Original Article

WIND ENGINEERING

Wind Engineering
1–25

DOI: 10.1177/0309524X16636318
wie.sagepub.com
SAGE

Development and application of a grid generation tool for aerodynamic simulations of wind turbines

Hamid Rahimi[1], Elia Daniele[2], Bernhard Stoevesandt[2] and Joachim Peinke[1,2]

Abstract

This manuscript presents an effort towards the introduction of a new grid generation tool for computational fluid dynamics (CFD) simulations of wind turbines. The tool was developed and designed to considerably reduce the duration and complexity of the grid generation process and, moreover, to enable users to perform an efficient CFD simulation for a complete wind turbine rotor. Furthermore, the tool is fully automatized, thus enabling the user to refine the grid along the desired direction. For the purpose of validation, the quality of the generated grid was checked via a grid independence test. Finally, the grid generator tool was evaluated by means of a series of simulations of the NREL phase VI rotor using a three-dimensional CFD Reynolds averaged Navier–Stokes method. The capability of CFD simulations based on the generated grid in capturing aerodynamic performances (such as the power) and the detailed flow characteristics (e.g. the surface pressure distributions) were investigated under various conditions. The obtained results demonstrate the quality and reliability of the developed grid generation tool.

Keywords

Wind turbines, blade aerodynamics, OpenFOAM, CFD, grid generation

Introduction

Wind energy is one of the largest and fastest growing resources in the field of renewable energies. Predictions indicate that wind energy will provide more than 20% of the global electricity supply by 2030 (Council, 2009). From a design point of view, the aerodynamic flow behavior around the blades is a dominant factor for the development of a more efficient wind turbine. Any advance in the understanding of the aerodynamic behavior of wind turbine blades can help, therefore, in reducing the uncertainty on loads, which leads to lower specific energy costs (optimal light weight blade design).

Computational fluid dynamics (CFD) have the potential to improve our understanding of the flow behavior, especially in the case of highly separated flow around a complex geometry. CFD solutions require discretization of the fluid domain, i.e. the generation of a numerical grid. This is a crucial point that can be much more time-consuming than the simulation of the wind turbine itself (Hughes et al., 2005).

Generating a proper grid could be regarded as an even harder task if the geometry to be discretized for the simulation presents a complicated shape, such as that of a wind turbine blade. Therefore, the generation of the grid has been considered as one of the most time consuming parts of CFD applications, due to the lack of fully automatic grid generation tools (Tendulkar et al., 2011). Thus, it can be observed that the grid quality has a significant role on the convergence rate, the

[1]ForWind, University of Oldenburg, Germany
[2]Fraunhofer IWES, Oldenburg, Germany

Corresponding author:
Hamid Rahimi, ForWind, University of Oldenburg, Ammerländer Heerstr. 136, 26129 Oldenburg, Germany.
Emails: hamid.rahimi@forwind.de; hamid.rahimi@uni-oldenburg.de

accuracy of the solution and the CPU time needed for a simulation. Besides that, to generate an appropriate grid there are several key points that must be considered, because they strongly affect the solution. Among these, the most important are: the skewness, the grid density, the adjacent cell length/volume ratio, the boundary layer discretization and the adaptive grid refinement (if available).

There are three different categories for classifying numerical grids: structured, unstructured, and hybrid. Assuming an equal number of cells, and provided that the grid lines are aligned with the flow direction, hexahedral meshes can lead to more accurate solutions than an unstructured grid (Bakker, 2002).

For the generation of structured grids, several methods have been proposed and tested over the years, such as medial surface subdivision (Price et al., 1995), H-morph (Owen and Saigal, 2000), plastering (Blacker and Meyers, 1993), grid-based (Schneiders, 1996), and feature-based (Liu and Gadh, 1997). All these methods have been widely investigated, exploring some of their drawbacks, such as the automation algorithm which leads to a bad quality grid if the complexity level of the geometry rises. However the block decomposition approach is still the most commonly used. The main idea behind this approach is the decomposition of the domain in several regions, each suited for structured grid generation. The boundary layer, which is close to the surface, should be captured with a good quality grid. To be more specific, close to the aerodynamic surface, block topology should be presented by quadrilateral blocks to achieve the goal of maximum similarity to the boundary layer, thus requiring the use of a structured grid.

In the CFD application a number of specialized software have been developed for grid generation (such as ICEM, GAMBIT, PointWise and OpenFOAM-utility blockMesh or snappyHexMesh), although they suffer from some disadvantages. For example, the drawback of the OpenFOAM-snappyHexMesh tool is the difficulty of quickly producing a grid of high quality, especially close to sharp edges, such as the trailing edge of an airfoil section. Furthermore, only a trial and error procedure could provide the accurate capture of the boundary layer. Another example is the OpenFOAM-blockMesh utility, which can also create structured meshes, allowing grid generation and manipulation for simple geometries; however its use is limited when dealing with complex geometries, where steep curves and kinks are present. Hence the construction of a large number of blocks for complex geometries is an elaborate task that should be automated.

In general, the generation of structured meshes requires a long time, because the user needs to distribute vertices and blocks around the CAD geometry. This is in most cases achieved by means of a general purpose macro or script integrated into the grid generation software. However, these always rely on a previous generation of CAD geometry, constituting the basis for three-dimensional (3D) grid generation. Therefore, a grid generator was developed and designed to enable users to perform efficient CFD simulations of the complete wind turbine rotor. This considerably reduces the duration and complexity of the grid generation process.

The grid generation tool presented in this work is designed to be connected with the OpenFOAM library (OpenCFD, 2015), an open-source CFD package with a collection of modifiable libraries written in C++, along with numerous mathematical models and tools to simulate different flow problems.

As will be presented in this paper, the current grid generation tool has the following features:

- complete automation for applicable geometries;
- input table based on the BEM tool input, i.e. the one used in load simulation as a research/industry standard;
- boundary layer refinement;
- overall high cell quality;
- y^+ adjustment based on Reynolds of each section along the blade;
- fast execution time.

Method

In a 3D coordinate system the blade is defined by set of airfoils along the span axis as shown in Figure 1.

Beside the geometrical shape of each airfoil, additional information is required to represent the shape of blade in a 3D coordinate system, namely:

- section span-wise position;
- section chord length;
- section twist angle;
- section alignment point, x_a / c, a fraction between 0 and 1 indicating which chord-wise position has the blade axis to pass through.

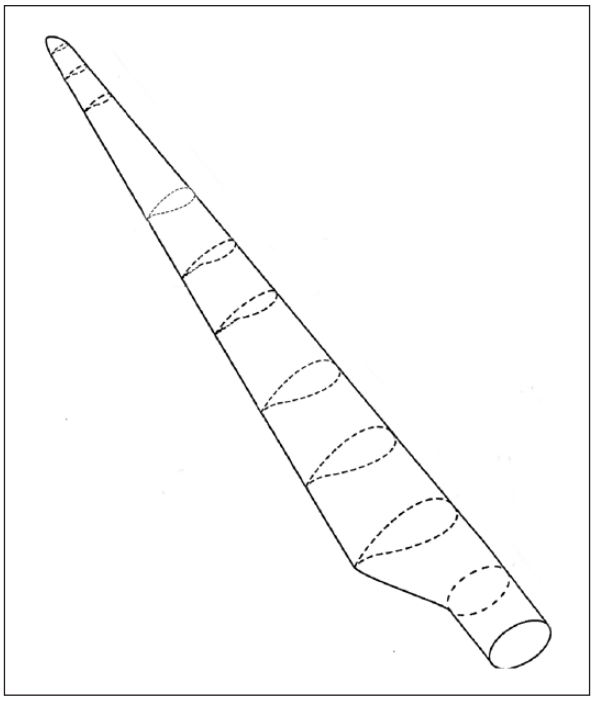

Figure 1. A sample representation of a blade with different airfoils along the span.

Two-dimensional grid generation

After discretization of the blade in a set of airfoils, a related set of a two-dimensional (2D) hexahedral O-grid (an O-shape boundary) is generated for each segment along the blade length. For this purpose, an hyperbolic partial differential equation (PDE) solver is used. The hyperbolic method solver was originally introduced by Steger and Sorenson (1980). The main advantage of this method is that the governing equations need to be solved only once for generating a grid. On the other hand, an exact specification of the outer boundary layer is not possible.

For the generation of the sectional grid several input data are needed:

- radius of the O-grid;
- number of points on the airfoil surface;
- number of points located on the airfoil normal surface;
- non-dimensional wall distance, y^+, from the airfoil surface;
- Reynolds number;
- ratio of clustering points near the trailing edge and the leading edge;
- number of points in the blunt trailing edge.

It should be noted that the velocity may vary from root to tip, especially for a large wind turbine. Therefore, the thickness of the boundary layer for each section along the span can be controlled based on the Reynolds number. Hence, each section should have a different y^+ value.

As a result, the 2D grid generator will read first the chord length and the twist angle. Afterward, it will scale and rotate the geometry of each airfoil accordingly. In hyperbolic grid generation, a grid is generated by moving outwards from the airfoil surface towards the radius of the outer boundary. Hence, hyperbolic grid generation creates the airfoil surface by reading the number of points which should be placed on the airfoil surface. The inner curve is marched outwards in the normal direction from a known number of points to a new number of points. This procedure starts from the airfoil surface. The boundary condition is the grid cell size and grid angle, which follows non-orthogonality conditions. Hence, by considering these constraints the grid can be marched to the next step. This procedure continues with a different number of points and spacings, determined by the number of points in the normal direction, the radius of the O-grid, y^+, and the Reynolds number. The marching continues until the specified number of levels/cells are generated. Finally, point spacings are reapplied to the grid to ensure that the correct value of y^+ is present everywhere. The grid generated with this process is shown in Figure 2.

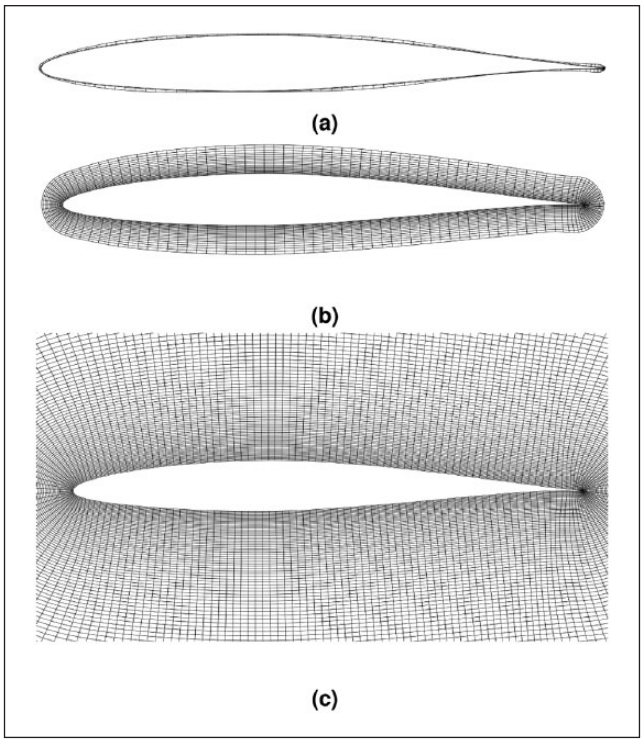

Figure 2. The representation of the generated grid for: (a) the first layer; (b) the first 20 layers; (c) all layers.

