [11] J. Saab and M. Pimenta, "Displacement Thickness Evaluation for BPM-Type Airfoil-TE Noise Prediction Model," in *6th International Conference on Wind Turbine Noise*, Glasgow, 2015.
- [12] J. Y. Saab, M. M. Pimenta, L. R. Piqueira, R. M. Augusto, D. Marten, G. Pechlivanoglou, C. N. Nayeri, C. O. Paschereit and N. Moesus, "Verification and Validation of the Airfoil Trailing-Edge Noise Prediction Module (PNoise) inside QBlade V0.95," POLI USP; TU Berlin, Sao Paulo; Berlin, 2016.
- [13] B. Petitjean, R. Drobietz and K. Kinzie, "Wind Turbine Noise Mitigation Technologies," in *Forth International Meeting on Wind Turbine Noise*, Rome, 2011.

- [14] Kitware Inc., VTK User's Guide: Install, Use and Extend The Visualization Toolkit, Columbia, 2010.
- [15] A. Utkarsh, The ParaView Guide: A Paraleel Visualization Application, Kitware, 2016.
- [16] P. J. Moriarty and A. C. Hansen, "AeroDyn Theory Manual," NREL, USA, 2005.
- [17] M. Kaltschmitt, W. Streicher and A. Wiese, Renewable energy: technology, economics, and environment, Springer, 2007.
- [18] H. Snel and J. G. Schepers, "Joint Investigation of Dynamic Inflow Effects and Implementation of an Engineering Method," ECN, Netherlands, 1995.
- [19] C. Bak, H. Aagaard Madsen and J. Johansen, "Influence from blade-tower interaction on fatigue loads and dynamics (poster)," in *Proceedings of Wind Energy for the New Millennium*, 2001.
- [20] M. J. Bhagwat and G. J. Leishman, "Stability, Consistency and Convergence of Time-Marching Free-Vortex Rotor Wake Algorithms," *Journal of the American Helicopter Society*, 2001.
- [21] T. Sant, "Improving BEM-based aerodynamic models in wind turbine design codes," TU Delft, Netherlands, 2007.
- [22] M. P. Scully and J. P. Sullivan, "Helicopter Rotor Wake Geometry and Airloads and Development of Laser Doppler Velocimeter for Use in Helicopter Rotor Wakes," Massachusetts Institute of Technology, USA, 1972.
- [23] P. G. Saffman, M. J. Ablowitz, J. Hinch and J. R. Ockendon, Vortex dynamics, UK: Cambridge University Press, 1992.
- [24] J. G. Schepers, "Final report of IEA Task 29, Mexnext (Phase 1), Analysis of Mexico wind tunnel measurements," ECN, Netherlands, 2012.
- [25] J. G. Schepers, K. Boorsma and X. Munduate, "Final Results from the Mexnext-I: Analysis of detailed aerodynamic measurements on a 4.5m diameter rotor placed in the large German Dutch Wind Tunnel DNW," in *The Science of Making Torque from Wind*, Oldenburg, Germany, 2012.
- [26] H. Snel, J. G. Schepers and N. Siccama, "Mexico Project: the database and results of data processing and interpretation," in *47th AIAA Aerospace Sciences Meeting*, USA, 2009.
- [27] D. Simms, S. Schreck, M. Hand and L. J. Fingersh, "NREL Unsteady Aerodynamics Experiment in the NASA-Ames Wind Tunnel : A Comparison of Predictions to Measurements NREL Unsteady Aerodynamics Experiment in the NASA-Ames Wind Tunnel : A Comparison of Predictions to Measurements," NREL, USA, 2001.

- [28] S. Mc Tavish, D. Feszty and F. Nitzsche, "Aeroelastic Simulations of the NREL Phase VI Wind Turbine using a Discrete Vortex Method," in *Canadian Aeronautics and Space Institute AERO Conference*, Canada, 2009.
- [29] A. Bianchini, G. Ferrara and L. Ferrari, "Design Guidelines for H-Darrieus Wind Turbines: Optimization of the Annual Energy Yield," *Energy Conversion and Management*, vol. 89, pp. 690-707, 2015.
- [30] F. Balduzzi, A. Bianchini, R. Maleci, G. Ferrara and L. Ferrari, "Critical issues in the CFD simulation of Darrieus Wind Turbines," *Renewable Energy*, vol. 85, pp. 419-435, 2016.
- [31] D. Marten, G. Pechlivanoglou, C. N. Nayeri, C. O. Paschereit, A. Bianchini, F. Balduzzi, G. Ferrara and L. Ferrari, "Effects of Airfoils' Polar Data in the Stall Region on the Estimation of Darrieus Wind Turbine Performance," in *ASME Turbo Expo 2016*, Seoul, South Korea, 2016.

