Modeling of wind turbine wake with a sliding mesh

by

J.F. van der Auweraert

to obtain the degree of Bachelor of Science
at the Delft University of Technology,

Student number:	4168445	
Project duration:	September 2, 2015 – December 2, 2015	
Thesis committee:	Prof. dr. S. Kenjeres,	TU Delft, supervisor
	BSc. J. Hennen	TU Delft, daytime supervisor
	Prof. dr. R. Mudde	TU Delft
	MSc. C. Haringa	TU Delft

This thesis is confidential and cannot be made public until January 1, 2016.

TU Delft

Abstract

At the Transport Phenomena group of the TU Delft the aim is to solve complex wind farm simulations using in-house code. This in-house code models the forces on the blades of the wind turbines using the Actuator Disk Model (ADM) instead of modeling the rotating blades. This is done, because modeling the forces is computationally less expensive. This approach might not be equally accurate compared to simulations of a wind turbine with rotating blades. Therefore, a wind turbine with rotating blades will be simulated to be compared with results using the forces approximation. This is done using the $k-\varepsilon$ model and a sliding mesh in Ansys Fluent.

To verify the results, wind tunnel measurements are used for comparison. The numerical domain is designed so that it is adequate to simulate the conditions of wind tunnel measurements. A number of different simulations were performed. One simulation was chosen from which the results are presented as the final results. This simulation did not predict the wake most accurate, however it was the only simulation where the results did not change when refining the mesh. Thus, it is the only grid independent simulation and other more accurate predictions of the wake were coincidental. The final results suggest that wind turbine simulations using a sliding mesh and the $k-\varepsilon$ model provide results that are qualitatively correct. In terms of magnitude, the results differ a factor 1 to 1.5 compared to the wind tunnel measurements. This is the result of an overprediction of wake in the final simulations.

In conclusion, wind turbine simulations using a sliding mesh provide qualitatively correct results. However, the final simulation overpredicts the wake. If this problem can be solved in further research this method seems very promising.

Contents

List of Symbols, Operators and Abbreviations

Symbol	**Units**	**Description**
ρ	$\mathrm{kg\,m^{-3}}$	Density
U	$\mathrm{m\,s^{-1}}$	Velocity
$\overline{U}$	$\mathrm{m\,s^{-1}}$	Time-averaged velocity
u'	$\mathrm{m\,s^{-1}}$	Fluctuating velocity component
p	$\mathrm{kg\,m^{-1}\,s^{-2}}$	Pressure
t	s	Time
μ	$\mathrm{kg\,m^{-1}\,s^{-1}}$	Dynamic viscosity
F	$\mathrm{kg\,m\,s^{-2}}$	Force
x	m	Position
k	$\mathrm{m^2\,s^{-2}}$	Turbulence kinetic energy
σ	−	Turbulent prandtl number
ε	$\mathrm{m^2\,s^{-3}}$	Turbulence energy dissipation rate
μ_t	$\mathrm{kg\,m^{-1}\,s^{-1}}$	Turbulent viscosity
L	m	Length
ω	$\mathrm{s^{-1}}$	Angular velocity
r	m	Position vector
d	m	Diameter

Operator	**Description**
∂	Partial derivative
δ	Dirac delta function
∇	Gradient

Abbreviation	**Explanation**
ADM	Actuator Disk Model
UDF	User Defined Function
$RANS$	Reynolds-Averaged Navier-Stokes
CFD	Computational Fluid Dynamics
HPC	High Performance Computing/Computer
GUI	Graphical User Interface
TUI	Text User Interface

1 Introduction

Whilst the global fossil fuel reserve is constantly diminishing the demand on renewable energy technologies is rapidly growing. This demand expresses itself in investments in new technologies, but furthermore emphasizes the importance to optimize current technologies. On the one hand, optimizing current technologies leads to a higher power output, which translates to higher revenues. On the other hand, more efficient renewable energy technologies lead to an even smaller carbon footprint. Making it interesting for both investors and environmentalists alike.

In the wind energy sector these optimizations can be found on the one hand in the design of the wind turbine itself and on the other hand in the placement of wind turbines in a wind farm. The placement of wind turbines in a wind farm is of importance. Each individual wind turbine extracts energy from the surrounding airflow, resulting in a lower-velocity wake behind the turbine. This wake will reduce the power output of nearby wind turbines, which leads to a complex optimization problem regarding the placement of individual wind turbines to ensure maximum power generation over an entire wind farm. Studies on the physics of wind turbine wake have been done [1]. To accurately perform the measurements in these studies requires high end expensive measurement techniques. In addition, it is very hard to control the conditions when performing these measurements. Therefore, in more recent years computational fluid dynamics (CFD) has become a popular tool to study wind turbine wake. Some studies even use CFD to study the energy harvesting in entire wind farms [2].

Using CFD to study the optimization problem regarding energy harvesting in wind farms requires efficient modeling. The wind turbines in these wind farms cannot be modeled using rotating blades, because this is too computationally expensive. At the Transport Phenomena group of the TU Delft the aim is to solve these complex simulations using in-house code [3] [4]. This in-house code models the forces on the blades using the Actuator Disk Model (ADM) instead of modeling the rotating blades. This is done, because modeling the forces is computationally less expensive. This approach might not be equally accurate compared to simulations of a wind turbine with rotating blades. Therefore, a wind turbine with rotating blades will be simulated in Ansys Fluent to be compared with results using the forces approximation.

2 Theory

The simulations done in this study are performed in Ansys Fluent 15.0, a Computational Fluid Dynamics (CFD) software. Ansys Fluent offers a wide range of theoretical models to simulate fluid flow. The theoretical models used for these simulations are the Reynolds-Averaged Navier-Stokes (RANS) equations in combination with the $k-\varepsilon$ model. Therefore, the theoretical framework regarding the RANS equations and the $k-\varepsilon$ model will be described.

2.1. The Reynolds-Averaged Navier-Stokes (RANS) Equations

A popular technique used to simulate turbulent flow in fluids is a set of equations called the Reynolds-Averaged Navier-Stokes (RANS) equations. This technique is based on the Navier-Stokes equations, which describe the conservation of momentum. A general form of the Navier-Stokes equations is the compressible Navier-Stokes momentum equation, which can be written in vector notation as

$$\frac{\partial}{\partial t}\left(\rho\vec{u}\right)+\nabla\cdot\left(\rho\vec{u}\vec{u}\right)=-\nabla p+\nabla\left(\mu\left(\nabla\vec{u}+\nabla\vec{u}^{T}\right)-\frac{2}{3}\mu\left(\nabla\cdot\vec{u}\right)\right)+\vec{F} \tag{2.1}$$

Note that $\vec{F}$ are the external forces applied to the fluid. The momentum equation is always solved together with the continuity equation

$$\frac{\partial\rho}{\partial t}+\nabla\cdot(\rho\vec{u})=0 \tag{2.2}$$

In RANS modeling, the variables are decomposed into time-averaged and fluctuating components. For example, when x is the streamwise direction the streamwise velocity would be given as

$$U_x=\overline{U_x}+u'_x \tag{2.3}$$

More generally, all variables can be expressed as

$$\Phi=\overline{\Phi}+\phi' \tag{2.4}$$

Where ϕ is an arbitrary flow variable. Then, by substituting expressions in the form of equation (2.4) for the flow variables in the continuity equation (2.2) and the momentum equation (2.1) the following set of equations are obtained [5].

$$\frac{\partial\rho}{\partial t}+\frac{\partial}{\partial x_i}(\rho U_i)=0 \tag{2.5}$$

$$\frac{\partial}{\partial t}(\rho U_i)+\frac{\partial}{\partial x_j}(\rho U_i U_j)=-\frac{\partial p}{\partial x_i}+\frac{\partial}{\partial x_j}\left[\mu\left(\frac{\partial U_i}{\partial x_j}+\frac{\partial U_j}{\partial x_i}-\frac{2}{3}\delta_{ij}\frac{\partial U_l}{\partial x_l}\right)\right]+\frac{\partial}{\partial x_j}\left(-\rho\overline{u'_i u'_j}\right) \tag{2.6}$$

Air flowing at low velocities can very well be assumed to be incompressible [6]. This simplifies equation (2.6) to

$$\frac{\partial}{\partial t}(\rho U_i)+\frac{\partial}{\partial x_j}(\rho U_i U_j)=-\frac{\partial p}{\partial x_i}+\frac{\partial}{\partial x_j}\left[\mu\left(\frac{\partial U_i}{\partial x_j}+\frac{\partial U_j}{\partial x_i}\right)\right]+\frac{\partial}{\partial x_j}\left(-\rho\overline{u'_i u'_j}\right) \tag{2.7}$$

These equations are the RANS equations. They are very similar to the exact Navier-Stokes equations, however the velocities and other solution variables now are time-averaged values. Furthermore, an additional term $\left(-\rho\overline{u_i' u_j'}\right)$ appears which is a result of the effects of turbulence. This term represents the Reynolds stresses.

To solve the RANS equations with sufficient accuracy the shear stresses in equation (2.6) have to be modeled appropriately. The Boussinesq hypothesis [7] is generally used to formulate a relation between the shear stresses and the mean velocity gradients.

$$-\rho\overline{u_i' u_j'} = \mu_t \left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) - \frac{2}{3}\rho k \delta_{ij} \tag{2.8}$$

This is the formulation used in the RANS models.

2.2. RANS in a Moving Reference Frame

To simulate the rotation of the blades, a sliding mesh is introduced, which will be further explained in the numerical method section. A sliding mesh uses a moving reference frame, which changes the mathematical formulation of the problem. The transformations required will be explained here [5].

2.2.1. Velocity Translation in a Moving Reference Frame

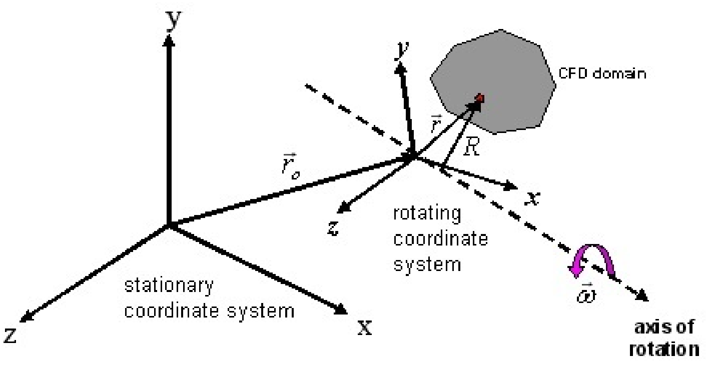


Figure 2.1: Stationary and rotating reference frames

Figure 2.1 provides a visual representation of the moving reference frame, which in this study is a rotating cylindrical disk with a constant angular velocity ω. The axis of rotation is defined by a unit vector perpendicular to the plane of rotation $\hat{a}$. In figure 2.1 this unit vector would point in the direction of the dotted axis of rotation vector. Then the angular velocity $\vec{\omega}$ can be expressed as

$$\vec{\omega} = \omega\hat{a} \tag{2.9}$$

The numerical domain is defined with respect to the rotating coordinate system such that a point in the CFD domain is given by the position vector $\vec{r}$ from the origin of the rotating coordinate system. The transformation from fluid velocities in the stationary to the moving reference frame are then given by

$$\vec{u}_r = \vec{u} - \vec{v}_r \tag{2.10}$$

With

$$\vec{v}_r = \vec{u}_t + \vec{\omega} \times \vec{r} \tag{2.11}$$

Here $\vec{u}_r$ is the relative velocity (the velocity observed from the moving reference frame), $\vec{u}$ is the absolute velocity (the velocity observed from the stationary frame), $\vec{v}_r$ is the velocity of the moving frame relative to the inertial reference frame, $\vec{u}_t$ is the translational frame velocity. The translational frame velocity is zero, because the moving reference frame rotates and does not move. Combining this information with (2.11) and (2.10) gives the relevant formulation of the relative velocity

$$\vec{u}_r = \vec{u} - \vec{\omega} \times \vec{r} \tag{2.12}$$

2.2.2. The Absolute Velocity Formulation

When solving the RANS equations in the moving reference frame their formulation changes. These new formulation can be found by using the new definition for relative velocity (2.12). The equations can be formulated in two different ways, the relative or absolute formulation. In this study the absolute formulation is used.
The continuity equation (2.2) is transformed as follows,

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{u}_r) = 0 \tag{2.13}$$

The momentum equation (2.1) is transformed to

$$\frac{\partial}{\partial t}\left(\rho \vec{u}\right) + \nabla \cdot \left(\rho \vec{u}_r \vec{u}\right) + \rho \left(\vec{\omega} \times \vec{u}\right) = -\nabla p + \nabla \cdot \left(\mu \left(\nabla \vec{u} + \nabla \vec{u}^T\right) - \frac{2}{3}\mu \left(\nabla \cdot \vec{u}\right)\right) + \vec{F} \tag{2.14}$$

Similar to the derivation of the RANS equations without a moving reference frame, the flow variables in the form of equation (2.4) are substituted in (2.13) and (2.14). This results in the absolute formulation of the RANS equations in a moving reference frame.

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_i}(\rho U_{i,r}) = 0 \tag{2.15}$$

$$\frac{\partial}{\partial t}(\rho U_i) + \frac{\partial}{\partial x_j}(\rho U_{i,r} U_j) + \rho\left(\vec{\omega} \times \vec{U}_i\right) = -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\left[\mu\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} - \right)\right] + \frac{\partial}{\partial x_j}\left(-\rho \overline{u_i' u_j'}\right) \tag{2.16}$$

2.3. The Standard $k-\varepsilon$ Model

When modeling turbulent flow, extra transport equations have to be introduced to accurately model the effects of turbulence. The most common approach is the $k-\varepsilon$ model, which is a two-equation model that allows the determination of a turbulent length and time scale. The standard $k-\varepsilon$ model is derived using the assumptions that the flow is fully turbulent and that the effects of the molecular viscosity are negligible. These assumptions are valid for fully turbulent flows. Incompressibility and a uniform temperature profile, thus no temperature gradient are also taken into account to simplify the equation. The variables in the $k-\varepsilon$ model, the turbulence kinetic energy, k, and its rate of dissipation, ε, are derived from the transport equations, which are [5]

$$\frac{\partial}{\partial t}(\rho k) + \frac{\partial}{\partial x_i}(\rho k U_i) = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] + G_k - \rho\varepsilon \tag{2.17}$$

$$\frac{\partial}{\partial t}(\rho \varepsilon) + \frac{\partial}{\partial x_i}(\rho \varepsilon U_i) = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right] + C_{1\varepsilon}\frac{\varepsilon}{k}G_k - C_{2\varepsilon}\rho\frac{\varepsilon^2}{k} \tag{2.18}$$

In equation (2.17), G_k is a representation of the production of turbulence kinetic energy as a result of the average velocity gradients. G_k is defined as

$$G_k = -\rho \overline{u_i' u_j'} \frac{\partial u_j}{\partial x_i} \tag{2.19}$$

The terms $C_{1\varepsilon}$ and $C_{2\varepsilon}$ in equation (2.18) are constants. Furthermore, the turbulent Prandtl numbers for k and ε are represented by σ_k and σ_ε respectively. The constants in equations (2.17) and (2.18) have values which have been determined through numerous experiments and are default values in the standard $k-\varepsilon$ model [8]. These values are $C_{1\varepsilon} = 1.44$, $C_{2\varepsilon} = 1.92$, $\sigma_k = 1.0$ and $\sigma_\varepsilon = 1.3$

2.4. Modeling Turbulent Viscosity

The turbulent viscosity, μ_t, is a quantity used to model momentum transfer caused by turbulent diffusion. This is similar to the way in which the molecular viscosity is used to model momentum transfer caused by molecular diffusion, also known as friction.

$$\mu_t = \rho C_\mu \frac{k^2}{\varepsilon} \tag{2.20}$$

The turbulent viscosity is calculated using (2.20), where $C_\mu = 0.09$ is a default constant [8]. With the turbulent viscosity, a turbulent viscosity ratio can also be determined. The viscosity ratio $\frac{\mu_t}{\mu}$ simply is the ratio between turbulent and molecular viscosity. It is a measure of how turbulent or laminar the flow is behaving. For small values the fluid flow will behave almost like a fully laminar flow, whereas for larger values the flow will be dominated by turbulent characteristics.