So far, for each section along the span of the blade a 2D hexahedaral grid is generated. The overview of this process is shown in Figure 3.

The next step is to merge the 2D structural hexahedral grid of two or more sections in order to reconstruct the blade geometry.

Spline and curve fitting

As mentioned above, in a 3D coordinate system the blade consists of different sectional grids. If these sectional grids are merged together without any further care, it results in an inaccurate representation of the blade surface and also numerical errors. Hence, to obtain accurate numerical results without the need for a large number of cells, it is mandatory to handle the boundary of the disordered or curved domains.

To overcome this problem, a finite difference formula on an arbitrarily spaced grid developed by Fornberg (1988) is used. He described a simple recursion formula which gives the weights for any order of derivative (including the 0th derivative, corresponding to interpolation), approximated to any order of accuracy on an arbitrary grid in one dimension.

The main anticipated application of this method was to dynamically change grids and also to generate tables of weights (such tables are included for the cases of one-sided and centered approximations at a grid point and at a 'half-way point' between grid points). For this purpose, $M > 0$ is the order of the highest derivative to be approximate, $N+1$ is the set of grid points, $\alpha_0, \ldots, \alpha_N$ are the x-coordinates, while the array $\delta_{i,i}^{i}$ and the scalars c_1, c_2 and c_3 are defined inside the algorithm.

Now M has to be approximated for each set of grid points (at $x-$coordinates $\alpha_0, ..., \alpha_N; N > 0$).

However, the problem is to find all the weights such that the approximations defined by

$$\frac{d^m f}{d_{x^m}}|_{x=x_0} \approx \sum_{v=0}^{n} \delta_{n,v}^{m} f(\alpha_v)$$

$$m = 0,1,...,M; n = m, m+1, ..., N \tag{1}$$

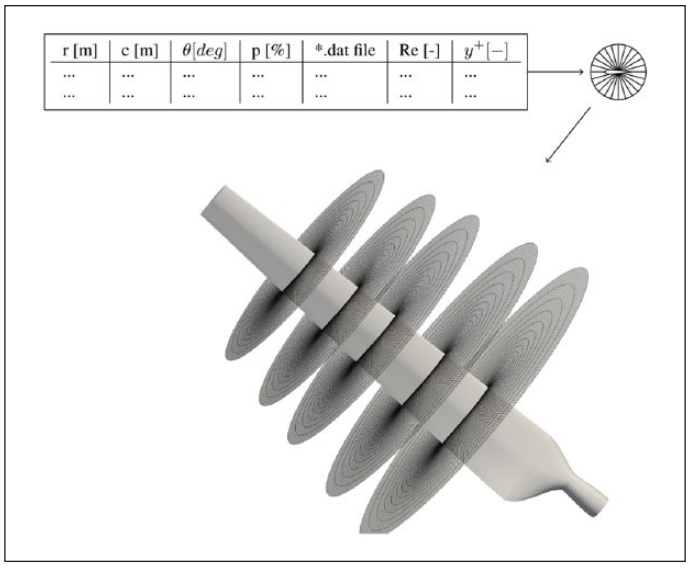

Figure 3. Overview of the process of creation of sectional grids along the span.

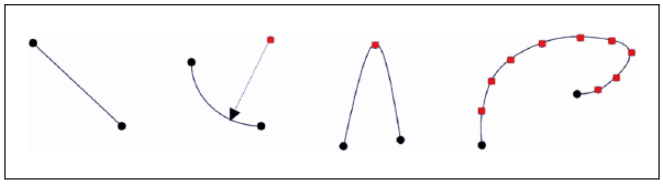

Figure 4. An example of the Forenberg method: generating splines between different points.

becomes of optimal formal order of accuracy (in general of order $n-m+1$, although it can be higher in special cases). The algorithm that is presented in detail in Appendix 1 is able to achieve this task. Figure 4 shows an example of the Fornberg method's application in 1D.

Using this method, the tool generates splines between different sections on the blade. Figure 5 shows the blade surface generated from 2D sectional grids, on the left without curve fitting and on the right after curve fitting with Fornberg's method.

It should be mentioned that the comparison between a CAD geometry and the geometry generated by the aforementioned method indicates no significant discrepancy.

Next, it is necessary to check the perpendicularity of each cell to the surface near the blade, otherwise the cell non-orthogonality of the grid will be too high. This is mainly the case for the root portion, because it usually consists of a cylindroidal shape that is transitioned over a certain length towards an airfoil shape. This portion is difficult to grid smoothly with high quality cells.

Therefore, the cross product of each spline which builds up the 2D grid in each cell and the spline generated on the previous step along the span has to be calculated, and the cells have to be relocated along the direction of the vector that is orthogonal to the surfaces. This goes on until the last layer in the 2D grid is reached. In Figure 6 a set of relocated splines for a few cells are shown.

Tip closure

The tip is critical, as it has a smaller section size than the other airfoil sections along the blade. Therefore, the cell density becomes very large, and producing high quality cells there is not straightforward. By choosing an appropriate block structure, two possibilities for the tip closure based on the initial geometry of the blade are offered: flat and rounded closure at the tip.

Flat closure. The flat tip is structured in a way that reduces the number of small cells and would project the tip grid points over a circular patch enclosed by the latest section O-grid's external radius. The user has the possibility to change an

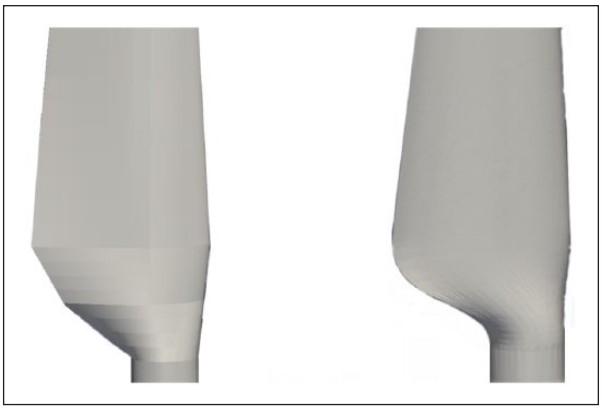

Figure 5. Blade surface generation from a 2D grid. Left: without curve fitting; right: the same case after curve fitting with Fornberg's method.

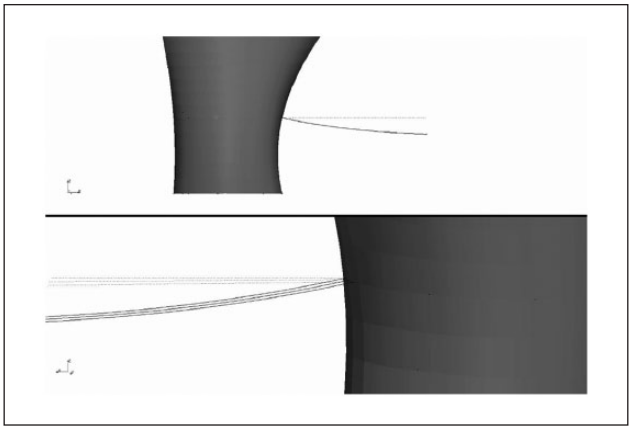

Figure 6. Cell perpendicularity process: dotted line shows the original line passing through the cells; solid line shows the generated splines perpendicular to the surface.

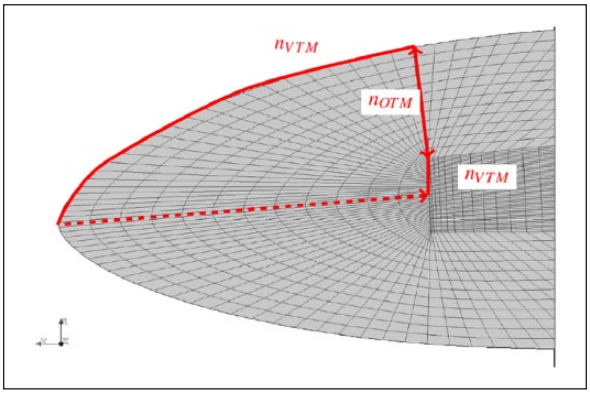

Figure 7. Flat tip parameter setup.

integer indicating the number of cells for the outer tip mask discretization (n_{OTM}). Another integer indicates the number of cells for the vertical (or normal to chord) tip mask discretization (n_{VTM}). These cells are associated with the corresponding airfoil surface cells on both the leading and trailing edge sides, see Figure 7 for an explanation of this parameter. More options are available and can be found in the user manual of the tool.

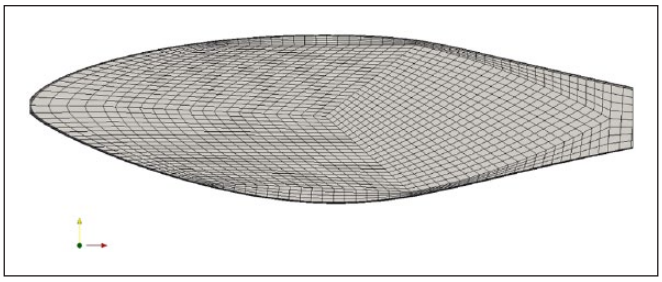

Figure 8. Flat tip grid view of the of the NREL phase VI blade.

