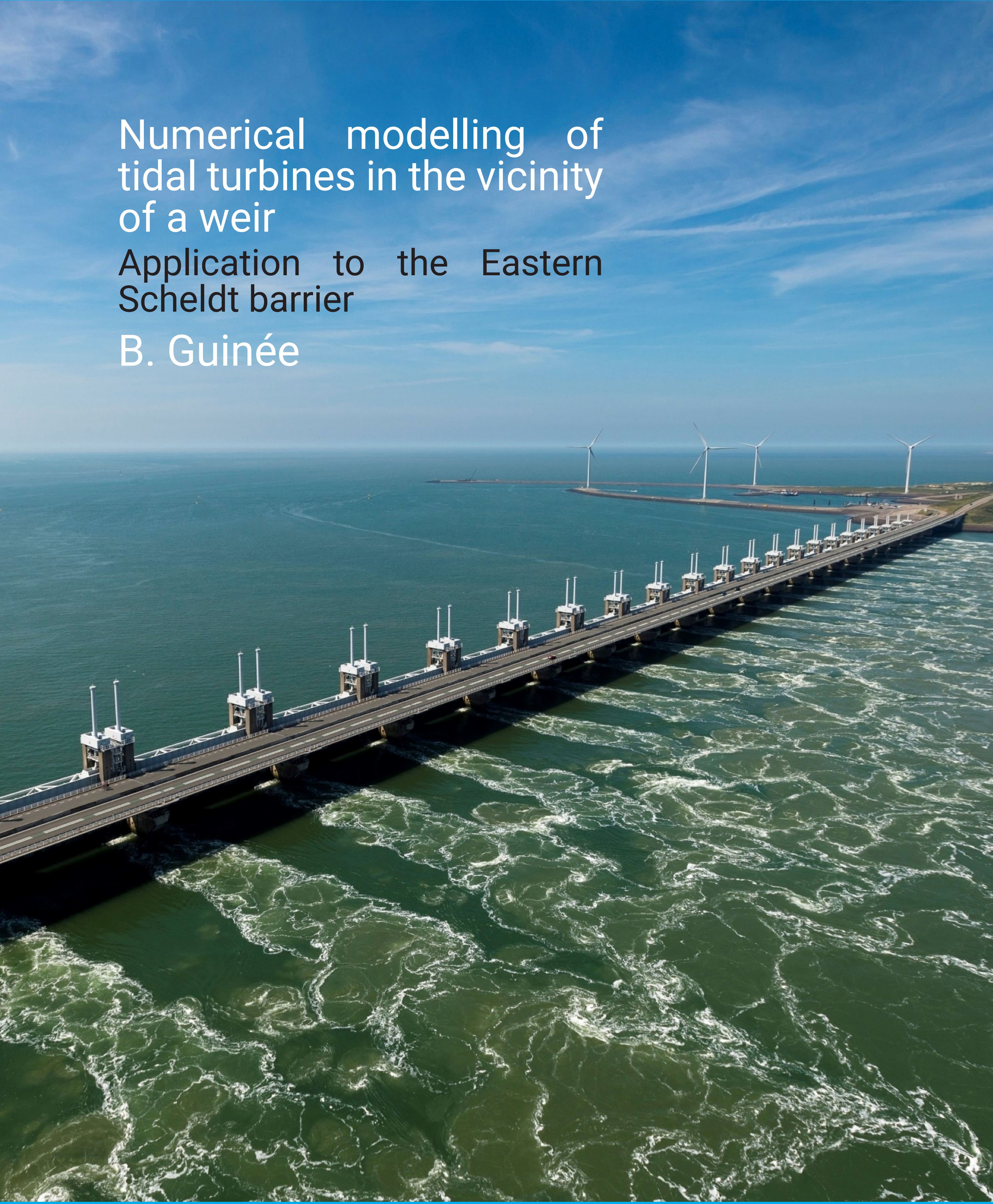
Numerical modelling of tidal turbines in the vicinity of a weir

Application to the Eastern Scheldt barrier

B. Guinée



TUDelft

Numerical modelling of tidal turbines in the vicinity of a weir

Application to the Eastern Scheldt barrier

by

B. Guinée

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

Student number:	4365275	
Project duration:	August, 2020 – June, 2021	
Thesis committee:	Dr. ir. R.J. Labeur,	TU Delft
	Dr. ir. M. Zijlema,	TU Delft
	Dr. S.R. de Roode,	TU Delft
	Ir. A. Bijlsma,	Deltares

Cover: VVV Zeeland

Preface

Before you lies the thesis 'Numerical modelling of tidal turbines in the vicinity of a weir'. This thesis is the last step to obtaining my Master of Science title in Hydraulic Engineering at the Delft University of Technology with the specialization of Environmental Fluid Mechanics. I have always been interested in innovative techniques and the last years I have developed an increasing interest in sustainability. These two are greatly combined in the subject of this thesis which made me very motivated to get the most out of this project. The combination of tidal turbines and numerical modelling made the subject challenging as I had to dive into two subjects at once, but I am very proud of the result.

I would never have reached this result without the people that supported me in writing this thesis. In first I want to thank the committee chairman Robert Jan Labeur for familiarizing me with his very own FinLab finite element model and always being open to help me. It was a pleasure to work together. Next, I want to thank the other committee members. I would like to thank Marcel Zijlema for his immediate action when this was needed, making it possible for me to finish this thesis. I also would like to thank Marcel for the personal meetings which we had in which we were able to discuss the details of the content of the thesis. I want thank Arnout Bijlsma for all his constructive feedback on as well the content of the thesis as the thesis structure. This was always very helpful. I would also like thank Arnout for bringing me in contact with the people of Deltares who provided me with all the measurement and simulation data of the Eastern Scheldt field case. I want thank Stephan de Roode for always being enthusiastic during the committee meetings, the provided feedback and questions helped me a lot to look critically at my own work. At last, I specially want to thank Merel Verbeek who provided me with all the data that I could wish for. Further, Merel's enthusiasm about the subject tremendously motivated me and our meetings were always very helpful and pleasant.

B. Guinée
Delft, June 2021

Abstract

Tidal energy has a large potential to contribute to achieving the sustainability goals in the Netherlands, due to the long coastline and many estuaries. The investment costs for the implementation of offshore tidal energy are however, still a drawback. A possibility to lower the investment costs for tidal energy is to implement tidal energy extraction into existing coastal infrastructure. An example of this is the current pilot project in the Eastern Scheldt barrier where five Tocardo turbines are attached to the barrier's geometry. The flow passing the turbines in the barrier is contracted by the barrier's geometry and the weir of the foundation of the barrier. Deltares executed an environmental impact assessment for the pilot project with a measurement campaign and a computationally expensive but accurate blade resolved numerical model. In this thesis, a cheaper computational model is added to the these assessment tools. To do so, a turbine parameterization is introduced in the finite element model FinLab. The relatively cheaper computational model should make it possible to execute more extensive sets of simulations of the pilot project in the future.

FinLab solves the non-hydrostatic three-dimensional incompressible Navies-Stokes equations in an unstructured mesh with an optional moving free-surface. The used meshes are refined in certain areas of complex flow, such as the turbine parameterization, near the bed, the recirculation zone of the weir and the wake of the turbine. An extra source term is added to the Navier-Stokes equations on a chosen number of turbine integration points, to implement the turbine into the mesh. The placement of these integration points in a two-dimensional mesh is rather simple, the integration points are evenly distributed over the diameter of the turbine. In a three-dimensional mesh, a Fibonacci series is used to implement an actuator disc (AD) into the model domain. A benefit of the Fibonacci circle is that each turbine integration point represents a similar area.

The force in the turbine integration points is determined by three methods: an actuator disc with a uniform thrust distribution (uniform AD), a rotational averaged actuator disc blade element momentum method (AD-BEM) and a non-rational averaged actuator line blade element momentum method (AL-BEM). In the uniform AD method, a measured thrust force is divided equally over all turbine integration points. In the BEM methods axial and tangential forces are determined for each turbine integration point based on the local flow speed and experimental lift and drag coefficients. With these methods, it is possible to estimate the generated power and thrust by the turbine. A tip-correction is added to the AD-BEM method to achieve accurate results of the simulations. Furthermore, in both the AD-BEM and AL-BEM methods a small actuator disc is applied in the nacelle region to reproduce the nacelle's blockage of the flow.

The uniform AD, AD-BEM and AL-BEM parameterizations are validated with measurement data of flume experiments of a single turbine, flume experiments with different turbine-weir geometries and flume experiments with multiple turbines. The combination of these last two sets of experiments does very well represent the situation in the Eastern Scheldt barrier. As stated above, the AD-BEM and AL-BEM methods are able to estimate the turbine performance. The results of the AD-BEM simulations show a relative error of the turbine's generated power of within 20% in comparison with the flume experiments. The accuracy of the thrust fluctuates more. The AD-BEM method shows to be able to accurately estimate the location of optimum power harvesting in a combined turbine-weir geometry. Both the power and thrust accuracy depend on the mesh resolution and the turbine's rotational speed. Accurate results can be obtained by fine-tuning these settings. The AL-BEM method should be able to produce at least similar similar accuracy to the AD-BEM method when it comes to power and thrust. The AL-BEM method in this thesis is however, not yet able to do so.

The results of the simulations with the three methods are also compared with time-averaged velocity measurements of the flume experiments. In the near wake, the uniform AD method does not imply any wake rotation. The rotation of the wake is represented accurately in a time-averaged sense by the AD-BEM method. The AL-BEM method adds transient features such as the downstream trailing tip-vortices to the flow. The velocity shear and turbulent eddies in the near wake as caused by the different methods influence the TKE in the near wake and the recovery of the far wake. The uniform AD under-estimates the wake recovery. The extra velocity shear in the AD-BEM method improves the accuracy but still under-estimates the recovery rate. Resolving the transient features of the near wake in the AL-BEM method further improves the accuracy of the wake recovery. In the simulations with the weir, the geometry seems to be dominant and the wake recovery rate is more accurate than in the simulations without the weir.

The AD-BEM method is applied to the Eastern Scheldt field case. Five turbines are introduced in a mesh which represents Roompot 7 to 9 in a simplified manner. Due to the simplification of the mesh, the resistance of the barrier on the flow appeared to be too low. Therefore, the simulations are executed with an upstream discharge boundary. This discharge boundary makes the model less useful to estimate the environmental impact of the turbines in the barrier and at the moment not yet an alternative for the blade resolved model by Deltares when it comes to estimating these effects. With the discharge boundary, the AD-BEM model predicts the average thrust over the five turbines with a relative error of 3% and the average power with a relative error of 12%. This is only slightly less accurate than the results by the blade resolved model by Deltares, while the computational costs of the Eastern Scheldt field case simulations are only a fraction of the earlier executed blade resolved simulations by Deltares.

Nomenclature

α	[°]	Inflow angle on aerofoil
β	[°]	Inclination angle aerofoil
γ_0	[-]	Elliptical factor
Δ_e	[m]	Element size
ϵ	[$\mathrm{m}^2/\mathrm{s}^3$]	Energy dissipation rate
η	[m^2/s]	Dynamic viscosity
θ	[-]	Implicitness factor
θ	[rad]	Angle of radius of turbine integration point w.r.t. y-axis
θ_0	[-]	Coefficient to form elliptical factor
λ	[rad]	Total rotation angle of turbine blades w.r.t y-axis
μ_t	[m^2/s]	Turbulent viscosity
ν	[m^2/s]	Molecular viscosity
ρ	[kg/m_3]	Density of flowing matter
σ_k	[-]	Coefficient k-ϵ turbulence model
σ_k	[-]	Coefficient k-ϵ turbulence model
τ	[N/m^2]	Stress
ϕ	[°]	Inclination angle resulting velocity
ω	[rad/s]	Rotational speed turbine
A	[m^3/s]	Blade swept area of the turbine (rotor)
a	[-]	Induction factor
B	[-]	Blockage ratio of turbine
c	[m]	Chord length
$\mathrm{C_D}$	[-]	Experimental drag coefficient
C_μ	[-]	Turbulence model coefficient
$\mathrm{C_L}$	[-]	Experimental lift coefficient
$\mathrm{C_P}$	[-]	Power coefficient
$\mathrm{C_s}$	[-]	Smagorinsky constant
$\mathrm{C_T}$	[-]	Thrust coefficient
$\mathrm{c_1}$	[-]	Coefficient k-ϵ turbulence model
$\mathrm{c}_{\mathrm{p}\epsilon}$	[-]	Coefficient k-ϵ turbulence model
$\mathrm{c}_{\mathrm{d}\epsilon}$	[-]	Coefficient k-ϵ turbulence model
D	[m]	Diameter of turbine
dr	[m]	Width of annulus
$\mathrm{F_A}$	[kN]	Axial force with respect to the turbine
$\mathrm{F_D}$	[kN]	Drag force created by flow over aerofoil
$\mathrm{F_L}$	[kN]	Lift force created by flow over aerofoil
$\mathrm{F_T}$	[kN]	Tangential with respect to the turbine
Fr	[-]	Froude number
h	[m]	Water depth
$\mathrm{h_w}$	[m]	Height of weir
k	[W]	Kinematic energy
$\mathrm{k_s}$	[m]	Nikuradse roughness
$\mathrm{l_m}$	[m]	Mixing length
P	[W]	Power extracted from flow by turbine
p	[N/m^2]	Pressure
R	[m]	Turbine radius
Re	[-]	Reynolds number
r	[m]	Radial position on turbine blade
S	[kN]	Momentum source

S_a	[kN]	Axial momentum source
S_k	[W]	Kinetic energy source term
S_t	[kN]	Tangential momentum source
S_w	[kN]	Additional downwash momentum source
t	[s]	Time
u_∞,u_0	[m/s]	Flow velocity in the undisturbed region upstream
u_a	[m/s]	Axial flow velocity on turbine
u_r	[m/s]	Resulting velocity on aerofoil
u_{rw}	[m/s]	Resulting velocity on aerofoil including additional downwash
u_t	[m/s]	Tangential velocity on turbine
u_w	[m/s]	Additional downwash velocity
x^*	[-]	Ratio of distance to weir over turbine diameter
z	[m]	Vertical position in water column

Abbreviations

AD	Actuator disc
ADCP	Acoustic Doppler current profiler
ADV	Acoustic Doppler velocimetry
AL	Actuator line
BEM	Blade element momentum
BEMT	Blade element momentum theory
CFD	Computational fluid dynamics
CG	Continuous Galerkin
DG	Discontinuous Galerkin
FEM	Finite element model
GIS	Galerkin interface stability
LES	Large eddy simulation
RANS	Reynolds averaged Navier-Stokes
TKE	Turbulent kinetic energy
TSR	Tip-speed ratio

Contents

1 Introduction

1.1. Tidal Energy

Climate change is one of the major problems we are facing as a society today. The electricity generation by burning fossils fuels plays a large part in this process. The Dutch government strives to generate all electricity from renewable sources in 2050 (Ministerie van Economische Zaken en Klimaat, 2020). To reach this goal more resources should be investigated, one of them being extracting energy from the tides.

Tides are the rise and fall of seawater levels driven by the gravitational pull of the sun and the moon on the water bodies on earth. Tidal flows are complicated by Coriolis, the bathymetry of the seas and the presence of landmasses while daily inequalities are caused by the relative motions of astronomical bodies. Nevertheless, due to the predictability of the tide generating forces, tides are very predictable themselves, which is an advantage of tidal energy compared to, e.g., solar and wind energy.

Energy can be extracted from the tides by either using the tidal range or the tidal flow. When using the tidal range a dam must be constructed to create a head difference, energy can be extracted by allowing the water to flow through the dam, while passing turbines. An example of this technique is the project in La Rance (France), which was the first tidal power station in the world and still is one of the largest today (Borthwick, 2016). When using the tidal flow to generate electricity, free stream turbines can be placed in the flow to extract energy from it.

In the Netherlands, due to the long coastline and many estuaries, tidal energy has a large potential (estimated to be between 20 to 128 MW over 9 locations (Scheijgrond, 2015)). Unfortunately, the costs of offshore tidal energy are still a drawback, in the UK the levelized cost of tidal energy is estimated 2 to 5 times higher than the cost of wind and solar energy (Borthwick, 2016). A possibility to lower the investing cost of tidal energy could be to implement tidal energy extraction into existing coastal infrastructure. An example of this is the current project in the Eastern Scheldt barrier. The Eastern Scheldt is a basin in the southwest of the Netherlands which is protected by the Eastern Scheldt barrier as a part of the Dutch delta works. The barrier can completely close off the Eastern Scheldt from the sea. When the barrier is open, water driven by the tides flows at high speeds through the gates of the barrier. This high velocity flow makes the barrier very suitable for the extraction of energy. In 2015 Tocardo placed a pilot project of five turbines in one gate of the barrier (Leopold & Scholl, 2019), a picture of this can be seen in figure 1.1.

There is still a long way to go before tidal turbines can be implemented into coastal infrastructure on a large scale. Research should be executed to determine, among others, an estimate of the energy production, the optimum mounting position for maximum energy extraction by the turbines, the stability of the coastal structures and the environmental impact of future tidal power stations. When tidal turbines are implemented in coastal infrastructure, the flow will be constricted by the sidewalls of its flow channels. In addition to this, in the case of the Eastern Scheldt barrier, a weir is present close to the location of the turbines. This weir contracts and complicates the flow around the turbine. A side view of the situation in the Eastern Scheldt in which the proximity of the weir clearly can be seen is given in figure 1.2. This thesis extends the already executed research about the flow through turbines in constricted flows near a weir.

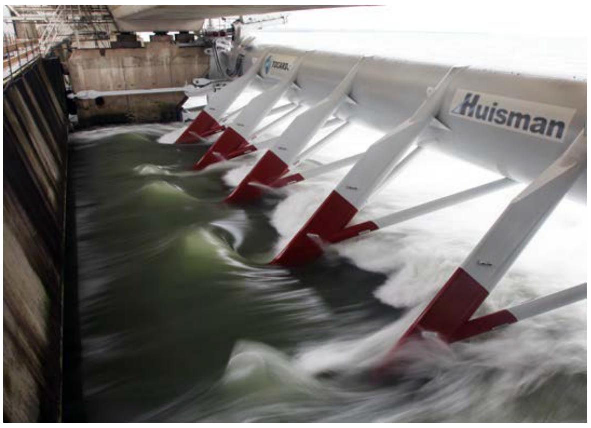


Figure 1.1: Turbines in operation in the Eastern Scheldt barrier (Leopold & Scholl, 2019)

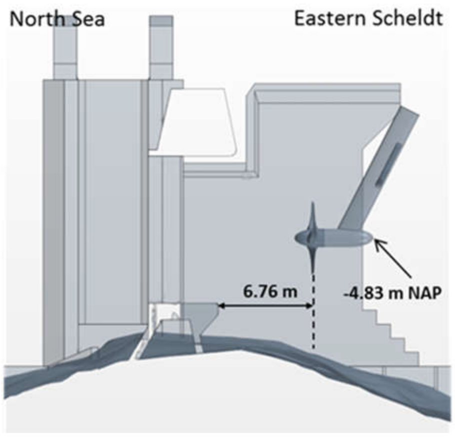


Figure 1.2: Sideview of turbine and weir combination in Eastern Scheldt (de Fockert & Bijlsma, 2018)

1.2. Previous works

This section shortly summarizes the previous executed studies on tidal energy extraction implemented in coastal infrastructure to find the research gaps. An extensive elaboration on more general turbine theories can be found in chapter 2.1 while a literature study on numerical modelling of tidal turbines can be found in chapter 2.2.

Verbeek et al. (2020b) investigated the influence of the proximity of a weir on the production of a tidal turbine by analysing field observations from the Eastern Scheldt barrier and by setting up a one-dimensional theoretical model of the flow passing a turbine in the proximity of a weir. The theoretical model can quickly give some rough design estimates in a very simplified environment, more about this model can be read in section 2.1.5. Furthermore, Verbeek executed experiments in a flume with a turbine-weir combination to research the influence of the weir on the power output and wake characteristics of the turbine (Verbeek et al., 2021) and the influence of the turbine-weir combination on the bed stability near the weir (Verbeek et al., 2020a). The measurements of these experiments were used to verify the one-dimensional model. More about the experiments can be read in section 4.1.1.

Guijt (2018) used discharge coefficients through the gates of the Eastern Scheldt barrier to represent turbines in a large scale flow model of the Eastern Scheldt basin to estimate the morphological effects of the implementation of turbines in the barrier. A discharge coefficient is multiplied with the discharge through a gate opening once turbines are implemented in it. These discharge coefficients were produced by a detailed CFD model made by Deltares (O'Mahoney et al., 2020). The complete geometry of one opening of the Eastern Scheldt barrier was modelled while three-dimensional (3D) drawings of the Tocardo turbines were loaded. The model produces very accurate flow patterns and discharge coefficients but requires enormous computational power.

1.3. Research gap

What lacks is a fairly detailed but relatively cheap three-dimensional numerical model, which can estimate the power extraction by and flow pattern around a turbine which is placed in the vicinity of a weir. A quicker and more general model could be a useful tool in future research to new turbine implementations in coastal infrastructure (bridges, weirs or storm barriers). When applied on the Eastern Scheldt situation, a quicker model could be used to do more experiments to determine, e.g., additional discharge coefficients, the optimum position within the gate openings for the turbines to be placed, wake characteristics etc.

1.4. Research question

The research gap can be translated in the following research question:

How can a horizontal axis tidal turbine be parameterized such that the turbine and flow properties are represented accurately in a three-dimensional numerical model?

The following sub-questions are defined to answer the research question:

- How accurate are the main characteristics of the turbine represented in the numerical model with the turbine parameterization?
- How accurate are the main characteristics of the flow around the turbine and the weir represented in the numerical model?
- How do the computational costs of the numerical model with the turbine parameterization compare to the computational costs of a blade resolved turbine model?

1.5. Research scope

The goal of this thesis is to implement a relatively cheap yet accurate computational model of a turbine in the finite element model FinLab. Similar to the computational-intensive model by Deltares (O'Mahoney et al., 2020) the model must be able to accurately represent the main flow characteristics and give a good approximation of the harvested energy. The final goal is to validate the model on the turbine implementation in the Eastern Scheldt barrier.

Accordingly, a turbine parameterization is implemented in FinLab. This turbine parameterization only represents horizontal axis tidal turbines. These are the most efficient kinds of turbines, see section 2.1.2 for a theoretical substantiation, and therefore the most commonly applied ones in the present day. In addition, horizontal axis tidal turbines are also applied in the Eastern Scheldt barrier. Therefore, when a turbine gets mentioned in this thesis, it can be interpreted as a horizontal axis tidal turbine.

In the Eastern Scheldt schematization, five turbines are placed side-by-side in one gate. Therefore, the numerical model is validated for the flow passing multiple turbines. Coastal infrastructure works are usually rather short in the streamwise direction while long in the direction perpendicular to the flow, which means that the placement of multiple turbines next to each other will probably occur more often in the future, making this validation process convenient. Array formations with multiple turbines in line are disregarded according to the same reasoning.

The process to a validated general flow model which is finally applied on the case of the Eastern Scheldt is an extensive and challenging one. To maintain a good overview this process is distinguished into three parts. In first, the model is validated in a simplified situation for the flow passing one turbine with a weir in its proximity. In second, the model is validated in for the flow passing multiple side-by-side placed turbines without a weir. In third, the model is applied on the situation in the Eastern Scheldt with multiple turbines and a detailed bed and barrier geometry.