2.5. The Turbulence Intensity

The papers used to compare the results of this study (see the numerical method) present the streamwise turbulence intensity instead of the turbulence kinetic energy. In this report results will be expressed by the turbulence kinetic energy. A method for converting these two quantities is presented here.
First the formula for the turbulence kinetic energy k is presented.

$$k = \frac{1}{2}\left(\overline{u_x'^2} + \overline{u_y'^2} + \overline{u_z'^2}\right) \tag{2.21}$$

Then, using the turbulence isotropy approximation equation (which is also used by Fluent)

$$\overline{u_x'^2} = \overline{u_y'^2} = \overline{u_z'^2} \tag{2.22}$$

equation (2.21) can be transformed into equation (2.23).

$$k = \frac{3}{2}\overline{u_z'^2} \tag{2.23}$$

The turbulence intensity, TI is given by equation (2.24)

$$TI = \frac{\sigma_u}{\overline{U}_{hub}} \tag{2.24}$$

where $\sigma_u = \sqrt{\overline{u_z'^2}}$, combining this with equation (2.24) and equation (2.23) gives (2.25) the relation for converting k to TI.

$$TI = \frac{\sqrt{\overline{u_z'^2}}}{\overline{U}_{hub}} = \frac{1}{\overline{U}_{hub}}\sqrt{\frac{2}{3}k} \tag{2.25}$$

The other way around, transforming (2.25) to(2.26) gives the relation for converting TI to k.

$$k = \frac{3}{2}\cdot\left(TI\cdot\overline{U}_{hub}\right)^2 \tag{2.26}$$

3
Numerical Method

The numerical simulation of the wind turbine is done using the Ansys CFD software. These simulations consist of four steps. First, modeling the geometry in Ansys Design Modeler. Second, creating a mesh of the modeled geometry in Ansys Meshing. Third, setting up the physical models, boundary conditions and solving the setup in Ansys Fluent. Finally, analyzing the results in Ansys CFD Post. Besides the Ansys software package, Matlab is also used for post processing the results.

3.1. The Numerical Domain

In this study we use a domain similar to the domain described in Wu and Porté-Agel (2010) [9]. This domain is adequate to simulate the conditions of wind-tunnel measurements performed by Chamorro and Porté-Agel (2010) [10]. This allows comparison with both wind-tunnel measurements and numerical simulations.

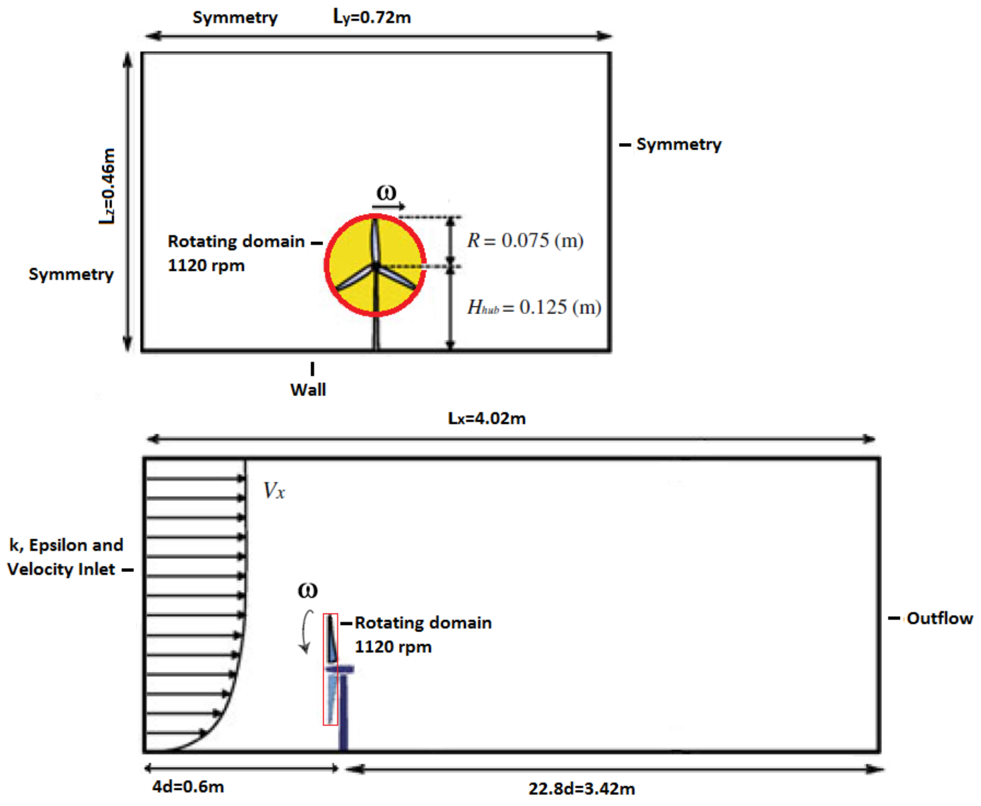


Figure 3.1: Schematic of the numerical domain used in this study including the boundary conditions

The height of the domain is L_z = 0.46 m, which matches the wind-tunnel boundary-layer depth H reported by Chamorro and Porté-Agel. The boundary-layer depth is a measure of the height of the atmosphere below which its behaviour is influenced by the ground. The domain has a streamwise length $L_x = 26.8d = 4.02$ m and a spanwise length $L_y = 4.8d = 0.72$ m, the turbine diameter $d = 0.150$ m is used as a reference here. The wind turbine is placed 4 rotor diameters or 0.6 m from the inlet and at the centre of the spanwise direction.
The goal is to make all conditions identical to the wind tunnel experiment. In the experiment a three-blade GWS/EP-6030x3 rotor is used, which is attached to a small motor. The hub height H_{hub} = 0.125 m, the rotor blade diameter, d = 0.150 m. The motor is cylindrical with a diameter d_m = 0.015 m and l_m = 0.03 m.
Similarly, the tower has a cylindrical shape with d_t = 0.005 m and l_t = 0.118 m. In the simulations rectangular flat blades are used for simplicity's sake.

3.2. Modeling and Meshing the Domain

To simulate this domain a geometry has to be created, this is done using Ansys Design Modeler. Roughly, there are three steps involved. First, the solid structures need to be created. Secondly, the domain needs to be generated. This can be as simple as creating a box with the dimensions specified earlier, 4.02x0.72x0.46 m. However, due to some meshing complications it was chosen to split this domain up into several subsections, as can be seen in figure 3.2.

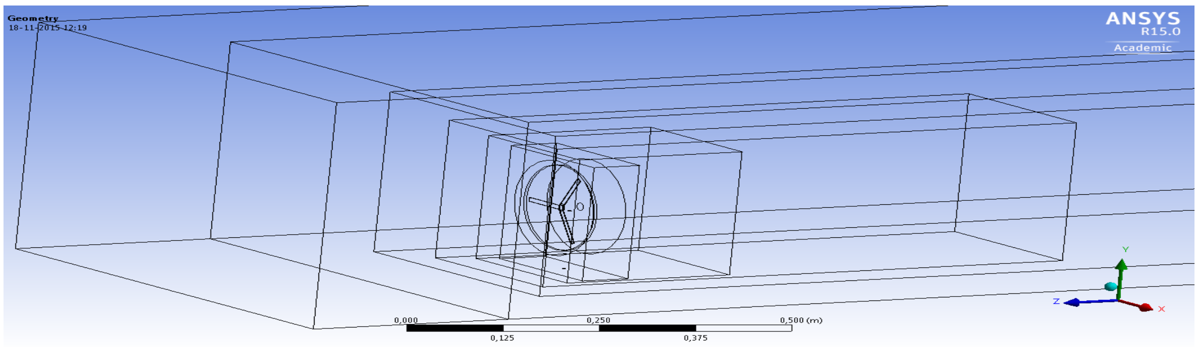


Figure 3.2: A wireframe representation of the geometry used in the final simulations

Finally, to match the geometry input required by Fluent, the solid structures have to be subtracted from the domain using boolean operations.

When the geometry is completed it can be imported into Ansys Meshing to create a mesh. A mesh is a structure of small cells, the simplest type being cubics. In each of these cells the RANS equations will be solved. There are two major factors which need to be taken into account when meshing.
The first factor is the mesh size. The RANS and $k-\varepsilon$ equations have to be solved in each cell. A reasonably fine mesh will easily exceed one million cells, thus when millions of differential equations need to be solved at every iteration it is clear that the mesh size is the bottleneck for computational speed. Therefore, a smart meshing approach needs to be used to find a mesh which, on the one hand is fine enough to provide a convergent solution, but on the other hand does not take an excessive amount of time to solve. To find this solution it is important to identify where the mesh needs to be fine and where a coarser mesh can be used. With some basic fluid mechanics knowledge one can imagine that the flow right behind the wind turbine is most turbulent and will require a fine mesh to capture all physical effects. On the other hand, the far field flow will barely be influenced by the wind turbine and a coarser mesh can be used.
The second factor to consider is the transition of cell size within the mesh. The problem which can occur if you transition from a really fine mesh to a really coarse mesh is a type of numerical interpolation error. Imagine that at the interface the fine mesh contains 10 times more cells than the coarse mesh. The large cell suddenly needs to implement 10 different inputs and compute the result with a single set of RANS equations. The error which now occurs is that part of the information is missed out, leading to a diffusion of physical quantities. In the early stages of this project the numerical interpolation error was very large. The numerical interpolation error is illustrated and explained in appendix A.
Taking both these factors into account resulted in the final mesh used, which is illustrated in the next section. An overview of the different stages of the development of the mesh is provided in the table below.

Different stages of the mesh development	**Details**
1. The numerical domain with a simple mesh, no refinements	0.2 million elements
2. Domain with one box close to the wind turbine	0.5 million elements
4. Another box is introduced even closer to the turbine	1 million elements
3. The box from 2. is extended, similar element size	1.4 million elements
4. Four boxes with a gradually increasing element size	2.6 million elements
5. A cylindrical mesh is introduced around the rotating domain	4.8 million cells, smallest cell size of 0.001m in the rotating domain and the stationary cylindrical domain closest to the rotating domain.

3.3. The Mesh

Figures 3.3-3.6 give a visual representation of the mesh used in the final simulations. Figure 3.3 shows the numerical domain, the black boxes correspond to areas with a different meshing size to reduce numerical diffusion. Figure 3.4 clearly shows that the transition between the meshing areas in the total domain is very gradual. Figure 3.6 illustrates that the transition between the rotating domain and the stationary domain is very smooth as well. This is ensured by the cylindrical mesh, a rectangular mesh does not provide a smooth transition. Note that two types of the rotating domain were used in the final cases. The large rotating domain is four times wider than the blades in the streamwise direction, with the blades located at the center. On the other hand, the small rotating domain is exactly as wide as the blades. Figure 3.6b shows the large rotating domain, where the blades are the dense mesh at the centre of the rotating domain.

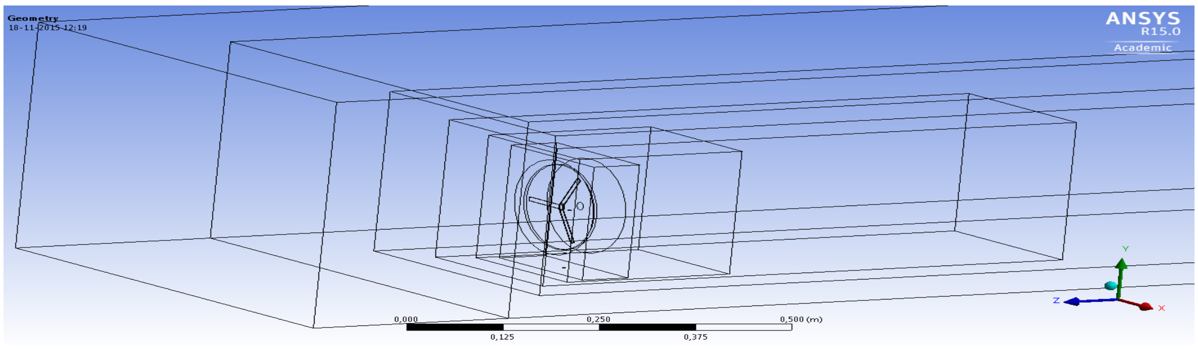


Figure 3.3: A wireframe view of the numerical domain around the wind turbine

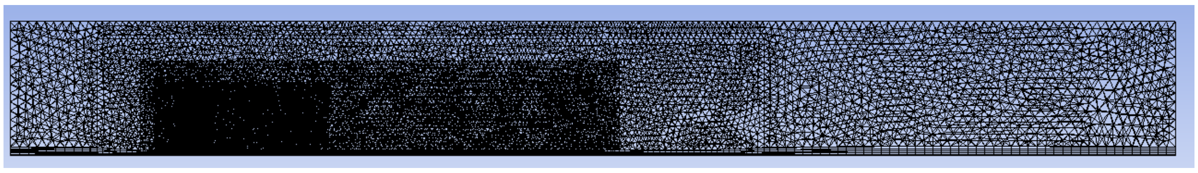


Figure 3.4: A sideview of the final mesh used, notice the smooth transitions to ensure that the numerical interpolation error is reduced to a minimum