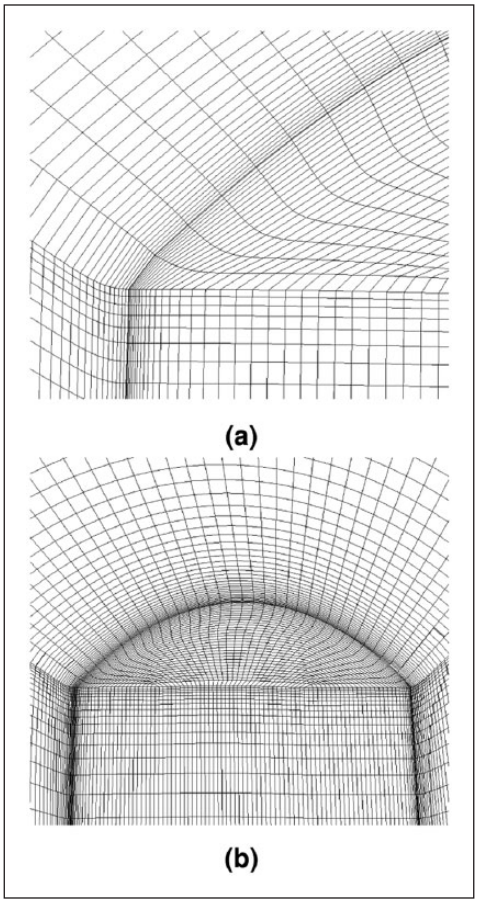

Figure 9. (a) Leading edge detail, and (b) side view of the 3D rounded grid for a slice going thorough the mid plane of the grid for the NREL phase VI blade.

An example of a final flat tip grid is shown in Figure 8.

Rounded closure. For the rounded tip closure, a spline going through the last 3 sections is used to generate the nail- or hill-shaped block structure as indicated in Figure 9.

As is the case of the flat tip, there are two options which give the user the possibility to modify the shape of the tip. First a real value, A_{end} between 0 and 1, indicating the percentage of spline joining the one before the last section and the last section of the blade from where the nail shape block topology starts. See Figure 10 for an explanation of this parameter.

The second option accepts a real value, Ahill, between 0 and 1, indicating the percentage of spline joining the nail shape block topology start point and the tip border edge. This value is applied on four points: the airfoil leading and trailing edges, and the airfoil upper and lower surface midpoints. In these locations the nail shape block vertices would be shifted

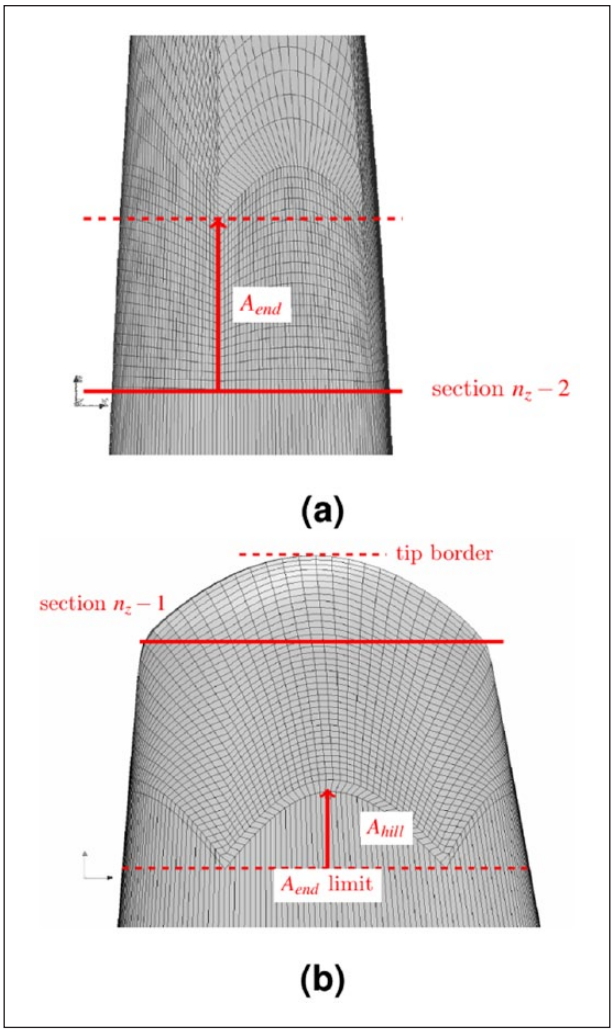

Figure 10. Definition of the parameter introduced by the (a) –tip-part-a-end, and (b) –tip-part-a-hill options.

by a quantity equal to the argument of this option, so that a hill-shaped topology would be realized on the blade surface. See Figure 10 for an explanation of this parameter.

An example of a final rounded tip grid is shown in Figure 11.

Final grid and background mesh

The generated structured grid is shaped as a cylindrical domain with a flat or spherical top surface containing the rotor blade. For the rest of the domain and hub a script has to be used to create a structured grid.

Using the advantage of the symmetrical geometry of the rotor, one could first generate the grid around one blade as well as the nacelle, and then assemble the entire rotor by rotating the resulting grid by an appropriate angle (e.g. 120° for a three-bladed rotor). The output of the developed grid generation tool is a structured grid in form of an OpenFOAM blockMesh dictionary. Examples of the grids which can be generated by this tool are shown in Figures 12 and 13. The time which is required to generate a grid with this tool is in the order of a few hours, including the preparation of the input data. This is far less than the time needed for OpenFOAM snappyHexMesh and commercial tools.

Grid quality check

Finally, the grid quality is checked to test certain geometrical criteria. The method of discretization has a large influence on the accuracy of the CFD simulation. When the simulation volume is subdivided into highly skewed or highly

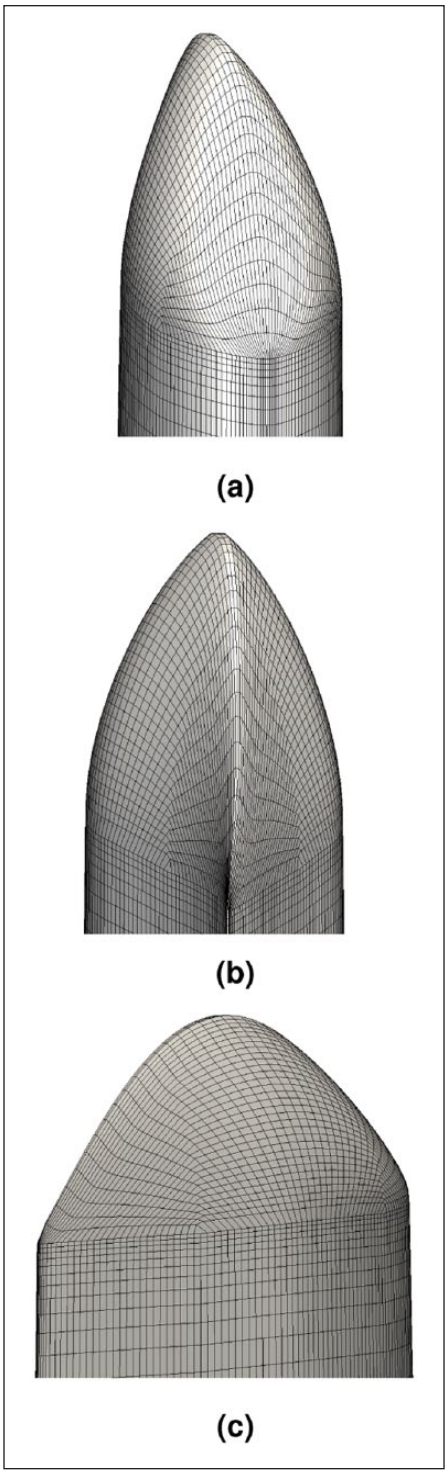

Figure 11. (a) Leading edge detail, (b) trailing edge detail, (c) side view of the 3D rounded grid of the NREL phase VI blade.

nonorthogonal cells, these discretized elements can produce a numerical solution which has significant errors due to the numerical method employed to approximate the differential equations. The properties of the grid (such as the boundary definition, aspect ratio, orthogonality and skewness) all passed this test without any errors. In the following, the major errors, such as the skewness and non-orthogonality, are briefly discussed.

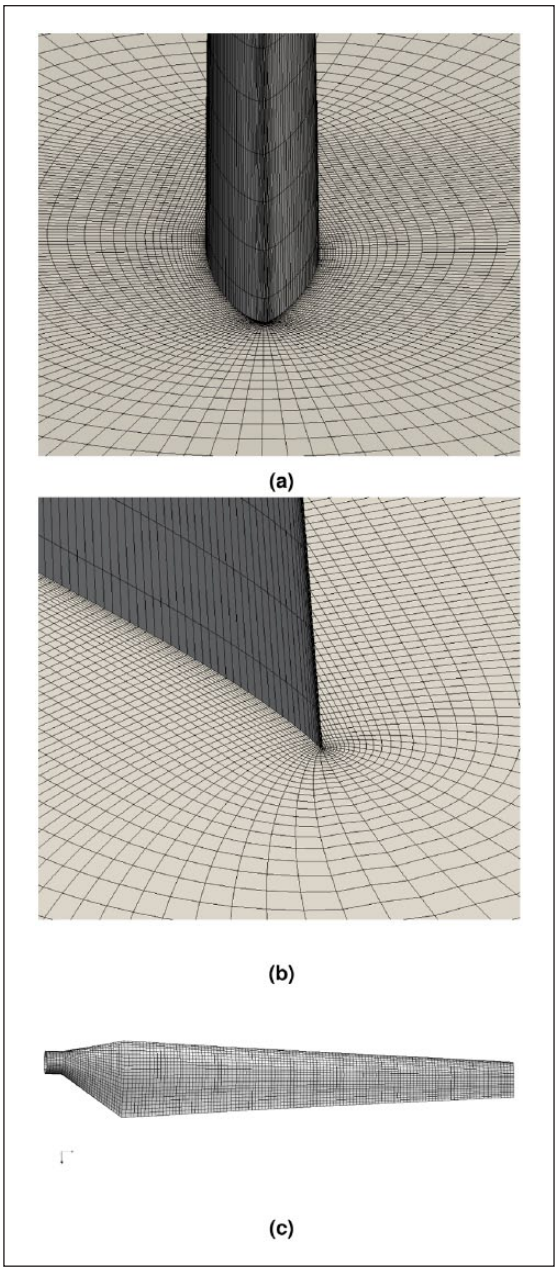

Figure 12. Section-wise (a,b) and side view (c) of the 3D grid of the NREL phase VI blade.

Skewness

The skewness defined in OpenFOAM measures discrepancies between the orientation of the cell-to-cell center vector and the cell-to-face center vector (Rhoads, 2014). The skewness error is computed for each face by taking the distance between adjacent cell centers, and the ratio of the distance between the current face center and the interpolated face center locations.