2
Literature study

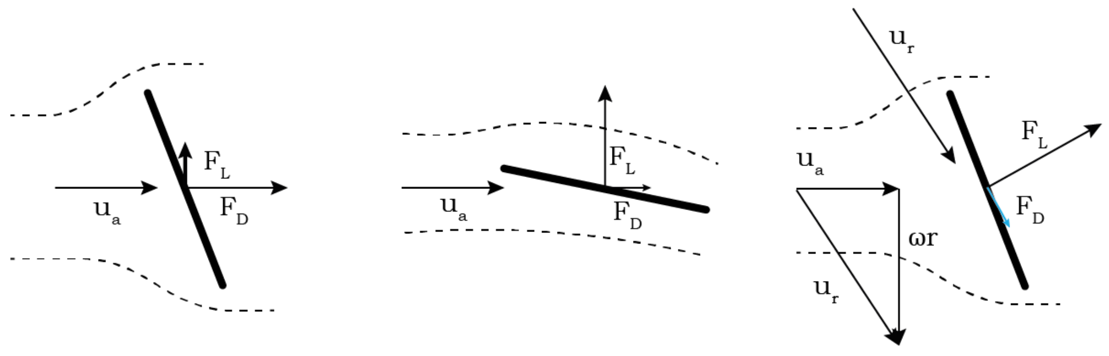
The literature study is split up in two mayor parts, a theoretical outline and a literature study on numerical modelling of turbines. The theoretical outline in section 2.1 describes the working principle of turbines and the expected flow features around turbines and weirs. The literature study on numerical modelling in section 2.2 describes several possible turbine parameterization methods and discusses the choices which can be made within the set-up of the numerical models in this thesis.

2.1. Theoretical outline

As stated above, this theoretical outline is set up to gain understanding of the processes which occur when a tidal turbine is placed in the vicinity of a weir. In first, in section 2.1.1 the working principle of turbines is discussed. Understanding this working principle is essential to correctly implement turbines into a numerical model. In the sections thereafter, the hydrodynamic processes around turbines and weirs are discussed. Understanding these processes is essential to understand the influence of the geometry on the performance of a turbine. In addition, the discussed hydrodynamic processes should be recognisable in the numerical model. This makes understanding these processes essential for validation of the flow patterns in the numerical model. At last, in section 2.1.7 a summarizing conclusion is drawn from the discussed theory.

2.1.1. Turbine blade theory

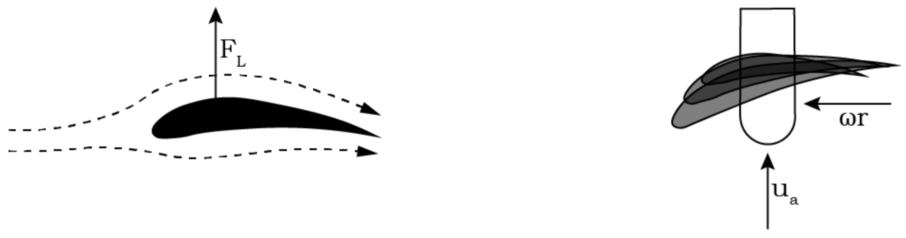
To make a start in the knowledge of harvesting power from flows, it is important to understand the essence of the blade theory. The most simple blade possible is just a flat plate. A plate can be used to harvest power by by placing it into a flow. When such a flat plate is placed perpendicular to a flow, it obtains a drag force (F_D). When slightly tilting the plate in the flow, apart from the drag force, a small lift force (F_L) is obtained, see figure 2.1a. Likewise, a highly tilted plated leads to a small drag and a large lift force, see figure 2.1b.



(a) Forces on a slightly tilted plate (b) Forces on a highly tilted plate (c) Forces on a slightly tilted rotating plate

Figure 2.1: Plate theory, adjusted from (van Bussel, 2011)

A rotational movement (ω) can be initiated by mounting such highly tilted plates to a horizontal axis. The rotational motion leads to a velocity of the plate perpendicular to the axial flow (u_a) as a function of the radius (ωr). Together, these lead to a resulting velocity (u_r) approaching the plate. Tilting the plate in a small angle with respect to this resulting velocity leads to a great amount of lift, see figure 2.1c. The resulting lift and drag forces on the blades are often rearranged in a force in the direction of the flow and a force in the direction of the rotating motion of the blade. These forces are then mentioned as thrust and torque respectively.



(a) Flow lines past an aerofoil

(b) Different orientations of aerofoil over the length of a turbine blade

Figure 2.2: Aerofoil theory, adjusted from (Wimshurst, 2018)

To further increase the lift force, modern turbine blades are constructed from a set of aerofoils. The flow passing an aerofoil causes the streamlines to converge more above the aerofoils than below the aerofoil. The accelerating flow above the aerofoil leads to a higher pressure underneath the aerofoil than above it (Bernoulli equation), a lifting force is created (Wimshurst, 2018). A typical aerofoil and its surrounding flow are shown in figure 2.2a. In addition, to further increase the efficiency, modern turbine blades are slightly twisted over their length. When rotating, the further away from the centre of rotation, the faster the blade moves. Consequently, the angle of attack of the resulting velocity with respect to the incoming flow increases towards the tip of the blade. Three aerofoils of a typical blade are shown in figure 2.2b, the smallest aerofoil (most perpendicular to the main flow) is the aerofoil most near the tip of the blade.

The number of blades does not influence the efficiency of the turbines. Turbines with many blades have a high torque due to the inertia of the blades and therefore rotate at a low rotational speed. Turbines with just two or three blades, have a small torque and therefore rotate at a much higher rotational speed. In the end, the turbines with the same blade swept area all produce the same amount of power if they work at their optimum tip-speed ratio (TSR). The TSR is the ratio of the speed of the tip of the blade over the incoming wind speed. When the blades are spinning too slow, a substantial part of the flow can pass through the gaps between the blades while no power is extracted from it. When spinning too fast, the blades will form a wall and avert flow through the blades.

2.1.2. Unconstricted flow

The basis for theoretical modelling of flow passing turbines was provided by Lanchester (1915) and Betz (1920). Both described a turbine as an actuator disc, with the size of the blade swept area, which extracts energy from a flow. Betz derived a one-dimensional theoretical model of the flow through this actuator disc, see figure 2.3 on the next page for an overview. With this model, Betz was able to derive a limit of the fraction of the incoming energy which can be harvested by a turbine from an unconstricted flow. Betz used conservation of mass, momentum and energy to derive a polynomial which represents the power extracted by the turbine over the kinetic energy of the flow upstream of the turbine. By differentiating this polynomial, the maximum fraction of energy to be possibly extracted by a turbine in an unbounded flow can be found. This limit appeared to be 16/27 of the incoming upstream energy as can be seen in equation 2.1. With A_R being the area of the turbine (rotor), ρ being the density of the fluid and u_∞ being the flow velocity in the undisturbed area upstream of the turbine.

$$P_{max} = \frac{16}{27}\rho\frac{1}{2}A_R u_\infty^3 \tag{2.1}$$

The limit of Betz is reached when the flow velocity downstream (u_4) in the wake of the turbine is equal one-third of the flow velocity in the undisturbed region upstream (u_∞) of the turbine. Doing the same for drag driven or vertical axis windmills the limit appears to be 5/27, which is the reason why horizontal axis turbines are the most applied form of turbines in the present day.

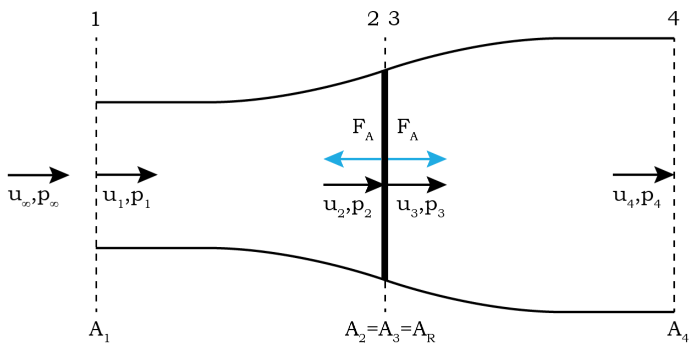


Figure 2.3: Actuator disc model by Betz, adjusted from (Ferreira, 2020)

The theory of Betz, with the actuator disc approximation, is used as a basis for many more theoretical studies and gives due to its simplicity good possibilities to verify these studies. Its full derivation is therefore given in appendix **??**. Note that, as the only difference is the density of the flow, all the wind energy theory as described above also applies to turbines extracting energy from water.

2.1.3. Constricted flow

The theory of Betz is only valid in a completely unconstricted flow, a flow without any boundaries in its proximity. For modern tall wind turbines, this could be a good approximation, but it is not for tidal turbines. Tidal turbines are almost always placed in the proximity of the bed, surface or shorelines. In the case of this thesis with the turbine placed within coastal infrastructure, this most certainly applies. Therefore some theoretical models of constricted flow near turbines are featured in the coming paragraphs.

Garrett and Cummins (2005) tried to come up with a theoretical model to describe the power potential of a tidal flow in a channel between two large bodies of water. In their model, the turbines were represented by fences of turbines which cover the complete cross-subsection of the channel. Accordingly, unlike the theory of Betz, no bypass is possible which makes the flow constricted. An illustrative overview of the theoretical model can be seen in figure 2.4, with x being the direction of the flow and $A(x)$ the area of the 'tidal fence'. In the paper, it is shown that an upper limit of the power extracted by turbines can be found based on friction as the turbines merely block the flow. This is an important claim as it means that there is a point at which adding more turbines in a constricted channel leads to less power production. In the paper, this process is referred to as 'chocking the flow'. An estimate of the maximum possible harvested fraction of energy from the maximum tidal head in the described channel appears to be between 20 and 24%.

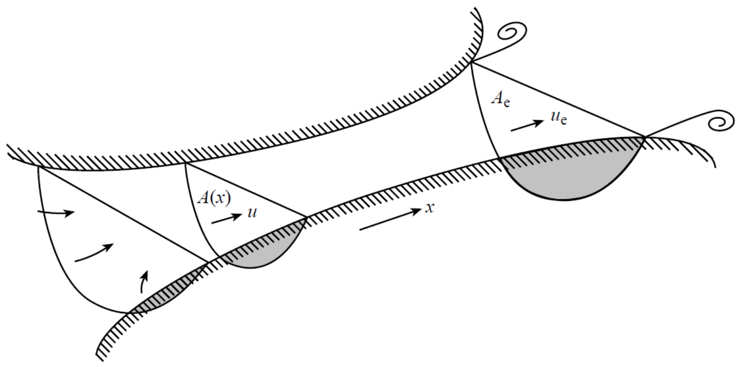


Figure 2.4: Model by Garrett and Cummins (2005)

Later, Garrett and Cummins (2007) extended their own work by allowing the tidal turbines to cover only a part of the cross-subsection of a tidal channel. Note that this still leads to a constricted flow as the sidewalls of the channel are still present in the proximity of the turbines. Garret and Cummins took a very similar approach as the approach by Betz (1920) for the model of unconstricted flow. Conservation of mass, momentum and energy (Bernoulli equation) are applied in a horizontally constricted flow. An illustrative overview of the modelled situation can be seen in figure 2.5.

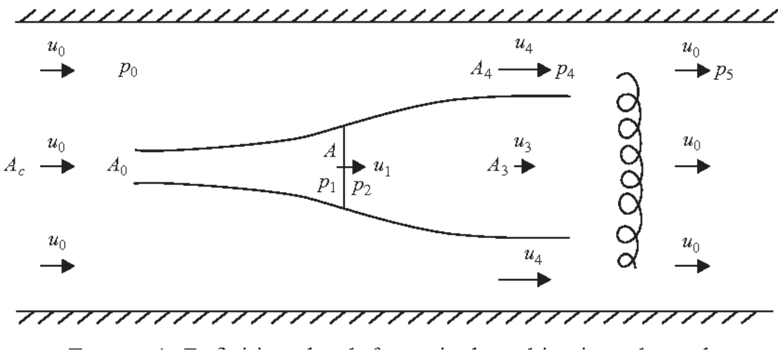


Figure 2.5: Model by Garrett and Cummins (2007)

To be able to verify their results with the model of Betz (1920), Garret and Cummins introduced a blockage ratio of the turbines: $B = A/A_c$. In this ratio A represents the blade swept area of the turbine and A_c represents the total flow area of the channel. Garret and Cummins were able to prove that the fractional power loss of the flow increases from 1/3 to 2/3 as the blockage ratio increases from limit case $B \to 0$ to the other limit case of $B \to 1$. This satisfies the theory of Betz.

$$P_{max} = \frac{16}{27}(1-B)^{-2}\rho\frac{1}{2}Au_0^3 \tag{2.2}$$

Furthermore, by derivation of equation 2.2, Garret and Cummins proved that the extracted power by the turbines maximizes at $u_3/u_0 = 1/3$ for all values of B, which also satisfies Betz theory. In this equation, it can be noticed that the maximum extractable power limit as derived by Betz (see equation 2.1) increases with the blockage ratio by the factor $(1-B)^{-2}$. This extra term states that an isolated turbine in bounded flow is more efficient than an isolated in an unbounded flow, but also again states that an upper bound for the generated power exist as too many turbines merely block the flow.

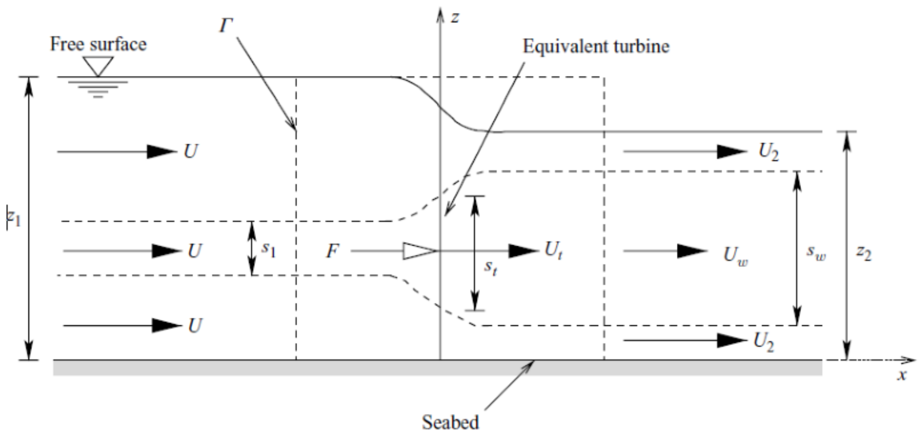


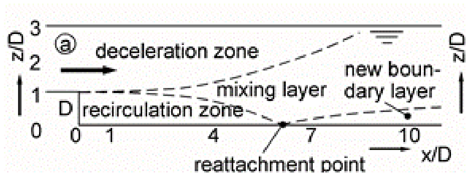
Figure 2.6: Model by Whelan, Graham and Peiro (Whelan et al., 2009)

The two theoretical models by Garret and Cummins are constricted by sidewalls of a channel, but not yet constricted by the proximity of the bed and the surface level. Whelan et al. (2009) set the first steps in adding a free-surface deformation as caused by a tidal turbine into the theoretical model, see figure 2.6. Vogel et al. (2016) extended this work by combining the vertical and horizontal bounded flow. To do so, a multi-scale model is applied, one on the turbine scale and one on the array scale. Again conservation of mass, momentum and energy is applied in the models while they are coupled through their boundary conditions. It was again explored that the blockage ratio of an array of turbines is proportional to the power extracted out of the flow. The turbines cause a change in the static and dynamic head of the flow, therefore it was concluded that the Froude number also has its impact on the fraction of extracted energy. The power extracted increases with increasing Froude number as the surface deformation and thus the blockage ratio also increases with increasing Froude number.

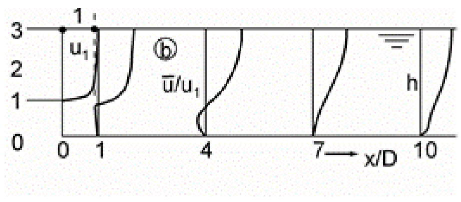
2.1.4. Flow over a weir

To be able to model the flow passing a turbine in the vicinity of a weir it is important to understand the flow over the weir too. It is important to note if the flow over a weir is sub- or super-critical as a hydraulic jump appears downstream of the weir when the flow is super-critical. In this thesis, the flow is assumed sub-critical. Typically, a logarithmic profile approaches the weir. Due to the higher bed, the flow accelerates which leads to a more rectangular flow profile and a lower surface level. The latter can be easily calculated by setting up a Bernoulli equation.

The flow at the backward-facing step at the downstream end of the weir separates if the step is steep. The flow becomes free with a mixing layer immediately downstream of the step and eventually becomes uniform again. When the turbine is located downstream of the weir, the latter of course depends on the distance between the weir end and the turbine and the height of the weir. A recirculation zone arises between the mixing layer and the bed and between the weir and the re-attachment point of the mixing layer. Downstream from the re-attachment point, a new boundary layer starts to grow until a new equilibrium is reached. The re-attachment point is typically located at a distance of five to seven times the height of the weir downstream of the weir end (Schiereck & Verhagen, 2016). An overview of the situation near a backward-facing step and its corresponding velocity profile over the vertical can be seen in figures 2.7a and 2.7b respectively.



(a) Flow situation



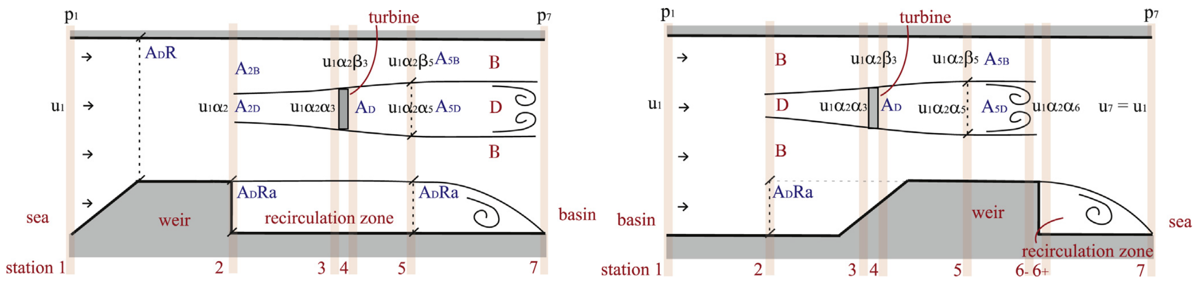
(b) Velocity profile

Figure 2.7: Flow situation near a backward-facing step (Nakagawa & Nezu, 1987)

2.1.5. Constricted flow with weir in bathymetry

The last step to completely modelling the flow of the researched situation of this thesis is combining the flow near the turbine with the flow over the weir. Verbeek et al. (2020b) extended the work of Garrett and Cummins (2007) and Vogel et al. (2016) by including the weir in the bathymetry and thus making a theoretical model of the described situation. The paper consists of field data analysis of the Eastern Scheldt barrier and data interpretation through theoretical modelling. In this subsection, the data interpretation and the theoretical model are discussed. The data acquisition setup is used to validate the numerical model for the Eastern Scheldt case and is therefore discussed in subsection 5.1.1.

An overview of the schematization of the model can be seen in figure 2.8. Again the flow is schematized by two stream paths, one passing through the actuator disc and one bypassing the disc at the top, bottom and lateral sides. From the field measurements, it can be concluded that the flow behind the weir separates, therefore a recirculation zone is added to the model. Due to the addition of the weir is, the bathymetry and model gets asymmetrical, therefore two cases are discerned. In the first case, the turbine is placed downstream of the weir. For the Eastern Scheldt turbines, this is the flood situation. In the second case, the turbine is placed upstream of the weir. For the Eastern Scheldt turbines, this is the ebb situation. An overview of the situations can be seen in figures 2.8a and 2.8b respectively. By using mass and momentum balances according to these figures, a system of four equations with four unknowns is set up which concludes as a theoretical model. This model can be used as the most simple design tool for the implementation of a turbine in the proximity of a weir. Again, it has been verified that when the weir height or both the weir height and the blockage ratio approach zero, the model converges to the model of Garrett and Cummins (2007) or Betz (1920) respectively.



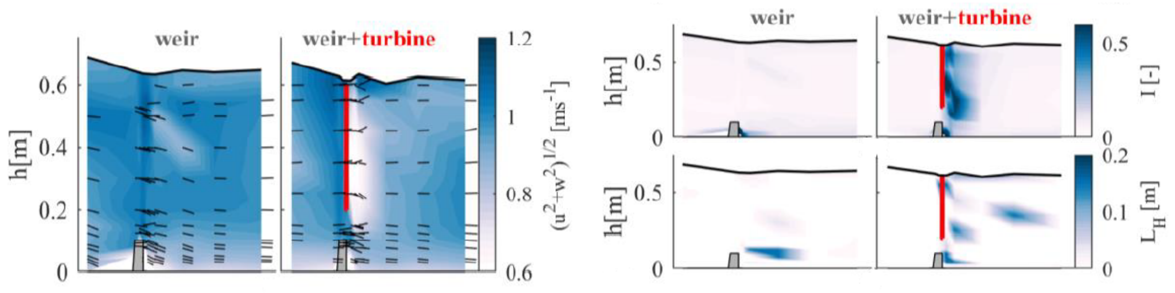
(a) Flood situation

(b) Ebb situation

Figure 2.8: Sketch of the one-dimensional approximation to the flow past a tidal turbine with a weir in its proximity (Verbeek et al., 2020b)

In addition to this paper, Verbeek conducted two sets of flume experiments. The first set focused on the effects of the turbines in the Eastern Scheldt on the bed protection of downstream of the barrier (Verbeek et al., 2020a). Bed protection stability is outside of the scope of this thesis and therefore further disregarded. The second paper focused on the harvested power and optimum weir-turbine geometries (Verbeek et al., 2021). The data of these experiments are the main tool to validate the numerical model, the setup is therefore described extensively in subsection 4.1.1.

Combining the conclusions of the three papers the flow patterns and turbine performance can be understood well. The weir and the turbine both increase the non-uniformity of the flow by vertically constricting the flow. These constrictions result in large flow velocities on top of the weir and below the turbine while leading to large velocity deficits behind the weir and the turbine. Downstream of the weir the flow expands over a length of approximately seven weir heights, the turbine typically generates a wake over a length of approximately ten turbine diameters. High turbulence intensities occur in the recirculation zone downstream of the weir and in the lower part of the turbine wake. The turbulent length scale, which is a measure of the eddy size, increases into the far wake as eddies tend to grow. In the recirculation zone downstream of the weir the eddies are suppressed by the turbine and moved closer to the bed. The turbine clearly suppresses the dissipation of energy in the recirculation zone, which could be a beneficial environmental effect. An overview of the complete flow situation with a turbine weir combination compared to the situation with only a weir as obtained from flume experiments can be seen in figure 2.9.



(a) The mean flow velocity along the centre line

(b) Upper: the turbulence intensity (I). Lower: turbulence length scale (L_H)

Figure 2.9: Comparison flow situations with and without turbine (Verbeek et al., 2020a)

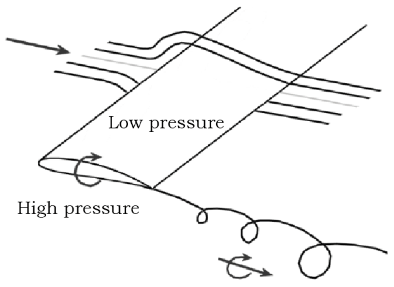
Analysing the turbine performance, the position of the turbine relative to the weir influences the wake geometry and therefore the power output. The weir increases the local blockage when placed upstream of the turbine and therefore increases the mass flux through the turbine and accordingly its power output. An optimum turbine position relative to the weir is found in the flume experiments, unfortunately, this position gives no certainty for the optimum weir-turbine geometry in the case of tidal flow. The flow in the flume experiments is generated by a constant discharge into the flume, this can be seen as a realistic river situation. However, in a tidal situation, the flow is generated by a varying water level, which leads to the earlier mentioned 'chocking of the flow effect'. This effect could lead to a different optimum positioning which should be investigated. The presented optimum position shows interestingly a geometry where the energy yield is high while the flow resistance is low, which could indicate that the flow is 'choked' just slightly. It occurs when the turbine is placed at the downstream end of the weir, the turbine then suppresses the flow separation and expansion downstream of the weir as earlier described above. The field data showed that the one-dimensional model is capable of estimating full-scale performance data.

2.1.6. Wake theory

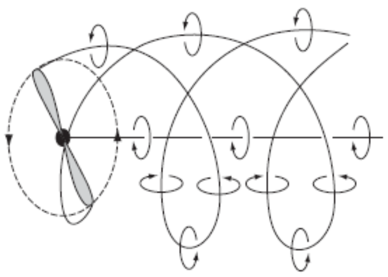
As described in the previous subsections, two wakes will form in the described flow. Downstream of the weir the earlier described recirculation zone arises while a wake will form downstream of the turbine. It is important to have a good understanding of the wake flows to be able to accurately model the flow passing the weir and the turbine. A wake flow is characterized by a large variation in velocity perpendicular to the flow direction while the variation in flow direction is small. In this subsection, the wake of the turbine is discussed.