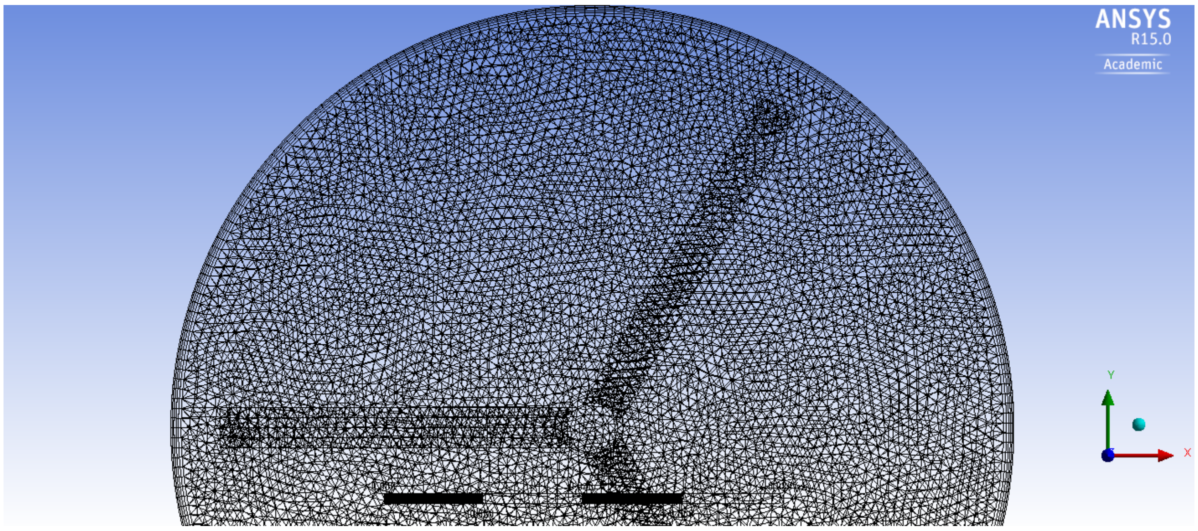
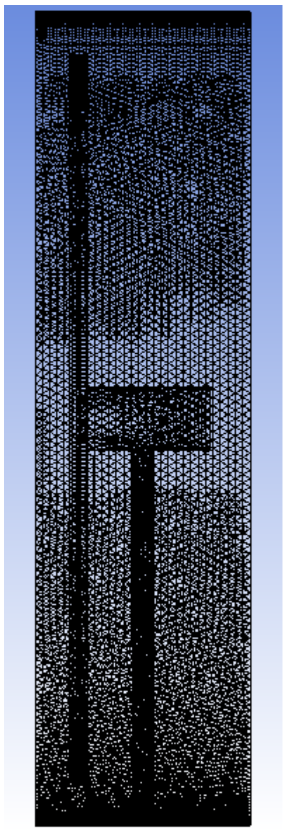


Figure 3.5: A view of the rotating domain in the streamwise direction


(a) An overview of the cylindrical part of the mesh

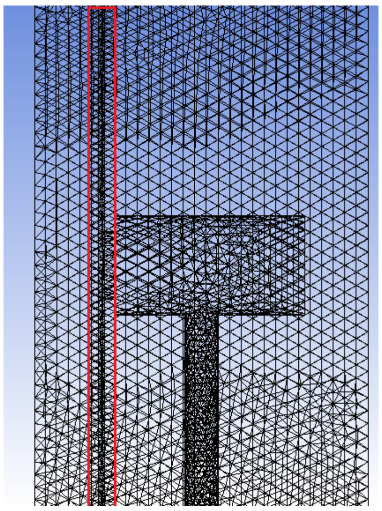

(b) A close up of the interface between the cylindrical part of the stationary mesh and the rotating domain. The rotating domain is highlighted in red.

Figure 3.6: A side view of the cylindrical part of the mesh

3.4. Numerical Solution Setup

The core of the Ansys CFD software package is Ansys Fluent, this is the software which performs the numerical calculations. Before performing these simulations it is important to make sure that Fluent interprets the problem correctly. This can be done by adjusting a range of settings, such as the boundary conditions and solution methods.

The rotation of the blades is the most important physical aspect to be modeled in the present study. This can be done in the cell zone conditions options, where there is a possibility to implement translation or rotation of different cell zones. The blades are located in a small cylindrical disk, the rotating domain in figure 3.1. This domain is specified as a sliding mesh with a rotational velocity of 1120 rpm [9]. A sliding mesh will rotate the entire disk at every time step, which leads to an actual rotation of the blades. The mathematical implementations of the sliding mesh are similar to the moving reference frame and can be found in section 2.2 of the theory. The feature which makes the sliding mesh unique when compared to a simple moving reference frame is the fact that it is time-dependent (transient). At every time step the entire mesh rotates and a convergent solution is found for that time step. This makes the simulations realistic, however also computationally expensive.

The solution process was done in two parts, first 5000 time steps with a time step size of 2 ms, 10 seconds in total. Afterwards, another 2000 time steps were calculated with a size of 1 ms, another 2 more seconds in total. The time step size was chosen by looking at how long one full rotation takes. This is $60/1120 = 0.0536$sec so with a time step size of 2ms a full rotation is split up in 27 steps. For 1ms this is 54 steps, it is unclear whether these are enough steps to obtain realistic results. Therefore, in the results section it is tested whether the time step size is small enough to get realistic convergent results .

The boundary conditions and solution methods need to be specified. The domain boundary conditions can be seen in figure 3.1. A velocity inlet is chosen for the inflow of air and the boundary condition outflow is specified at the outlet. At the inlet a custom velocity, k and ε profile can be chosen. A logarithmic velocity profile is specified with a friction velocity and surface roughness of 0.126 m s^{-1} and 0.09 mm respectively. The k and ε profiles are similar to the profiles of the in-house code, which have been customized to match the inlet conditions of the wind tunnel measurements. These profiles are included in appendix E. The ground is treated as a wall and the other sides of the domain are specified as symmetry. Fluent allows specification of other simulation conditions, such as solution methods and solution controls. These simulation conditions can be found in appendix C.

3.5. Implementing the Setup on a High Performance Computer

At the end of this project the simulations got to the point where they were so computationally expensive that the use of a High Performance Computer (HPC) was required to save time. To be able to run the setup on the HPC, also referred to as a cluster, some extra details have to be taken into account. It is important to note that there is no Graphical User Interface (GUI) on the cluster. Therefore, all Fluent commands need to be written in a journal using the TUI language. Furthermore, another script needs to be written which will tell the cluster to load Fluent with a sufficient amount of cores, 32 cores were used for the final simulations. In this script it is also essential to specify that Fluent will read the journal after being loaded. Once the script and journal are finished these two documents and the fluent case file can be uploaded to the cluster and it is possible to do the calculations. An example of the script and journal files used are shown in appendix D.

3.6. Post processing Results

Finally, once the calculations are completed, either local or on the cluster, the results can be extracted and implemented in CFD Post for analysis. CFD Post offers a wide variety of options to analyze the results. In this study, the main feature which was used to visualize the results were contour plots in the centre streamwise plane. This is the x z-plane at the center of the y axis (the turbine location). These contour plots give a good impression of the physics of the system and allow a quick check to see if the system behaves as expected. In Wu and Porté-Agel (2010), the exact results are presented as plots of a variable versus the height in the centre streamwise plane at different streamwise positions. To be able to compare the results with the experimental data it is necessary to extract these data sets. In CFD Post this is made easy, by creating vertical lines at equivalent positions and then plotting the required quantities such as streamwise velocity at these locations. The data used in these plots can then be exported in a data file. Finally, Matlab was used to plot these data files versus the experimental data to get an accurate comparison.

4 Results and Discussion

Throughout this study, simulations were performed with different mesh configurations. The final four simulations used a cylindrical mesh surrounding the rotating domain. From these four simulations the best results were chosen and are presented and discussed in this section.

4.1. Final Results

Four different cases using the cylindrical mesh were performed on the cluster. A description of these cases is given in the list below. Note that two types of the rotating domain were used, which are discussed in section 3.3.

- Blades only, large rotating domain
- Setup with Tower and Motor, large rotating domain
- More refined mesh quality test, large rotating domain
- **Final Result** Blades only, small rotating domain

Realistic profiles for the streamwise velocity and shear stress were observed in multiple cases. However, the results kept changing when refining the mesh, therefore they were not yet grid independent. The cases with the cylindrical mesh were the only cases where the results are be grid independent. Furthermore, the case with the small rotating domain is the only case which captures the turbulence kinetic energy well. Therefore, the case with the cylindrical mesh and the small rotating domain was chosen as the final result.
The final results are presented in this section and compared to the wind tunnel measurements. The streamwise velocity, turbulence kinetic energy and kinematic shear stress profiles will be compared. The turbulence viscosity ratio profiles are shown as well for both the Fluent simulations and the in-house code. Unfortunately, the turbulence energy dissipation was not reported by Wu and Porté-Agel, therefore the turbulence energy dissipation cannot be compared with the wind tunnel experiments and is not reported.
Contour plots of the results will be displayed in the middle vertical plane perpendicular to the turbine. These contours provide a quick image to check if the physics make sense. For accurate comparison to wind tunnel measurements, the results and measurements will be plotted at different streamwise locations in the centre streamwise plane.
Besides this comparison of the final results with the wind tunnel measurements three side-effects are investigated at the end of this section. First, the mesh quality is investigated. Second, the effects of the tower and motor are discussed. Finally, as this is a transient solution the effect of the time-step size and number of time steps is investigated.

4.1.1. The Streamwise Velocity

The time-averaged streamwise velocity contour, figure 4.1, looks quite realistic. There is a clear wake profile occurring behind the blades, which gradually decreases and the far field velocity profile seems to match the inlet conditions. Wen comparing figure 4.1 to figure 4.2, the wind tunnel measurements from Wu, it does become clear that the low velocity wake is too strong.
The plots in figure 4.3 confirm this. The shape of the wake pattern is correct, however the wake is too strong at all the different locations.

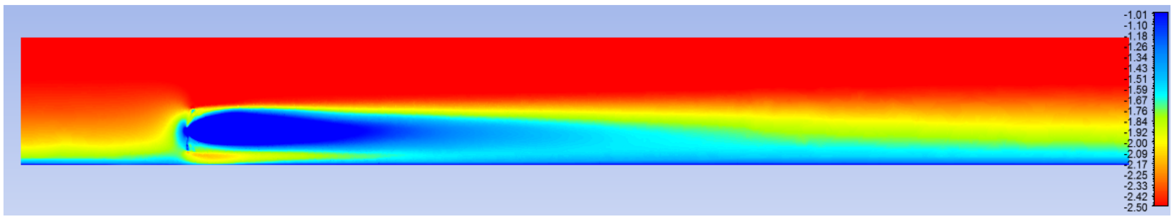


Figure 4.1: Contours of the time-averaged streamwise velocity. The domain is identical to figure 4.2

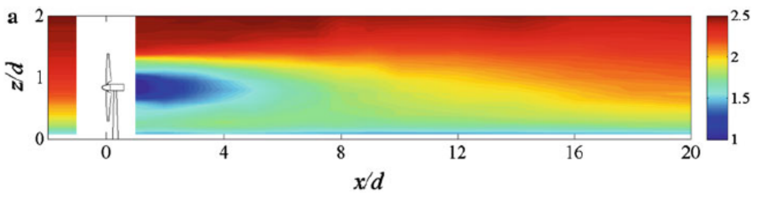


Figure 4.2: Contours of time-averaged streamwise velocity from wind tunnel measurements [9]

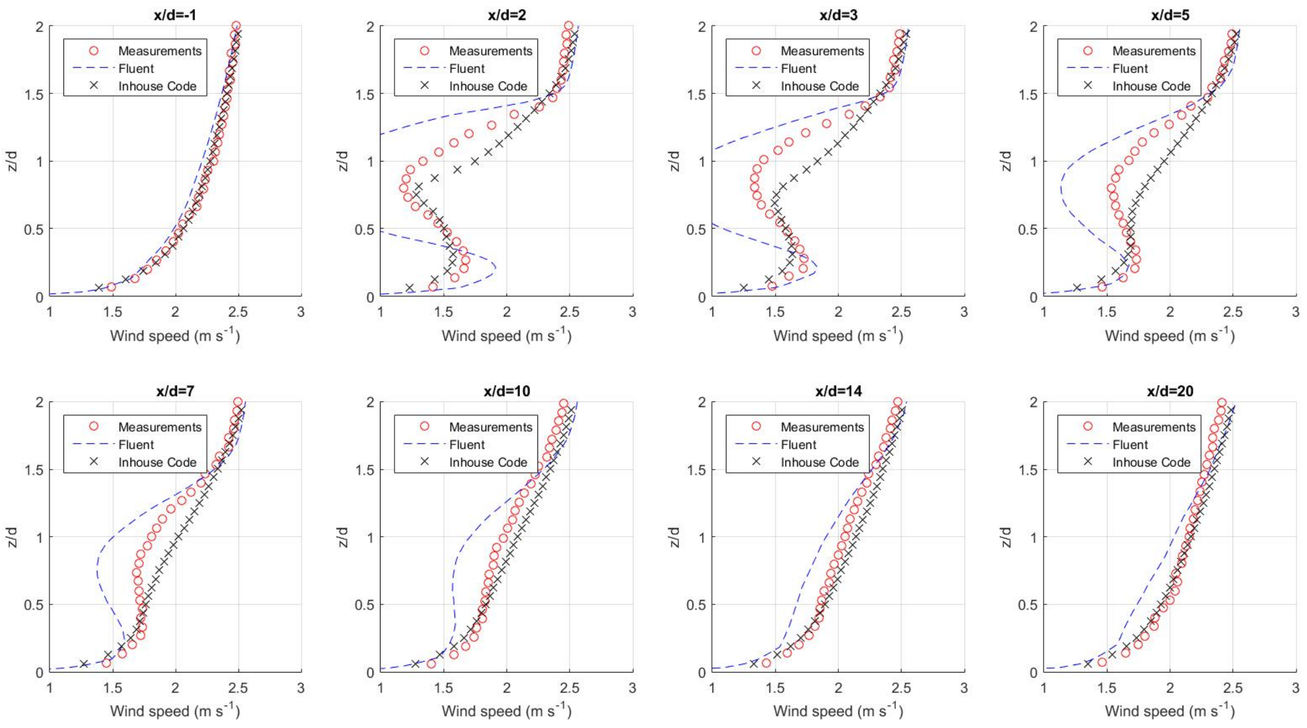


Figure 4.3: Wind speed profiles at different locations. Wind tunnel measurements [9] (*open circle*), results from the Fluent simulations (*dashed line*) and results from the in-house code (*crosses*).

4.1.2. The Turbulence Kinetic Energy

The turbulence kinetic energy profile is presented here. In the paper by Wu, the turbulence intensity $\sigma_u/\bar{u}_{hub}$ is presented. This quantity can easily be converted to the turbulence kinetic energy profile by using the isotropic turbulence approximation, see section 2.5 from the theory.

When comparing figure 4.4 to figure 4.5, we can see that the contours look similar. However, these are two related yet different quantities. The plots in figure 4.6 show the same quantity and from these plots we can see that the trends are captured well. In terms of magnitude the results are off by a factor 1.5 or less compared to the wind tunnel measurements. The far field (after x/d=10) and near wall k profiles are too small. This is probably a result of the way in which Fluent treats the inlet conditions for k and ϵ, because the in-house code uses the same inlet conditions and figure 4.4 shows that they match the wind tunnel results better.