The skewness has a specific impact for convective derivatives due to the fact that, in the discretization of the convective term, these derivatives require the evaluation on the face to compute the flux towards the adjacent cell (Rhoads, 2014)

$$(V.\nabla)\phi \rightarrow \int_V \nabla.(V\phi)dV \rightarrow \sum_f S_f.(V_f\phi_f) \tag{2}$$

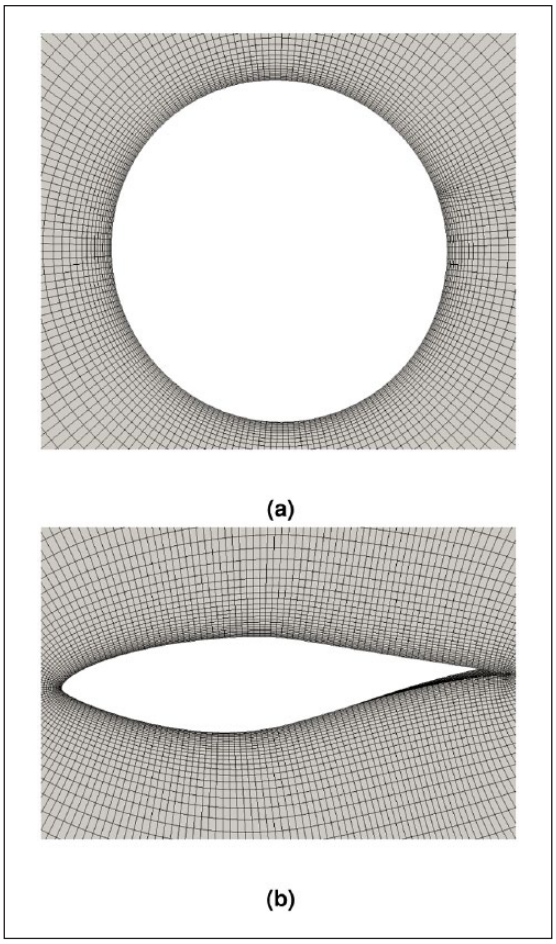

Figure 13. Geometrical transition from a circle (a) to an airfoil shape (b) for the 3D grid of the NREL phase VI blade.

where V is the fluid velocity, and ϕ is the desired variable. The maximum skewness admissible for OpenFOAM is 4.0, and for the grid generated here this value is 1.18, as shown in Figure 14. The distribution of the skewness error for the complete 3D grid in Figure 14 shows that most of the grid faces in the domain have a skewness of less than 0.3.

Non-orthogonality

The grid non-orthogonality can be measured by using the angle between the connecting line of the two cell centers and the normal of their common face.

In OpenFOAM non-orthogonality causes a significant contribution to the error, due to the way that gradients are computed at cell faces (by taking the difference between values at neighboring cell centers and dividing by the appropriate distance). Hence, this results in an approximation for the gradient in the direction of the vector connecting the two cell centers (Rhoads, 2014). However, when a term being computed in the system requires the dot product of the gradient with the face area, this can introduce significant error due to the misaligned face normal and cell-center vectors (Rhoads, 2014).

The parameter NonOrthogonalCorrectors in the OpenFOAM fvSolution dictionary helps in correcting for non-orthogonality, especially in cases where it is above 70°. The grid non-orthogonality average value for the presented grid is about 15°, far less than the critical value. In Figure 15 the visualization of two slices with the maximum non-orthogonality at root and tip is shown, together with the distribution of the non-orthogonality for the complete 3D grid of the NREL phase VI blade. As shown, the higher non-orthogonality is near the root and tip, due to the highest grading on the surface. However, the number of affected cells is still very few compared to the total number of cells.

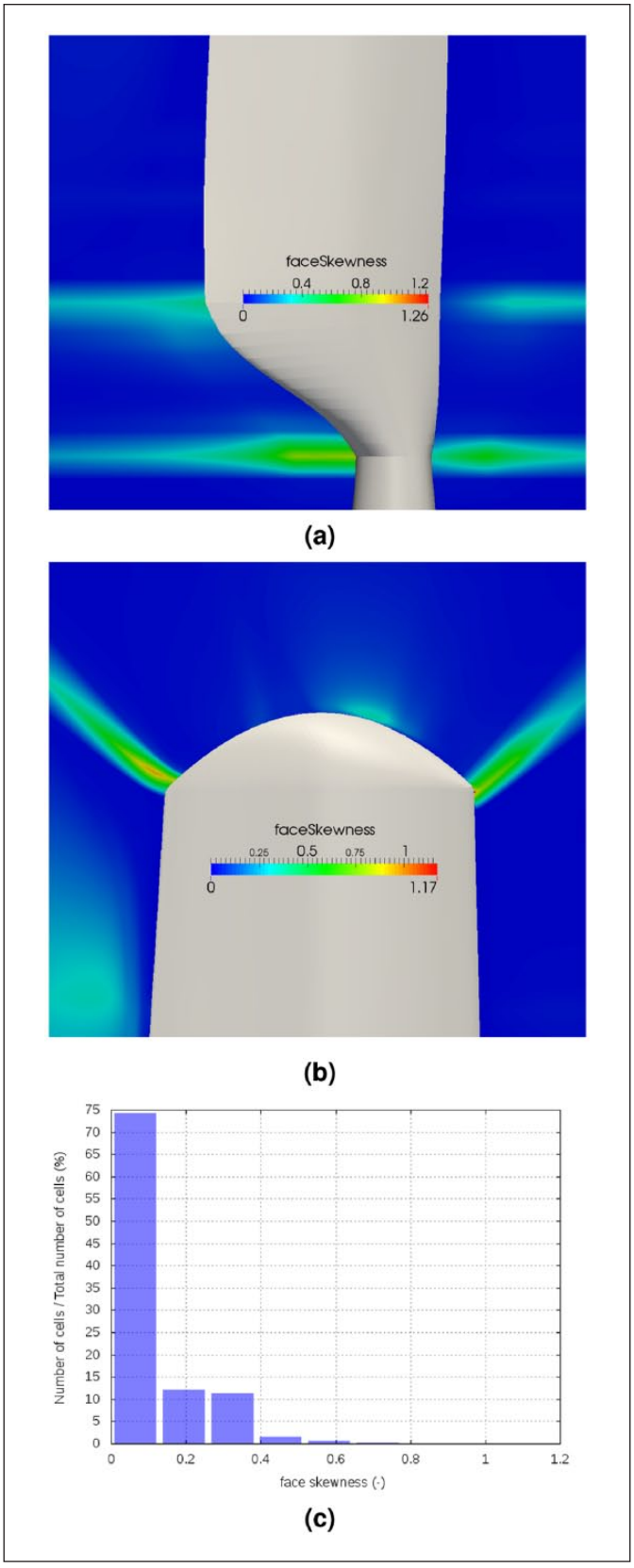

Figure 14. Visualization of two slices with the maximum skewness at the root (a) and tip (b). The distribution of the skewness error for the complete 3D grid of the NREL phase VI blade is shown in (c).

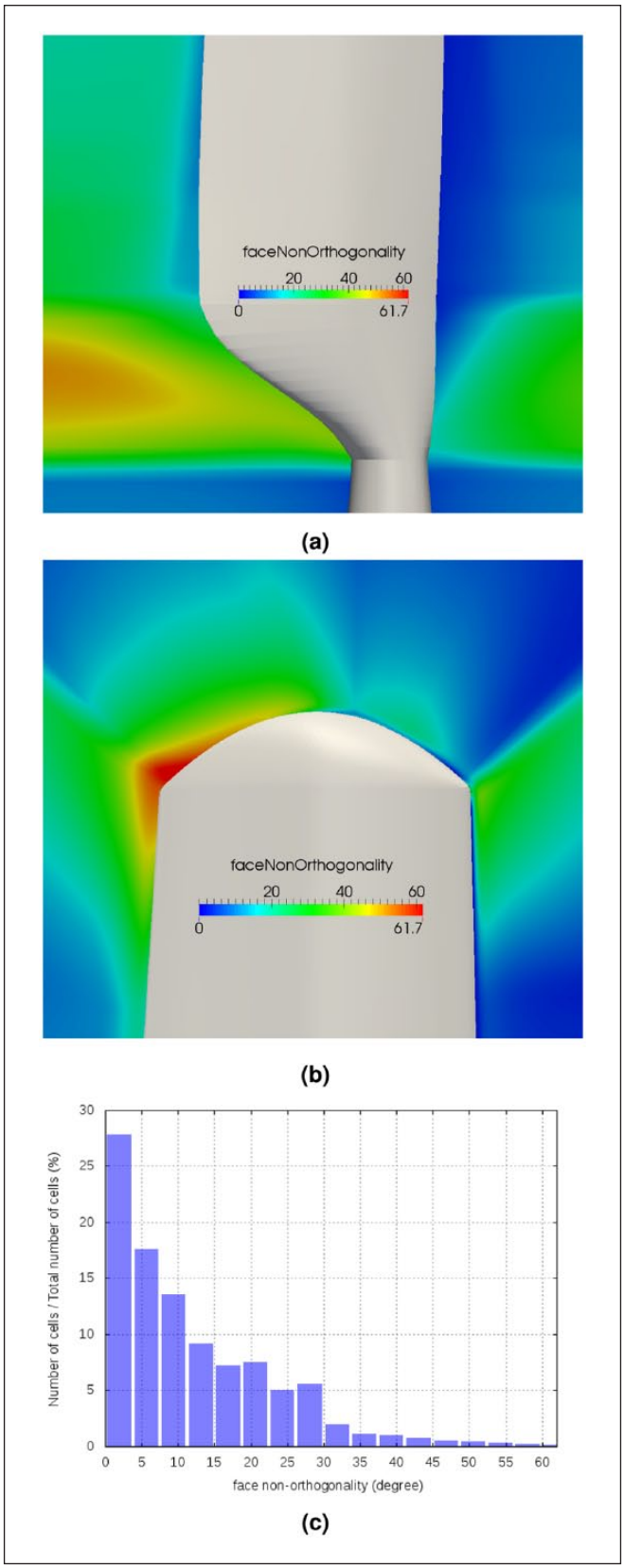

Figure 15. Visualization of two slices having the maximum non-orthogonality at root (a) and tip (b). Distribution of the non-orthogonality for the complete 3D grid of the NREL phase VI blade is shown in (c).

Table 1. Specification of the NREL Phase VI wind turbine (Hand et al., 2001).

Parameter	Value
Number of blades (-)	2
Rotor radius (m)	5.029
Rotational speed (Ω) (rpm)	71.9
Cut-in wind speed (m/s)	6
Rated power (kW)	19.8
Power regulation	Stall
Rotational direction	*CCW*
Pitch angle (deg)	3
Tower height (m)	11.5

Table 2. Operational conditions for the simulations.