A distinction can be made between the near wake and far wake. The near wake is the subsection close to the turbine where its influence is directly felt (Schepers, 2016). The far wake is considered as the area where the turbine is only felt globally and turbulent mixing is dominant. The near wake usually ends at a distance of two to four times the diameter of the turbine. A more specific distinction can be made when looking into the pressure gradient. The pressure gradient is a direct consequence of the energy extraction as explained in the actuator disc theory. In the near wake, the pressure gradient can't be neglected, while in the far it can.

The near wake is determined by the detailed rotor hydrodynamics. The lift by an aerofoil is created due to the higher pressure beneath the aerofoil than above. The same pressure difference initiates a flow near the tip of the blade. Near the tip, the water is able to escape by flowing around the tip, from the high pressure area beneath the aerofoil to the low pressure area above it (Abdulrahim, 2014). This flow decreases the total lift capacity of the blade, as it induces downwash. Together with the approaching flow, this outwards flow over the blade initiates vortices at its tip. These tip-vortices are convected downstream with the local flow velocity. The flow pattern of these tip-vortices can be seen in figure 2.10b. In the near wake, the velocity profile is not uniform. The lowest flow velocities occur near the nacelle and the tips of the blades while the support structure of the turbine can have a significant effect upon the flow field too (Myers & Bahaj, 2009). Due to turbulent mixing, the minimum velocity shifts to the centre line of the wake. This occurs at a distance of three to four turbine diameters downstream of the turbine. Downstream of this point, the far wake starts.



(a) Initiation of tip-vortices, adjusted from (Abdulrahim, 2014)



(b) Downstream convection of tip-vortices in near wake (Burton et al., 2002)

Figure 2.10: Initation and induction of tip-vortices

In the far wake, the tip-vortices are broken down by turbulent mixing and the minimum velocity is located at the centre line of the wake. Although experiments have shown that the velocity profile does not necessarily have to be axis-symmetric due to the influence of the bed and the free surface (Myers & Bahaj, 2009). Turbulent mixing with the ambient flow causes the velocity deficit to decrease over distance. In most research areas the far wake characteristics are far more important than the near wake characteristics. But due to the influence on each other, a complex near wake flow could influence the accuracy of the far wake flow. The far wake develops according to the self-similarity principle and can be approximated by a Gaussian shape. This principle implies that the velocity profiles are of the same shape at different cross-subsections in the wake (Uijttewaal, 2020). Scaling can be done by introducing a parameter for the maximum velocity deficit in the wake and the length scale over which this velocity deficit occurs.

Ouro and Stoesser (2019) studied the impact of a realistic incoming turbulence field through the turbine by the introduction of two dunes in the bathymetry of a CFD simulation and running a precursor simulation. The result showed that the turbulence induced by the dunes enhanced a faster recovery of the wake deficit due to an increased turbulence mixing. A similar process could be expected when the turbine is placed near the weir. In figure 2.11 an overview is given of the flow in the near wake with the tip-vortices and the flow in the far wake in these harsh environmental conditions induced by the dunes.

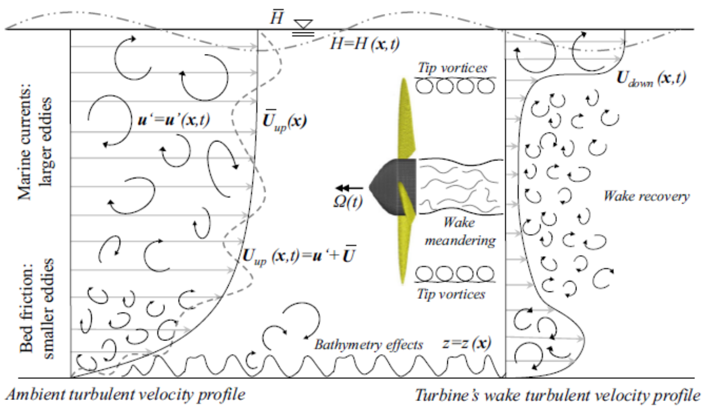


Figure 2.11: Near and far wake flow in harsh environmental (Ouro & Stoesser, 2019)

Wakes of turbines expand and grow in size as described by the self-similarity principle. Therefore, when multiple turbines are placed side-by-side, it can be expected that the wakes of the different turbines will interact with each other. Stallard et al. (2013) conducted an experimental study to the flow characteristics of the merging wakes of multiple side by side placed turbines in a flume. The experiments are used to validate the numerical model, the setup is therefore described more extensively in subsection 4.2.1.

Concluded from the studies is that if the turbines are placed at a distance larger or equal to three times the diameter of each other, the wakes behave very similar to the wake of an isolated turbine. When the velocity reduces by 80% in the near wake, the velocity will be recovered to a deficit of 20% of the ambient velocity at a distance of ten diameters downstream. For turbines placed within a distance of two diameters of each other, wakes start to merge within a distance of four diameters. The outermost wakes become slightly asymmetric while the centre wake narrows due to increased turbulence intensities. This leads to a faster wake recovery in the region between the wakes. From a distance of six diameters downstream and on, an almost constant velocity deficit can be noticed over the wakes. An overview of the development of merged wakes can be seen in figure 2.12.

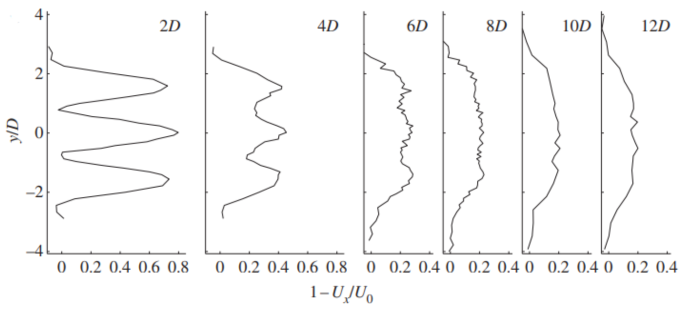


Figure 2.12: Wake development of side by side placed turbines (Stallard et al., 2013)

2.1.7. Conclusion

This theoretical outline describes the working principle of tidal turbines, which is helpful to parameterize the turbine into the numerical model, and describes the main characteristics of the flow around turbines and weirs which is helpful to validate the numerical model. Summarizing, the flow around the aerofoils of turbine blades initiate a lift and a drag force on the blade and the flow. These again lead to a thrust force and the power generating torque force. Both these forces increase over the blade radius.

There is an upper bound to the amount of power which can be extracted by turbines in a flow. In an unconstricted flow, this maximum is equal to 59.3%. This limit increases in a constricted flow where the blockage ratio of the turbine is higher. In contrast to this, there is also a friction based upper bound of the amount of power which can be extracted in a constricted tidal flow. If a tidal flow is 'chocked' by the turbines, the power generation will be higher when fewer turbines are placed. Placing the turbines downstream of a weir further increases the energy extraction by the turbines as a higher mass flux is pushed through the turbines while at the same time the turbine suppresses energy losses in the recirculation zone of the weir.

The forces performed on the flow by a turbine cause a wake to form behind the turbine. Turbine wakes can be divided into a near and a far wake. In the near wake complicated time-varying flows occur. The near wake rotates due to the rotation of the blades and contains downstream trailing tip-vortices due to the pressure difference over the blades. In the far wake an axial velocity deficit occurs which can be observed far downstream.

2.2. Literature study on numerical modelling of tidal turbines

This literature study on numerical modelling of tidal turbines is set up to be able to make a well-considered decision on how to parameterize the turbine in FinLab. In section 2.2.1, various parameterizations methods of turbine blades in numerical models are discussed. Section 2.2.2 discusses several parameterization methods of the turbine nacelle and support structures. Section 2.2.3 discusses several turbulence closure models. In section 2.2.4, the advantages and disadvantages of different combinations of the parameterization methods with the turbulence closure models are discussed. At last, in section 2.2.5 it is concluded which combination best suits the scope of this thesis.

2.2.1. Turbine parameterizations

Several proven techniques exist to represent a tidal turbine in a numerical model. The computational cost and accuracy of the flow model strongly depend on the choice of representation which makes it an important decision. The most common parameterization methods are discussed in this subsection.

Blade element momentum theory

In blade element momentum theory (BEMT) the blade swept area is split into several annuli. An illustrative representation of this segmentation of the blade swept area can be seen in figure 2.13. For each annulus, an iterative process is executed to find a force balance between the forces of the flow and the forces on the blade. The shape of the blade is taken into account when determining the lift and torque forces. The angle of attack of the flow on each blade subsection determines the lift and torque forces using experimental data of the aerofoils of the used blade. BEMT is only able to determine the forces on the blades, it lacks the ability to model the flow and wake formation around the turbine. Therefore BEMT is mainly used as a fast design tool for rotors and turbines. The accuracy of a BEMT model can be further increased by applying one of the many empirical corrections for the model. E.g., Whelan et al. (2009) came up with the earlier described free-surface correction to correct for its proximity while Prandtl (Burton et al., 2002) came up with a correction which can be applied to correct for the losses caused by tip-vortices.

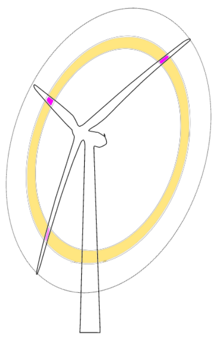


Figure 2.13: actuator disc divided into subsections of annuli in BEM (Ferreira, 2020)

Actuator disc

Actuator disc (AD) models, as discussed thoroughly for theoretical models in theoretical outile in section 2.1, can also be implemented in numerical models. An actuator disc with the size of the blade swept area is then implemented into a CFD model. When the flow passes the disc, it is subjected to extra source terms in the momentum equation. In CFD-AD approaches, the force subjected to the flow by the turbine is averaged over one full rotation of the turbine. Most simply, this can be done by introducing one force to flow averaged over the disc. This method is mentioned as the AD method in the remaining of the thesis. Due to the averaging over the turbine rotation period, AD approaches are not able to resolve for the, in subsection 2.1.6 described, tip-vortices.

A CFD-AD model can be made more detailed applying the splitting technique of the BEMT on the actuator disc, it is then named AD-BEM. In AD-BEM the actuator disc is split in annuli with a different, aerofoil specific, source term working on each annulus. In other words, the applied source terms now vary over the radius of the actuator disc. By resolving the flow around the turbine it is no longer required to execute the iterative process of BEMT to find the equilibrium between the loading of the flow and the loading on the blades. AD-BEM combined with a RANS model applied on tidal turbines was introduced in the study of Malki et al. (2013) and applied by Masters et al. (2013).

Actuator line

In an actuator line (AL) model, the momentum source terms are modelled only on several lines representing the turbine blades. The model is originally proposed by Sorensen and Shen (2002) and among others applied to tidal turbines by Baratchi et al. (2017). A typical representation of the AL model in a Cartesian grid is shown in figure 2.14. AL models work very similar to AD-BEM. The same blade element function are used to determined the generated thrust and torque by the blade, the method is therefore mentioned as AL-BEM in the remaining of this thesis. In contrast to the AD approaches, the extra momentum terms are not averaged over one rotation but are only applied at the mesh elements where a blade element is present at the given time-step. Due to this, AL models result in an over time-evolving solution which can capture tip-vortices and quite accurately model the near wake. A drawback of the AL-BEM method is that this leads to a computational more expensive model, due to the requirement of a finer mesh and smaller time-step due to the rotating blades and LES requirement.

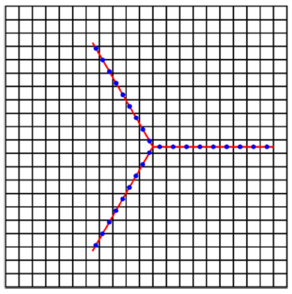


Figure 2.14: A schematic of uniform distribution of AL-BEM integration points with a Cartesian grid (Baratchi et al., 2019)

Blade resolved

In blade resolved techniques, the body of the turbine is embedded in the flow domain. Blade resolved techniques can produce a simulation of the flow around the blades with great accuracy by using a full CFD approach along the geometry of the blades. Often sliding-mesh techniques are used to position the turbine blades according to its TSR. An example of this approach is the STAR-CCM+ model of the tidal turbines in the Eastern Scheldt barrier by Deltares (Tralli et al., 2015). Such models can produce very accurate results when a sufficiently small element size is applied near the turbine. A drawback of blade resolved model is the large computational cost due to this small element size and the required re-allocation of the mesh with every time-step (Ouro & Stoesser, 2019). Different details can be captured by either using a RANS or an LES model for the blade resolved approach (Afgan et al., 2013).

2.2.2. Nacelle and support structure modelling

Implementing the nacelle of the turbine can be of great influence for the accuracy of the near wake flow pattern. Amongst others, Olczak et al. (2016) showed that the velocity deficit in the near wake in the centre of the blade swept area is not captured when not implementing the nacelle. In addition, modelling the nacelle leads to a more accurate estimation of the turbine performance as it can prevent the flow from bypassing through the nacelle area. Churchfield et al. (2015) introduced methods for simplified modelling of the nacelle and support tower of wind turbines in actuator line methods. They introduced three different actuator lines methods to represent the support tower and an actuator disc and actuator line to represent the nacelle. They concluded that the results in the near wake improved with 5-7% when using actuator line based tower and nacelle models, with the actuator line as an aerofoil with a cylindrical drag coefficient. These actuator approaches again can be implemented as source terms into the momentum equation for the mesh elements in which the support tower and nacelle are present. Simplified methods like these only help approximating the flow pattern and not actually resolve for boundary layers near the support tower and nacelle.

Cheng and Changhong (2019) introduced an immersed boundary method to include the effect of the nacelle which does resolve the boundary layers. In the immersed boundary method cells are categorised as either fluid points, solid points or forcing points (at least one neighbouring fluid and solid point). In this way, a sub-grid boundary can be created over which a wall-flow can be modelled. Apsley et al. (2018) applied a similar technique by introducing partly blocked cells. The cells in which the support tower and the nacelle are located are blocked for the percentage which they are covered by either of them. Partly blocking the flow makes the model very useful for finite volume approaches. Both these methods require a higher mesh resolution near the nacelle and support tower than the approach by Churchfield et al. (2015). At last, the nacelle and tower can, of course, be completely implemented in the mesh. This is done by, amongst others, Edmunds et al. (2017). Completely resolving the flow around the nacelle and tower of course leads to the highest accuracy results. Implementing these geometries into the mesh do, however, require a very high mesh resolution and do lead to flexibility constraints when moving the turbine through the mesh.

2.2.3. Turbulence modelling

An appropriate turbulence model should be chosen for the simulations. In FinLab, RANS and LES are available. RANS is available with a constant eddy, Bakmhetev mixing length model and a k-ϵ model. An explanation of why turbulence models are needed together with an elaboration of these available turbulence models is given in appendix B.4.

Shives and Crawford (2014) showed that, when RANS is used, the k-ω SST turbulence model leads to the most accurate results in the wake of an actuator disc parameterization. In addition, amongst others, Anwar-ul-Haque et al. (2007), Jehad et al. (2015) and Bijlsma et al. (2017) showed that the recirculation of a weir is better approximated by the k-ω SST turbulence model than by the k-ϵ turbulence model. The k-ϵ turbulence model tends to overestimate the mixing in the shear layers in both situations. As the k-ω SST turbulence model is not available in FinLab, the drawbacks of the k-ϵ model should be accepted.

When more accurate results are desired, LES can be considered. Sanderse et al. (2011) concluded from a literature review on earlier conducted CFD simulations on the wake forming of wind turbines that LES overall performs better than RANS. LES is able to solve the anisotropy of the turbulence in the wake which improves the results compared to RANS. This is confirmed by Churchfield et al. (2013) for the wakes of tidal turbines. LES resolves the large anisotropic turbulent motions while using a sub-grid model to represent the smaller isotropic motions. The error of such a model decreases with higher mesh resolutions (Chow & Moin, 2003). Sanderse et al. (2011) concluded that the computational cost of a single wake simulation of LES compared to RANS is usually two orders of magnitude larger due to this required higher mesh resolution. This could increase the computational time from hours to days. LES and the two-equation models would lead to a similar computational time when used on the same mesh resolution.

When choosing between RANS or LES, it is useful to take the applied turbine parameterization method into account. LES leads to a time-varying simulation, while RANS leads to a solution averaged over the time-step. The actuator disc approaches are averaged over one turbine circulation and therefore are most logically combined with a RANS method. The actuator line method evolves in time and is, therefore, more logically combined with LES. E.g, the tip-vortices which are formed in actuator line models would be averaged out by RANS but resolved by LES.

2.2.4. Comparison of the models

An overview of the different turbine parameterizations in combination with different turbine models and their corresponding capabilities, coming from a paper by Olczak et al. (2016), is given in table 2.1. Theoretical models such as discussed thoroughly in the theoretical outline in section 2.1 are included for completeness as 'Linear momentum'. The other categories give turbine models with their corresponding CFD technique. In the last column, the computational cost is given a score.

Ehrich et al. (2018) executed a study to compare three of the above-mentioned methods with each other and showed how enormous the difference in computational cost can be. The methods in the study by Ehrich et al. were a blade resolved (detached delayed eddy) simulation (36 million cells), a LES actuator line (22 million cells) and a BEMT simulation. The differences in computational times were clear, the simulations took 20 days on 360 cores, 25 hours on 360 cores and two minutes on one core respectively. An example of the computational time for a AD-BEM model is given by Malki et al. (2013). Their coupled model (1 million cells) took 1.3 hours to run on 24 processors. Ouro et al. (2019) conducted a LES AL-BEM study which took 15 days on 51 processors. The mesh consisted of a coarse mesh near the boundaries of 20 million cells and a fine mesh near the turbine of 53 million cells fine grid. The mesh was constructed according to the threshold of the minimum amount of calculation point on the blade axis as set up (Sorensen & Shen, 2002). Of course, the computational time depends on a lot more than just the chosen parameterization: processor speed, number of elements, the efficiency of the CFD model etc. but as each method requires a certain amount of detail, the mentioned computational times should give a good indication of what can be expected.

Other studies which have examined the difference between the models are amongst others, Martinez et al. (2012), Martínez et al. (2014) and the earlier mentioned study by Olczak et al. (2016). Summarizing their conclusions, linear momentum theory (theoretical modelling) can be very useful as a quick design tool, while also being useful to get a good understanding of the physics. BEMT can be a good design tool for quick turbine blade designs isn't capable of determining flow and wake patterns. CFD combined with an implemented AD as a momentum sink is the cheapest numerical method. It can give a good approximation of the flow in the far wake but has its drawbacks in the near wake. Splitting the AD into annuli leads to AD-BEM which is the second cheapest numerical method. It is more detailed than the AD method as it determines the thrust and torque as a function of the radius. Also, the shape of the blade can be taken into account when determining the power production. In addition, several corrections can be applied to further increase the accuracy of the model. AD-BEM is concluded to give a good approximation of the power estimation while also approximating the far weak to a decent accuracy. The AL-BEM method is a lot more accurate in the near wake than the AD methods. Due to the accurate positioning of the blades, the model is able to capture the tip-vortices. The major drawback of the model is computational expensiveness. AD-BEM and AL-BEM show similar accuracy in the far wake. The blade resolved CFD methods are by far the most accurate methods in each subsection of the flow and could therefore be very useful if certain details are desired to be captured. Nevertheless, again the major drawback is the very large computational cost.

Table 2.1: Summary of capabilities and typical applications of turbine wake modelling approaches, courtesy of (Olczak et al., 2016)

Method	Predicted				Modelled				
	Mean loads	Unsteady loads	Far wake	Near wake	Wake rotation	Blade scale flow	Onset shear	Onset turbulence	Computational cost
Linear momentum	-	-	-	-	Zonal idealisation				Low
BEMT	✓	-	-	-	No wake				Low
AD (RANS)	-	-	-	-	No	No	Yes	No	Medium
AD-BEM (RANS)	✓	-	✓	-	Yes	No	Yes	No	Medium
AL-BEM (RANS)	✓	-	✓	(a)	Yes	No	Yes	No	High
AL-BEM (LES)	✓	✓	✓	(a)	Yes	No	Yes	Yes	Very high
RANS blade resolved	✓	✓	✓	(a)	Yes	No	Yes	Yes	Very high
LES blade resolved	✓	✓	-	(a)	Yes	Yes	Yes	Yes	Very high

(a) - validation needed

2.2.5. Conclusion

Several numerical turbine parameterizations methods are discussed in this literature study. When comparing the scope of this thesis with the capabilities of the different described models, it can be concluded that an AD combination with a RANS flow model seems the most appropriate option. The time-averaging character of both the AD methods and the RANS method seems to be a good combination for the creation of a relatively quick model while producing reasonable accurate results in the far wake. The k-ϵ turbulence model seems to be the most appropriate option within FinLab to close the system of equations. The AD model is chosen with both a uniform and a BEM force module. The uniform AD is able to quickly form quite accurate flow fields, while the AD-BEM model produces accurate estimates of the power production by the turbines. The relatively low computational costs of both the turbine parameterization and the turbulence model give opportunities to model several turbines next to each other in one model as is the case in the Eastern Scheldt barrier.

In the AD-BEM model, a tip-correction and nacelle module are needed. Both should increase the accuracy of the model. Due to its location with respect to the blades, the nacelle is expected to have a larger influence on the turbine performance than the support structure. Therefore, the focus is first placed on the implementation of the nacelle into the model. Most AD-BEM models from literature implemented the nacelle in the mesh. This also seems to be the most accurate method. It is however, a computationally heavy method as the mesh near the nacelle must be small enough to form an accurate boundary layer. This method also compromises the flexibility of the turbine implementation. A quality of the AD implementation is the fact that the AD can be placed on each location within the mesh, without having to adjust the mesh. Therefore, a simpler thrust force based method, similar to the method by Churchfield et al. (2015), seems the best option to represent the nacelle and support structure.

3 Method

To model the flow passing a turbine which is placed in the vicinity of a weir, several turbine parameterizations are implemented into the finite element model FinLab. FinLab is developed by (Labeur, 2009) at the Delft University of Technology. FinLab solves the non-hydrostatic three-dimensional incompressible Navies-Stokes equations in an unstructured mesh with an optional moving free surface. The model is extensively elaborated in appendix B.

In the conclusion of the literature study on numerical modelling of tidal turbines in section 2.2.5, it is concluded that the uniform AD and AD-BEM turbine parameterization methods together with a RANS k-ϵ model seem to best match the scope of this thesis. These turbine parameterizations are, therefore, implemented into FinLab. The results of the flume simulations in chapter 4 do, however, show that both the uniform AD and AD-BEM methods lead to an underestimation of the velocity shear in the near wake which seems to be critical for the far wake recovery. An AL-BEM turbine parameterization together with a LES turbulence model is better able to estimate the velocity shear in the near wake and is therefore also implemented into FinLab.

In this chapter, these turbine parameterizations and their implementations in FinLab are elaborated. In first, in section 3.1, the implementation method of the turbine in the mesh is elaborated. Thereafter, the uniform AD, AD-BEM and AL-BEM force modules are elaborated in section 3.2. Note that in all equations and figures in this chapter and the remaining of the thesis, the x-axis is defined as the axial axis through the turbine, the y-axis is defined as the horizontal axis through the turbine and the z-axis is defined as the vertical axis through the turbine.

3.1. Turbine implementation in mesh

Due to the possibility to create and solve for unstructured meshes, the mesh can be refined in certain areas of complex flow, such as the AD, near the bed, the recirculation zone of the weir and the wake of the turbine. One of the used refined 2DV meshes can be seen in figure 3.1. The meshes are expanded over the y-axis to create a three-dimensional mesh. The size of the mesh elements in the FinLab model can determine the level of detail in the model. The more mesh elements the smaller and more detailed motion can be captured but also the more expensive the model gets.