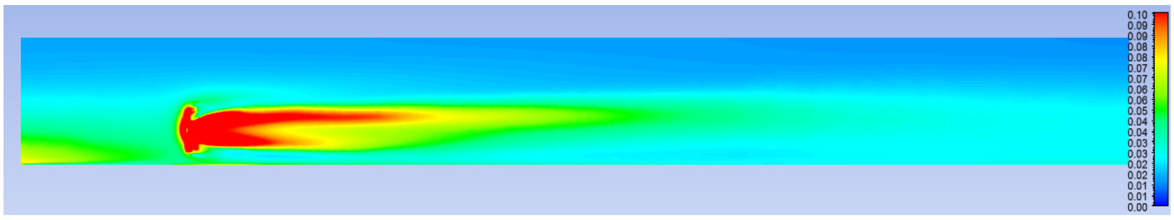


Figure 4.4: Contours of the turbulence kinetic energy. The domain is identical to figure 4.5

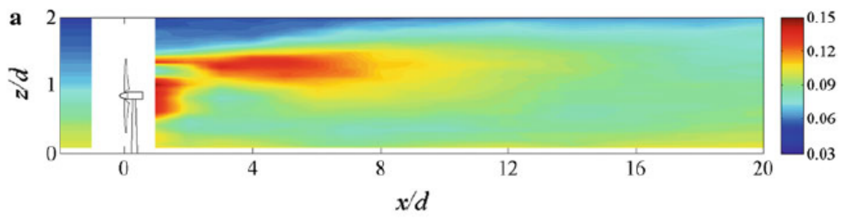


Figure 4.5: Contours of the streamwise turbulence intensity from wind tunnel measurements [9]

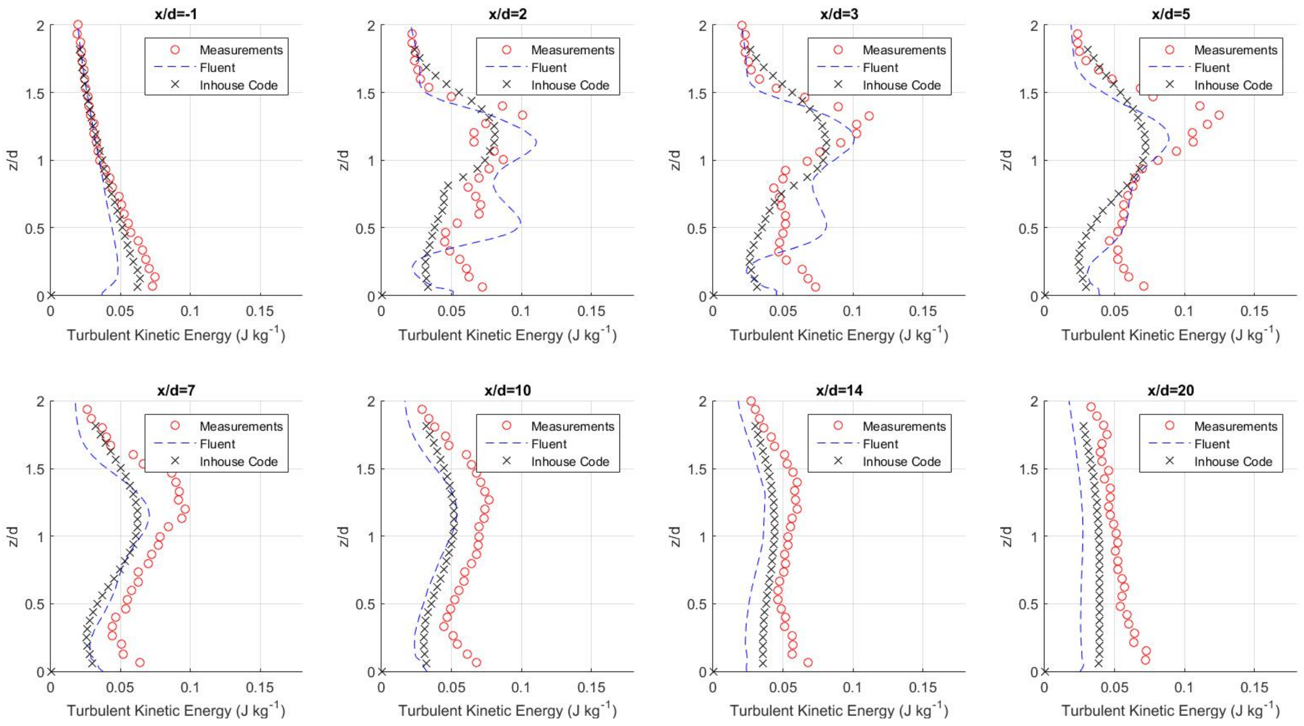


Figure 4.6: Turbulence kinetic energy profiles at different locations. Wind tunnel measurements [9] (*open circle*), results from the Fluent simulations (*dashed line*) and results from the in-house code (*crosses*).

4.1.3. The Kinematic Shear Stress

Comparing figure 4.7 with figure 4.8, it can be seen that both our results and the wind tunnel measurements provide similar contour plots. Figure 4.8 confirms that the trends are very similar, the only difference is that the magnitude of the near wake shear stress is larger. This could be caused by the fact that the near-wake streamwise velocity is smaller than in the wind tunnel experiments. This will cause an increase in mixing between the high velocity layer and the low velocity wake. This in turn results in a larger magnitude of the kinematic shear stress.

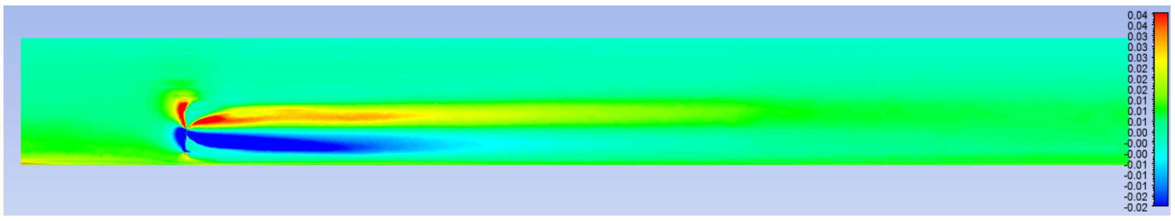


Figure 4.7: Contours of the kinematic shear stress. The domain is identical to figure 4.8

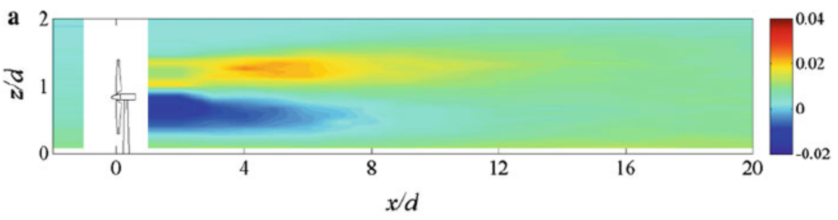


Figure 4.8: Contours of the kinematic shear stress $\overline{u'w'}$(m^2 s^{-2}) from wind tunnel measurements [9]

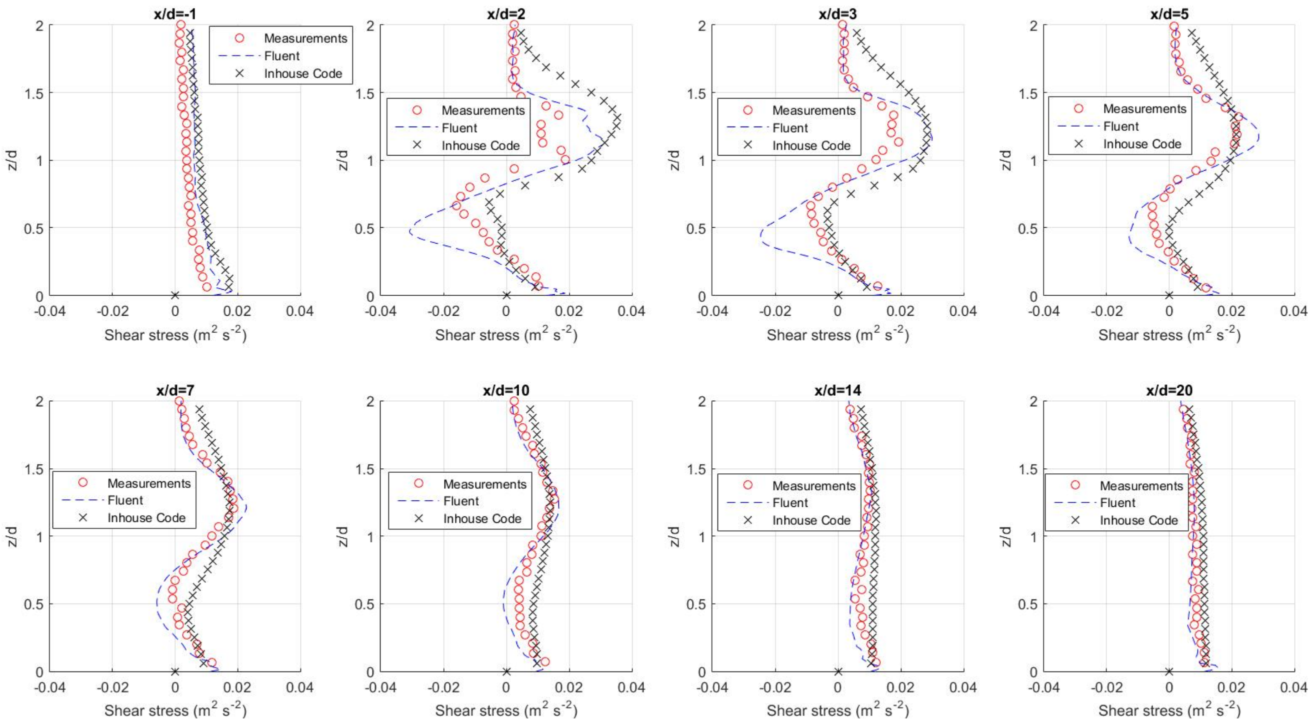


Figure 4.9: Shear stress profiles at different locations. Wind tunnel measurements [9] (*open circle*), results from the Fluent simulations (*dashed line*) and results from the in-house code (*crosses*).

4.1.4. The Turbulence Viscosity Ratio

The turbulence kinetic energy is presented for the Fluent results and the in-house code. The wind tunnel measurements do not include measurements of the turbulence energy dissipation ratio, therefore the viscosity ratio cannot be determined. Figures 4.10 and 4.11 show that the results from this study and the in-house code are rather different. Our results show a decrease in the entire area behind the blades. The in-house code shows an increase above the centre of the blades and a decrease below the centre of the blades.

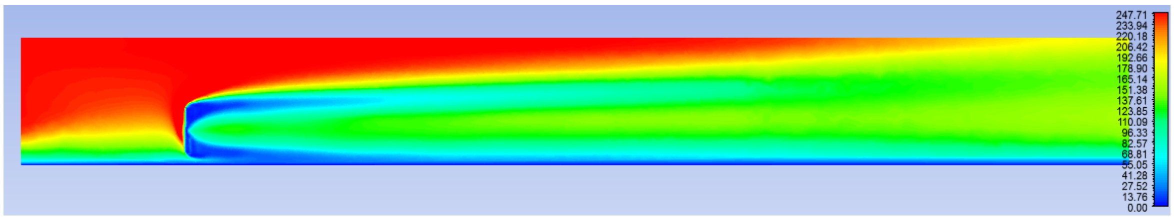


Figure 4.10: Contours of the viscosity ratio. The domain is identical to figure 4.11

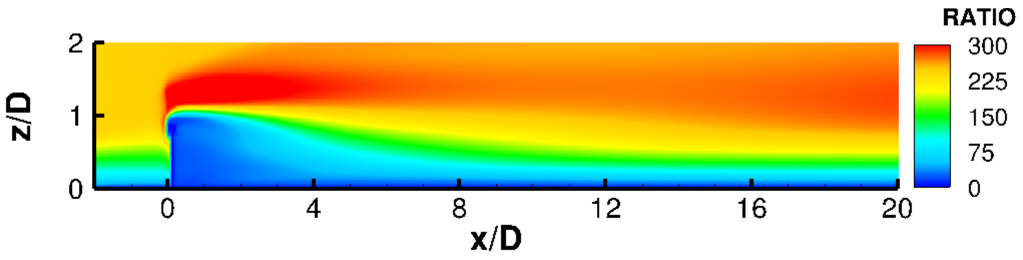


Figure 4.11: Contours of the turbulence viscosity ratio from in-house code

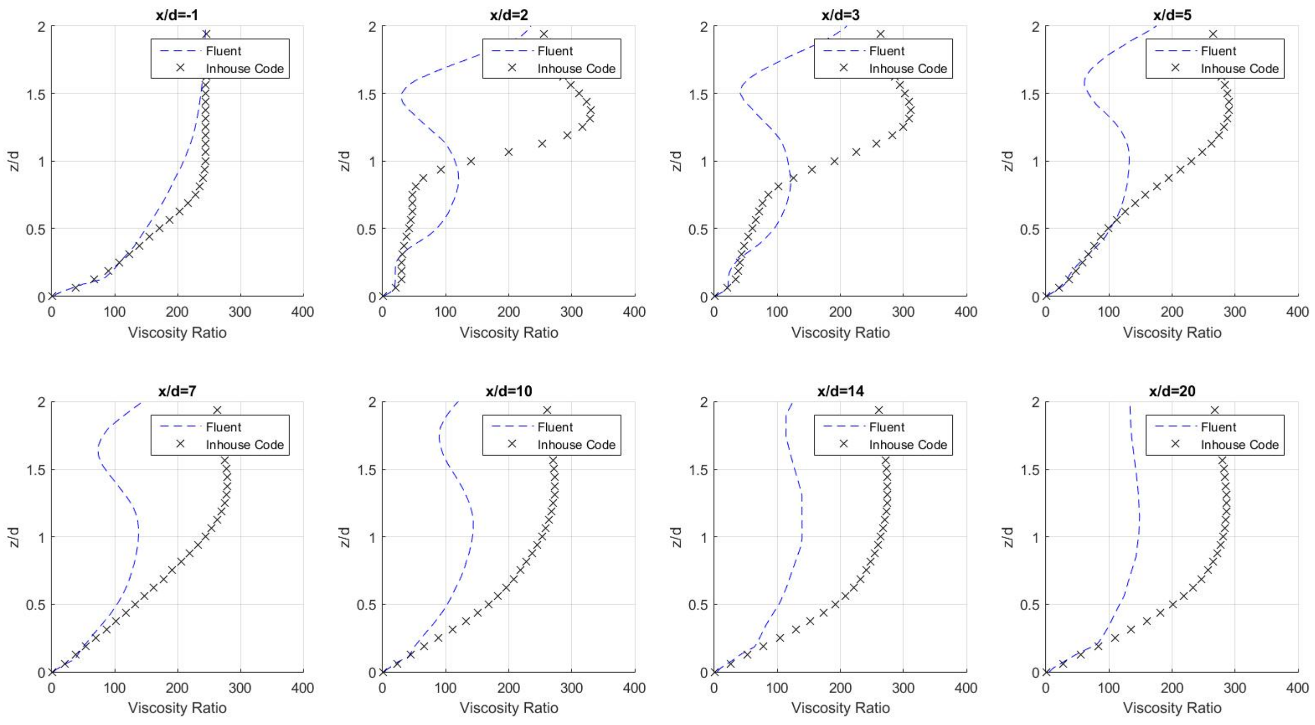


Figure 4.12: Profiles of the viscosity ratio at different locations. Results from the Fluent simulations (*dashed line*) and results from the in-house code (*crosses*).