Case	Ω (rpm)	*V* (m/s)	Air density (kg/m^3)
1	71.9	7	1.246
2	72.1	10	1.246
3	72.1	13	1.246
4	72.1	15	1.224
5	72.0	20	1.221
6	72.1	25	1.220

Analysis of the grid generation tool: Test case definition

The validation of the simulations using the NREL phase VI wind turbine are conducted in OpenFOAM in order to assess the capabilities of the presented tool. The total aerodynamic blade power, the sectional force and pressure coefficient distribution at different span-wise sections along the blade are compared to the NREL phase VI rotor experimental results (Michelsen, 1992; Mo et al., 2013; Sørensen et al., 2002; Yelmule and VSJ, 2013). In 2000 NREL released the experimental results of the Phase VI wind turbine which was tested in the NASA/Ames 24.4 m × 36.6 m wind tunnel. The NREL VI turbine is stall-regulated, with a rated power of 19.8 kW. The detailed results of the experiment are available on the NREL website (NREL, 2015). For this work, the cone angle of the rotor is 0° and the pitch angle of the blade is 3°. Table 1 provides the technical settings of this wind turbine. The S809 airfoil with 21% thickness, designed by Somers, is used for the blade construction (Somers, 1997). The numerical simulations are performed using an incompressible RANS method. The rotor was in the upwind configuration and the tower influence on the rotor aerodynamics will be neglected. Only one of the blades is simulated in this case, while the second blade is accounted for by means of periodic conditions. The other information needed for CFD computation, including the blade geometry and operational parameters of the rotor (RPM, air density), is listed in Table 1. Table 2 lists all computational runs performed for this test case. The SST turbulence model of Menter and Esch (2001) is used, as it exhibits a good performance for 2D separated flows (Daniele, 2014; Menter, 1992; Wilcox, 1991).

Using the presented structured grid generator a cylindroidal numerical grid around the blade geometry is obtained. The overall domain is a cylindroid with an area close to the cross section of the NASA/AMES wind tunnel. The grid has two main components, which are shown in Figure 16. The first part is the cylindroid that contains the structured grid of the blade. The second part is the rest of the computational domain containing the wake region. These two different grids are connected using the arbitrary mesh interface (AMI) technique available in OpenFOAM (Farrell and Maddison, 2011). The cylindroidal structured grid has about six-million cells while the outer domain has about two million. Moreover, the y^+ value is around one on the entire blade surface.

The test cases have four different boundaries, namely: inlet, outlet, the surrounding domain and the blade. The boundary conditions' configuration is listed in Table 3. Simulations are performed with a single reference frame (SRF) approach with a turbulent intensity of 1%.

To perform the numerical simulations the facility for large-scale computations in wind energy research (FLOW) at the University of Oldenburg (FLOW, 2015) was used, where for each run 96 CPU cores are used. The computations were run for approximately 20,000 iterations, which took 15 hours. Steady simulation were performed, and the convergence of the equation residuals was obtained. The flow is assumed to be incompressible and the solver uses a smoother (smoothSolver

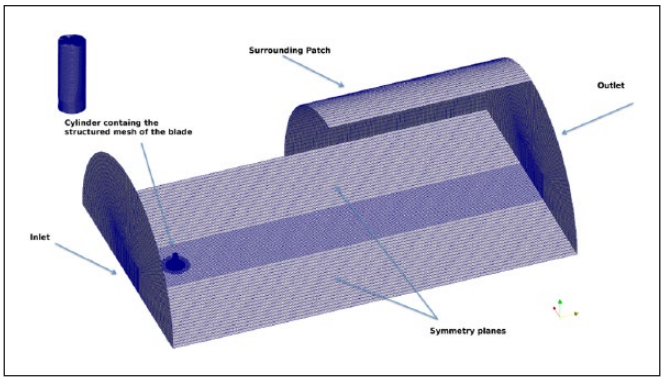

Figure 16. Details of the computational domain used for wind turbine simulation.

Table 3. Boundary conditions for the computational simulations using the $k-\omega$ SST model. I is the turbulence intensity and ν is the kinematic viscosity. The boundary conditions for k and ω are from Menter and Esch (2001).

Field\Boundary	Inlet	Outlet	Blade	Surrounding
k	Dirichlet uniform $k = \frac{3}{2}(u_\infty I)^2$	Neumann	Wall function	slip
ω	Dirichlet uniform $\omega = \frac{k}{10\nu}$	Neumann	Wall function uniform $\omega = \frac{k}{10\nu}$	slip
Pressure	Neumann	Dirichlet	Neumann	slip
Velocity	Dirichlet uniform (0 u_∞ 0)	Neumann	Dirichlet	slip

in OpenFOAM's lexicon) for the pressure and velocity equations. The second-order upwind scheme was adopted to discretize the convective terms. A grid independence test was performed and is presented in more details in the next section.

Grid independence test

Different grids were built up in order to examine the grid-independent behavior of the solution for both span-wise and chord-wise direction for further refinement. The integral aerodynamic blade power and pressure coefficient curves for a wind speed of 10 m/s are used for this purpose. Two different methods are used, namely grid systematic refinement and the grid convergence index (GCI). For all the grids used for this purpose, the structured grid near the blade was changed, while the background grid size was kept.

Grid convergence index

The concept of the grid convergence index (GCI) was proposed by Roache (1994) and derives from the general Richardson extrapolation method. This method was used in CFD application in Celik et al. (2008) and Neittaanmäki et al. (2004). The GCI is a measure of the numerical error produced by a numerical scheme based on calculations for different grid sizes. In this case, grid M_2 was systematically refined from grid M_3 with a factor of $r_{32} = 1.2750$, and analogously grid M_1 from M_2 with a factor of $r_{21} = 1.4059$. Table 4 shows the results of the application of the GCI procedure. Based on these calculations, the GCI for the two finest grids was less than 1% for both monitored variables (integral aerodynamic blade power and thrust). As the refinement from M_2 to M_1 merely produced a 0.17% discretization error, but increased the computational cost by a factor of two, the M_2 (M_{GCI}) grid was used as an input for seeking a grid independent solution in the following section.

Table 4. Calculations of discretization error with GCI. M_{1-3} are the number of cells for each case. r_{21} and r_{32} are the refinement factors of grids 2 to 1 and 3 to 2 respectively. $\Phi :=$ is the monitored variable for the GCI method, namely, power and thrust for the three different grids ($\Phi_{M_{1-3}}$).

Parameter	$\Phi :=$ Power (kW)	$\Phi :=$ Thrust (kN)
M_1 (10^6)	25.7	
M_2 (10^6)	9.2	
M_3 (10^6)	4.4	
$r_{21}(-)$	1.4059	
$r_{32}(-)$	1.2750	
Φ_{M_1}	5.825	27.196
Φ_{M_2}	5.817	27.105
Φ_{M_3}	5.454	27.878
GCI_{fine} (%)	0.1758	0.1112

Table 5. Grid properties in the span-wise direction. Note that Sp_3 is M_2, which are the results of GCI method.

Case	Number of cells (10^6)	Type of refinement
Sp_1	7.2	span-wise
Sp_2	7.9	span-wise
Sp_3	9.2	span-wise

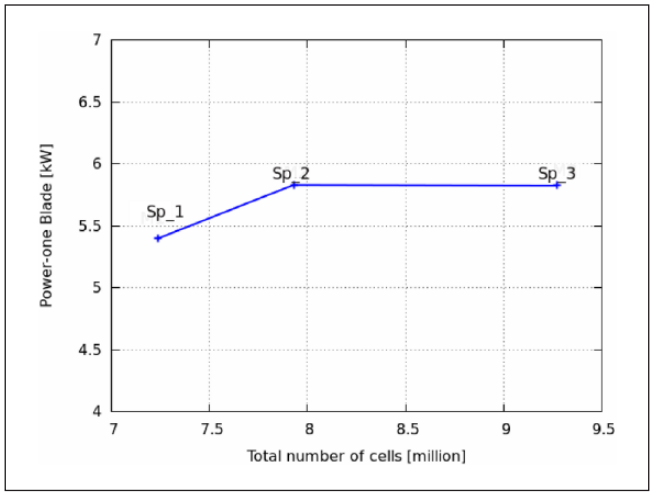

Figure 17. The integral aerodynamic blade power for a wind velocity of 10 m/s. The power convergence was achieved after refinement.

Grid resolution

One of the most common procedures in searching for a grid independent solution is to increase the grid resolution, based on the prior result of the problem, until the convergence for the monitored variable is achieved. For this purpose, the output of the final grid (the M_2 with 9.2×10^6 cells) from the previous section (GCI method) was changed systematically, first by –20% and –40% of the initial span-wise cell number. The grid characteristics are summarized in Table 5.

In Figures 17 and 18 the results of the power and pressure coefficients were found to have different values depending on the span-wise cell size. The convergence monitoring quantities (integral aerodynamic blade power and sectional pressure coefficient distribution) were achieved during grid refinement.

After checking the grid independence in the span-wise direction, grid Sp_2 was used as an input to check the convergence in the chord-wise direction by ±20% of the initial grid Sp_2 chord-wise cell number. The grid characteristics are summarized in Table 6.