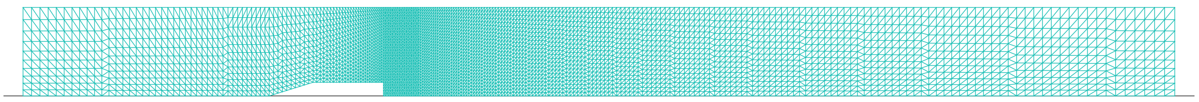


Figure 3.1: 2D mesh with weir

To implement the AD into the mesh, an extra source term is added to the momentum equation of the Navier-Stokes equations on a chosen number of turbine integration points. The placement of these integration points in a two-dimensional mesh is rather simple, the integration points are evenly distributed over the diameter of the turbine. In a three-dimensional mesh, a Fibonacci series is used to implement the AD in the model domain (Labeur, personal communication). A Fibonacci circle is an often in nature arising pattern, an example is the placement of seeds within a sunflower. The seeds in the sunflower grow from the centre of the flower and are placed such that each seed occupies a similar area almost independent of the distance to the centre. This phenomenon can also be used when creating the AD. When the turbine integration points are chosen according to the Fibonacci series, each source term is applied to a similar area.

In a Fibonacci series, each number is the sum of the previous two numbers in the series. For the first ten number this leads to: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55. When drawing a contiguous line of a quarter circle in a rectangular grid of the size of each step (1x1, 1x1, 2x2, etc.), the well-known Fibonacci spiral arises. The ratio of sequential numbers in the limit in the Fibonacci series is very close to the golden ratio of 1.618, formed of adding one to the irrational number 0.618. Plotting points in a two-dimensional plane with a rotation equal to this irrational number between every two sequential points forms the Fibonacci circle in which two points will never coincide with each other. In figure 3.2, a Fibonacci circle with 1254 turbine integration points is plotted in the geometry of the flume.

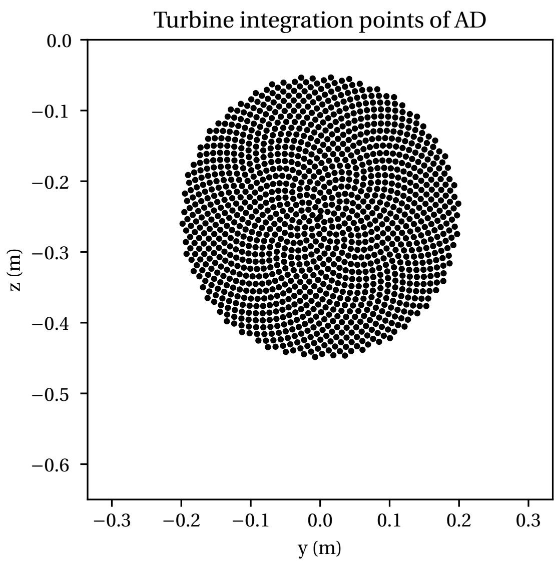


Figure 3.2: AD with 1254 integration points

The turbine integration points in both a two-dimensional or a three-dimensional mesh are defined independently of this mesh. A simple nested loop is created which loops over all mesh elements for each integration point, to determine in which mesh elements the integration points are located. FinLab uses unstructured meshes with triangular or tetrahedral elements for two- and three-dimensional meshes respectively. These triangles and tetrahedrons are again formed by respectively three or four nodes. In a three-dimensional space, the three node coordinates: x_n, y_n, z_n with $n = 1,..,3$ form a triangular plane. The side at which the fourth node x_4, y_4, z_4 is positioned relative to this plane can be determined by the sign of the determinant of the volume matrix of the formed tetrahedron (Hollasch, 1994).

$$A = \begin{bmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 \\ x_4 & y_4 & z_4 & 1 \end{bmatrix}$$

The turbine integration point $\mathrm{x_t}$, $\mathrm{y_t}$, $\mathrm{z_t}$ can now be introduced and shifted through the volume matrices as shown on the next page. If the fourth coordinate in each matrix is positioned on the same side as the fourth coordinate of the element matrix, it can be stated that the turbine integration point is positioned inside the tetrahedron. If one of the determinants is equal to zero, the volume of the particular defined tetrahedron is equal to zero and the turbine integration point is located on one of the planes of the tetrahedron. The same principle works for a two-dimensional mesh, with triangles instead of tetrahedrons, and thus one volume matrix less.

$$B = \begin{bmatrix} x_t & y_t & z_t & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 \\ x_4 & y_4 & z_4 & 1 \end{bmatrix} C = \begin{bmatrix} x_1 & y_1 & z_1 & 1 \\ x_t & y_t & z_t & 1 \\ x_3 & y_3 & z_3 & 1 \\ x_4 & y_4 & z_4 & 1 \end{bmatrix} D = \begin{bmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_t & y_t & z_t & 1 \\ x_4 & y_4 & z_4 & 1 \end{bmatrix} E = \begin{bmatrix} x_1 & y_1 & z_1 & 1 \\ x_2 & y_2 & z_2 & 1 \\ x_3 & y_3 & z_3 & 1 \\ x_t & y_t & z_t & 1 \end{bmatrix}$$

3.2. Force module

The forces of the blades are applied to the flow by adding a momentum source to the Navier-Stokes equations. The added force in the source term of one mesh element equals the sum of forces applied by all integration points positions in the element. As stated in the introduction of this chapter, three turbine parameterizations are used to determine these applied forces per turbine integration point. These are a very simple uniform AD model, a AD-BEM model and a AL-BEM model.

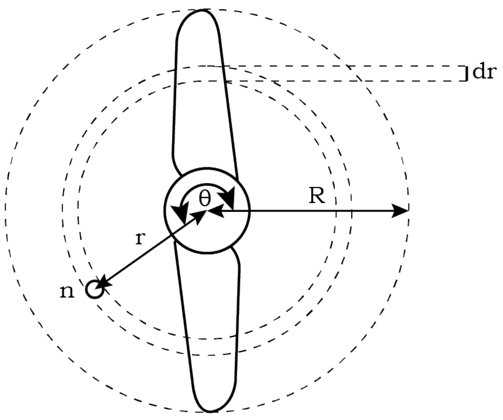
3.2.1. Uniform AD

In the uniform AD model, an overall thrust force (F_A) is equally distributed over the number of turbine integration points in the Fibonacci circle (N_t), see equation 3.1. In the experiments executed by Verbeek et al. (2021), this force has been measured, making this method easy applicable.

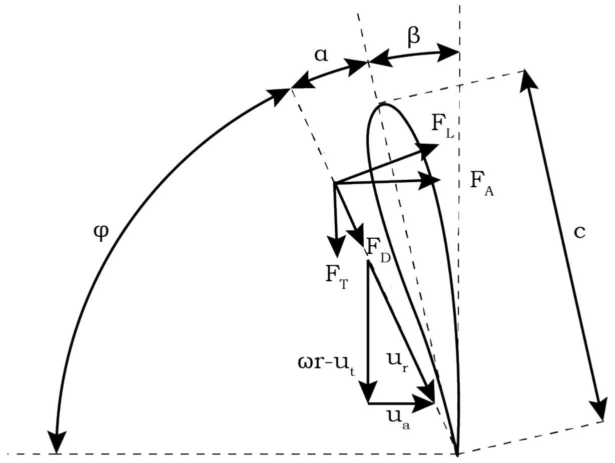
$$S_x = F_A / N_t \tag{3.1}$$

3.2.2. AD-BEM

The approach of the AD-BEM module in this thesis is based on the AD-BEM model by Edmunds et al. (2017), who based their approach on the method by Griffiths and Woollard (1978). In the method, in each time-step an axial and a tangential force are determined in all turbine integration points. These forces are based on the local angle of attack of the flow on the blade in combination with the local flow velocity.



(a) AD-BEM discretization applied on the Fibonacci circle



(b) AD-BEM model, inspired by (Edmunds et al., 2017)

Figure 3.3: Overview of the AD-BEM model

In figure 3.3b, an overview of the in the model defined forces on an aerofoil are given. The figure illustrates the velocities which are used to determine the loading on the blade element. Three numerically solved velocities are defined: the axial velocity at the blade (u_a), the tangential velocity near the blade section (u_t) caused by the presence of the turbine and the velocity due to the rotation of the blade (ωr). These three can be combined into a resultant velocity approaching the aerofoil (u_r) according to equation 3.2.

$$u_r^2 = u_a^2 + (\omega r - u_t)^2 \tag{3.2}$$

The inflow angle (ϕ) under which the resultant flow velocity approaches the aerofoil can be determined according to equation 3.3.

$$\phi = tan^{-1}((\omega r - u_t)/u_a \tag{3.3}$$

To determine the geometric angle of attack (α) under which the resultant flow velocity approaches the aerofoil, the twist of the blade (β) must be taken into account, see equation 3.4. The twist, as described in section 2.1.1, is a blade characteristic function over the radius.

$$\alpha = 90° - \beta - \phi \tag{3.4}$$

Lift and drag coefficients (C_L and C_D respectively) are experimentally determined as a function of this geometric angle of attack (α). Using the resultant velocity together with these coefficients and the area which is represented by one turbine integration point (A_c), the lift (F_L) and drag (F_D) forces of the aerofoil section can be determined following equations 3.5 and 3.6 respectively.

$$dF_L = \frac{1}{2}\rho u_r^2 C_L(\alpha) A_c \tag{3.5}$$

$$dF_D = \frac{1}{2}\rho u_r^2 C_D(\alpha) A_c \tag{3.6}$$

Using the inflow angle (Φ) these lift and drag forces can be transformed into an axial (F_A) and tangential (F_T) force, according to equations 3.7 and 3.8 respectively.

$$dF_A = dF_L \sin(\Phi) + dF_D \cos(\Phi) \tag{3.7}$$

$$dF_T = dF_L \cos(\Phi) - dF_D \sin(\Phi) \tag{3.8}$$

The partial forces from equations 3.7 and 3.8 are in reality only generated when one of the blades is positioned within the area represented by the turbine integration point. This is, of course, only the case during a certain fraction of one rotation, while the AD-BEM method determines axial and tangential forces on every integration point in every time-step. The fraction of time-steps at which the forces should be applied can be determined by calculating the blade solidity (s). The blade solidity is the relative area of the blades within the annulus represented by the turbine integration point. The thickness of this annulus (dr) is estimated by calculating the diameter of the area represented by one turbine integration point (A_c), see equation 3.9. The annulus, together with this area are illustrated in figure 3.3a.

$$dr = 2\sqrt{A_c/\pi} \tag{3.9}$$

$$r_1 = r - 0.5dr \tag{3.10}$$

$$r_2 = r + 0.5dr \tag{3.11}$$

The area of the blades can now be calculated by multiplying this thickness with the radius specific chord and the number of blades. Dividing this over the area of the annulus gives the blade solidity.

$$s = \frac{N_b\, c\, dr}{\pi(r_2^2 - r_1^2)} \tag{3.12}$$

The applied momentum sources can now be calculated by multiplying the forces as determined in equation 3.7 and 3.8 with this blade solidity. With equation 3.13, the axial force is applied in the x-direction, the tangential force is applied to the y- and z-direction with equations 3.14 and 3.15 respectively. The angle of the radius of the turbine integration point with the y-axis (θ) as defined in figure 3.3a, is used to apply the tangential force normal to the blade radius in the yz-plane. Note that the force is distributed over the points defining the triangles or tetrahedrons by dividing the source term by the dimension ($\mathrm{n_{dim}}$) plus one.

$$S_x = dF_A \frac{s}{\rho(n_d+1)} \tag{3.13}$$

$$S_y = -\sin(\theta)\, dF_T \frac{s}{\rho(n_d+1)} \tag{3.14}$$

$$S_z = \cos(\theta)\, dF_T \frac{s}{\rho(n_d+1)} \tag{3.15}$$

Tip correction

As stated in section 2.2.1, AD-BEM simulations are not able to capture the, in section 2.1.6 described, tip-vortices due to its averaging character over one blade rotation. But, as explained, these tip-vortices decrease the total lift capacity of a blade. Therefore, a tip-loss correction should be applied to the AD-BEM method to achieve accurate modelling of the power production. Often in BEMT and AD-BEM approaches, Prandtl's tip-correction is applied which reduces the induction on the blades near the tip and root. The momentum sources as stated in equations 3.13, 3.14 and 3.15 are then multiplied with the correction factor is given in equation 3.16. This leads to more accurate modelling of the power coefficient. However, when applied on a AD-BEM model, the momentum extraction on the flow is reduced too, leaving the flow unaffected by the presence of the blade near the tip.

$$f_{tip} = \frac{2}{\pi}\left(\exp\frac{-N_b(R-r)}{2r\cdot\cos(\phi)}\right) \tag{3.16}$$

Edmunds et al. (2017) introduced a new approach based on Prandtl's lifting line model. In the theory, an additional velocity (u_w) is added to the BEM module. This velocity represents the downwash as induced by the tip-vortices, it redefines the angle of attack on the aerofoil which leads to a decreased lift. In figure 3.4 an overview of the new situation is shown. The resultant velocity (u_r) and the angle of attack (α) are defined in figure 3.3b. This downwash velocity at the tip must lead to the efficient angle of attack (α_e) being equal to the angle of attack at which the created lift force equals zero. Most simply this can be approached by increasing the source terms which are applied to the flow towards the tip. An increased force to the flow leads to a new flow equilibrium with a decreased force applied to the blade near the tip. This can be achieved by introducing an elliptic function which gives a value of zero at the hub and a value of one at the blade tip, see equations 3.17 and 3.18.

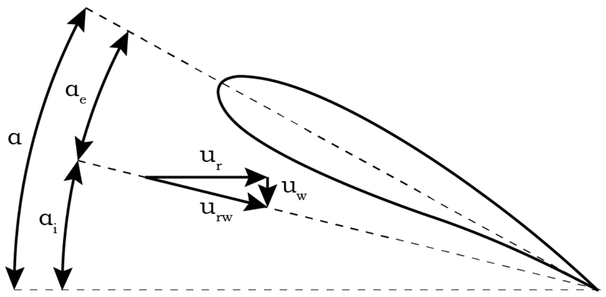


Figure 3.4: Redifining the angle of attack using the additional downwash, inspired by (Edmunds et al., 2017)

$$\theta_0 = (r/R) \tag{3.17}$$

$$\gamma_0 = 1 - \sin(\theta_0) \tag{3.18}$$

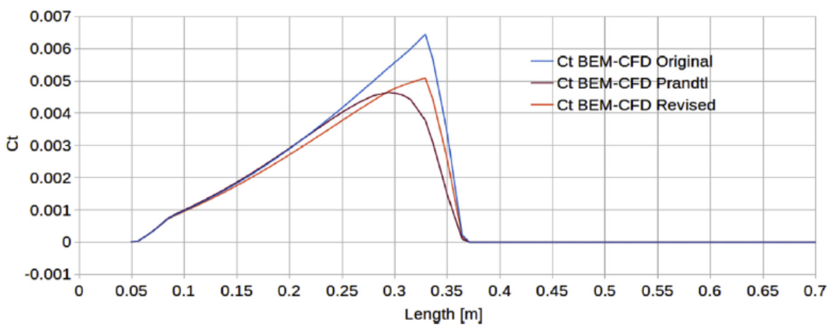
Applying this elliptic function to the source terms as defined in equations 3.13, 3.14 and 3.15 leads to the source terms including the tip-correction as given in equations 3.19, 3.20 and 3.21.

$$S_x = (1+\gamma_0)\, dF_A \frac{s}{\rho(n_d+1)} \tag{3.19}$$

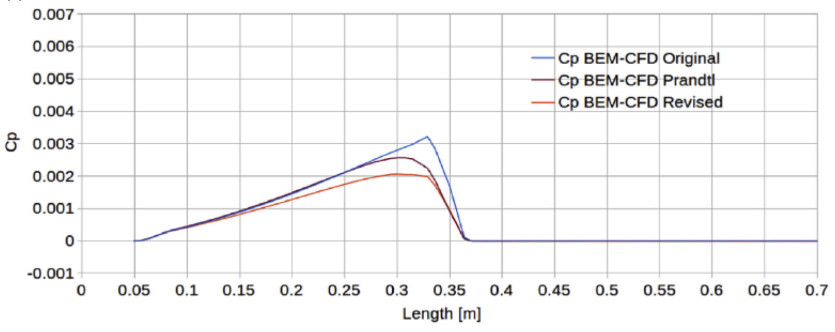
$$S_y = -(1+\gamma_0)\,\sin(\theta)\,dF_T\,\frac{s}{\rho(n_d+1)} \tag{3.20}$$

$$S_z = (1+\gamma_0)\,\cos(\theta)\,dF_T\,\frac{s}{\rho(n_d+1)} \tag{3.21}$$

Results of the tip-correction approach by Edmunds et al. (2017) showed more accurate results for the power coefficients compared to AD-BEM with no or Prandtl's tip-correction. The wake deficit also showed better agreement with experimental results. Compared to Prandtl's tip-correction, the new method leads to a larger velocity deficit in the wake, which agrees better with experimental results. Furthermore, with this method, it is no longer needed to introduce an extra turbulence generation term in the turbine area as the additional source term induced turbulence automatically. In figure 3.5 the thrust and power coefficient over the blade radius are shown without a tip-correction, with a Prandtl tip-correction and with the correction by Edmunds et al. These results from the paper by Edmunds et al. (2017) are used to validate if the applied tip-correction work properly in section 4.1.3.



(a) Thrust coefficient



(b) Power coefficient

Figure 3.5: Comparison of tip correction methods (Edmunds et al., 2017)

3.2.3. AL-BEM

The AL-BEM method in this thesis is based on the method as introduced by Sorensen and Shen (2002). The AL-BEM method has the exact same working principle as the AD-BEM method as used in this thesis. However, instead of applying the by the blade solidity averaged forces every time-step, the blade forces are only applied at the time-step specific location of the blade. This means that equation 3.12, in which the blade solidity is applied, is skipped.

In the AL-BEM method, due to its transient nature, the velocity sampling position is a topic of discussion in literature. In this thesis, the velocity is sampled at the centre chord line of the blade position in the previous time-step. Nathan (2018) argues that this is the position where the velocity is the least distorted by the induced velocities surrounding the blade. Other velocity sampling methods exist where the velocity is, e.g., sampled at an upstream location (Shen et al., 2009) or by using a Lagrangian method following the blades movement (Yang & Sotiropoulos, 2018). To sample the velocities at the chord centre line of the blade, a new set of turbine integration points is introduced into the mesh. In figure 3.6 on the next page, these new points are shown. In the present time-step, the chord centre chord line of the blades are located at the dark blue lines. The velocity is sampled at the lighter blue lines, the location of the centre chord line in the previous time-step. The time-step is chosen such that the blade rotates exactly to the next centre chord line in one time-step.

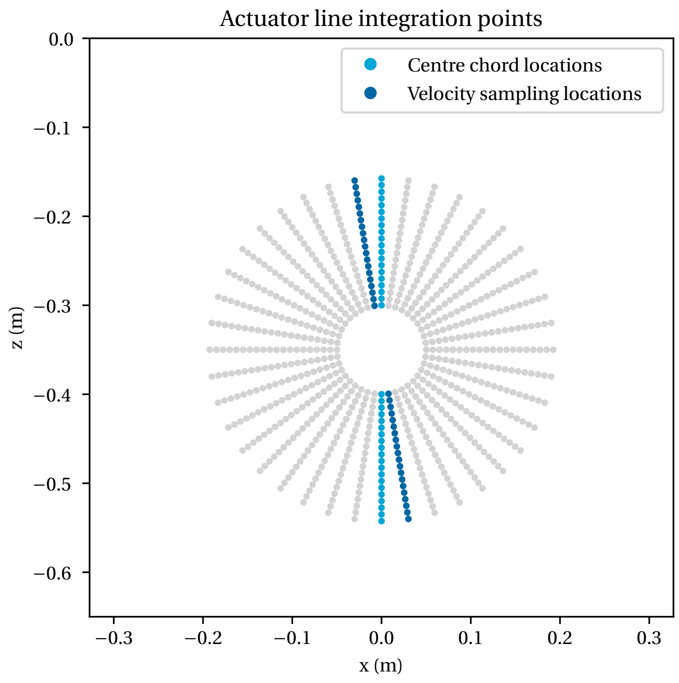


Figure 3.6: AL turbine integration points

In the AL-BEM method as introduced by Sorensen and Shen (2002), the blade force is distributed into the mesh by an Gaussian distribution function. In this thesis a different method is chosen. The presence of the Fibonacci circle routine gives the opportunity to distribute the force according to the shape of the blade. To do so, in each time-step, only the turbine integration points on which the turbine blades are currently located are activated. The BEM formulas are executed in these integration points with the velocities sampled as described above. These velocities are interpolated over the radius of the blade. The sampled velocities at the chord centre lines are transferred to all MPI partitions to make it possible to use velocities from other turbine integration points in the BEM equations.

To find the turbine integration points in the Fibonacci circle at which the turbine blade is currently located, the total rotation sum (λ) is determined. This is done by taking the product of the rotational speed (ω) and the time-step, see equation 3.22.

$$\lambda = \sum \omega \cdot dt \tag{3.22}$$

The position of the blades can be easily found by taking into account the chord distribution of the blade. To do so, two new angles can be introduced. The angle from the y-axis to the innermost point of the blade (λ_1), and the same angle to outermost point of the blade (λ_2). According to the chord distribution of the blade, these angles differ for each radial position. The integration point is located on the blade at this particular time-step when the angle between the radius of the turbine integration point and the y-axis (θ) is positioned between λ_1 and λ_2. This process can, of course, be executed for all the blades. See figure 3.7 on the next page for a further explanation. Figures 3.8a and 3.8b on the next page show the activated turbine integration points at the start of the first iteration for the Tocardo T2 turbine and the Goëttingen 803 turbine respectively. These are the turbines used in the flume experiments by Verbeek et al. (2021) and Stallard et al. (2013) respectively.

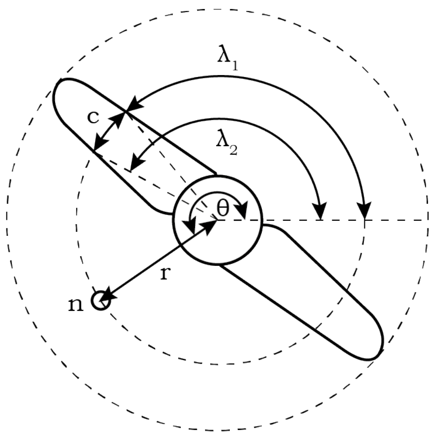
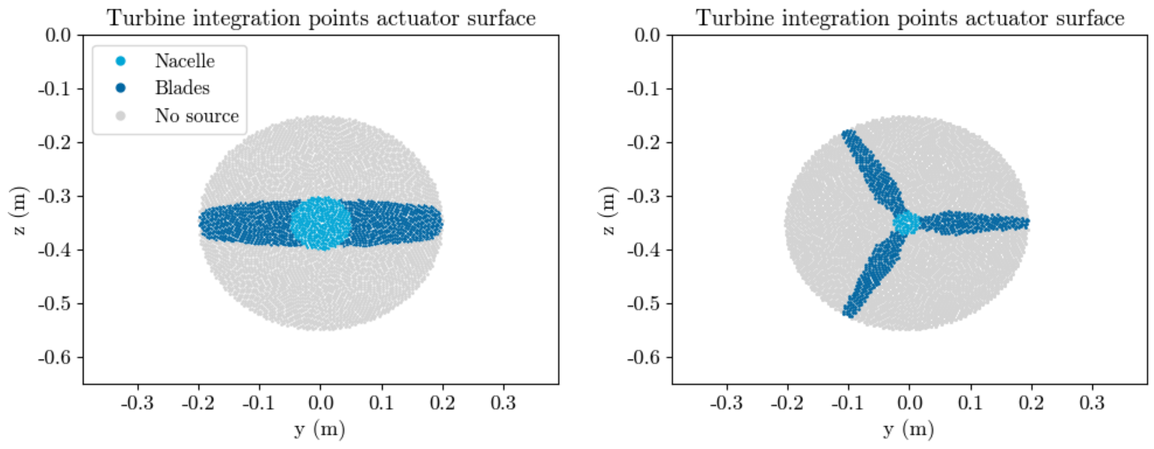


Figure 3.7: Position of blade in AL-BEM method



(a) Tocardo T2

(b) Goëttingen 803

Figure 3.8: Activated turbine integration point at at the start of the iterations

3.2.4. Nacelle modelling

The AD-BEM and AL-BEM methods require a nacelle module to prevent the flow from bypassing through the nacelle area. As concluded in the conclusion of the literature study on numerical turbine parameterizations in section 2.2.5, a thrust force based nacelle implementation seems to best match the scope of this thesis. In other words, on the points within the Fibonacci circle which are located within the diameter of the nacelle, only a thrust force is applied. Over the iterations of the simulation, this thrust force is adjusted such that the axial flow velocity becomes zero in one of the nodes of the element in which the centre of the nacelle is located. Note that only using the turbine AD to model the nacelle leads to a two-dimensional blockage of the flow. The flow will not be blocked in the nacelle area downstream of the turbine blade. Due to the downstream distance between the blades and the support structure, it is expected that the support structure has a lower influence on the turbine performance than the nacelle. The support structure is therefore not modelled at all.