List of QBlade Publications

QBlade general:

- D. Marten, G. Pechlivanoglou, C.N. Nayeri, C.O. Paschereit, "Integration of a WT Blade Design Tool in XFLR5", Proceedings of DEWEK 2010, Bremen, Germany, 2010
- D. Marten, J. Wendler, G. Pechlivanoglou, C. N. Nayeri, and C. O. Paschereit, "QBlade : An Open Source Tool For Design And Simulation Of Horizontal And Vertical Axis Wind Turbines," IJETAE, 3 special issue, pp. 264–269, 2013
- D. Marten, J. Wendler, G. Pechlivanoglou, C.N. Nayeri, C.O. Paschereit, "Development and application of a simulation tool for vertical and horizontal axis wind turbines," in Proc. of ASME Turbo Expo, GT2013-94979, 2013
- M. Lennie, D. Marten, G. Pechlivanoglou, C.N. Nayeri, C.O. Paschereit, „ Development and Validation of a Modal Analysis Code for Wind Turbine Blades”, in Proc. of ASME Turbo Expo 2014, GT2014-27151, 2014

Implementation and Application of the Lifting Line Free Vortex Wake Model:

- D. Marten, M. Lennie, G. Pechlivanoglou, C.N. Nayeri, C.O. Paschereit, „Implementation, Optimization and Validation of a Nonlinear Lifting Line Free Vortex Wake Module Within the Wind Turbine Simulation Code QBlade”, Journal of Engineering for Gas Turbines and Power 138(7), December 2015
- D. Marten, G. Pechlivanoglou, C.N. Nayeri, C.O. Paschereit: Integration of an Unsteady Nonlinear Lifting Line Free Vortex Wake Algorithm in a Wind Turbine Design Framework. EWEA 2015 Annual Meeting, Paris, France, 2015

- D. Marten, G. Pechlivanoglou, C.N. Nayeri, C.O. Paschereit, „ Nonlinear Lifting Line Theory Applied to Vertical Axis Wind Turbines: Development of a Practical Design Tool”, in Proc. of the ISROMAC 2016, Honolulu, Hawaii, USA, 2016
- D. Marten, G. Pechlivanoglou, C.N. Nayeri, C.O. Paschereit, A. Bianchini, F. Balduzzi, G. Ferrara, L. Ferrari: Effects of Airfoils' Polar Data in the Stall Region on the Estimation of Darrieus Wind Turbine Performance, ASME Turbo Expo 2016, June 13- 17, ASME paper GT2016-56685, Seoul, South Korea
- J. Saverin, D. Marten, G. Pechlivanoglou, C.N. Nayeri, C.O. Paschereit: Coupling of an Unsteady Lifting Line Free Vortex Wake Code to the Aeroelastic HAWT Simulation Suite FAST, ASME Turbo Expo 2016, June 13- 17, ASME paper GT2016-56290, Seoul, South Korea
- J. Saverin, J. Wendler, D. Marten, G. Pechlivanoglou, C.N. Nayeri, C.O. Paschereit: Aeroelastic simulation of multi-MW wind turbines using a free vortex model coupled to a geometrically exact beam model, TORQUE 2016, October 5-7, Munich, Germany
- M. Lennie, D. Marten, G. Pechlivanoglou, C.N. Nayeri, C.O. Paschereit: Modern Methods to Investigate the Stability of a Pitching Floating Platform Wind Turbine, TORQUE 2016, October 5-7, Munich, Germany

Dynamic Stall:

- J. Wendler, D. Marten, G. Pechlivanoglou, C.N. Nayeri, C.O. Paschereit: Implementation and Validation of an Unsteady Aerodynamics Model for Horizontal and Vertical Axis Wind Turbines Within the Simulation Tool QBlade, ASME Turbo Expo 2016, June 13- 17, ASME paper GT2016-57184, Seoul, South Korea

Application:

- S. Niether, B. Bobusch, D. Marten, G. Pechlivanoglou, C.N. Nayeri, C.O. Paschereit, „Development of a Fluidic Actuator for Adaptive Flow Control on a Thick Wind Turbine Airfoil”, Journal of Turbomachinery 06/2015, DOI:10.1115/1.4028654, 2015
- D. Marten, H. Spiegelberg, G. Pechlivanoglou, C.N. Nayeri, C.O. Paschereit, C. Tropea, Configuration and Numerical Investigation of the Adaptive Camber Airfoil as Passive Load Alleviation Mechanism for Wind Turbines, 33rd AIAA Applied Aerodynamics Conference 2015, Dallas, Texas, 2015
- S. Vey, D. Marten, G. Pechlivanoglou, C.N. Nayeri, C.O. Paschereit: Experimental and Numerical Investigations of a Small Research Wind Turbine, 33rd AIAA Applied Aerodynamics Conference, Dallas, Texas, 2015