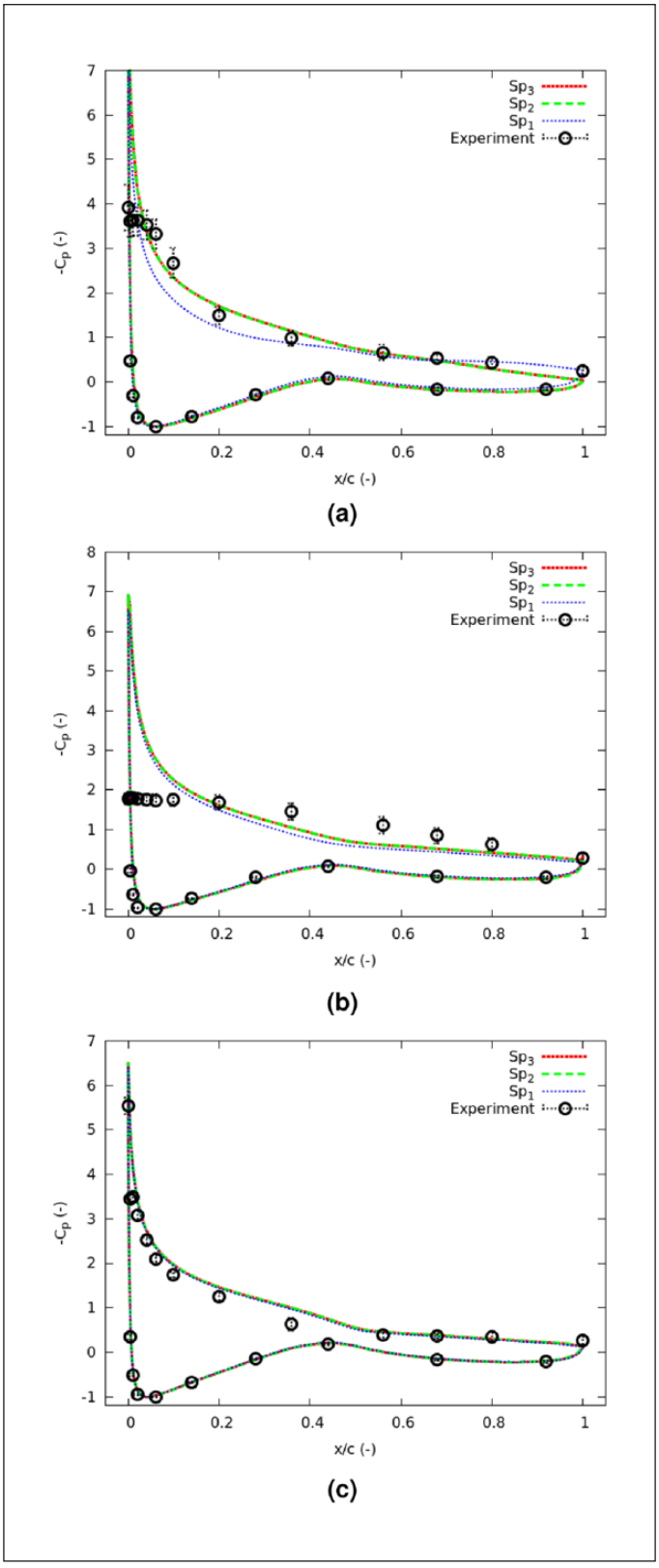

Figure 18. Pressure distributions at 10 m/s wind speed for several span-wise sections: open circles represent measurement data ± the standard deviation and grid refinement cases for the span-wise direction. (a) r / R = 30%. (b) r / R = 47%. (c) r / R = 63%.

Table 6. Grid properties in the chord-wise direction.

Case	Number of cells (10^6)	Type of refinement
Ch_1	7.03	chord-wise
Ch_2	7.93	chord-wise
Ch_3	9.28	chord-wise

The pressure coefficient curve presented in Figure 19 shows the convergence under grid refinement when changing the chord-wise cell size using the span cell size achieved from the previous calculation. It should be noted that the discrepancies of the simulation and experimental results at locations $r/R = 30$ and $r/R = 47$ persisted. From Figure 19, it is clear that between grid Ch_2 and Ch_3, the results are comprised within a small difference. Therefore, Ch_2 was selected as a final grid for all the simulations.

Results and discussion

For the purpose of validation the power produced by the wind turbine (integral load) and the pressure distributions at different span-wise sections along the blade (sectional force) are compared with the experimental results. In all plots measurements are shown with ± the standard deviations to indicate the variation over one revolution. The standard deviation of the measurements comes from several sources, including the disturbance from tower passage, dynamic loads, etc.

Integrated load

Figure 20 shows the comparison of the mechanical power computed from the CFD simulations and that extracted from the NREL experiments for different wind speeds. The results show a similar behavior. The $k-\omega$ SST turbulence model reproduces the power as a function of the incoming wind speeds. Moreover, there is an acceptable agreement for higher values of wind speed, where the turbine is in a stall-regulated condition.

Several studies of the aerodynamics behavior of NREL VI have been presented, where different software, either with a structured or unstructured grid, was used. Sørensen et al. (2002) used a structured grid within the Ellipsys3D CFD code (Michelsen, 1992) with 6 million cells and the $k-\omega$ SST turbulence model. In general, the experimental behavior was well captured, although the stall initiation at 10 m/s wind speed was not matched. Yelmule and VSJ (2013) used ANSYS CFX (CFX, 2015) and a structured grid, for which the number of grid cells used for the simulation was not reported. However, the results are in good agreement with experimental data. Nevertheless, it should be noted that the stall prediction at 10 m/s is not captured (red line in Figure 20). Mo and Lee (2012) applied a structured grid in ANSYS Fluent (Fluent, 2015) with 3 million cells. The computed power is in good agreement with the experimental measurements, except for the wind speed of 25 m/s (gray line in Figure 20). Song and Perot (2015) used an unstructured grid with OpenFOAM and reported good agreement with experimental results for wind speeds lower than 10 meter per second. For wind speeds higher than 10 m/s larger differences were observed between the simulations and experiments (green line in Figure 20). In the aforementioned works it is stated that the differences in the higher wind speed regime is probably due to the limitations of the RANS model in describing the stall behavior. However in this manuscript, the CFD simulations are able to reproduce with good agreement the behavior of the turbine, even for higher wind speeds.

Since the mechanical power is a quantity obtained by integration over the area of the blade, it is sensitive to span-wise compensation errors. Hence, to check the capability of the simulation, based on the proposed grid generator, in order to correctly predict the aerodynamic behavior of the wind turbine, it is necessary to evaluate the chord-wise pressure distribution along several sections.

Sectional force

Reasonably good agreement is observed when comparing the sectional force coefficients (normal and tangential) along the blade span-wise direction with experiments. The coefficients are calculated for five different span-wise locations, as shown in Figures 21 and 22.

The stall is present at 10 m/s up to the mid-span of the blade, where a deviation of the numerical calculation from the experimental value is observed in Figure 22a. At this wind speed value, the flow undergoes an extreme instability. When the wind speed increases, the stall area spreads, as predicted in Figure 22 by the extension of the tangential force coefficient. This can be explained considering the accuracy issue of many turbulence models when dealing with the flow separation (Sørensen et al., 2002).

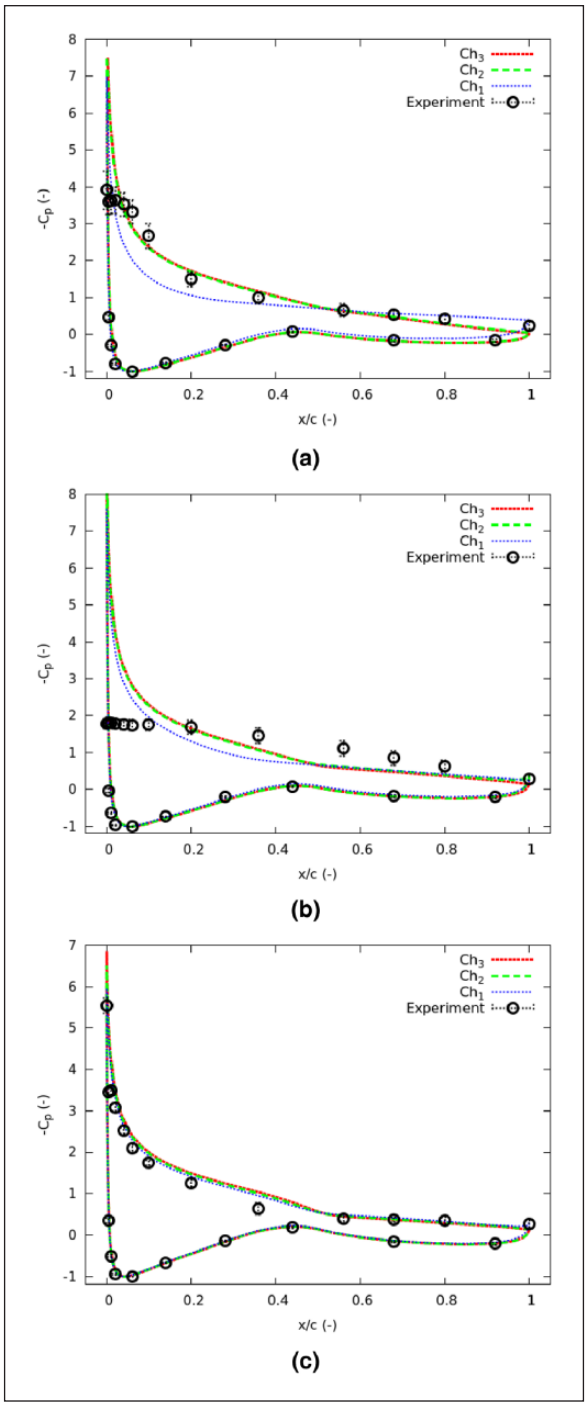

Figure 19. Pressure distributions at a wind speed of 10 m/s for several span-wise sections: open circles represent measurement data ± the standard deviation and grid refinement cases for the chord-wise direction. (a) r / R = 30%. (b) r / R = 47%. (c) r / R = 63%.

Pressure Distributions

In this section, the comparison between experimental and numerical pressure distribution is shown for the sections located at r / R =30, 47, 63, 80 and 95% along the span-wise direction. The stagnation point pressure values are used to normalize the pressure distribution. For the wind speed values of 7 and 10 m/s, good agreement is observed for all sections, as shown in Figures 23 and 24.

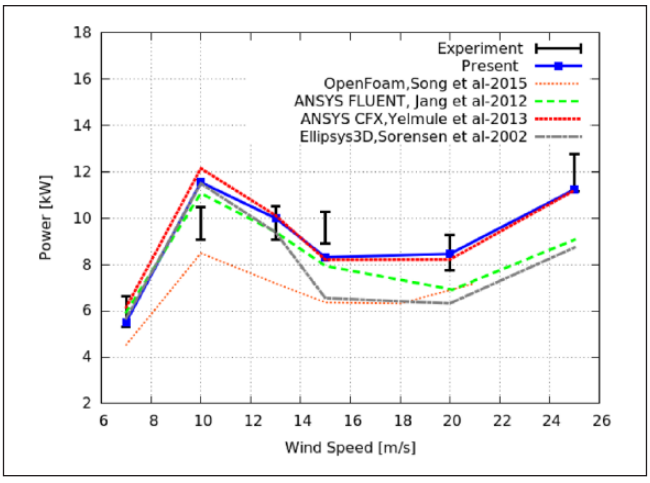

Figure 20. Distribution of the power for experimental results (± the standard deviation) and CFD simulations.

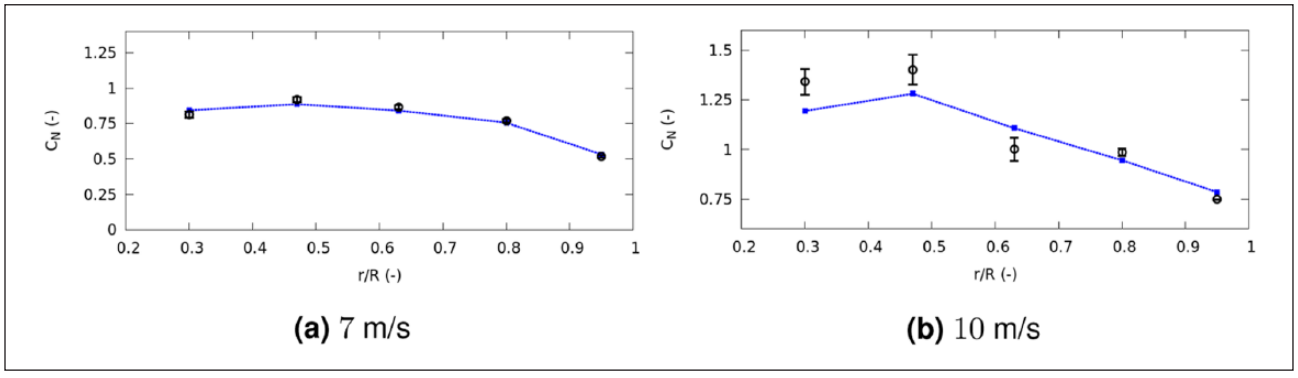

Figure 21. Span-wise distribution of normal force coefficients for different wind speeds: open circles represent measurement data ± the standard deviation, and full curves represent the computational data.