4.2. Testing Side Effects

When testing the side effects most visual examples are shown with contour plots of k. The reason why k is used and not all quantities or a different quantity is because k was the quantity which converged the slowest per grid cell in Fluent. Thus, k was the bottleneck for the convergence of the entire simulation and is sufficient when testing the convergence of side effects such as the mesh quality.

4.2.1. Testing the Mesh Quality

To make sure that the size of the mesh elements does not make a difference, an even more refined mesh than the cylindrical mesh was created and compared. See appendix B, figures 6.9 and 6.10, for a visual representation of this mesh. The cylindrical mesh used to calculate the results consisted of roughly 4.8 million cells. On the other hand, the mesh used to test mesh quality consisted of 7.5 million elements. These 2.7 million extra cells are located in the first and second cylinder to ensure an even better transition from the rotating domain to the stationary domain.

When looking at figures 4.10 and 4.13, the turbulence kinetic energy profile with the two meshes, there is no visible difference. This indicates that the mesh quality does not influence the results. Although it has to be noted that the cell size of the rotating domain was not refined, because this would require the entire mesh to be refined even further to avoid big mesh size transitions. This in turn would lead to an even longer computational time, which was not available.

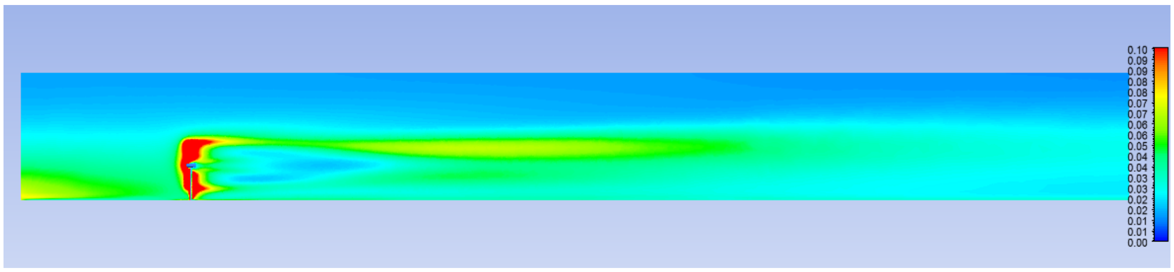


Figure 4.13: The turbulence kinetic energy contour with the cylindrical mesh

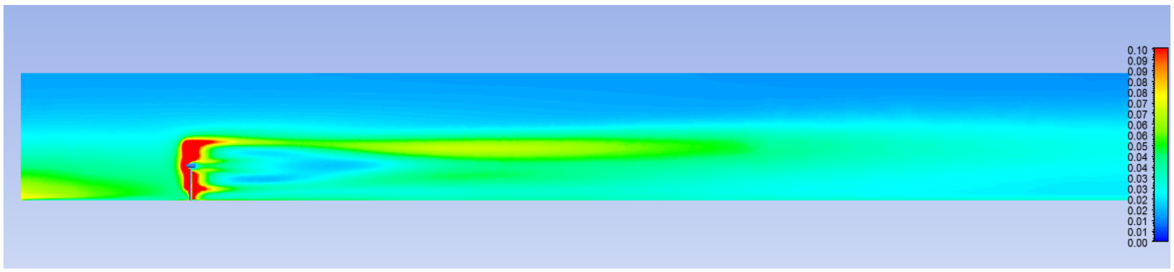


Figure 4.14: The turbulence kinetic energy contour with the refined quality testing mesh

4.2.2. The Effects of the Tower and Motor

By comparing figures 4.15 with 4.17 and 4.16 with 4.18 the effects of the tower and motor can be seen. Clearly, the effect of the tower and motor is very small. The only visible difference is observed in the area behind the tower and motor, as is expected. Therefore, the final result which only include the blades can be treated as a representative result.

Blades only

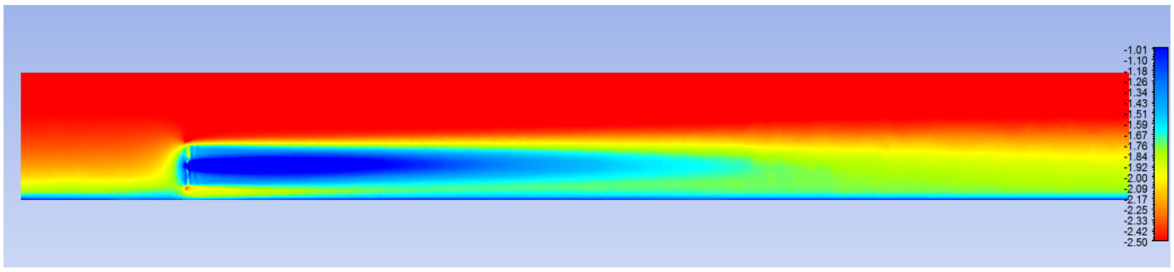


Figure 4.15: Contours of the streamwise velocity without the tower and motor

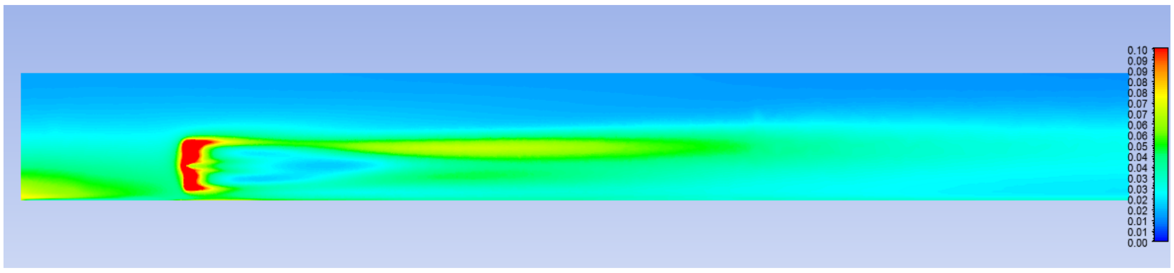


Figure 4.16: Contours of the turbulence kinetic energy without the tower and motor

Tower and Motor

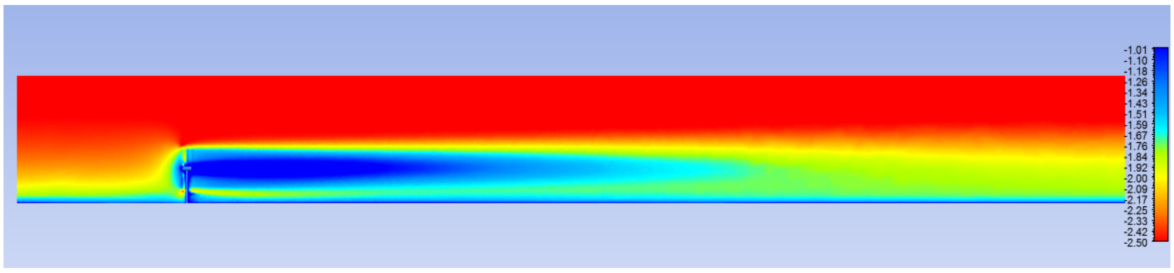


Figure 4.17: Contours of the streamwise velocity with the tower and motor

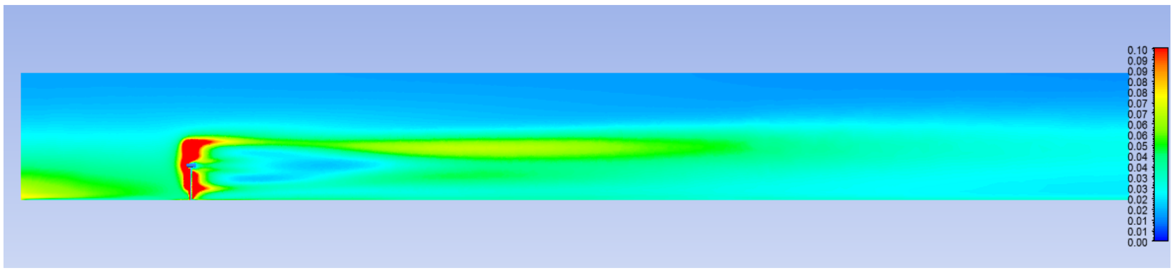


Figure 4.18: Contours of the turbulence kinetic energy with the tower and motor

4.2.3. Testing the Effects of the Time Step Size

To make sure that the results are correct the time step size and number of time steps have to be investigated. The results were recorded at 5, 10 and 12 seconds. The contour plot of the turbulence kinetic energy is shown to illustrate how the solution converges and depends on the time stepping. Figures 4.19 and 4.20 are at 5 and 10 seconds respectively, with a time step size of 2 ms. The result does not seem to change from 5 to 10 seconds, therefore the solution is convergent and we can be sure that enough time steps have been simulated. On the other hand, when switching to a smaller time step, figure 4.21 the solution does change slightly. Thus, the solution is not yet time step size independent, although it is safe to assume that decreasing the time step size further will only result in very small changes of the results.

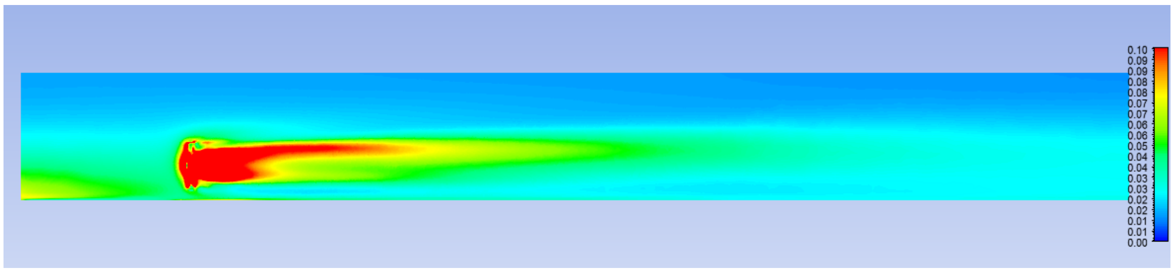


Figure 4.19: Contours of the turbulence kinetic energy after 5 seconds

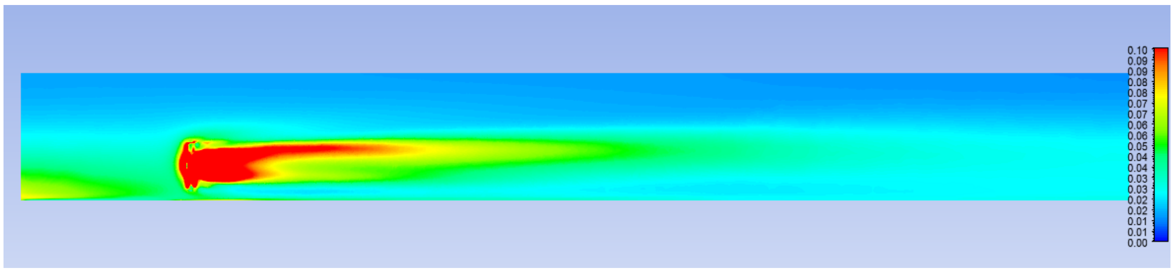


Figure 4.20: Contours of the turbulence kinetic energy after 10 seconds

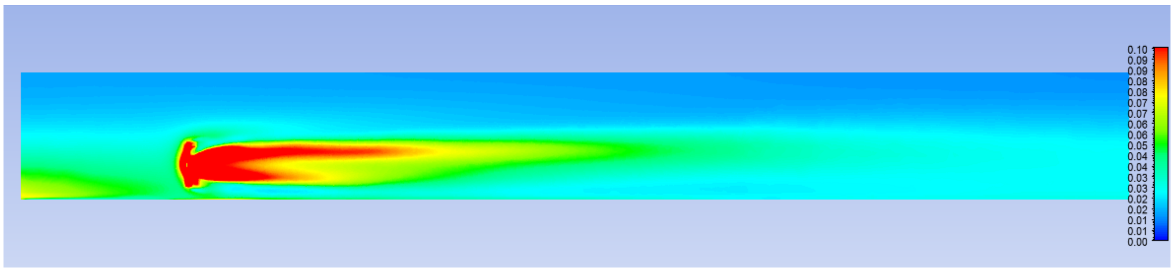


Figure 4.21: Contours of the turbulence kinetic energy after 12 seconds, the time step size is also halved

4.3. Discussion

First of all, the reason why the case with the small rotating domain was chosen over the other cylindrical mesh cases as the final result is illustrated here. We use the $k-\epsilon$ model to model turbulence, thus it is essential that the turbulence kinetic energy profile is correct. Figures 4.22 shows that when using the case with the large rotating domain the turbulence kinetic energy profile in the near wake is missing almost completely. Figure 4.23 on the other hand does show that the turbulence kinetic energy profile is captured quite well, as shown in section 4.1.2. Therefore, the case with the small rotating domain was chosen.

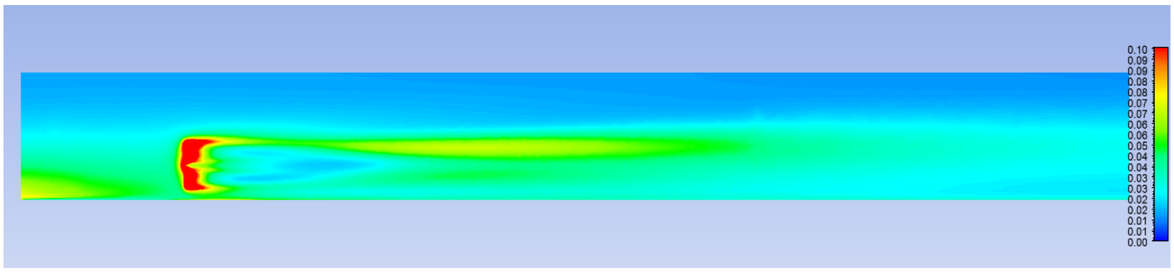


Figure 4.22: Contours of the turbulence kinetic energy with the large rotating domain

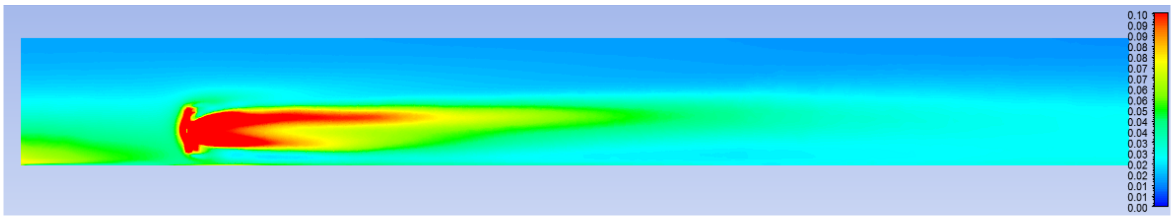


Figure 4.23: Contours of the turbulence kinetic energy with the small rotating domain

The streamwise velocity, turbulence kinetic energy and kinematic shear stress profiles were compared with wind tunnel measurements to verify the results. The final results suggest that the physics of this problem are captured, however the magnitude of the reported quantities is incorrect. This can be caused by several reasons, which are discussed here.