To determine the thrust force in the centre nacelle turbine integration point, the nacelle thrust coefficient is adjusted by equation 3.23. The applied source term in this centre nacelle point is given in equation 3.24. This thrust coefficient is transferred to all MPI partitions using MPI communication to obtain a smooth velocity profile across the nacelle area. Equations 3.17 and 3.18 are used again to form an elliptical function towards the edge of the nacelle to form a smooth transition from the nacelle flow to the flow through the blades.

$$C_{nac} = \begin{cases} C_{nac} \cdot 1.001 & \text{if all}(u_{nac} > 0) \\ C_{nac} \cdot 0.999 & \text{if any}(u_{nac} < 0) \end{cases} \tag{3.23}$$

$$S_x = \frac{1}{2}\gamma_0 C_{nac} \rho u_{ref}^2 A_{cel} \tag{3.24}$$

3.2.5. Thrust and torque

In the AD-BEM and AL-BEM methods, values for the thrust and torque coefficients of the turbine are obtained to validate the results of the numerical simulations in FinLab with the experimental data. The thrust coefficient is defined as the ratio of the axial force obtained by all the turbine integration points over the incoming flow force in the undisturbed region upstream, see equation 3.25.

$$C_T = \frac{F_A}{\frac{1}{2}\rho {u_1}^2 A_R} \tag{3.25}$$

$$F_A = \sum_{n=1}^{n=N_t} dF_A \tag{3.26}$$

The power coefficient is defined as the ratio of the power extracted by the turbine over the incoming kinetic energy of the flow. The total power (P) is defined as the sum of all, with equation 3.8 determined, torque forces multiplied by the radial rotational velocity in all the N_t turbine integration points of the actuator disc. This is shown in equation 3.28.

$$C_P = \frac{P}{\frac{1}{2}\rho u_1^3 A_R} \tag{3.27}$$

$$P = \sum_{n=1}^{n=N_t} r\omega dF_T \tag{3.28}$$

Note that the thrust and power coefficients equations do not differ between the AD-BEM and AL-BEM methods. The only difference is that in the AL-BEM method some integration points add zero force to the thrust and power summations while in the AD-BEM method all points add to the summations.

4 Validation of turbine model with flume experiments

In this chapter, the uniform AD, AD-BEM and AL-BEM methods are validated by comparing the simulation results with the experimental data of the flume experiments by Verbeek et al. (2021) and Stallard et al. (2013). In first, in section 4.1, the uniform AD and AD-BEM methods are validated by the experimental data by Verbeek et al. (2021), with and without the weir. Thereafter, in section 4.2, the AD-BEM method is validated for the situation with the five side-by-side placed turbines with the experimental data by Stallard et al. (2013). Note that the combination of the experiments with the weir by Verbeek et al. (2021) and the experiment with the five side-by-side placed turbines by Stallard et al. (2013) lead to a good approximation of the situation in the Eastern Scheldt barrier. At last, in section 4.3, it is investigated if the AL-BEM method improves the simulations results by validating the method with the experimental data by Verbeek et al. (2021) without the weir.

Note that in the complete chapter the experimental data is always added to the plots when it is available. All plots are always taken through the centre turbine axis. A plot of the flow velocity in x-direction over the z-axis is for example taken in the centre x- and y-axis of the turbine.

4.1. Validation of AD simulations with experimental data by Verbeek et al.

In this section the uniform AD and AD-BEM turbine parameterizations in FinLab are validated by comparing the results of the simulations with experimental data by Verbeek et al. (2021). The section is split up in five parts. In section 4.1.1, the experimental setup by Verbeek et al. (2021) is described. In section 4.1.2, the numerical setup of the simulations is elaborated. In section 4.1.3, the uniform AD and AD-BEM models are validated and a sensitivity analysis is executed for AD-BEM method by using data of the experiments by Verbeek et al. (2021) without the weir. In section 4.1.3, the AD-BEM model is applied on the experiments by Verbeek et al. (2021) with the weir. At last, a general conclusion of the accuracy of the uniform AD and AD-BEM models is drawn in section 4.1.5.

4.1.1. Experimental setup

Verbeek et al. (2021) executed experiments with the scale model of the Tocardo T1 turbine in a flume. Experiments with and without the weir were performed to find the influence of the weir on the performance of the turbine. More about the conclusions of these experiments can be read in subsection 2.1.5. An impression of the geometry of the experiments by Verbeek et al. (2021) can be seen in figure 4.1. In each experiment a constant water level (h_a) was used of either 0.65 or 0.70 m, together with the turbine diameter (D) of 0.4 m. A streamwise consistent turbulence intensity of 0.05-0.07 is generated at the inflow which is consistent with typical offshore data and in line with earlier experimental models. The turbine is rotated at a constant TSR. To describe the position of the turbine relative to the weir in the experiments with the weir, the dimensionless distance $x*$ is introduced. With $x*$ being the distance of the turbine from the weir divided by the diameter of the turbine, see equation 4.1. This dimensionless distance has a negative value when the turbine is positioned upstream of the weir and a positive value when the turbine is positioned downstream of the weir.

$$x* = x/D \tag{4.1}$$

The results of the experiments include flow data at vertical transects throughout the complete flume at the horizontal centre line of the flow. These velocity measurement are obtained by an ADV, an overview of these measurement points can be seen in figure 4.1. A central data-acquisition system recorded water levels and turbine data (thrust, torque and tip speed). The inflow velocity of the flume in combination with the measured thrust and power is used to determine the power and thrust coefficients. See equations 3.25 and 3.27 for a further clarification. In figure, 4.1 it can be seen that the load cell of the thrust measurements is located at the end of the turbine mast, which means that the combined thrust on the blades, nacelle and mast is measured.

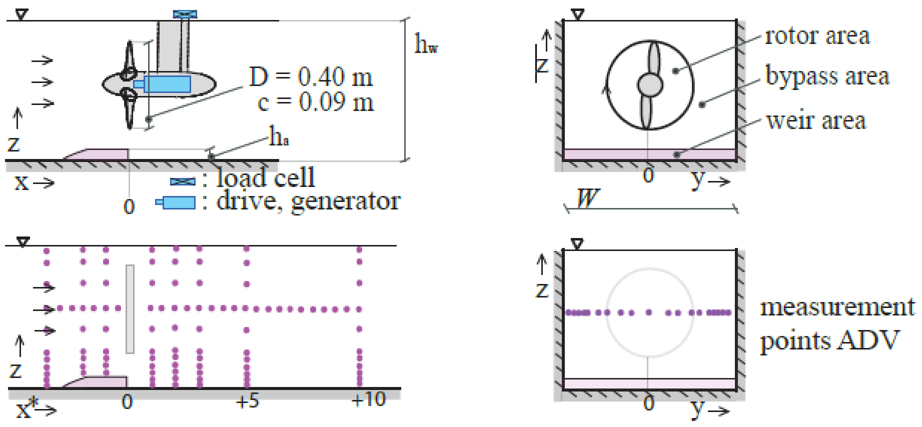


Figure 4.1: Side (left) and front (right) view of the geometry (top) and data points (bottom) of the experiments by Verbeek et al. (2021)

4.1.2. Numerical setup

All the simulations mentioned in this section are executed in the geometry of the flume experiments by Verbeek et al. (2021) with- and without the weir. All simulations are executed with an upstream velocity boundary and a downstream water level boundary. The simulations are executed with a symmetry boundary instead of a moving free surface due to stability problems caused by the combination of the high velocity bypass flow around the turbine near the surface and the small element size at this location. The turbine is parameterized by the uniform AD method and the AD-BEM with scaled data of the Tocardo T2 turbine instead of the used scaled Tocardo T1 turbine.

4.1.3. Results of simulations without the weir

The validation of the model without the weir is split up in two parts. In first, a sensitivity analysis is executed for the AD-BEM model. The thrust and power generated by the BEM module are compared to the measurements by Verbeek et al. (2021). In second, the resulting flow profiles throughout the flume of the simulations are compared with the measured flow profiles by Verbeek et al. (2021). This is done for both the uniform AD and the AD-BEM methods.

Thrust and power

The simulations of the sensitivity analysis of the thrust and the power of the AD-BEM method are executed with a upstream velocity boundary of 0.79 m s^{-1} and a TSR of 4.0 (except for the TSR analysis). In the BEM module, the axial force is split up in the force on the blades and the force on the nacelle. The streamlined geometry of the nacelle of the Tocardo turbine is made such that it exerts a low thrust force on the flow. It is however not known how low this thrust force is. Therefore, in this validation the thrust coefficients as produced by the AD-BEM model are given both with and without the nacelle thrust. Furthermore, the BEM moudule does not include the thrust on the support structure while the measured thrust coefficient by Verbeek does.

The power output of the BEM module is the exact extracted power from the flow. In the measurements by Verbeek et al. (2021) the power output is the power as delivered to the grid. According to Tocardo and Deltares (O'Mahoney et al., 2020), this power is objected to a mechanical loss of about 10%. The power coefficients as measured in the flume experiments are therefore all corrected for this mechanical loss.

Mesh sensitivity

As described in the scope of the thesis, a relatively quick model is desired. Therefore, a mesh sensitivity analysis is executed with decreasing mesh element sizes to find the largest mesh element size for which the results of each method converge. In the BEM module, the exerted forces on the blades and the flow depend on the local flow velocity and flow direction and therefore its level of detail depends on the mesh element size.

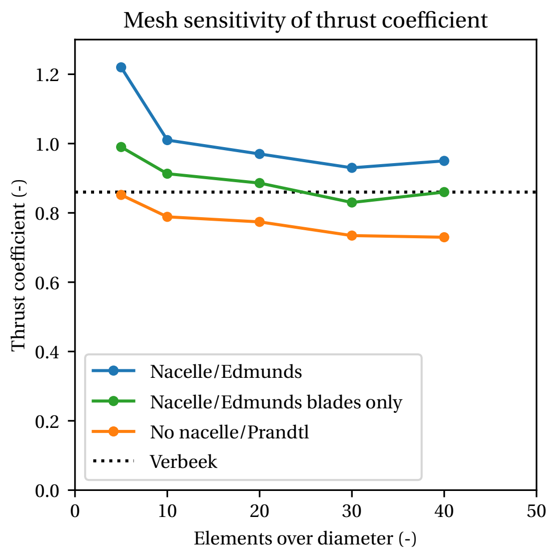
The nacelle model converges to a zero flow velocity in the element in which the centre of the nacelle is located. This only leads to a realistic flow if enough elements are present in the nacelle to construct a boundary layer. Therefore, simulations are executed for a situation with and without the nacelle module. The Prandtl tip-correction model is completely mesh independent, as it simply decreases the force in the BEM module near the blade tip. The tip-correction as introduced by Edmunds et al. (2017) is expected to be mesh dependent, see subsection 3.2.2 for further substantiation. To obtain a less mesh-sensitive run and a more mesh-sensitive run the Prandtl tip-correction is applied on the runs without a nacelle and the Edmunds et al. correction is applied on the runs with the nacelle module.

The mesh is refined near the turbine. Therefore, in this mesh sensitivity analysis, only the number of elements near the turbine is altered. Runs are executed for each method with 5, 10, 20, 30 and 40 elements over the diameter of the turbine. Due to the linear form of the BEM equations, the number of turbine integration points hardly influences the computational cost of the simulation. Therefore, to make sure that a well distributed circle is formed by the Fibonacci circle, a minimum of 5024 points is used. This is equal to one integration point per element for the finest mesh. An overview of the executed runs in the mesh sensitivity study of the AD-BEM methods is given in table 4.1.

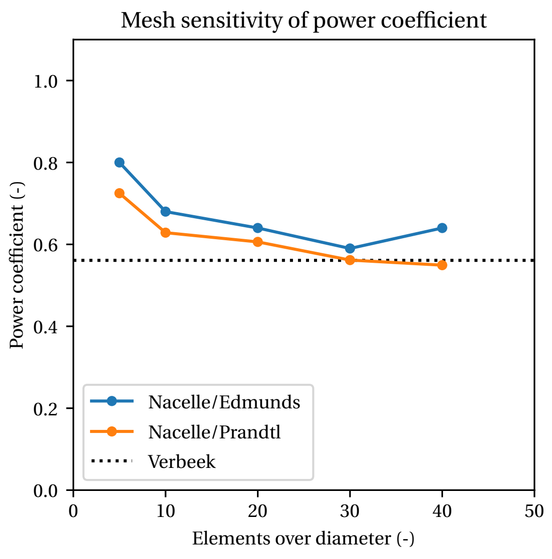
Table 4.1: Mesh size, mesh sensitivity study

N_t/D	dy,dz	N_t
5	0.08 m	5024
10	0.04 m	5024
20	0.02 m	5024
30	0.013 m	5024
40	0.01 m	5024

The produced thrust and power by the turbine in the simulations of the mesh-sensitivity analysis are shown in figures 4.2a and 4.2b on the next page respectively. The measured value of thrust and power by Verbeek et al. (2021) is plotted by the dotted line. The thrust coefficients from the simulations with the Edmunds/nacelle simulations are plotted with and without the nacelle thrust. The figures show that the both the methods, do not reach a completely mesh independent state, however they do converge towards the measured value by Verbeek et al. (2021). Furthermore, the figures do show that the Prandtl/no-nacelle model indeed does perform better in a coarse mesh. The thrust and power of the Edmunds/nacelle simulations are always higher as the nacelle method pushes more flow through the blades. For this method, the error of the thrust and power coefficient converges at about 20 elements over the diameter. In these simulations, the nacelle module does not seem to exert an extraordinary force on the flow. To reach maximum accuracy, the remainder of the validation simulations in this section are executed with 40 elements over the diameter.



(a) Thrust coefficient



(b) Power coefficient

Figure 4.2: Results of mesh sensitivity study

Tip-correction

To validate if both the tip-correction methods work correctly, three simulations are executed. One without a tip-correction, one with the Prandtl tip-correction and one with the Edmunds et al. tip-correction. The thrust and power coefficients of these three cases are plotted over the blade radius in figures 4.3 and 4.4 respectively. The coefficients are determined by binning the forces from the turbine integration points in 20 annuli and dividing over the total incoming thrust or power in the specific annuli.

From the figures it can be concluded that the two different tip-correction models work as expected. Both methods decrease the turbine performance towards the tip. The shape of all the three lines in the figure correspond very well with the shapes of the lines in figure 3.5. Due to the better physical representation of the tip-vortices, the choice is made to use the tip-correction by Edmunds et al. (2017) for the remaining AD-BEM simulations in this section.

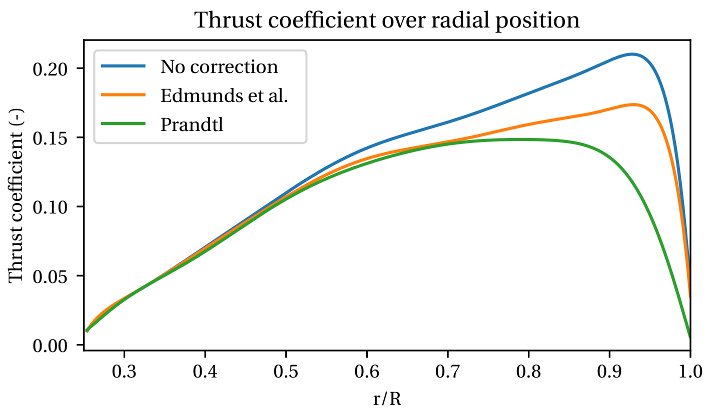


Figure 4.3: Thrust coefficient plotted over the blade radius without a tip-correction, the Prandtl correction and the Edmunds et al. correction

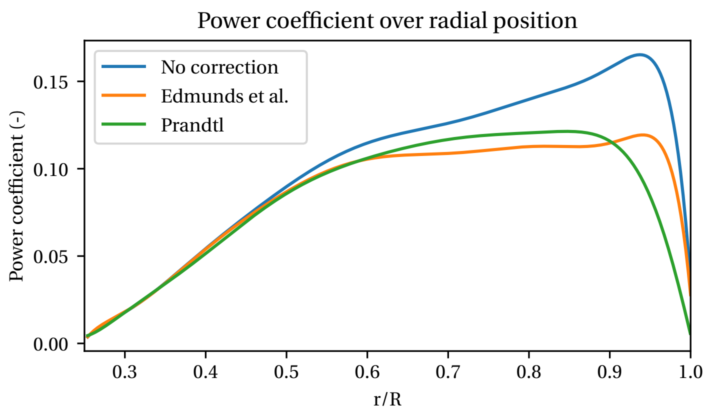
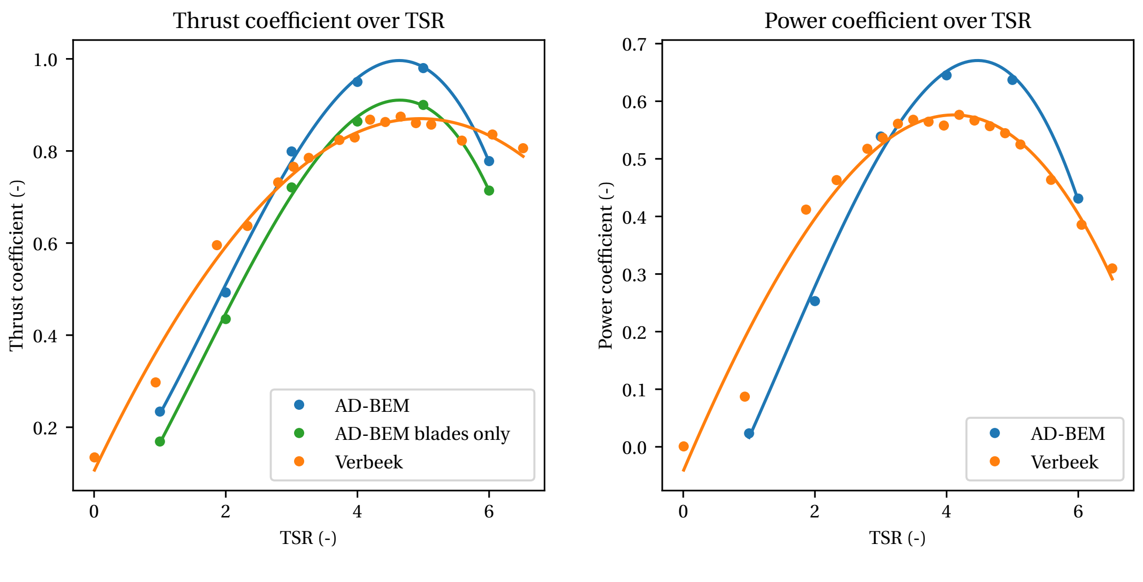


Figure 4.4: Power coefficient plotted over the blade radius without a tip-correction, the Prandtl correction and the Edmunds et al. correction

Tip-speed ratio

To validate if the influence of the TSR on the BEM module is correct, simulations are executed with a TSR of 1, 2, 3, 4, 5 and 6. The measured thrust and power coefficient are compared with the produced coefficients from the simulation in figures 4.5a and 4.5b respectively. A third-degree polynomial is plotted through the data of Verbeek et al. (2021) and the data of the simulations. The runs are executed with the Edmunds et al. tip-correction and the nacelle model. The thrust coefficients are therefore again plotted with and without the nacelle thrust.

Figures 4.5a and 4.5b show that optimum tip-speed ratio seems to be shifted to a slightly higher tip-speed ratio in the simulations. In the lower TSR regime the BEM module seems to under-predict the performance of the turbine. This is however expected as the main assumption of the BEM module is an axis-symmetric flow. The lower the TSR, the less this assumption is met. In the optimum TSR regime, the BEM module seems to over-predict the turbine performance. Again, the nacelle module does not seem to exert an extraordinary large force on the flow. Therefore, the nacelle subroutine is used in the remainder of the AD-BEM simulations in this thesis.



(a) Thrust coefficient (b) Power coefficient

Figure 4.5: Thrust and power coefficient of simulations at different speed TSR's compared with the measurements by Verbeek et al. (2021)

Flow fields

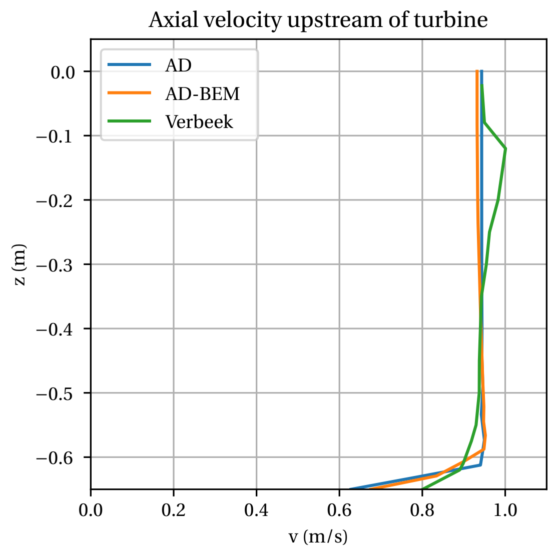
As can be read in the scope of this thesis, the goal is to produce a relatively cheap numerical flow model of a turbine which is not only able to generate accurate bulk coefficients of the turbine but also able to generate accurate flow fields. Therefore, in this section, the flow fields in the approach area, near wake and far wake are observed. Run 6.1 from the paper by Verbeek et al. (2021) is used to validate the velocity fields. An overview of this run is given in table 4.2. All the flow velocity measurements are averaged over time to filter out the transient fluctuations. Simulations are executed with both the AD-BEM and the uniform AD method. The measured thrust by Verbeek et al. (2021) is used as the input for the uniform AD method. Up- and downstream positions relative to the turbine are given in diameters (D), with one turbine diameter being equal to 0.4 m for the scale model of the Tocardo T1 turbine.

Table 4.2: Overview of the experiment by Verbeek et al. (2021) without the weir which is used to validate the flow fields of the FinLab model

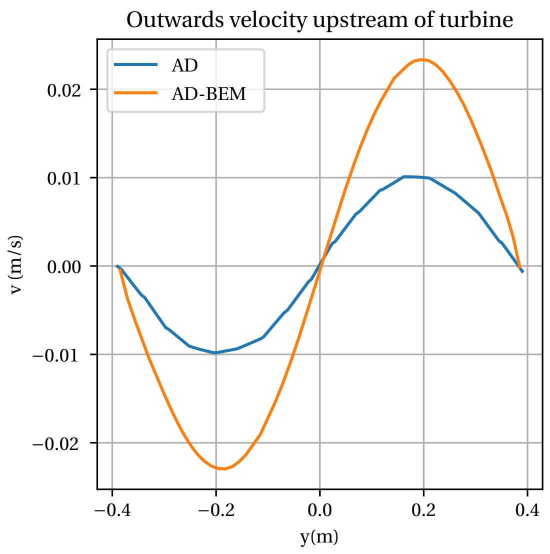
Run	h	u_0	h_w	x*	h_t	TSR	Thrust
Run 6.1	0.65 m	0.92 m s^{-1}	-	-	-0.35 m	3.12	31.3 N

Approach area

In the undisturbed region upstream a logarithmic flow profile must be present. Figure 4.6a shows that this profile is indeed present with an accurate velocity amplitude for both methods at a distance of 2D upstream of the turbine. Furthermore, it is expected that just upstream of the turbine the flow moves outwards. As stated in the theorem by Betz (1920), the streamtube widens due to the presence of the turbine. In figure 4.6b the velocity in the y-direction is plotted over the y-axis at 0.5D upstream of the turbine. This figure clearly shows that the flow moves outwards in both methods.