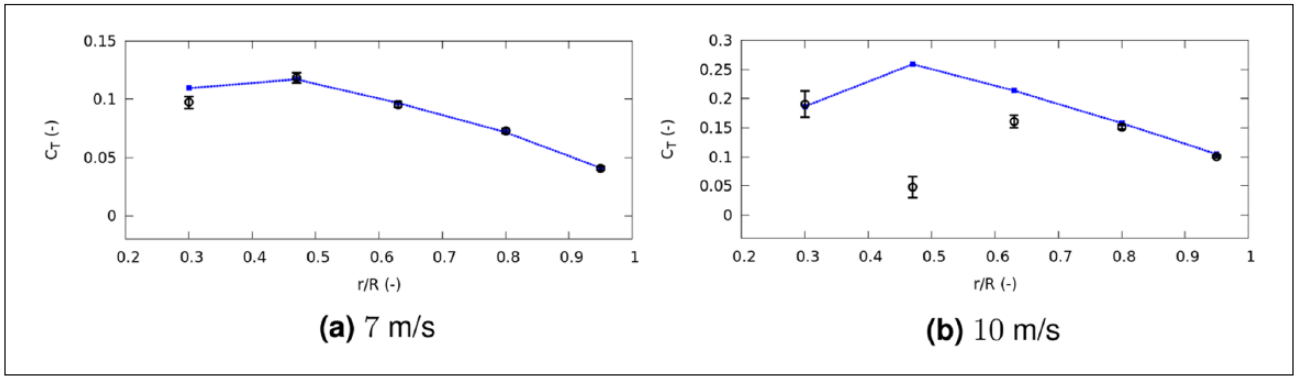

Figure 22. Span-wise distribution of tangential force coefficients for different wind speeds: open circles represent measurement data ± the standard deviation, and full curves represent the computational data.

Conclusions

An automatized grid generation tool for wind turbine blade CFD simulation has been presented. This grid generator can be used for the fast and reliable investigation of 3D rotor aerodynamics and blade load calculations. The quality of the grid and its effects on the results were checked by a test-matrix of cases for the NREL Phase VI rotor undergoing upwind

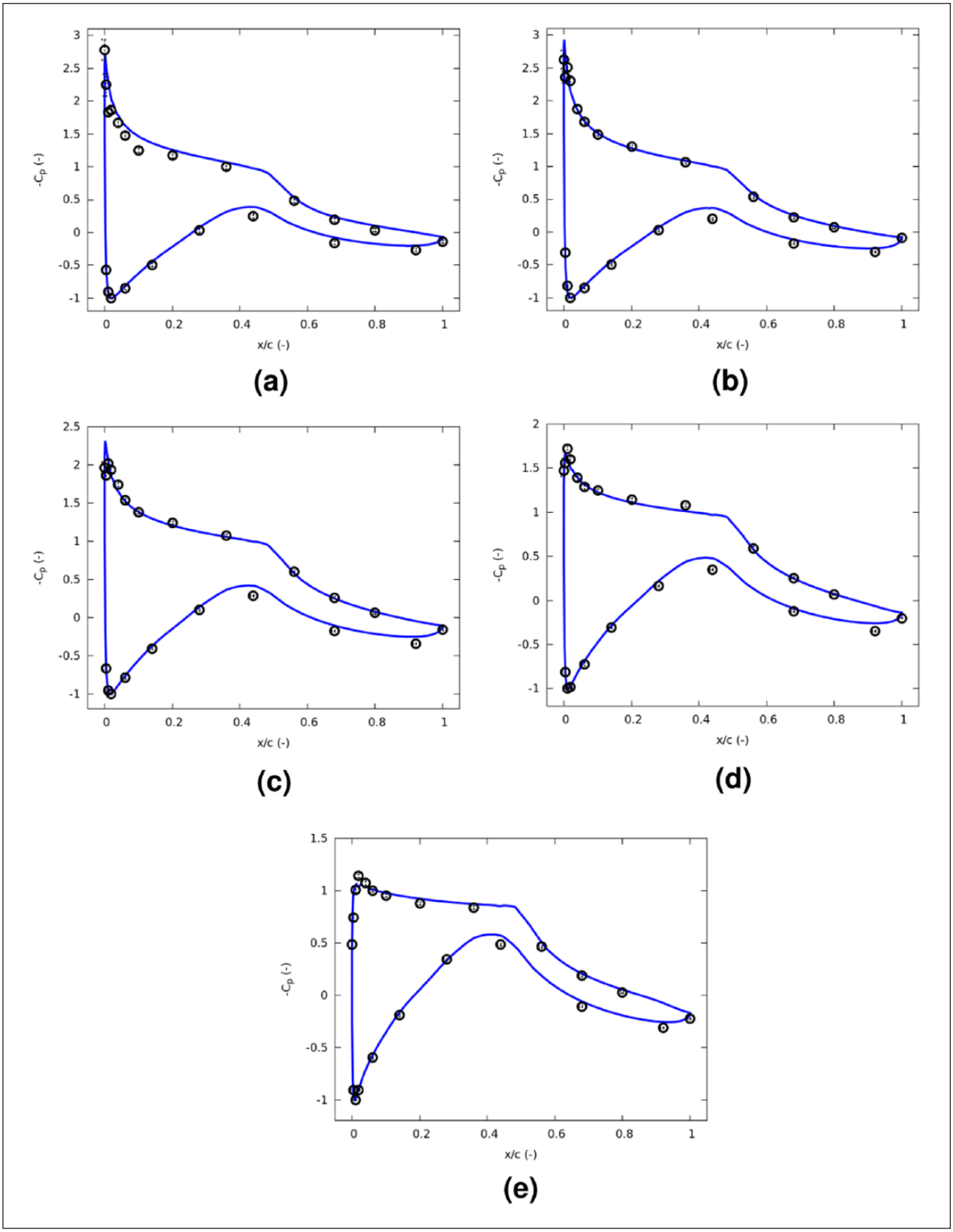

Figure 23. Pressure distributions at 7 m/s wind speed: open circles represent measurement data ± the standard deviation, and full curves represent the numerical result. (a) $r / R = 30\%$. (b) $r / R = 47\%$. (c) $r / R = 63\%$. (d) $r / R = 80\%$. (e) $r / R = 95\%$.

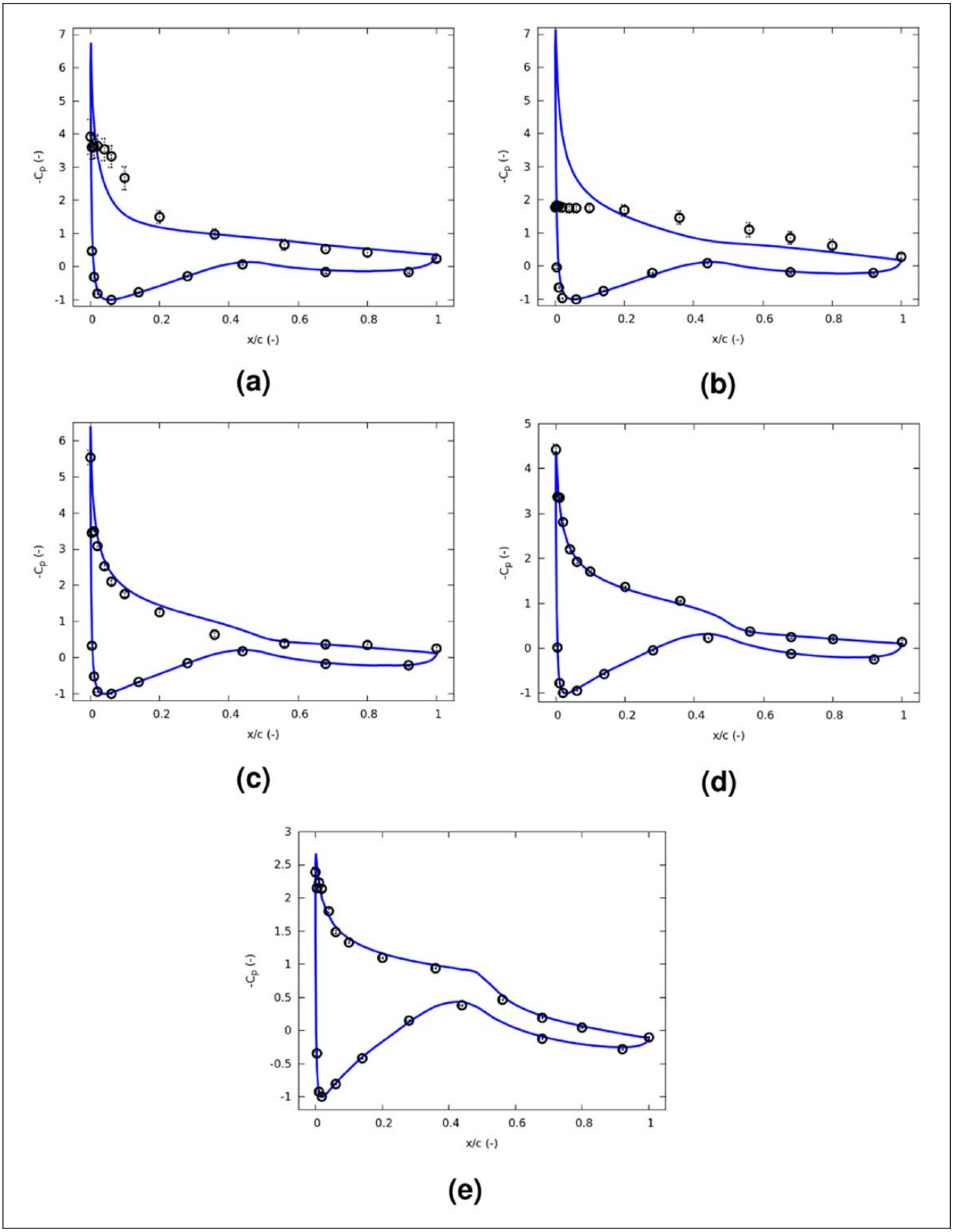

Figure 24. Pressure distribution at 10 m/s wind speed: open circles represent measurement data ± the standard deviation and full curves represent the numerical result. (a) $r / R = 30\%$. (b) $r / R = 47\%$. (c) $r / R = 63\%$. (d) $r / R = 80\%$. (e) $r / R = 95\%$.

operating conditions. A grid independence test was conducted for these cases. The CFD results were compared with the measurements from a wind tunnel and other CFD tools with different types of grids.