First, it can be the case that the inlet profiles for k and ϵ are incorrect, this is also suggested in the section 4.1.2. Another reason why the results are incorrect might be the sliding mesh. The rotational speed of the sliding mesh 1120 rpm is incredibly high, because commercial wind turbines only achieve about 30-60 rpm [11]. The large rotational velocity used in the simulations might cause Fluent to obtain errors in the $k-\epsilon$ differential equations.

The way in which the rotating domain is simulated might also cause errors in the results. Right now, the rotational domain includes the air in between the blades. This means that the air rotates automatically, whereas realistically we want the blades to move the air around. This might be a negligible effect, because we do expect the simulation to converge to a realistic result at every time step, however it is interesting to investigate.

The most important reason that the results are incorrect is the fact that the $k-\epsilon$ model is used. There are more accurate models for solving complex rotational problems in the turbulent domain, such as large eddy simulation (LES) or hybrid LES/RANS.

5 Conclusions

5.1. Conclusions from the Results

The aim of this project was to simulate a wind turbine as realisticly as possible using the $k-\varepsilon$ model and a sliding mesh in Ansys Fluent. The final results suggest that wind turbine simulations using this method provide results that are qualitatively correct. In terms of magnitude, the results differ a factor 1 to 1.5 when compared to the wind tunnel measurements. This is the result of an overprediction of wake in the final simulations.
In conclusion, wind turbine simulations using a sliding mesh provide qualitatively correct results. However, the final simulations overpredict the wake. If this problem can be solved in further research this method seems very promising.

5.2. Suggestions for Further Research

There are several suggestions which can improve the results of this study. The first suggestion is to model a larger setup, with a much smaller rotational speed. This could be done to investigate the effect of the high rotational velocity of the sliding mesh. These simulations can also be used to check if this numerical setup is size independent.
Moreover, the blades were modeled as flat rectangular plates. Using a realistic model of the blades might lead to more accurate results.
To verify the mesh quality completely a finer mesh can be implemented in the rotational domain. This would mean that the entire mesh needs to be refined to ensure that there are no big interface jumps and will cause an even longer computational time.
The specified k and ε inlet profiles can be looked at as well. These profiles appear to be slightly incorrect and can be tested in an empty domain to match the far field and near wall results of the wind tunnel measurements better.
Finally, other numerical models than the $k-\varepsilon$ model can be tested to see if they lead to more accurate results.

6 Appendix

A. Numerical Interpolation Error

An example of numerical interpolation error in the streamwise velocity is presented in figures 6.1-6.3. The black lines represent sections of the domain, which have different mesh refinements (cell sizing). In figure 6.1 a small domain with a more refined mesh is used. In figure 6.2 this refined mesh is extended and in figure 6.3 extra sections are added to obtain smaller jumps in the cell sizing. A maximum factor of 2 in cell size was allowed between the different sections.
When comparing figures 6.1 to 6.2 it can be observed that part of wake is missing in figure 6.1. This dissipation of wake starts right behind the refined mesh, which clearly is an example of numerical interpolation error. Now, comparing to figure 6.3 where the big jumps in mesh sizing have been prevented as well it is clear that a way more realistic wake profile is captured. These figures illustrate the importance of appropriate meshing and the effects of numerical interpolation errors when this is not the case.

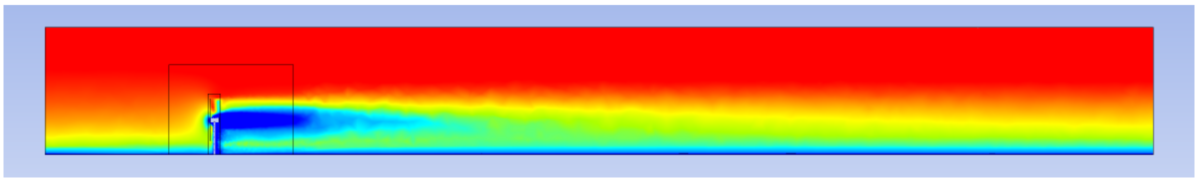


Figure 6.1: Contours of streamwise velocity where there is a very small section with a refined mesh

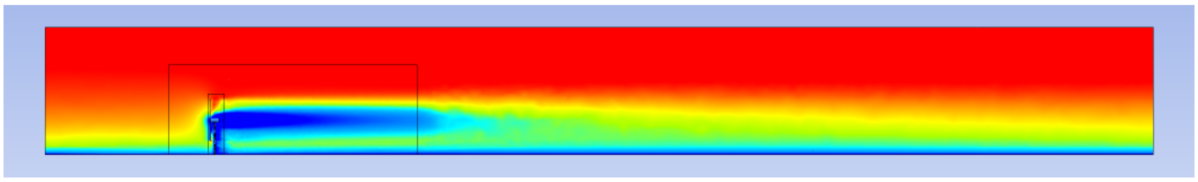


Figure 6.2: Contours of the streamwise velocity where the refined mesh section is extended

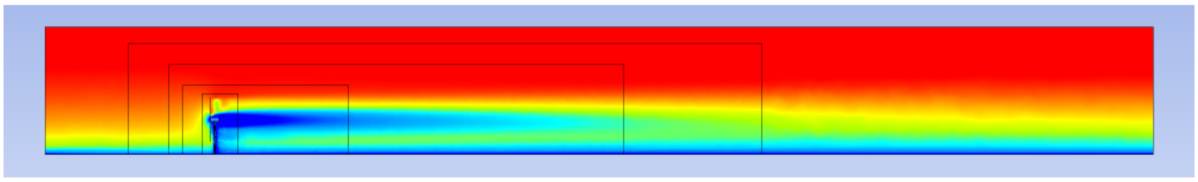


Figure 6.3: Contours of the streamwise velocity, here different mesh cell sizes are present in the different sections

B. Meshes from extra cases

Different meshes than the mesh used in the final case are presented here.

The Final Non-Cylindrical Mesh

The table at the end of section 3.2 shows how this mesh compares to the final mesh. This case is stage 4 of the mesh development, with 2.6 million elements compared to the 4.8 milllion elements of the final mesh.

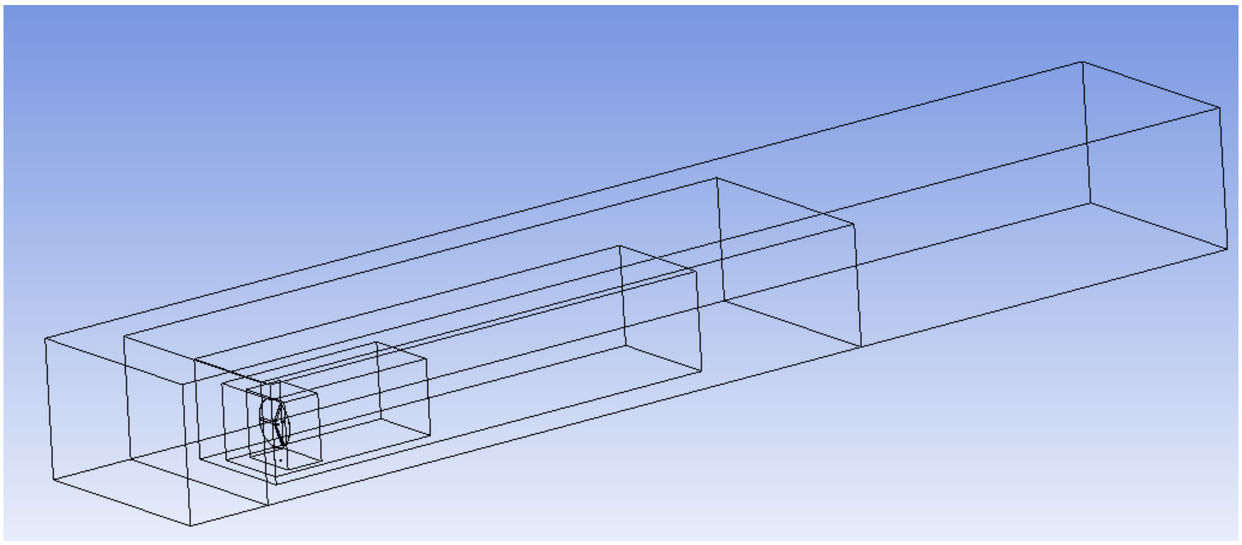


Figure 6.4: The numerical domain, the black boxes represent areas with different meshing size

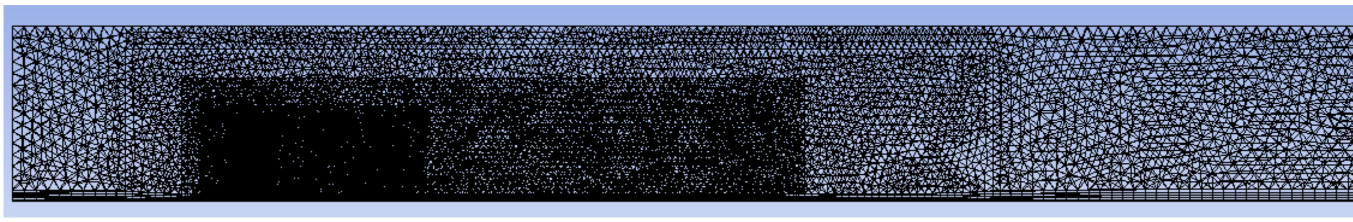


Figure 6.5: A side view of the mesh

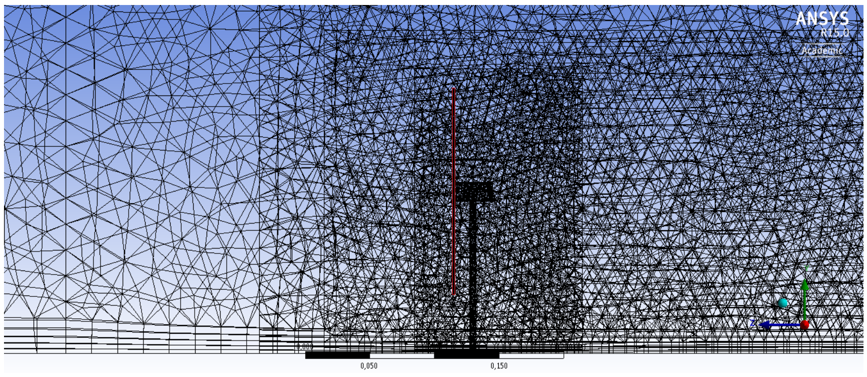


Figure 6.6: A close up of the meshing areas, the rotating domain is represented by the red area

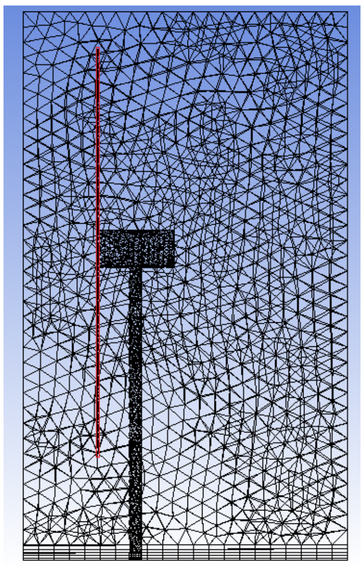


Figure 6.7: The meshing area closest to the wind turbine, the rotating domain is represented by the red area

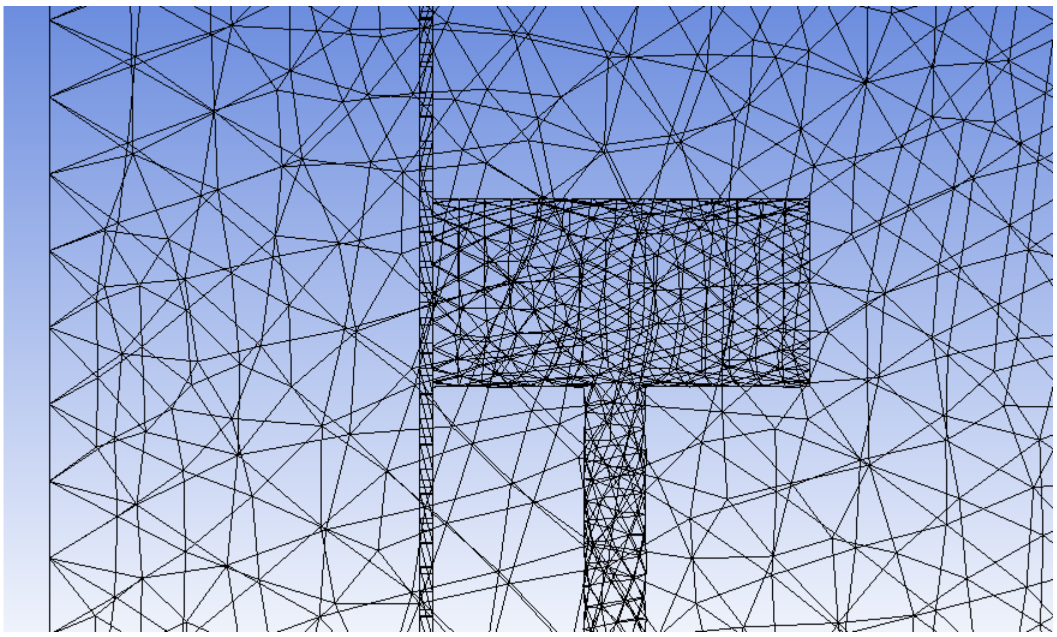


Figure 6.8: A close up of the transition from the non-rotational to the rotating mesh