List of Figures

Figure 1. Illustration of blade and wake modeling with the LLFWVW algorithm.....	3
Figure 2. Flowchart of one time step loop of the implemented lifting line algorithm	4
Figure 3. The dynamic stall hysteresis loop.....	6
Figure 4. Validation of the UAM with OSU test data (circles) [8]; left: varying mean AoA; right: varying dimensionless frequency.....	7
Figure 5. Sources of airfoil self-noise. Image created after [9]	8
Figure 6. A-weighted 1/3 octave band spectrum from a modern, large WT. Contributions of the relevant flow induced airfoil self-noise mechanisms, reproduced after [9]	9
Figure 7. QBlade v0.9 Main Toolbar.....	9
Figure 8. Control Bar of the Lifting Line Module	10
Figure 9. 3D View of the LLFWVW Module.....	11
Figure 10. Cut plane dialog and newly created cut plane, upstream of the rotor parallel to the rotor plane	12
Figure 11. Velocity distribution in the rotor plane; Top row: X-, Y-, Z-component; bottom row: Axial-, Tang-, Radial-component	13
Figure 12. Exemplary velocity distribution in VAWT H-rotor mid (equilibrium) plane, from left to right: X-, Y- and Z-component	14
Figure 13. Exemplary velocity distribution in horizontal mid-plane; from top to bottom: X-, Y-, and Z-component	14
Figure 14. The “Export Vel Field” button and the export dialog	15
Figure 15. ParaView visualization of streamlines and the vorticity isosurface of a VAWT wake, colored by the velocity magnitude; helix blades of the rotor shown in red	16
Figure 16. Graph Options(left) and Graph context menu (right)	16
Figure 17. Graph View of the LLFWVW Module	17
Figure 18. Visualization of the different parts of the polar decomposition.....	18
Figure 19. The “Graph Settings” menu.....	18
Figure 20. The “360 Polar Extrapolation Module” GUI in QBlade showing attached and separated lift and the separation function f	19
Figure 21. 1 st tab of LLFWVW simulation setup dialog	20
Figure 22. 2 nd tab of LLFWVW simulation setup dialog	21
Figure 23. Definition of platform angles and coordinate system in QBlade	23
Figure 24. Turbulent wind field marched through domain under Taylor’s frozen turbulence hypothesis	25
Figure 25. Definition of the horizontal inflow and yaw angle; top down view	26
Figure 26. Definition of the rotor teeter and pre-cone angle; side view	27
Figure 27. Definition of X- and Y- roll angles for VAWT	27
Figure 28. Definition of azimuthal angles for HAWT (left) and VAWT (right) rotors	28
Figure 29. Vortex system mirrored at the ground plane; showing velocity distribution in the rotor mid-plane from side view	30
Figure 30. Schematic view of the tower shadow model, overlaid with velocity distribution from QBlade calculation, drawing reproduced from [5]	30
Figure 31. Different tower heights for VAWT rotors.....	31

Figure 32. Simplified wake structure.....	31
Figure 33: Example of wake zones and parameters with exemplary values given.....	32
Figure 34. Examples for different “Max Wake Age” values for a VAWT; from top left: 2 revs, 4 revs, 6 revs	34
Figure 35. HAWT Case: The effect of maximum wake age on accuracy (Cp Betz value was used as integral value to compare accuracy)	34
Figure 36. VAWT case: The effect of maximum wake age on the accuracy of the power coefficient.....	35
Figure 37. HAWT case: The effect of full wake age on the accuracy of the power coefficient.....	35
Figure 38. Illustration of the implemented approach for the concentration of trailing vorticity.....	37
Figure 39. An example for the VAWT wake reduction - top: full wake; middle: 35% reduced; bottom: 70% reduced.....	38
Figure 40. The effect of wake reduction on accuracy (Mean Cp was used as integral value to compare accuracy)	38
Figure 41. Schematic view of the influence of the core size on the tangential velocity induced by a vortex element	40
Figure 42. Schematic view of the influence of Sc and $\delta\nu$ on the vortex core size.....	40
Figure 43. Validation with PHASE VI data and the SmartRotor code from [23]. Left: Power output for different yaw cases; right: power curve.....	46
Figure 44. Validation with MexNext-I data from [12]. Left: radial traverse 0.15m behind and parallel to rotor plane showing axial velocity component under 30° yaw; right: axial traverse through rotor plane showing axial-component of velocity under 30° yaw	46
Figure 45. Validation with MexNext-I data from [12]. Left: azimuthal variation of normal force under 30° yaw; right: normal force variation over the blade radius	46
Figure 46. Comparison of averaged power coefficient (left) and torque coefficient profile (right) at TSR=2.8; without unsteady aerodynamics model.....	47
Figure 47. Comparison of averaged power coefficient (left) and torque coefficient profile (right) at TSR=2.8; using unsteady aerodynamics model.....	47

List of Tables

Table 1. QBlade velocity cut plane export format.....	12
Table 2. Definition of polar decomposition parameters.....	19
Table 3. Sample QBlade Simulation file, to be copy/pasted in a text file	23
Table 4. Sample AeroDyn type hub-height wind file, to be copy/pasted in a text file	24
Table 5: Exemplary values of the Hellmann exponent.....	26
Table 6. Rotor Output (Global) Variables.....	43
Table 7. Blade Output Variables.....	44