(a) Axial flow profile 2D upstream of turbine for situation without weir

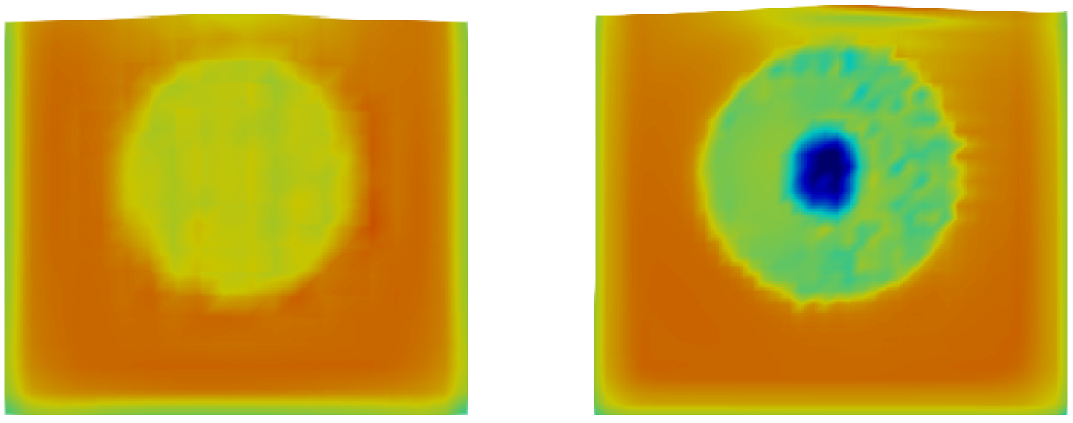


(b) Velocity in y-direction 0.5D upstream of turbine

Figure 4.6: Axial flow velocities upstream of turbine

Near wake

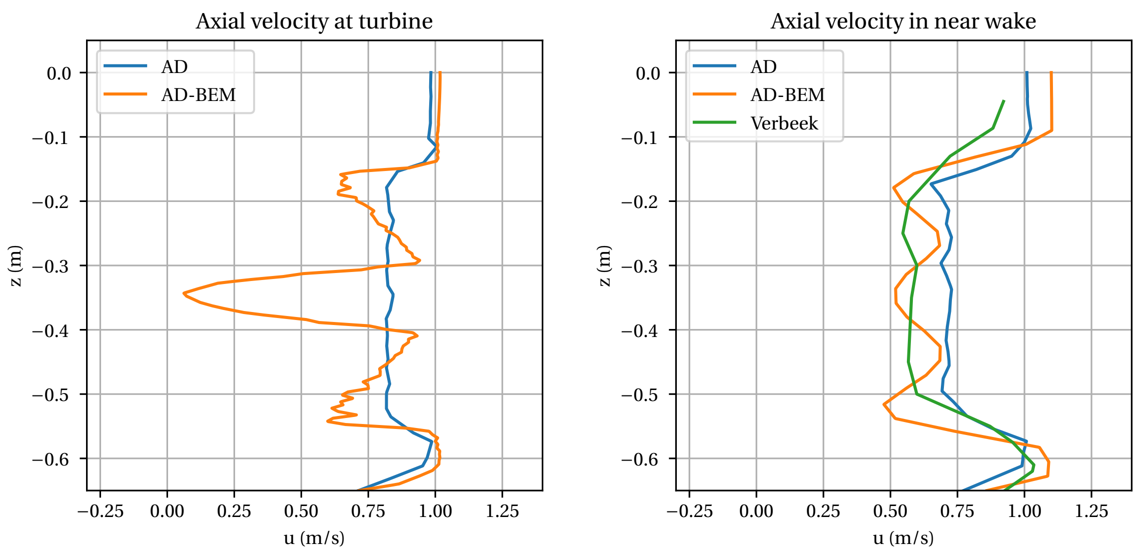
In figures 4.7a and 4.7b, the axial velocity field of the uniform AD and AD-BEM method respectively are shown at the location of the turbine for the situation without the weir. Note that these are from a different simulation to show the free-surface effect. In figure 4.8a, the axial velocity profiles of the two methods are shown at the turbine. In the figures, the difference between the two methods can be clearly seen. As expected, the axial flow velocity through the uniform AD is almost uniform. The axial flow velocity through the turbine with the AD-BEM model approaches zero in the area of the nacelle. The flow velocity increases near the nacelle and decreases again towards the tip. In the theoretical outline it is explained that this is the expected profile due to the increasing relative velocity of the blade towards the tip.



(a) Uniform AD method

(b) AD-BEM method

Figure 4.7: Axial flow velocity through actuator disc for the situation without weir



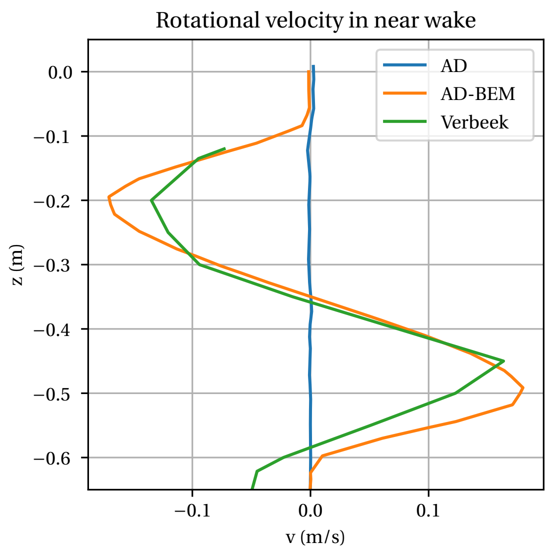
(a) At turbine

(b) 1D downstream of turbine

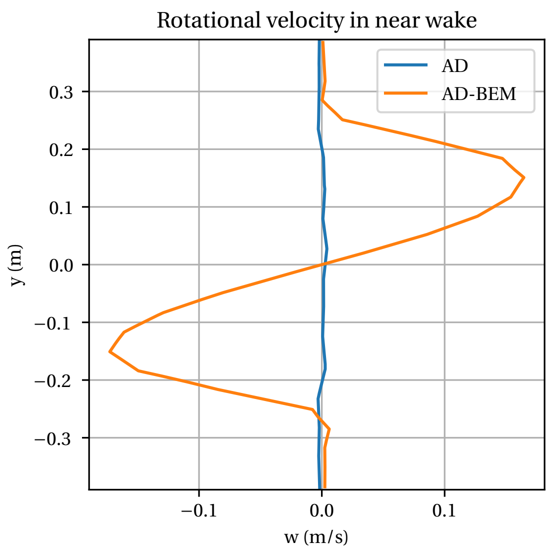
Figure 4.8: Axial flow profiles in the near wake for the situation without weir

Of course, as stated in the literature study, with the applied AD methods, the transient features of the near wake are not present. However, it can be checked if the time-averaged characteristics of the near wake agree with the time-averaged velocity measurements by Verbeek et al. (2021). In figure 4.8b, the axial velocity in the near wake is plotted for the uniform AD simulation, AD-BEM simulation and the measurements. The overall amplitude of the axial velocity profiles seem to correspond reasonably well with the measurement by Verbeek et al. (2021).

The rotational velocity profile 2D downstream of the turbine is plotted for the situation in figures 4.9b and 4.9a over the z-axis and y-axis respectively. It can be seen that the AD-BEM method estimates the time-averaged rotational movement well. The uniform AD simulation of course does not initiate a rotating movement of the flow at all. The rotational movement of the flow can also be observed in the free-surface elevations in figures 4.7a and 4.7b. The free-surface of the uniform AD simulation is deformed symmetrically while the free-surface of the AD-BEM simulation is asymmetrically deformed towards the side of the rotation.



(a) Velocity profile over z-axis at 1D downstream



(b) Velocity profile over y-axis at 1D downstream

Figure 4.9: Rotational velocity profiles over the z-axis and y-axis in the near wake for the situation without weir

Far wake

As stated in subsection 2.1.6 of the theoretical outline, a specific distinction between the near and far wake can be made by the influence of the pressure gradient. In the near wake, the pressure gradient can not be neglected, while in the far wake it can be neglected. According to theory, the far wake should start around 4D downstream of the turbine. In figure 4.10 the pressure gradient of the uniform AD and the AD-BEM method are plotted for the situation without the weir over the centre axis of the turbine from 1D upstream to 4D downstream. It can be seen that the pressure gradients of simulations correspond well with this theory. The difference in amplitude of the pressure difference of both methods can be explained by the presence of the nacelle in the centre axis of the turbine in the AD-BEM method.

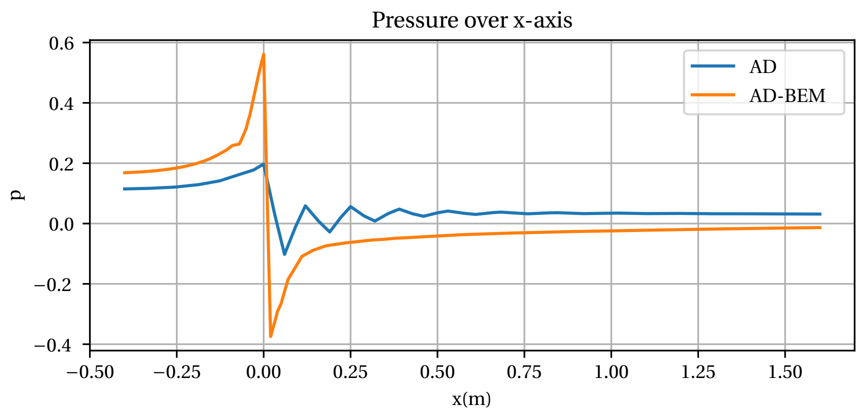
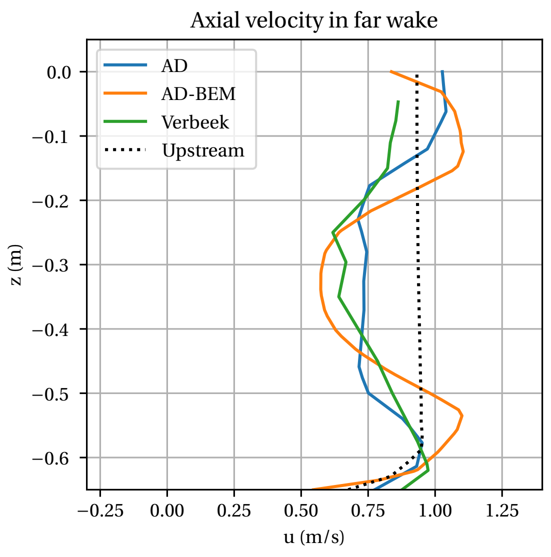
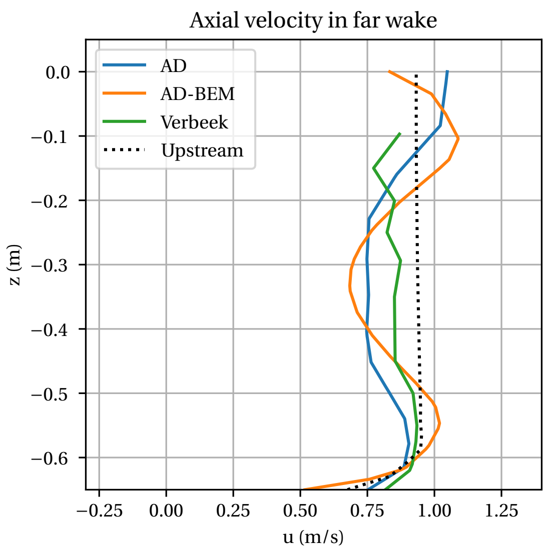


Figure 4.10: Pressure difference through centre axis of turbine

In the far wake the presence of the turbine should only be felt globally. A smooth velocity deficit should be formed as with the sharp velocity gradients should be flattened out by the turbulence in the near wake. In figures 4.11a and 4.11b the axial flow profiles in the far wake over the z-axis are plotted for the situation without the weir. The AD-BEM velocity profile upstream of the turbine is shown as a recovery reference. In the figures, it can be seen that for both methods a smooth wake has formed, the shape of the profiles of both methods correspond well with the measurement by Verbeek et al. (2021).



(a) Velocity profile over z-axis at 5D downstream



(b) Velocity profile over z-axis at 10D downstream

Figure 4.11: Velocity profiles over z-axis in the far wake for the situation without the weir

In figures 4.11a and 4.11b the velocity in the centre of the far wakes of both the uniform AD and the AD-BEM simulation seems to be too low in comparison with the flume measurements. This lower velocity in the centre wake is compensated with a higher velocity in the bypass regions. The combination of these two leads to a too slow wake recovery in the simulations. This can also be seen in figure 4.12, which shows the axial velocity throughout the domain for the simulations and the flume measurements. The AD-BEM simulation is both executed with an Prandtl and a Edmunds et al. tip-correction to show the effect of the different method on the wake recovery. Note that the flume measurements is probably slightly distorted as the velocity recovers to a larger velocity than the inflow velocity. Still, the overall wake recovery in the experiments seems to be faster than in the simulations. The extra velocity shear in the AD-BEM simulation in comparison to the uniform AD simulation leads to a faster and more accurate wake recovery. The extra velocity shear in the AD-BEM simulation with the Edmunds et al. tip-correction in the comparison to the simulation with the Prandtl tip-correction does however not lead to a faster and more accurate wake recovery.

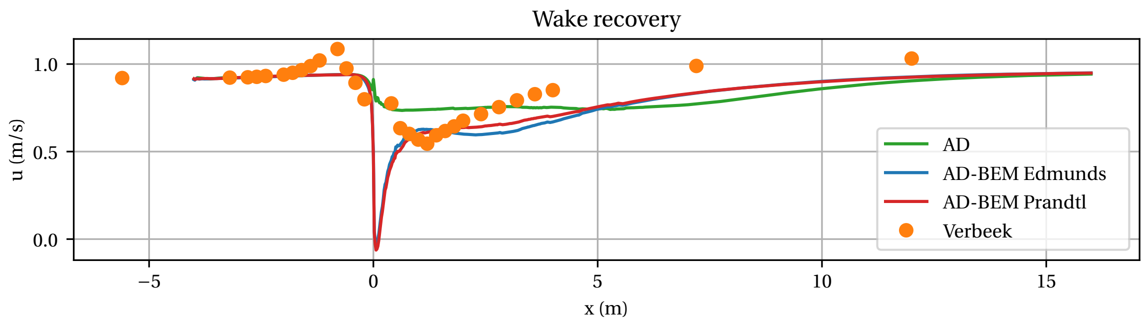


Figure 4.12: Axial velocity throughout the flume through centre axis of turbine

4.1.4. Results of simulations with the weir

Verbeek et al. (2021) executed a large set of experiments with different weir-turbine geometries. Three of these experiments are reproduced by simulations to validate the AD-BEM model. The simulations of these experiments with the weir are only executed with the AD-BEM method due to the higher accuracy of the model in comparison with the uniform AD model and the possibility of the model to estimate the turbine performance. In each of the three experiments the flow passes a 0.1 m high weir. One experiment is mimicked with the turbine at the upstream end of the weir ($x* = -2$), one with the turbine at the downstream end of the weir ($x* = 0$) and one with the turbine 1D downstream of the weir ($x* = 1$). An overview of the three executed simulations is shown in table 4.3, the names of the runs are for convenience taken the same as in the paper by Verbeek et al. (2021). The simulations are again executed with the tip-correction by Edmunds et al. in combination with the nacelle module. The used mesh contains 20 elements over the diameter of the turbine.

Table 4.3: Overview of the experiments by Verbeek et al. (2021) with the weir which are used to validate the flow fields of the CFD-BEM model

Run	h	u_0	h_w	x*	h_t	TSR
Run 4.8	0.65 m	0.89 m s^{-1}	0.10 m	0	-0.25 m	3.12
Run 5.7	0.65 m	0.89 m s^{-1}	0.10 m	1	-0.25 m	3.49
Run 4.10	0.65 m	0.90 m s^{-1}	0.10 m	-2	-0.25 m	2.52

Thrust and power

In table 4.4 an overview is given of the accuracy of the produced thrust and power coefficients by the AD-BEM model in comparison to the measurements by Verbeek et al. (2021). The power coefficients as measured in the flume experiments are again corrected for the estimated 10% mechanical loss. The thrust coefficients of the AD-BEM model in the table include the thrust on the nacelle.

The results show that the thrust coefficients as generated by the AD-BEM of all three runs are too low. The very high thrust of run 4.8 in the flume experiments is hard to explain and therefore also not reproduced by the AD-BEM model. The too low thrust and power coefficients produced by the model in run 4.10 seem to be caused by a too large bypass flow in combination with the low TSR of 2.52. In figure 4.24 it can be seen that the BEM module underestimates the performance of the turbine at such a TSR. It must be noted that the TSR in this case is still determined by using the inflow velocity while the flow is accelerated on the weir. The higher velocity near the turbine makes the actual TSR even lower. The power generation in runs 4.8 and 5.7 is accurate. Overall, the AD-BEM model seems to be able to estimate the position of maximum power extraction well with respect to the position of the weir.

Table 4.4: Overview of the accuracy of thrust and power coefficients produced by the BEM model

		C_T			C_P		
Run	x*	BEM	Verbeek	Error	BEM	Verbeek	Error
Run 4.8	0	0.98	1.64	-40%	0.74	0.77	-3.9%
Run 5.7	1	1.00	1.03	-2.9%	0.72	0.72	0.4%
Run 4.10	-2	0.75	1.22	-39%	0.48	0.58	-25%

Flow fields

Figures 4.13, 4.14 and 4.15 show the axial velocity flow fields of the area around the weir and turbine of the AD-BEM simulations with the turbines placed at x*=-2, x*=0 and x*=1 respectively. In these figures it can be clearly seen that the turbine at x*=-2 indeed has a lower influence on the flow than the other two. A large bypass occurs under the turbine. In the cases with the turbine at x*=0 and x*=1, an accelerating flow on top of the weir can be recognised. In all three figures a recirculation area behind the backward-facing step is noticeable.

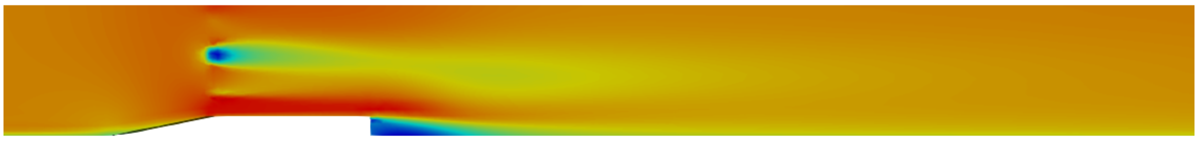


Figure 4.13: Axial velocity flow field with turbine at x*=-2

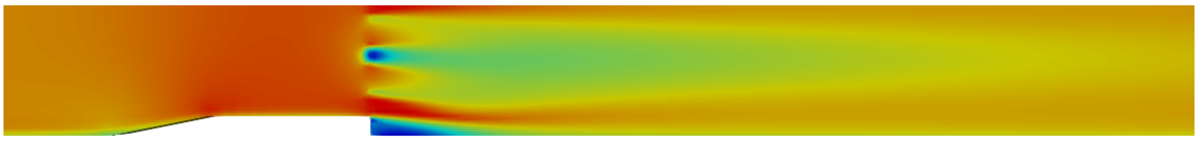


Figure 4.14: Axial velocity flow field with turbine at x*=0

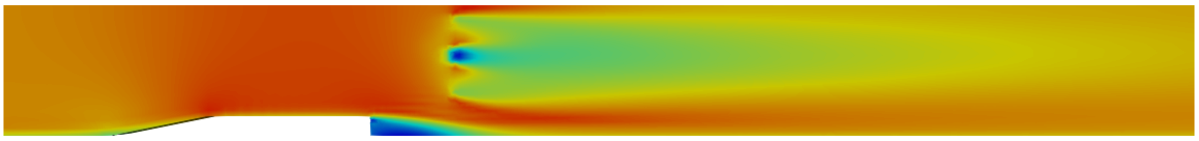
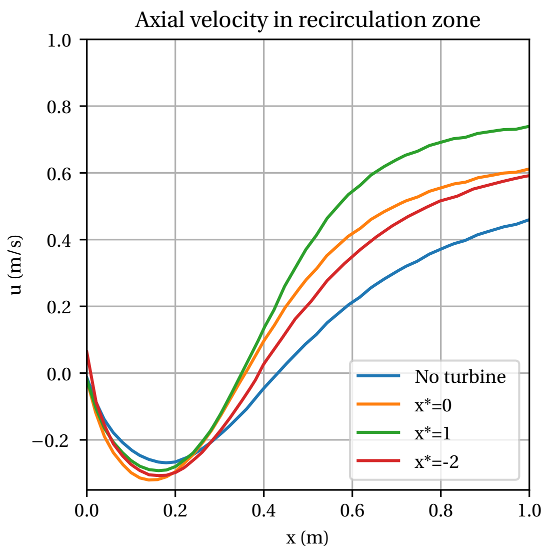


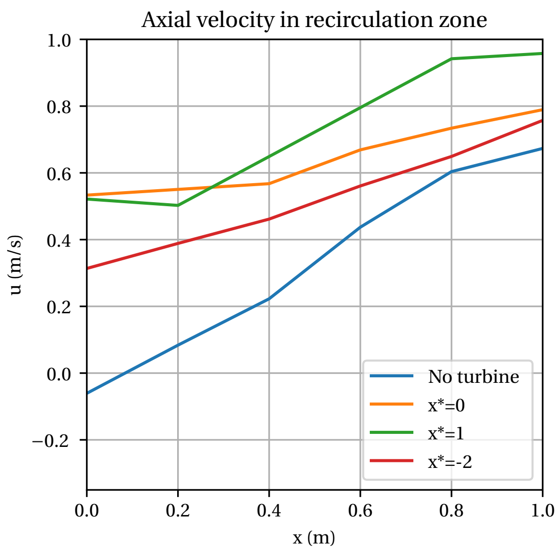
Figure 4.15: Axial velocity flow field with turbine at x*=1

Recirculation zone

In figure 4.16a the axial flow velocity on the bed in the recirculation zone of the AD-BEM simulations is plotted. The amplitude of the recirculating velocity is hard to validate due to the lack of accurate flume measurements in this area. The same applies to the position of the reattachment point. It is however expected that the presence of the turbine could suppress the flow separation and energy losses in the recirculation zone. Again, in figure 4.16a, it can be obtained that in the AD-BEM simulations, the recirculation flow is the most suppressed in the simulation with the turbine located at $x* = 1$. The same can be seen in figure 4.16b in which the flume measurements close to the bed are plotted.



(a) Data from CFD-BEM model



(b) Measurements by Verbeek

Figure 4.16: Axial velocity at bed in recirculation zone

Axial velocity

To validate the overall velocity profile of the AD-BEM simulations, the axial velocity through the centre axis of the turbine can again be compared to the measurements by Verbeek et al. (2021). This is shown in figures 4.17, 4.18 and 4.19, for the experiments with the turbine at $x* = -2$, $x* = 0$ and $x* = 1$ respectively. Overall the figures show a good correspondence with the flume measurements. The accelerating flow on the weir seems to correspond well. In the approach flow some fluctuations can be noticed in the flume measurements of mainly the situation with the turbine at x*=1, these seem to be some measurement errors and are therefore not shown in the simulation results.