The Mesh Used to Test Mesh Quality

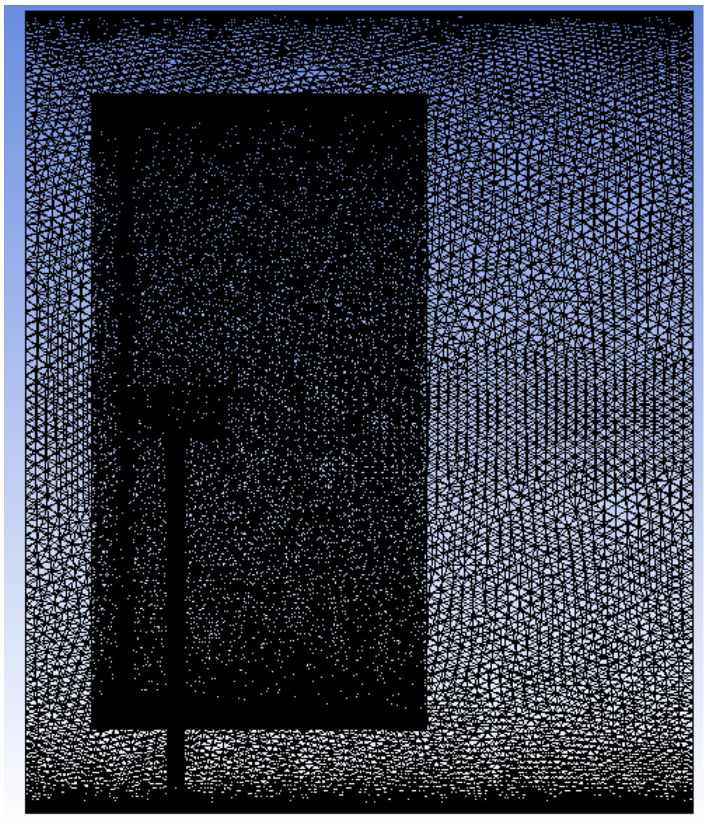


Figure 6.9: The meshing area closest to the wind turbine

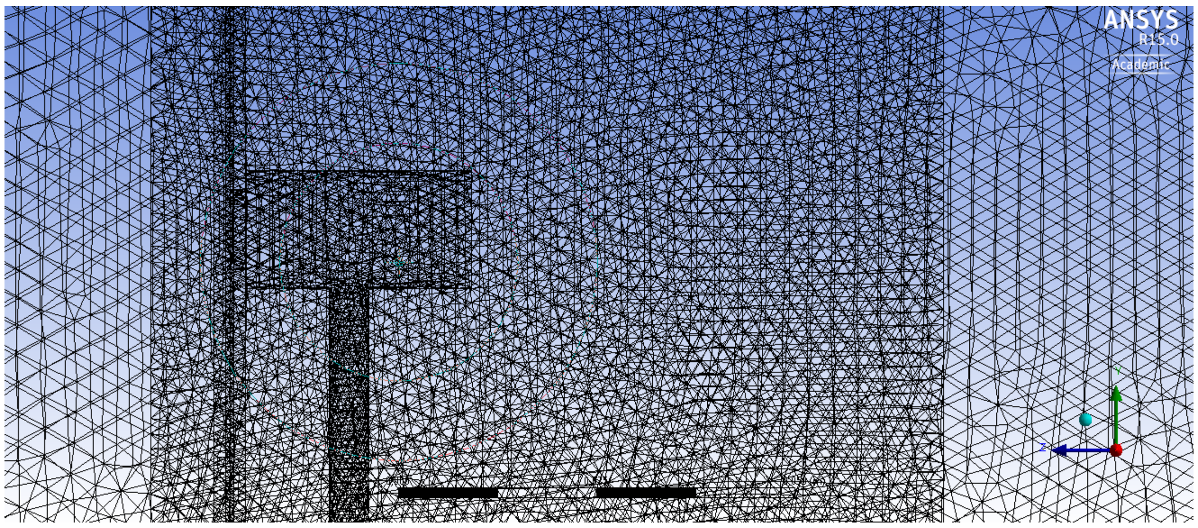


Figure 6.10: A close up of the transition from the non-rotational to the rotational mesh

C. Fluent Settings

The settings used in the Fluent software are displayed here. Ansys Fluent 15.0 was used for the simulations.

General, Solver	
Type	pressure-based
Velocity formulation	absolute
Time	transient
Models	
Viscous model	k-epsilon
K-epsilon model	standard
Wall functions	standard
Materials	
Fluid	air
Solid	nylon (density 1150kg/m^3)
Boundary Conditions	
Blade	wall
Tower	wall
Motor	wall
Bottom face	wall
Inlet	velocity inlet (for profiles see appendix E.)
Outlet	outflow
Side and top faces	symmetry
Reference values	standard Fluent settings
Solution methods	
Scheme	SIMPLE
Gradient	least squares cell based
Pressure	second order
Momentum	second order upwind
Turbulence kinetic energy	first order upwind
Turbulence dissipation rate	first order upwind
Transient eigenvalues	first order upwind
Solution controls	
Pressure	0.3
Density	1
Body forces	1
Momentum	0.3
Turbulence kinetic energy	0.3
Turbulence dissipation rate	0.2
Turbulence viscosity	0.1

D. High Performance Computing

There are two scripts which are used. One is the clusterscript, which tells the cluster to start up Ansys Fluent with a specified number of cores. The second script is a journal file (.jou), which is written in the Fluent TUI language. This script gives the commands to set up the boundary conditions in Fluent. Text with a ; are comments in the journal file.

The clusterscript

```
#PBS -l nodes=1:ppn=32
#PBS -N DefinitiefBladenRotdomeinKlein
module load fluent/15.0
export FLUENTLM_LICENSE_FILE=27021@flexserv-f2.tudelft.nl
cd $PBS_O_WORKDIR
NP=‘wc -l < $PBS_NODEFILE‘
fluent 3ddp -t$NP -g -cnf=$PBS_NODEFILE -i Journal.jou >& output.$PBS_JOBID
```

The journal file

```
; read case file
;rc /home/jvanderauweraert/clustermap/DefinitiefBladenRotdomeinKlein/DefinitiefBladenRotdomeinKle
; compile UDFs
/define/user-defined/compiled-functions/compile
"libudf"
yes
"Udfs2.c"
""
""
/define/user-defined/compiled-functions/load
"libudf"
; set inlet conditions
/define/boundary-conditions/velocity-inlet inlet
no
yes
yes
no
0
yes
no
0
no
0
yes
yes
"udf"
"log_velocity::libudf"
yes
yes
yes
"udf"
"tkeudf::libudf"
yes
yes
"udf"
"epsilonudf::libudf"
;MRF to mesh motion
/grid/modifyzones/mrftoslidingmesh
19
```

```
;initialize solution
/solve/initialize/hyb-initialization
; solve transient (time steps iterations)
/solve/set/time-step
0.005
solve dual-time-iterate
; number of time steps
1000
; number of iterations per time step
30

wd DefinitiefBladenRotDomeinklein_5sec.dat
wc DefinitiefBladenRotDomeinkleinOutput.cas

; solve transient (time steps iterations)
/solve/set/time-step
0.005
solve dual-time-iterate
; number of time steps
1000
; number of iterations per time step
30

wd DefinitiefBladenRotDomeinklein_10sec.dat
wc DefinitiefBladenRotDomeinkleinOutput_10sec.cas

; solve transient (time steps iterations)
/solve/set/time-step
0.002
solve dual-time-iterate
; number of time steps
1000
; number of iterations per time step
50

wd DefinitiefBladenRotdomeinKlein_mindt50itpts.dat
wc DefinitiefBladenRotdomeinKlein_12sec.cas
exit
yes
```

E. Custom Inlet Profiles

The streamwise velocity, turbulence kinetic energy and turbulence dissipation rate can be customized at the inlet with the use of UDFs. The UDFs used here attempt to mimic the initial conditions in Wu and Porté-Agel [9]. The code for these UDFs is shown below.

Streamwise velocity UDF

```
#include "udf.h"

DEFINE_PROFILE(log_velocity,thread,index)
{
real z[ND_ND];
real y, x , h , ustar , znul;
face_t f;
ustar = 0.126 ;  /*friction velocity in m/s */
znul = 0.00009;  /* surface roughness in m */
h= 0.125  ;       /* axis origin z-height in m */
begin_f_loop(f,thread)
{
F_CENTROID(z,f,thread);

y= (z[1]+h);  /*non-dimensional y coordinate */
if (y < znul) y=znul;

F_PROFILE(f,thread,index) = (ustar/0.41)*log(znul/y);
}
end_f_loop(f,thread)
}
```

Turbulence kinetic energy UDF

```
#include "udf.h"

DEFINE_PROFILE(tkeudf,thread,index)
{
real z[ND_ND];
real y, x, h;
face_t f;
h= 0.125  ;       /* axis origin z-height in m */
begin_f_loop(f,thread)
{
F_CENTROID(z,f,thread);

y= (z[1]+h);  /*non-dimensional y coordinate */

if (0 < y && y < 0.0203)
{
F_PROFILE(f,thread,index) = 1.5*(2.2*(0.02*y/0.15 + 0.0988))*(2.2*(0.02*y/0.15 + 0.0988));
}
else if (0.0203 < y && y < 0.4460)
{
F_PROFILE(f,thread,index) = 1.5*(2.2*(0.0071*(y/0.150)*(y/0.150) - 0.0422*y/0.150 + 0.1071))*(2.2
}
else
{
F_PROFILE(f,thread,index) = 0.0143;
}
}
end_f_loop(f,thread)
}
```

Turbulence dissipation rate UDF

```
#include "udf.h"

DEFINE_PROFILE(epsilonudf,thread,index)
{
real z[ND_ND];
real y, x, h, Fmax, F, tke;
face_t f;
Fmax=120;
h= 0.125  ;       /* axis origin z-height in m */
begin_f_loop(f,thread)
{
F_CENTROID(z,f,thread);

y= (z[1]+h);  /*non-dimensional y coordinate */
F=MAX(2*MIN(915.4082*y+18.0330,Fmax),1.0);

if (0 < y && y < 0.0203) {
tke= 1.5*(2.2*(0.02*y/0.150 + 0.0988))*(2.2*(0.02*y/0.150 + 0.0988));
}
else if (0.023<y && y < 0.4460) {
tke = 1.5*(2.2*(0.0071*(y/0.150)*(y/0.150) - 0.0422*y/0.150 + 0.1071))*(2.2*(0.0071*(y/0.150)*(y/0.150
}
else {

tke = 0.0143;
}

F_PROFILE(f,thread,index) = 0.09*tke*tke/(F*0.0000182/1.205);
}
end_f_loop(f,thread)
}
```

F. Results of the Non-Cylindrical Mesh

The results of the final non-cylindrical mesh are presented here.

The Streamwise Velocity

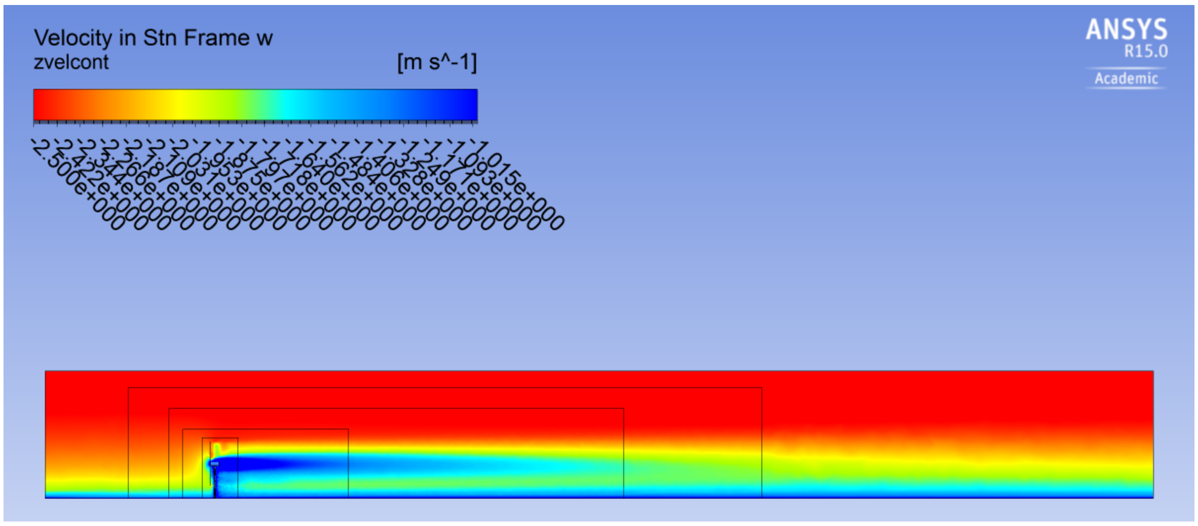


Figure 6.11: Contours of the streamwise velocity, the black lines indicate different meshing areas, which are irrelevant for these results

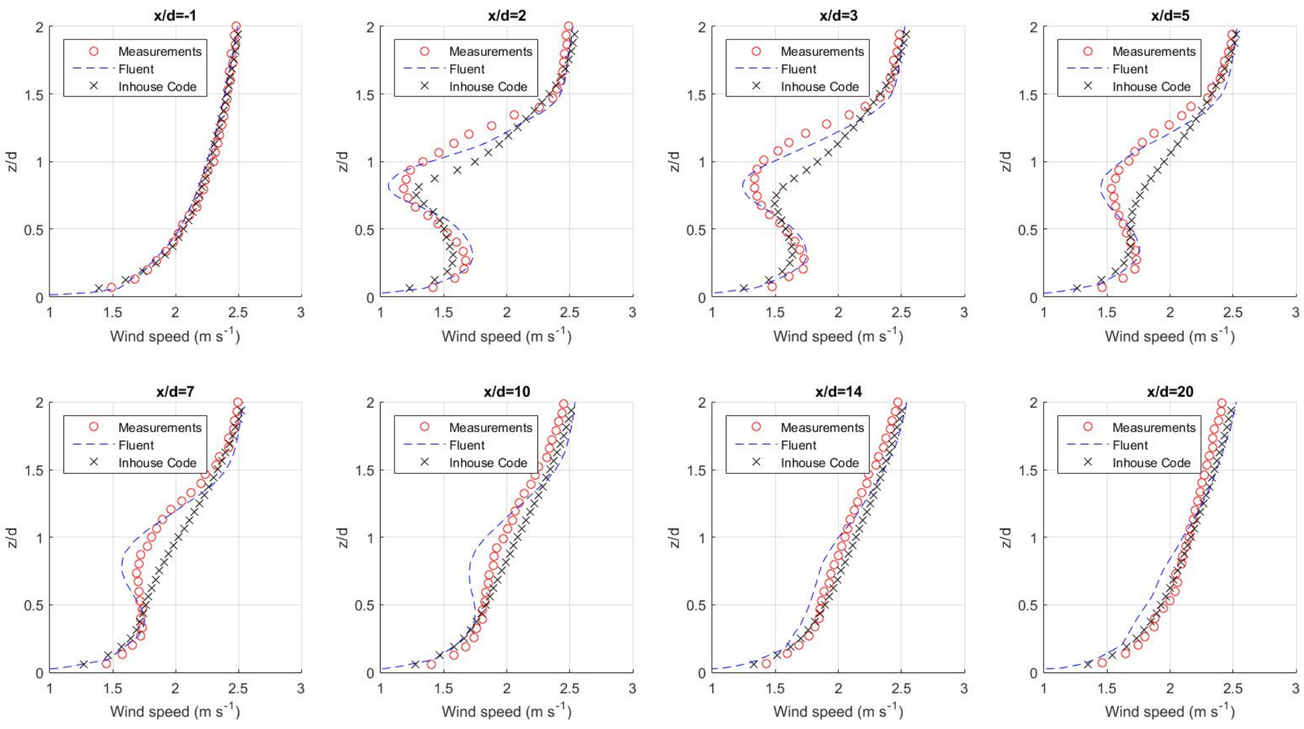


Figure 6.12: Wind speed profiles at different locations. Wind tunnel measurements (*open circle*), results from the Fluent simulations (*dashed line*) and results from the in-house code (*crosses*).