Results show that the CFD simulations performed using the grid generated via the presented tool exhibit no significant difference to the results obtained by other CFD tools in a similar condition. In general, good matching of the experimental data trend is shown, including the power and pressure coefficients at different sections along the blade. An advantage of the presented grid generation tool is the ability to easily produce a structured grid, thus significantly reducing the time for generating the grid when compared to the snappyHexMesh utility in OpenFOAM.

Taking advantage of the input table based on that of the industry standard load calculation tool, the presented code could significantly reduce the time needed to generate several structured grids along the blade in order to evaluate the effect of different geometrical transitions in the complex region such as root and tip, especially when compared to the basic snappyHexMesh OpenFOAM utility. Further improvements are necessary in order to implement a smoothing function to overcome possible errors in the grid.

The presented tool is available in its beta-version, with no fee to be paid for universities or research centers, assuming that the terms of use of Fraunhofer society are accepted. The users are invited to collaborate as beta-testers/users referring bugs or errors. For more information please contact the corresponding author or Dr Elia Daniele (mailto:elia.daniele@iwes.fraunhofer.de).

Acknowledgements

We would like to thank the computer time provided by the Facility for Large-Scale Computations in Wind Energy Research (FLOW) at the University of Oldenburg and Dr Jonas Schmidt and Fredrik Vestermark of Fraunhofer IWES Oldenburg. Experimental results were provided by Dr Scott Schreck, from the National Renewable Energy Laboratory at Boulder, CO. His kind assistance is deeply appreciated.

Funding

The present work is funded within the framework of the joint project ‘Smart Blades’ (grant number 0325601D) by the German Federal Ministry for Economic Affairs and Energy (BMWi) under the decision of the German Federal Parliament.

References

Bakker A (2002) An introduction to computational fluid dynamics. Available at: http://www.bakker.org/dartmouth06/engs150/07-mesh.pdf (accessed 29 July 2015).

Blacker TD and Meyers RJ (1993) Seams and wedges in plastering: A 3-d hexahedral mesh generation algorithm. *Engineering with Computers* 9(2): 83–93.

Celik IB, Ghia U, Roache PJ, et al. (2008) Procedure for estimation and reporting of uncertainty due to discretization in CFD applications. *Journal of Fluids Engineering - Transactions of the ASME* 130(7): 1–4.

CFX A (2015) Ansys cfx. Available at: http://www.ansys.com/Products/Simulation+Technology/Fluid+Dynamics/Fluid+Dynamics+Products/ANSYS+CFX (accessed 29 July 2015).

Council GWE (2009) Global wind 2009 report. Technical report, GWEC Global Wind Energy Council (GWEC).

Daniele E (2014) 3D steady simulation of a 7.5 mW wind turbine with OpenFOAM. In: *10th European Fluid Mechanics Conference*, Copenhagen, Denmark, September 2014.

Farrell P and Maddison J (2011) Conservative interpolation between volume meshes by local galerkin projection. *Computer Methods in Applied Mechanics and Engineering* 200(1–4): 89–100.

FLOW (2015) The HPC user wiki of the University of Oldenburg. Available at: http://wiki.hpcuser.uni-oldenburg.de/index.php/Main_Page/ (accessed 29 July 2015).

Fluent A (2015) Ansys fluent. Available at: http://www.ansys.com/Products/Simulation+Technology/Fluid+Dynamics/Fluid+Dynamics+Products/ANSYS+Fluent (accessed 29 October 2015).

Fornberg B (1988) Generation of finite difference formulas on arbitrarily spaced grids. *Mathematics of Computation* 51(184): 699–706.

Hand MM, Simms D, Fingersh L, et al. (2001) Unsteady aerodynamics experiment phase vi: wind tunnel test configurations and available data campaigns. *Technical report,* National Renewable Energy Laboratory, USA.

Hughes TJ, Cottrell JA and Bazilevs Y (2005) Isogeometric analysis: Cad, finite elements, nurbs, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering* 194(39): 4135–4195.

Liu SS and Gadh R (1997) Automatic hexahedral mesh generation by recursive convex and swept volume decomposition. In: *6th International Meshing Roundtable, Sandia National Laboratories*, Park City, UT, 13–15 October, pp.217–231.

Menter F and Esch T (2001) Elements of industrial heat transfer predictions. In: *16th Brazilian Congress of Mechanical Engineering*. Available at: http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19930013620.pdf

Menter FR (1992) Performance of popular turbulence models for attached and separated adverse pressure gradient flows. *AIAA Journal* 30(8): 2066–2072.

Michelsen J (1992) Basis3d-a platform for development of multiblock pde solvers. *Technical Report AFM 92-05*, Technical University of Denmark, Lyngby, Denmark.

Mo JO and Lee YH (2012) Cfd investigation on the aerodynamic characteristics of a small-sized wind turbine of nrel phase vi operating with a stall-regulated method. *Journal of Mechanical Science and Technology* 26(1): 81–92.

Mo JO, Choudhry A, Arjomandi M, et al. (2013) Large eddy simulation of the wind turbine wake characteristics in the numerical wind tunnel model. *Journal of Wind Engineering and Industrial Aerodynamics* 112(0): 11–24.

Neittaanmäki P, Rossi T, Majava K, et al. (2004) A verification exercise for two 2-d steady incompressible turbulent flows. *Proceedings of the ECCOMAS*, Jyväskylä, Finland, 24–28 July, pp.1–20.

NREL (2015) Wind research home page. Available at: http://www.nrel.gov/wind/ (accessed 29 July 2015).

OpenCFD (2015) OpenFOAM - the open source computational fluid dynamics (cfd) toolbox. Available at: http://OpenFOAM.com (accessed 29 July 2015).

Owen SJ and Saigal S (2000) H-morph: An indirect approach to advancing front hex meshing. *International Journal for Numerical Methods in Engineering* 49(1–2): 289–312.

Price MA, Armstrong CG and Sabin M (1995) Hexahedral mesh generation by medial surface subdivision: Part i. solids with convex edges. *International Journal for Numerical Methods in Engineering* 38(19): 3335–3359.

Rhoads J (2014) Effects of grid quality on solution accuracy. In: *OpenFOAM workshop 2014*. Available at: http://afinemesh.files.wordpress.com/2014/07/ofw20141.pdf (accessed 1 May 2015).

Roache PJ (1994) Perspective: A method for uniform reporting of grid refinement studies. *Journal of Fluids Engineering* 116(3): 405–413.

Schneiders R (1996) A grid-based algorithm for the generation of hexahedral element meshes. *Engineering with Computers* 12(3-4): 168–177.

Somers D (1997) *Design and experimental results for the S809 airfoil*. NREL. Available at: http://www.osti.gov/scitech/servlets/purl/437668 (accessed 1 May 2015).

Song Y and Perot J (2015) Cfd simulation of the nrel phase vi rotor. *Wind Engineering* 39(3): 299–310.

Sørensen NN, Michelsen J and Schreck S (2002) Navier–stokes predictions of the nrel phase vi rotor in the nasa ames 80 ft × 120 ft wind tunnel. *Wind Energy* 5(2-3): 151–169.

Steger JL and Sorenson RL (1980) Use of hyperbolic partial differential equations to generate body fitted coordinates. *NASA, Langley Research Center Numerical Grid Generation Tech*, pp.463–478 (SEE N 81-14690 05-64).

Tendulkar S, Beall M, Shephard MS, et al. (2011) Parallel mesh generation and adaptation for cad geometries. In: *Proceedings of the NAFEMS world congress*, Boston, MA, 23–26 May, pp.2011–2012.

Wilcox D (1991) A half century historical review of the k-w model. In: *29th Aerospace Sciences Meeting*, AIAA, Reno, NV, 7–10 January 1991, vol. 91, pp.2066–2072.

Yelmule MM and VSJ EA (2013) CFD predictions of NREL phase VI rotor experiments in NASA/AMES wind tunnel. *International Journal of Renewable Energy Research* 3(2): 261–270.

Appendix

The Fornberg (1988) method can be implemented with the following metode. For this purpose: $M > 0$, is the order of the highest derivative, $N+1$ are a set of grid points, $\alpha_0, \ldots, \alpha_N$ are $x-$coordinates and array $\delta_{i,i}^{i}$ and scalar c1, c2 and c3 are defined for calculations inside the algorithm.

$$
\begin{aligned}
&\text{Enter } M, N, x_0, \alpha_0, \alpha_1, \alpha_2, ..., \alpha_N \\
&\delta_{0,0}^{0} := 1 \\
&c1 := 1 \\
&\text{for } n := 1 \text{ to } N \text{ do} \\
&\quad c2 := 1 \\
&\quad \text{for } \nu := 0 \text{ to } n-1 \text{do} \\
&\quad\quad c3 := \alpha_n - \alpha_\nu \\
&\quad\quad c2 := c2.c3 \\
&\quad\quad \text{if } n \leqslant M \text{ then } \delta_{n-1,\nu}^{n} := 0 \\
&\quad\quad \text{for } m := 0 \text{ to } \min(n, M) \text{ do} \qquad \frac{n!}{r!(n-r)!} \\
&\quad\quad\quad \delta_{n,\nu}^{m} := ((\alpha_n - x_0)\delta_{n-1,\nu}^{m} - m\delta_{n-1,\nu}^{m-1} / c3 \\
&\quad\quad\quad \text{next } m \\
&\quad\quad \text{next } \nu \\
&\quad \text{for } m := 0 \text{ to } \min(n, M) \text{ do} \\
&\quad\quad \delta_{n,n}^{m} := \frac{c1}{c2}(m\delta_{n-1,n-1}^{m-1} - (\alpha_{n-1} - x_0)\delta_{n-1,n-1}^{m-1}) \\
&\quad\quad \text{next } m \\
&\quad c1 : c2 \\
&\quad \text{next } n
\end{aligned}
\tag{3}
$$

The order in which α_ν is given is significant, since the weights corresponding to all leading subsets of the α_ν s are calculated. There is no restriction on x_0 coinciding with any α_ν.