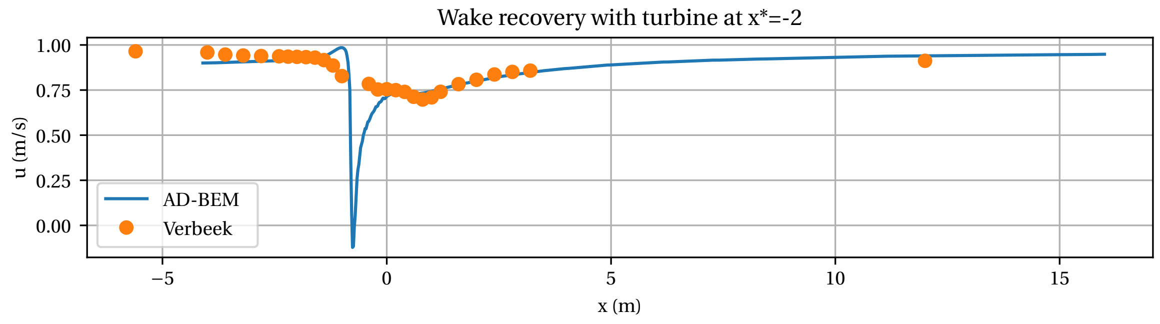


Figure 4.17: Wake recovery with turbine at x*=-2

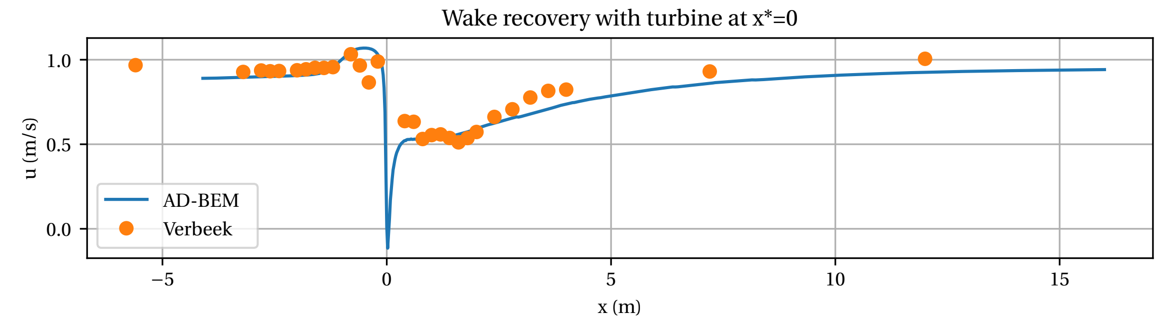


Figure 4.18: Wake recovery with turbine at x*=0

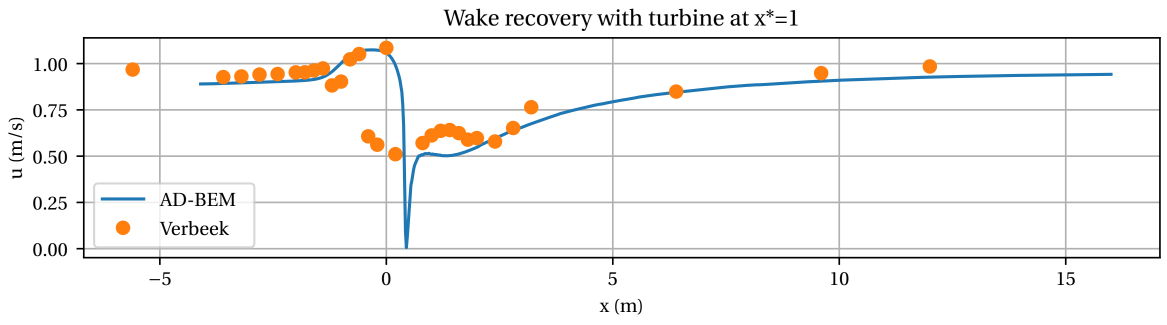


Figure 4.19: Wake recovery with turbine at x*=1

The wake recovery of the AD-BEM model in the geometry with the weir seems to correspond better with the flume measurements than in the simulation without the weir. The velocity deficit as caused by the weir thus seems to be dominant over the velocity deficit as cause by the turbine. Note that the flow velocity again recovers to a higher velocity than the inflow velocity in the flume experiments with the turbine at $x* = 0$ and $x* = 1$. This is not reproduced by the simulations.

4.1.5. Conclusion

In this section, simulations of the experiments by Verbeek et al. (2021) are made with the uniform AD and AD-BEM models. The sensitivity studies of the AD-BEM method show that the accuracy of the method depends on the element size, number of integration points, TSR, tip-correction, application of a free-surface and the choice of nacelle representation. The mesh-sensitivity study shows converging results from a minimum of 20 elements over the diameter of the turbine. Therefore, the remaining AD-BEM simulations in this thesis are all executed with 20 elements over the diameter. The nacelle model seems to perform a realistic thrust force on the flow and is therefore also used in the remaining of the AD-BEM simulations. Both the tip-corrections seem to work properly by decreasing the efficiency of the blades near the tip. It is, however, up for debate which tip-correction best fits the Tocardo T2 turbine. In contrast to results in the paper by Edmunds et al. (2017), their tip-correction does not lead to a faster wake recovery in the simulations in this thesis. The Edmunds et al. tip-correction is, however, still chosen to be better option as the theory is physically more sound in comparison with the Prandtl tip-correction. The remaining AD-BEM simulations in this thesis are therefore executed with the Edmunds et al. tip-correction.

The flow fields resulting from the simulations with the uniform AD and AD-BEM methods of the experiment of Verbeek et al. (2021) without the weir give a clear picture of the possibilities of both models. In the near wake, both the methods mimic the time-averaged axial velocity deficit well. The AD-BEM method also reproduces the time-averaged rotation of the near wake well and adds an extra velocity deficit in the nacelle area. Both these features lack in the uniform AD simulations. In the far wake the results of both methods are less accurate. Both the methods produce a too slowly recovering wake. The rotating wake and nacelle cause an extra velocity shear in the wake of the AD-BEM model. This extra velocity shear leads to a higher turbulent kinetic energy (TKE) in the near wake of the AD-BEM method which makes the wake of the AD-BEM method recover faster than the wake of the uniform AD method. This extra velocity shear does however not make the wake recovery rate accurate. The too slowly recovering wakes could be caused by the time-averaging nature of both the AD and RANS methods. The combination of both models filter out the transient features of the near wake. These transient features could cause the extra velocity shear an thus extra TKE in the near wake which could lead to a faster and more accurate recovering wake.

Due to the higher accuracy and the possibility to model the turbine performance, the simulations of the experiments by Verbeek et al. (2021) with the weir are only executed with the AD-BEM method. The results of the simulations show that the thrust of the turbines is under-predicted by the AD-BEM model in comparison with the measurements. Why this occurs is not completely clear. The simulations do show an accurate estimation of the power produced by the turbines. The model seems to be able realistically predict the location of maximum power extraction in a turbine-weir geometry. In the flow fields resulting from the simulations with the weir, the geometry seems to be of dominant influence on the flow over the turbine and the wake recovery rate is more accurate than without the weir. A reason for this could be that the flow around the weir leads to a large velocity shear and thus a large TKE. The larger TKE could lead to a faster mixing of the flow.

As stated in subsection 2.1.5 of the theoretical outline, Verbeek et al. (2021) hints to the existence of a turbine-weir geometry at which the energy yield is optimum while the added flow resistance by combined geometry is limited. Verbeek et al. (2021) concludes that this is the case when the turbine is placed at the weir end ($x* = 0$). The AD-BEM simulations hint that the optimum turbine location seems to be located slightly further downstream. In the AD-BEM simulations the turbine location of optimum power harvesting is indeed at $x* = 0$. However, the recirculation zone is most suppressed in the simulation with the turbine located one diameter downstream of the weir end ($x* = 1$). At this location the power generation by the turbine is only slightly lower. The turbine-weir geometry with optimum power harvesting and limited flow resistance could therefore be a geometry with the turbine placed somewhere between $x* = 0$ and $x* = 1$.

4.2. Validation of AD simulations with experimental data by Stallard et al.

As stated in subsection 2.1.6 of the theoretical outline, Stallard et al. (2013) executed experiments to examine the wake-wake interactions of multiple turbines placed next to each other in a flume. In this section, these experiments are used to validate the AD-BEM model for this situation. The section is split up in four parts. In section 4.2.1, the experimental setup by Stallard et al. (2013) is described. In section 4.2.2, the numerical setup of the simulations is elaborated. In section 4.2.3, the results of the simulations are discussed. At last, in section 4.2.4 a conclusion is drawn from these simulation results.

4.2.1. Experimental setup

In the experiments by Stallard et al. (2013) the flume was 12 m long and 5 m wide. The flow was generated by an incoming flow velocity of 0.47 m s^{-1}, water level of 0.45 m and quite high average turbulence intensity of 10 %. Time-varying velocities were measured by using an ADV at 40 depths on 18 downstream coordinates, creating 18 velocity profiles. Several similar scale model turbines were used with a diameter of 0.27 m, representing an in field size of 19 m. Several experiments were executed with a single turbine up to five turbines placed by side, in these experiments different spacing between the turbines was used. Furthermore, experiments with a second row of turbines placed downstream of the first row were executed. To validate the AD-BEM model, the case with five turbines placed side by side at 1.5D distance of each other is chosen (with D here equal to 0.27 m). This case corresponds the best with the situation in the Eastern Scheldt barrier. An overview of the used geometry can be seen in figure 4.20, note that in this figure only three turbines are present.

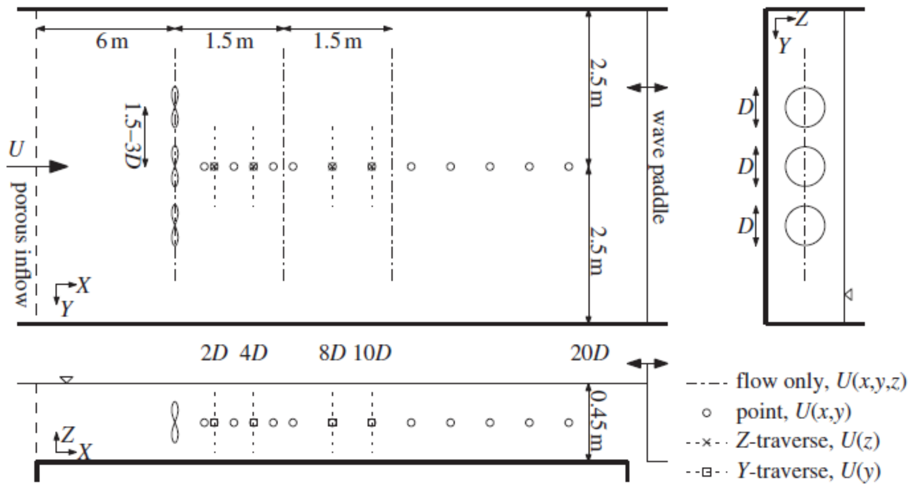


Figure 4.20: Arrangement of flume indicating key dimensions and global coordinate system. Three rotors and velocity measurement positions indicated for the central rotor. Velocity measured at identical locations downstream of each rotor. (Stallard et al., 2013)

4.2.2. Numerical setup

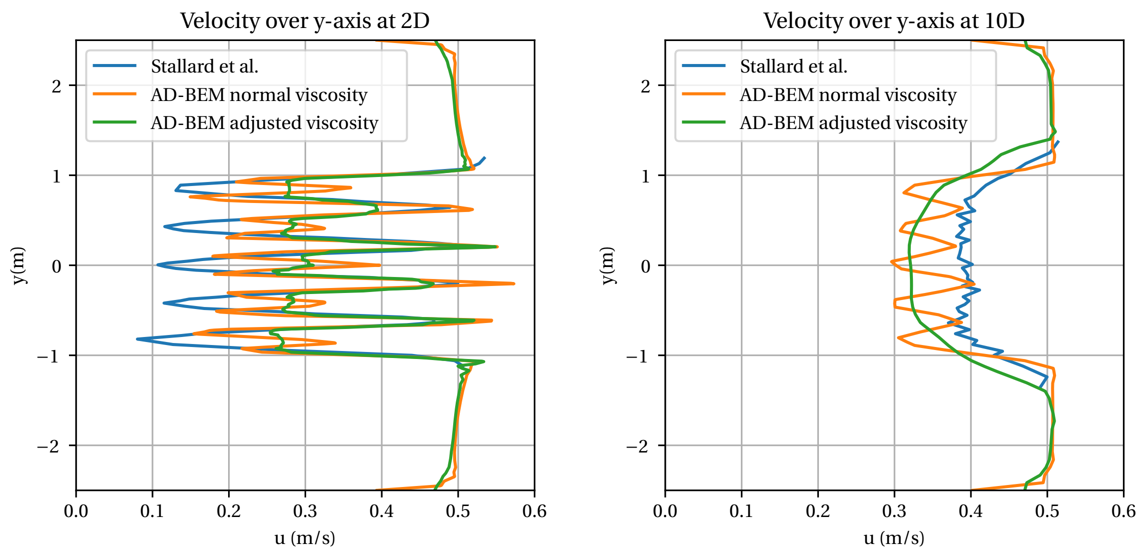
In the experiments by Stallard et al., Goëttingen 803 scale models were used, therefore, different geometry files are set up by using data from other flume experiments by Stallard et al. (2015). Figure 3.8b in chapter 3.2.3 gives an impression of the geometry of this turbine. Furthermore, an upstream velocity boundary is used together with an downstream water level boundary. The free-surface is again represented by a non-moving symmetry boundary. The nacelle module which pushes the flow in the nacelle to zero is used together with the Edmunds et al. tip-correction. Again, 20 elements are used again over the diameter of the turbine. An overview of the executed simulations can be seen in table 4.5.

Table 4.5: Simulation of the experiments by Stallard et al. (2013)

Run	Turbines	h	u_0	h_t	TSR	R
Stallard	5	0.45 m	0.47 m s^{-1}	-0.225 m	4.5	0.135 m

4.2.3. Results of simulations

The AD-BEM simulation predicts an average C_T of 0.95 over the five turbines, including nacelle. Stallard et al. (2013) measured values 0.87 with a range 0.05, which leads to an error of about 9%. The average C_P of the simulation is equal 0.37. Stallard et al. (2013) measured a value of 0.33, which leads to an error of 12%. Note that the power coefficient is not corrected for mechanical loss.



(a) At 2D (b) At 10D

Figure 4.21: Axial flow profiles in simulations of the experiments by Stallard et al. (2013)

In figures 4.21a and 4.21b the velocity profiles over the y-axis are plotted at 2D and 10D downstream of the turbine. At 2D the velocity profile of the AD-BEM simulation follows the measurement quite accurate. The largest difference seems to be the velocity deficit in the nacelle region which seems to recover faster in the simulation than in the measurement. At 10D the measurements shows a completely different profile than the simulation. In the measurement one joint wake is formed, while each individual turbine is still recognisable in the wake of the simulation.

This could be caused by the high turbulence intensities in the experiments and by the working mechanism of the k-ϵ turbulence model as implemented in FinLab. The high turbulence intensities of about 10% can lead to a lot of mixing in the experiments and thus a fast merging of the individual wakes. In the k-ϵ model in FinLab, the size of the eddies can grow up to maximum length scale at the location in the mesh. In a flat flow situation as is the case in the experiments by Stallard et al. (2013), the maximum size of the eddies is constraint by the depth of the domain. In reality, the eddies could grow up in horizontal direction until the horizontal size off the domain.

The larger horizontal eddies can be reproduced by increasing the horizontal viscosity with a factor 100 with respect to the vertical viscosity (Labeur, personal communication). The results of a simulation with such an adjusted viscosity are also shown in figures 4.21a and 4.21b. At 10D it can be seen that the simulation correctly predicts a completely merged wake. Furthermore, the velocity deficit is larger in the simulation than in the measurement. This again confirms that the wake in the AD-BEM simulations recovers too slow when no weir is present in the geometry. In figures 4.22 and 4.23 on the next page, the overall flow fields are shown of respectively the simulation without and with the adjusted viscosity.

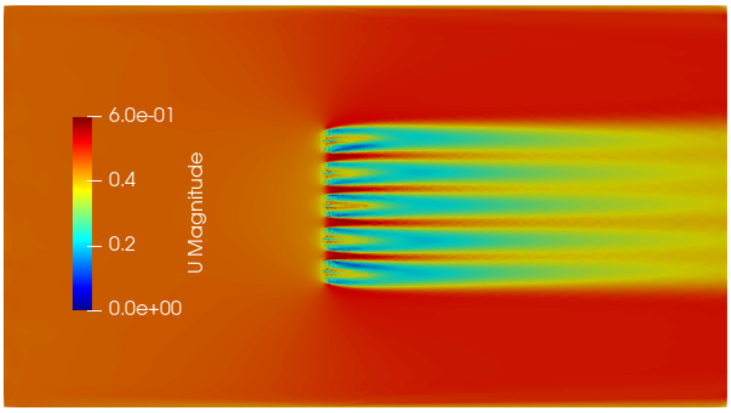


Figure 4.22: Flow field of simulation of experiment by Stallard et al. (2013) with normal viscosity

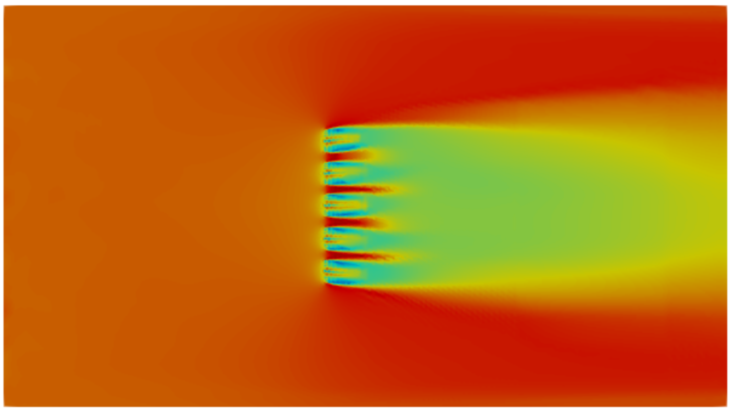


Figure 4.23: Flow field of simulation of experiment by Stallard et al. (2013) with adjusted viscosity

4.2.4. Conclusion

In this section, simulations of the flume experiments with five side-by-side placed turbines by Stallard et al. (2013) are made with the AD-BEM model. Out of these simulations it can be concluded that in a flat flow with multiple turbines, the mixing as determined by the k-ϵ model is too low. Increasing the turbulent viscosity in the horizontal plane of a flat flow increases the mixing accuracy of the wakes. However, from these results, no conclusion can be drawn of the wake mixing in a mesh with more equally sized length scales. The amplitude of the velocity deficit in the wake does not become more accurate by increasing the turbulent viscosity. The overall wake recovery rate is again too low. This confirms the conclusion from the simulations of the experiments by Verbeek et al. (2021) without the weir.

4.3. Validation of AL simulations with experimental data by Verbeek et al.

One flaw of both the AD methods stands out throughout results of the simulations experiments by Verbeek et al. (2021) and Stallard et al. (2013). The wake of the turbine seems to recover too slow in the simulations without the weir. In both the uniform AD and AD-BEM simulations, the forces exerted on the blades and flow are averaged over one full rotation. With this method, transient features of the flow such as the tip-vortices and the fluctuations in the near wake are lost. These transient features cause an extra velocity shear and thus a higher TKE in the near wake and could therefore lead to faster recovering wakes. The AL-BEM method does resolve these transient features and is therefore expected to improve the accuracy of the wake recovery.

In this section, the accuracy of the AL-BEM method is compared with the accuracy of the uniform AD and AD-BEM methods by validating the AL-BEM method with the experimental data by Verbeek et al. (2021). Note that in this thesis, the AL-BEM method is only added to show the possibilities of the code, detailed validation could be executed as future work. This section is split up in three parts. In section 4.3.1, the numerical setup of the AL-BEM simulations is elaborated. In section 4.3.2, the results of the AL-BEM simulations are discussed. At last, in section 4.3.3, a conclusion is drawn from the results of these simulations.

4.3.1. Numerical setup

AL-BEM simulations are executed in the flume geometry of Verbeek et al. (2021) for both turbines. An inflow speed of 0.9 m/s with a random velocity fluctuation of 6%. Simulations are executed with both the Tocardo T2 and Goëttingen 803 turbines. To solve for an accurate transient solution of the large-scale turbulent fluctuations, a finer mesh is used together with the fractional step method and LES, see appendix B for a substantiation of this choice. In the nacelle integration points, the nacelle routine which brings the axial flow velocity to zero is applied. No tip-correction is applied as the tip-vortices should be solved within the LES model.

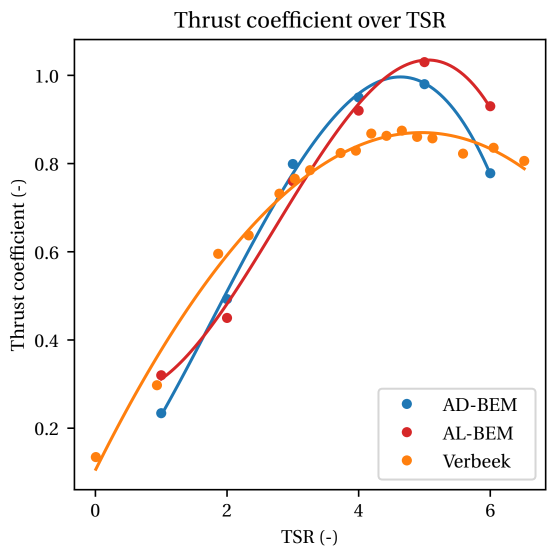
4.3.2. Results of simulations

Before, validating if the AL-BEM indeed does produce a more accurate wake recovery, it is important to validate if the AL-BEM method produces realistic thrust and power estimates. This can be done by again performing a TSR's sensitivity analysis. Note that by doing this, the mesh sensitivity analysis is passed while it is expected that the AL-BEM method requires a higher mesh resolution than the AD-BEM method.

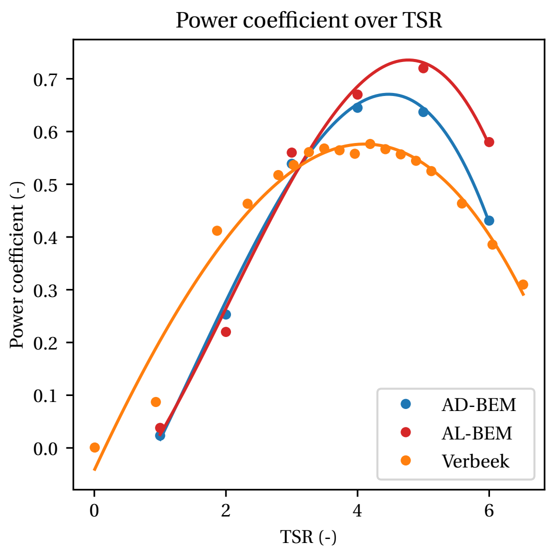
Another flaw of the AD-BEM model as concluded throughout in the simulations of the experiments by Verbeek et al. (2021) in section 4.1.3, is the under-prediction of thrust and power at low TSR's. The AD-BEM does perform worse at low TSR's as the main assumption of the model is that the flow field through the disc can be represented by a time-averaged flow. The faster the blades spin the more accurate this assumption is. The AL-BEM model does not make this assumption and should therefore perform better at low TSR's. To validate this, simulations are executed with the AL-BEM method with a TSR of 1, 2, 3, 4, 5 and 6.

Figures 4.24a and 4.24b on the next page show respectively the thrust (including nacelle) and power coefficients for these TSR's as measured by Verbeek et al. (2021) and produced by the AD-BEM and AL-BEM models. Unfortunately, both figures show that the performance the AL-BEM does not perform better in the low TSR region compared to the AD-BEM model. The over-prediction of the turbine performance in the optimum TSR region is also even higher than in the AL-BEM method. This is probably due to the introduced uniform distribution of the force over the blades. The Gaussian force distribution as used by Sorensen and Shen (2002) introduces the largest force at the location where the velocity is sampled. This leads to a lower flow velocity at the sampling points and thus a lower turbine performance. The distribution function could be added in future work.

As stated in the introduction of this section, the main flaw of the AD-BEM model is that the wake seems to recover too slow in the simulations without the weir. See figures 4.12 and 4.21b for an overview of the wake recovery in the simulations of the experiments of Verbeek et al. (2021) and Stallard et al. (2013) respectively. It is expected that this too slow recovery of the wake is caused by the nature of the combination of the actuator disc approach with the k-ϵ turbulence model. The combination of these two leads to a near wake in which the transient eddies are filtered out and averaged over time. These transient features do however contribute to the recovery of the wake and are solved for in the AL-BEM model. It is therefore interesting to see if the wake recovers faster and more accurate in comparison with the measurements by Verbeek et al. (2021). In figure 4.25 the axial flow velocities throughout the flume are shown for the uniform AD, AD-BEM and AL-BEM simulations and the measurements by Verbeek et al. (2021). The figure shows that the wake does indeed recover slightly faster and more accurate in the AL-BEM simulation.