The Turbulence Kinetic Energy

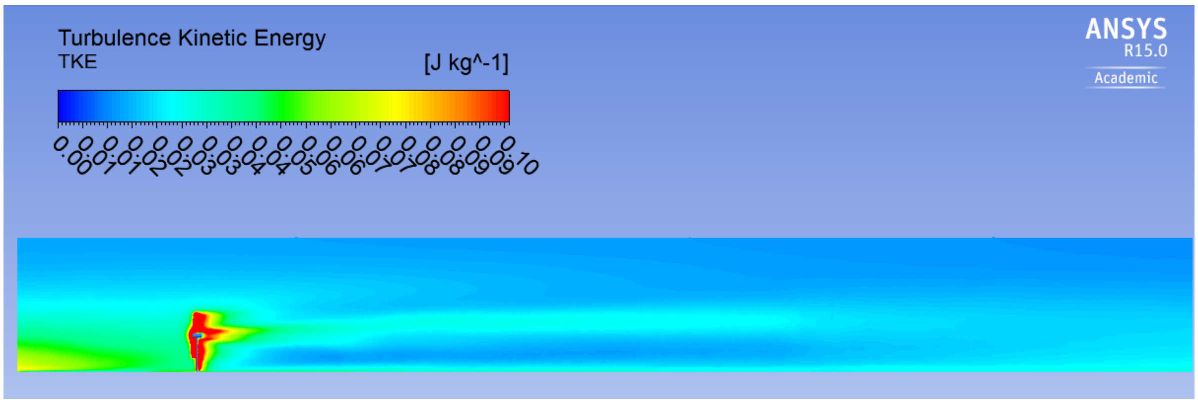


Figure 6.13: Contours of the turbulence kinetic energy

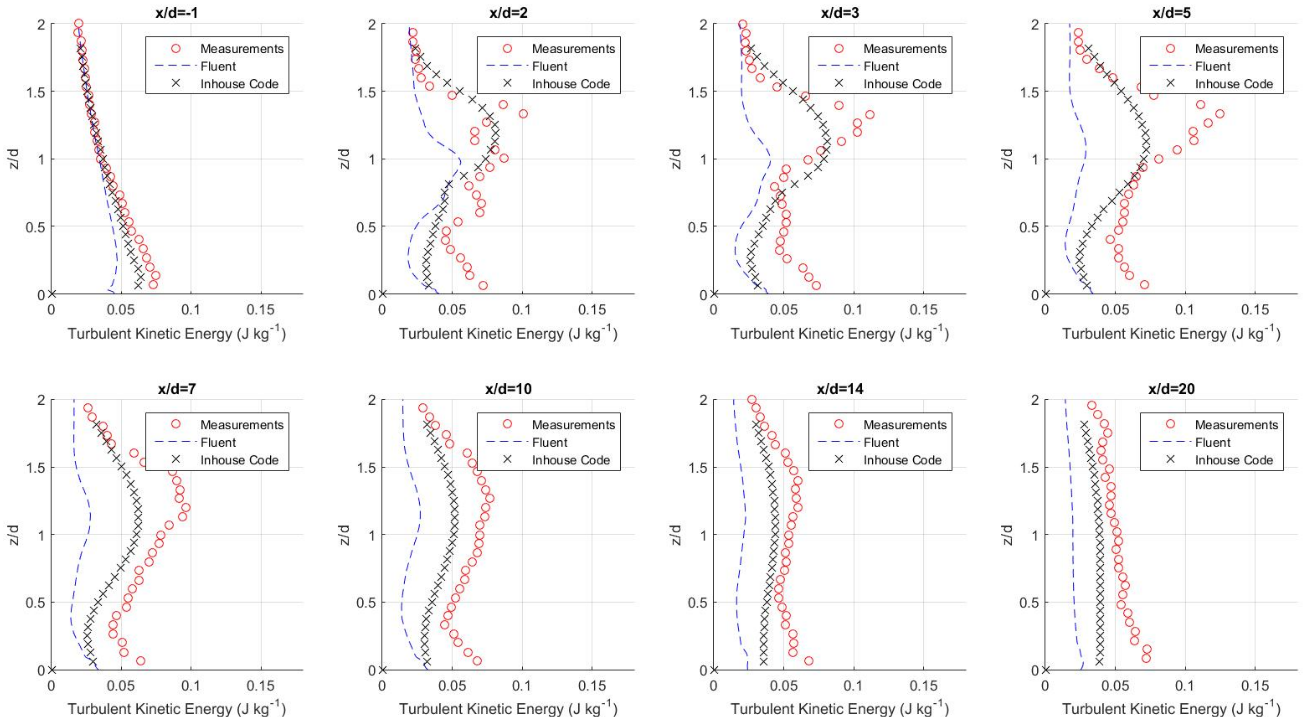


Figure 6.14: Turbulence kinetic energy profiles at different locations. Wind tunnel measurements (*open circle*), results from the Fluent simulations (*dashed line*) and results from the in-house code (*crosses*).

The Kinematic Shear Stress

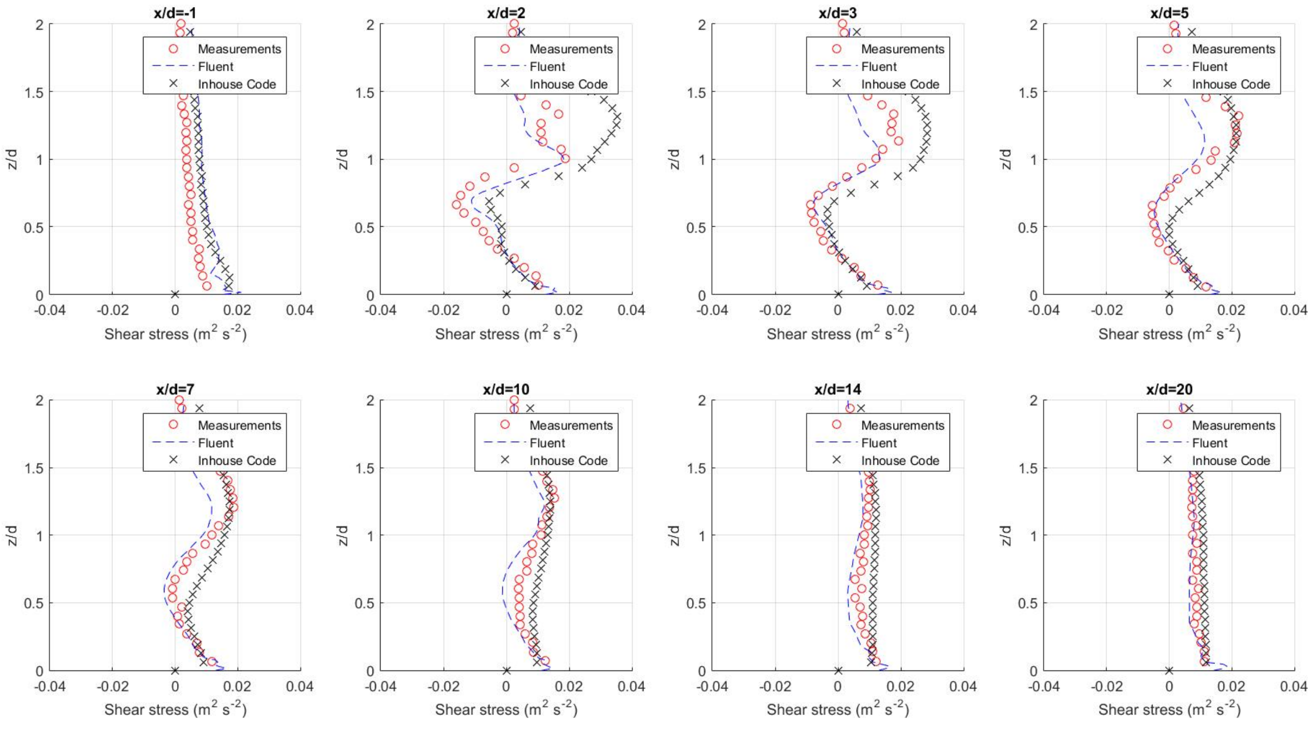


Figure 6.15: Shear stress profiles at different locations. Wind tunnel measurements (*open circle*), results from the Fluent simulations (*dashed line*) and results from the in-house code (*crosses*).

The Turbulence Viscosity Ratio

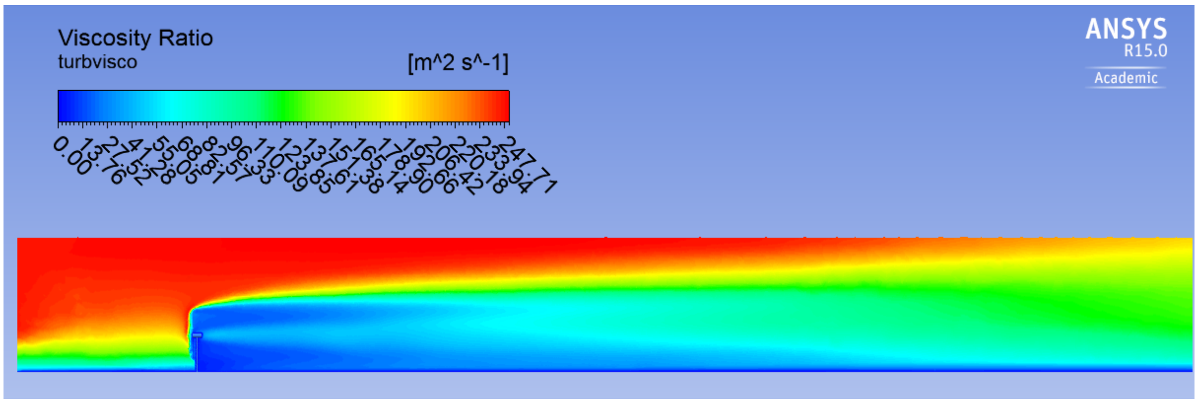


Figure 6.16: Contours of the turbulence viscosity ratio

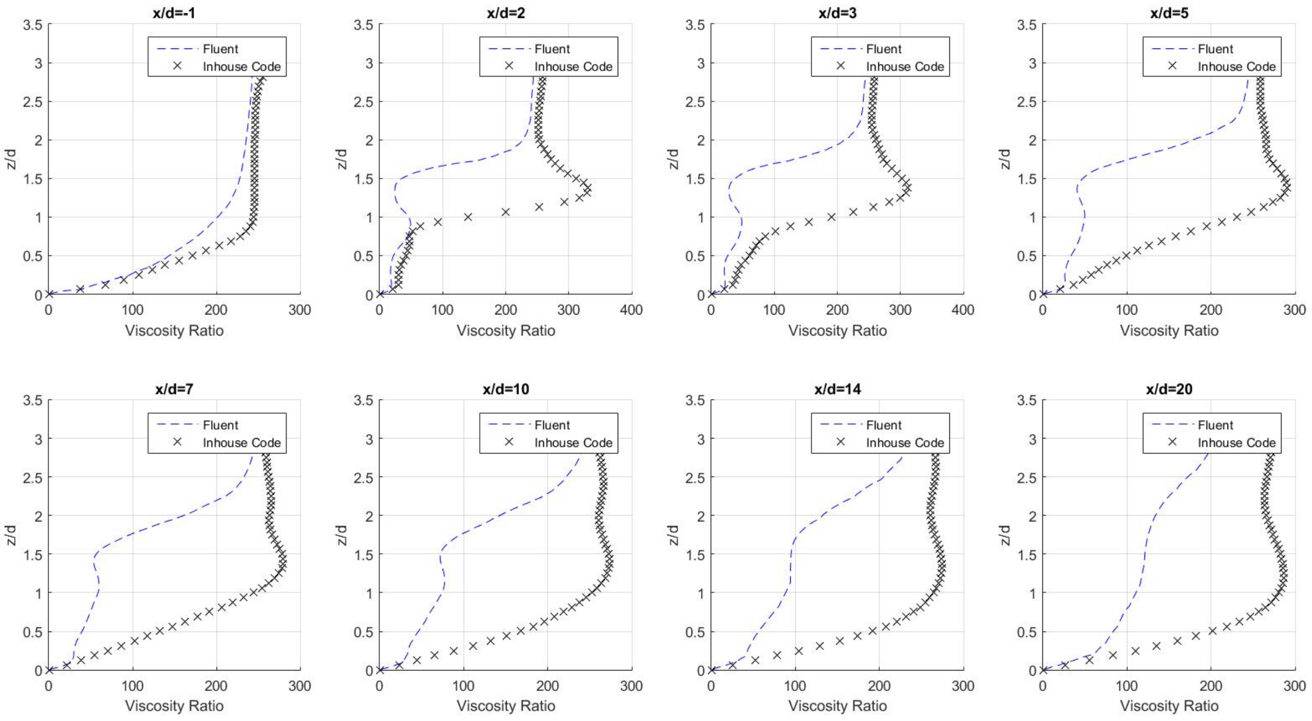


Figure 6.17: Profiles of the viscosity ratio at different locations. Results from the Fluent simulations (*dashed line*) and results from the in-house code (*crosses*).

Bibliography

[1] Vermeer et al. Wind turbine wake aearodynamics. *Progress in Aerospace Sciences*, 2003.

[2] Barthelmie et al. Modelling and measuring flow and wind turbine wakes in large wind farms offshore. 2009.

[3] S. Kenjeres, S. de Wildt, and T. Busking. Capturing transient effects in turbulent fflow over complex urban areas with passive pollutants. *Internal Journal of heat and fluid flow*, 2015.

[4] S. Kenjeres and B. ter Kuile. Modelling and ssimulation of turbulent fflow in urbun areas with vegetation. *Journal of wind engineering and industrial aerodynamics*, 2013.

[5] Ansys Inc. *Ansys Fluent 15.0 Theory Guide*, November 2013.

[6] Donald F. Young. *A brief introduction to fluid mechanics.* John Wiley & Sons Inc., 2010.

[7] J. O. Hinze. *Turbulence.* McGraw-Hill Publishing Co., 1975.

[8] B.E. Launder and D.B. Spalding. *Lectures in mathematical models of turbulence.* Academic Press, 1972.

[9] Y.-T. Wu and F. Porte-Agel. Large-eddy simulation of wind-turbine wakes: evaluation of turbine parametrisation. *Boundary-Layer Meteorology*, 2010.

[10] L.P. Chamorro and F. Porte-Agel. Effects of thermal stability and incoming boundary-layer flow characteristics on wind-turbine wakes: a wind-tunnel study. *Boundary-Layer Meteorology*, 2010.

[11] Office of Energy Efficiency and Renewable Energy. The inside of a wind turbine.