(a) Thrust coefficient



(b) Power coefficient

Figure 4.24: Thrust and power coefficient of simulations and measurements at different speed TSR's

Table 4.6: Overview of the experiment by Verbeek without the weir which is also used in subsection 4.1.3 to validate the flow fields of the AD-BEM model

Run	h	u_0	h_w	x*	h_t	TSR
Run 6.1	0.65 m	0.92 m s^{-1}	-	-	-0.35 m	3.12

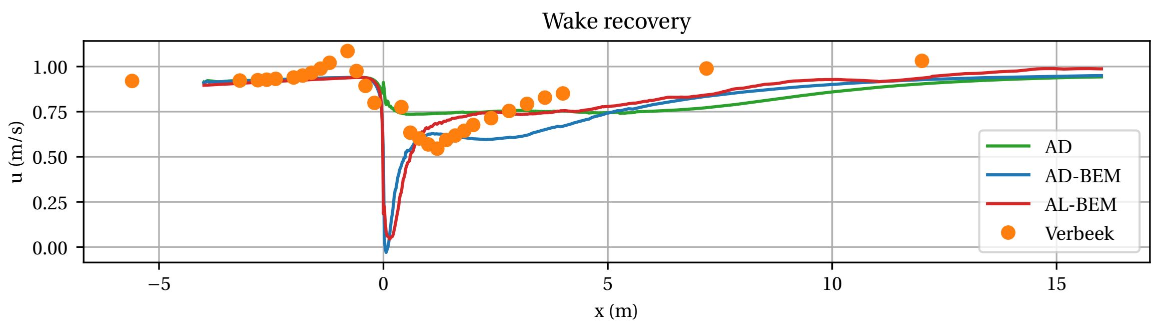


Figure 4.25: Wake recovery of all three methods

Figures 4.26a, 4.26b, 4.27 and 4.28 on the next page show that the transient features of the near wake are present in the results of the AL-BEM model. Iso-volume vorticity threshold (Q-criterion) plots are used to show the turbulent structures, pressure Iso-volume plots are used to show the structure of blades and nacelle (dark blue). The colors on the vortices represent the axial flow velocity. In the front view figures the blades can be clearly recognised. In the side view figures, the downstream trailing and dissipating tip-vortices can be clearly recognised. The tip-vortices in the simulation of the Goëttingen 803 turbine follow up closer to each other due to the higher TSR and extra blade.

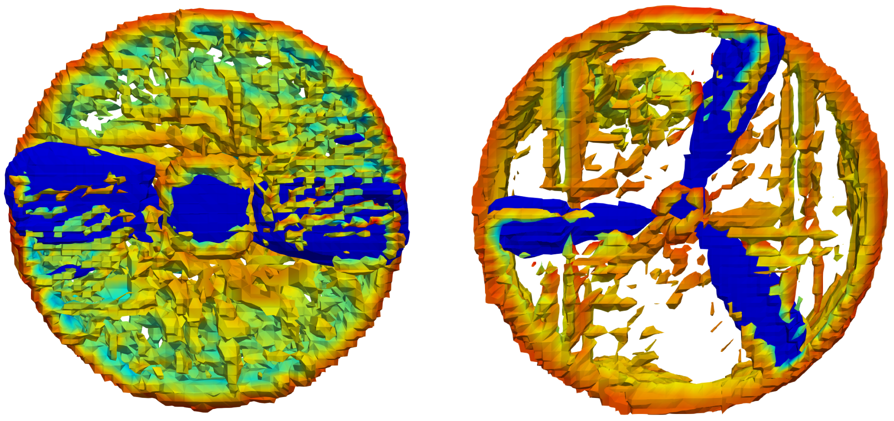

(a) Tocardo T2 (b) Goëttingen 803

Figure 4.26: Iso-volume plots of the front view of the turbines in the AL-BEM simulations

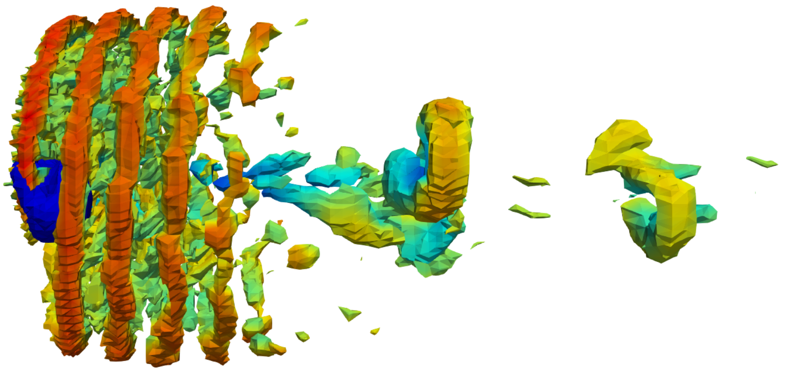


Figure 4.27: Iso-volume plot of the side view of the Tocardo T2 turbine in the AL-BEM simulation

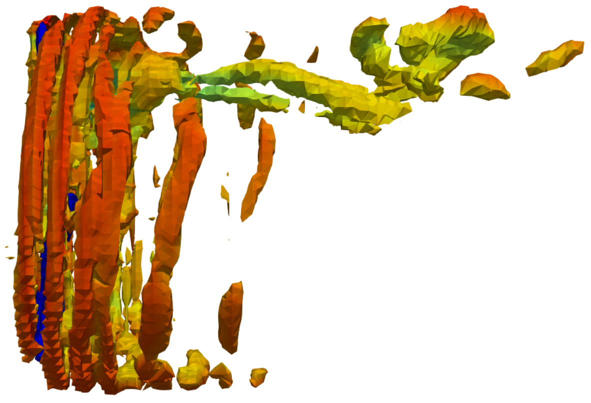


Figure 4.28: Iso-volume plot of the side view of the Goëttingen 803 turbine in the AL-BEM simulation

4.3.3. Conclusion
In this section, the time-varying AL-BEM method is partly validated and compared with the results of the uniform AD and AD-BEM methods. This chapter is added to this thesis for two reasons: to show the possibilities of the code and to check the influence of the transient near wake features on the recovery rate of the wake. As stated in the previous sections, the wake of the turbine recovers too slow in the uniform AD and AD-BEM simulations without the weir. It is expected that this is caused by the time-averaging nature of both the AD and RANS methods which filters out the transient near wake features. This causes a too low TKE in the near wake which leads to too little mixing of the flow and thus a too slowly recovering wake. The results of

the first simulations with the AL-BEM method look promising. In the flow fields of the results of the AL-BEM simulations the transient near wake features are present. As expected, this does lead to a slightly faster and more accurate wake recovery in comparison with the uniform AD and AD-BEM simulation.

The first simulation results also show that future work is needed to completely validate the AL-BEM method. In the first simulations, the performance of the turbine is over-estimated in the optimum TSR range in the first simulations with the AL-BEM method. This is most probably due to the introduced uniform force distribution over the blade. An addition of a force distribution function over the blade chord could improve this.

5 Case study: Pilot project Eastern Scheldt barrier

5.1. Introduction to case

The final goal of this thesis is to validate the numerical model on the pilot project of the five turbines in the Eastern Scheldt barrier. The Eastern Scheldt barrier is a storm surge barrier in the south-west of the Netherlands. It is a flexible barrier which only closes during high water storm surges. In normal conditions the openings of the barrier allow the tidal flow to enter the Eastern Scheldt estuary. The 9 km long barrier consists of a total of 64 openings divided into three parts. In each opening, the tidal flow is constricted by the pillars of the barrier and a weir. This leads to an acceleration of the flow up to a flow speed of 4 m s^{-1}, making the situation very suitable for the extraction of tidal energy. In 2015 Tocardo installed 5 turbines in the Roompot 8 opening. An overview of the complete situation in this opening is sketched very well by Verbeek in figure 5.1. As the turbines cause an extra blockage of the flow in the barrier, environmental and structural impacts are expected. With the permit for the five turbines, Tocardo got the obligation to monitor these impacts. To do so, flow measurements were executed by Tocardo and Deltares made a detailed blade resolved CFD model.

5.1.1. Flow measurements

Flow profiles were measured before and after the placement of the turbines. Before the placement, two three-beam ADCP's were mounted at the gate pillar pointed sideways and at the weir pointed upwards. Flow velocities near the turbine after implementation of the turbines were measured by two horizontal ADCP's mounted at the turbine hubs and struts of the centre and two outer turbines. The flow near the turbines is measured in both operating and non-operating situation. In figure 5.1 the locations of these ADCP's are shown within the barrier. Water level data is measured by gauges in the harbours up and downstream of the barrier.

5.1.2. Blade resolved simulations by Deltares

Deltares used a blade resolved CFD model to determine the environmental impact and power production of the turbines in the Eastern Scheldt (O'Mahoney et al., 2020). A 3D model of the barrier geometry and turbines was implemented into the CFD software STAR-CCM+. The blades were implemented into the mesh by a heavily refined rotating mesh across the turbines. To correctly capture all the flow effects near the turbine, the barrier and the bed, a mesh with a total of 55 million elements and a time-step of 0.05 s were used. This led to an overall simulation time of 4 weeks on 40 cores. The performance of the turbine in the simulations showed good agreement with the measurements. Flow profiles in the upstream area and in a situation without turbines corresponded very well with the ADCP measurements. The simulation showed an overestimation of the wake recovery compared to the measurements.

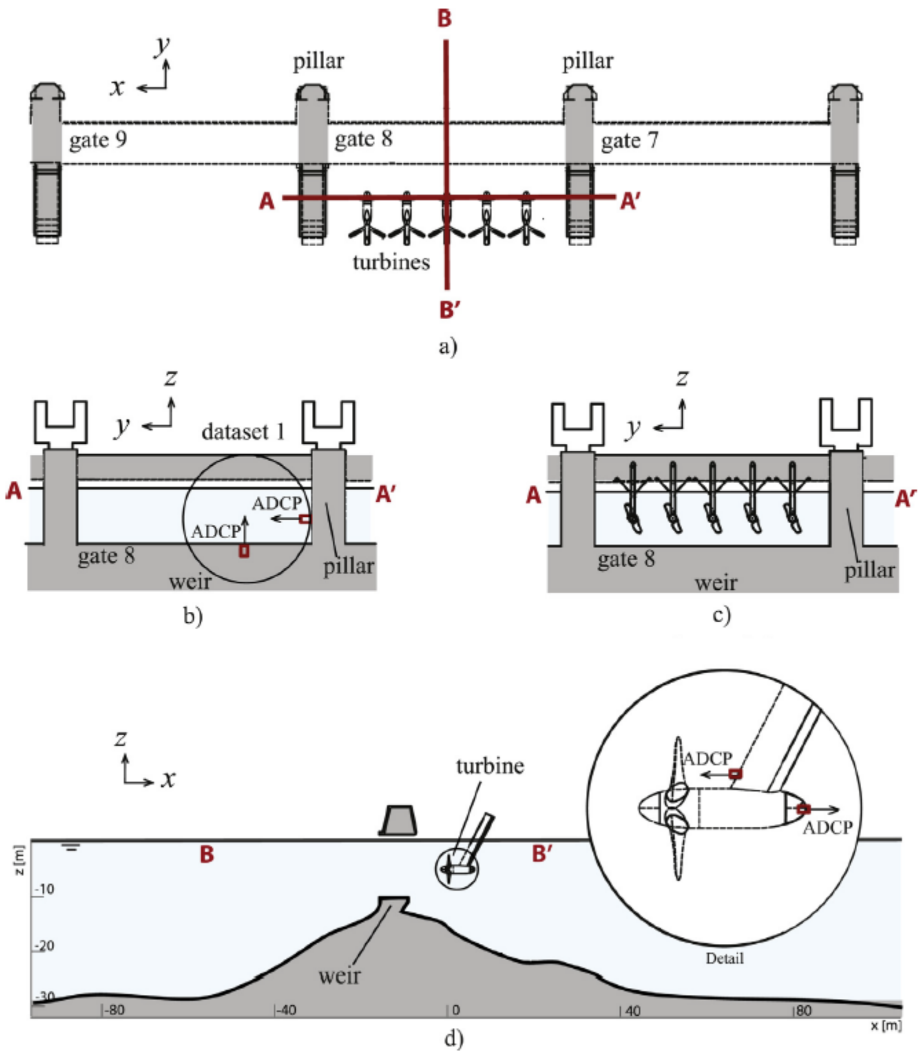


Figure 5.1: a) A schematic drawing of the gate with turbines in top-view indicating cross-sections A and B, b) cross-section A showing the gate from the side (looking to the east) with the positions of the two ADCP's mounted during the 2011 survey prior to turbine installation, c) cross-section A gives a side view of the turbines to the east, d) cross-section B showing the bathymetry over the gate from the side (looking north) and a detail of the turbine with ADCP's (Verbeek et al., 2020b)

5.1.3. Simulations by Svasek Hydraulics

Svasek Hydraulics executed simulations in the numerical flow solver FINEL of the flow through Roompot gates 6 to 10 (Talstra, 2017). A tetrahedral mesh extending 300 m up- and downstream of the barrier containing the bathymetry of the Eastern Scheldt and the weir was used. The barrier is represented by excluding a rectangular box from the mesh at the location of the pillars. Two water level boundaries representing spring tide were used at the up- and downstream end of the domain. The results of the simulation show an overestimation of the flow velocity over the weir. This could be explained by the simplified barrier geometry which causes a too small friction on the flow.

5.2. Numerical setup

In the setup of the experiments, a good balance between computational effort and detail of the solution is sought for. The computational model with the turbine parameterization is only useful as an addition to the model of Deltares when comparable results are obtained in a fraction of the time.

5.2.1. Mesh

The mesh from the simulations of Svasek Hydraulics (Talstra, 2017) is used as the basis of the mesh. Due to the smaller domain of interest in this thesis, only the part containing Roompot 8 and the half of Roompot 7 and 9 of the mesh by Svasek is used. This is similar to the mesh used by Deltares. The mesh by Svasek is unfortunately too coarse, with a minimum element size of 1.25 m. Therefore, a finer mesh is constructed over which the bathymetry of the mesh of Svasek is interpolated. The largest elements of the finer mesh are located at the upstream and downstream boundaries. At these locations, 8 elements are present over the depth with a ranging size between 4 and 5 m. The mesh is heavily refined towards the barrier and turbine. The smallest elements are located at the location of the turbine with a size of 0.28 m. This is equal to 20 elements over the diameter, which was taken as the minimal requirement to be able to obtain good results while using the nacelle model. The total number of elements in the mesh is equal to 1.4 million, which is only a fraction of the 55 million used by Deltares.

As stated before, the results of the simulations by Svasek showed an overestimation of the flow velocity above the weir. To improve the results, extra friction is added to the flow by increasing the steepness of the weir and introducing the foundation of the barrier in the mesh. In the mesh by Svasek, the weir is smooth while in reality, it is almost a rectangular concrete block. Subsequently, in the results of Svasek the flow only separates at the weir end, while in the more accurate simulations of Deltares, the flow already separates at the start of the weir.

The pillars of the barrier are constructed on a foundation of a large concrete block with a total length of 44.1 m. These foundation blocks lack in the mesh by Svasek while they almost certainly add extra friction to the flow. The foundation blocks are added to the mesh by adjusting the depth of the mesh according to the drawings of the three-dimensional model by Deltares. The steep gradient near these foundation blocks (and the weir) do however introduce some bad quality mesh elements.

5.2.2. Boundary conditions

The boundary conditions of the simulations are set such that the WT4 situation of the paper by Deltares is mimicked (O'Mahoney et al., 2020). In this situation during high flood tide, the water flows from the North Sea into the Eastern Scheldt with a constant discharge of 2278 m^3 s^{-1} due to a head difference of 0.54 m over the barrier.

On the sides of the flow domain (northern and southern boundary) a symmetry boundary is applied. On the free surface, a moving free surface boundary is applied. This is possible due to the larger scale of the domain (13x larger than flume experiments) in combination with the relatively less increased flow velocity (2.5x larger than flume experiments). The downstream water level (Eastern Scheldt side) is set at 0.58 m above NAP. On the upstream boundary (North Sea side) either a discharge boundary or a water level boundary of 1.12 m above NAP is applied. For the bed and the walls of the barrier, a closed wall boundary is used with a quite large Nikuradse roughness of 0.1 m. Svasek concluded that a variation in the Nikuradse roughness does not influence the flow velocity above the weir significantly (Talstra, 2017).

5.2.3. Turbine parameterization

As stated above, a good balance between computational effort and detail of the solution is sought for. Out of the models as used in this thesis, the AD-BEM method seems to be the best match for this purpose. The uniform AD model is not able estimate the turbine performance and less accurate in the near and far wake than the AD-BEM method. The AL-BEM method comes with a lot higher computational cost and is in this thesis less accurate in estimating the turbine performance.

The turbines are implemented in the mesh at 6.77 m downstream of the end of the weir. The turbines have a radius of 2.635 m and are represented by 1256 turbine integration points each. The AD-BEM method is used with the tip-correction by Edmunds et al. and the nacelle model which brings the flow to zero. Due to the saltwater inflow from the North Sea, the turbine forces are calculated with a density of 1025 kg m^{-3}. The turbines are all set to a rotational speed of 44 RPM as determined by Deltares for this specific case (O'Mahoney et al., 2020). The standard k-ϵ turbulence model is used. The increasing depth downstream of the weir gives the eddies the possibility to grow which diminishes the need to adjust the k-ϵ model in the horizontal plane.

5.3. Results of simulations

An overview of the results of the AD-BEM model is given in figure 5.6, 5.7, and 5.8. The model converges to a stable solution in about 10 hours on 16 processors.

5.3.1. Thrust and torque

In the AD-BEM simulation with the upstream discharge boundary, the average thrust on the five turbines, including the thrust on the nacelles, equals 114.92 kN. In the Eastern Scheldt, barrier thrust measurements are executed on turbine 3 and turbine 5 (de Fockert & Bijlsma, 2018). These measurements are executed on the turbine strut without the location on this strut being clear. It can therefore not be stated if the thrust on the support structure is (partly) taken into account. For the WT4 case, 105.04 kN was measured at turbine 3 and 117.68 kN at turbine 5, which equals an average thrust of 111.36 kN. This leads to an average error of 3.2%.

The power output of all five turbines is measured. Deltares corrected these measurements for mechanical losses (de Fockert & Bijlsma, 2018). The averaged corrected power output over the five turbines in the barrier during this specific case is 219.34 kW. The average power generated by the five turbines in the AD-BEM model, with the upstream discharge boundary is 246.76 kW. This leads to an average error of 12.5%. The blade resolved model for the Eastern Scheldt domain by Deltares (O'Mahoney et al., 2020) reaches an average relative error of -1.7% and -7.0% for respectively the thrust and the power. Which makes the blade resolved model slightly more accurate than the AD-BEM model.

5.3.2. Flow fields

In the simulation with the upstream discharge boundary, the upstream water level converged to a level of 0.96 m above NAP. This equal head difference over the barrier of 0.38 m instead of the actual 0.54 m. According to equation 5.1 this head difference in combination with the wet surface above the weir and the upstream discharge leads to a discharge coefficient of about 1.0, which is of course too high. Figure 5.2 confirms the too high discharge coefficient of the barrier in the simulation. In the figure, the axial flow velocity throughout the domain of the blade resolved model by Deltares and the AD-BEM model with the upstream water level boundary are plotted. In this simulation, the overall flow velocity in the domain is too high, which means that the friction in the mesh is too low.

$$Q = A\mu\sqrt{2g\Delta h} \tag{5.1}$$

In figure 5.3, the axial flow velocity throughout the domain of the blade resolved model by Deltares and the AD-BEM model with the upstream discharge boundary are plotted. The figure shows that the AD-BEM model with the upstream discharge boundary better approximates the flow than the model with the water level boundary. The wake recovery in this AD-BEM simulation is slower than in the blade resolved model by Deltares. Deltares compared their results with the measurements and concluded that the wake recovery in their model was faster than the wake recovery in the measurements. The accuracy of the wake recovery in the AD-BEM model can therefore not be stated.

Figure 5.4, shows the axial flow velocity throughout the domain of the STAR-CCM+ model by Deltares and the FinLab model without turbines. This simulation, with an upstream discharge boundary, again shows the lack of friction in the proximity of the barrier. A higher flow velocity is reached on top of the weir in the model by Deltares. This higher velocity influences the flow downstream of the barrier. This is again confirmed in figures 5.5a and 5.5b, in which the velocity profiles at 3.3 m downstream of the end of the weir of the AD-BEM model without turbines are compared with ADCP measurements. Figure 5.5a shows that the flow separation at the downstream end of the weir is represented in the FinLab model. The figure also shows that there is still some error in the geometry of the barrier in the model as the FinLab velocity profile extends deeper than the ADCP measurement.

At last, figure 5.8 from the AD-BEM model with the discharge boundary, seems to confirm that the adjusted k-ϵ model as discussed in chapter 4.2 is indeed not necessary in these simulations. The wakes of the five turbines merge within a distance of 5 to 10 turbine diameters downstream. It is however, not clear if this is caused by the increasing depth of the geometry or the developing horizontal boundary layers from the separating flow downstream of the pillars.

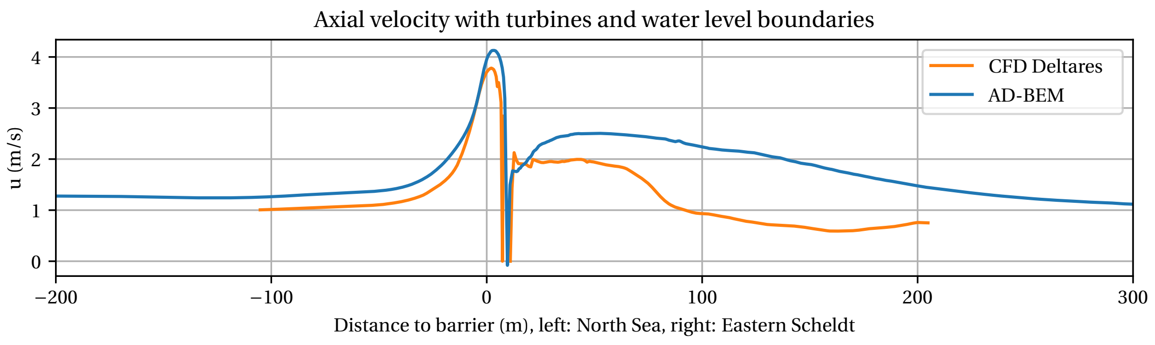


Figure 5.2: Axial velocity throughout flow domain in the situation with turbines and an upstream water level boundary

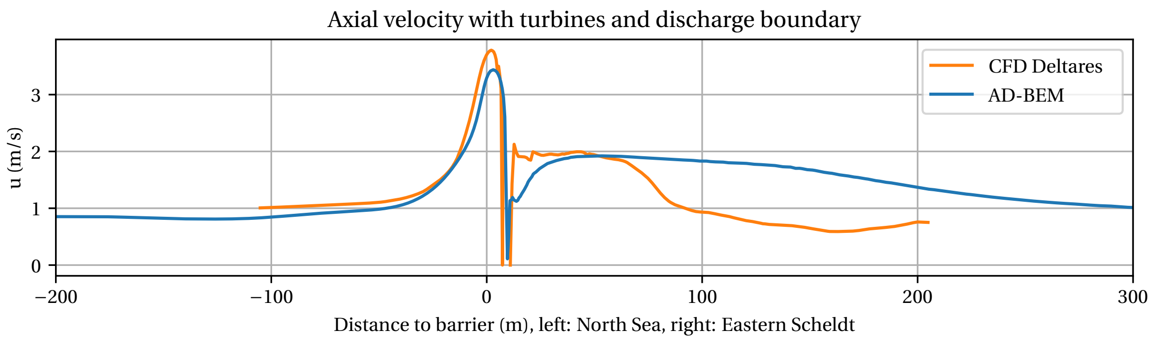


Figure 5.3: Axial velocity throughout flow domain in the situation without turbines and an upstream discharge boundary

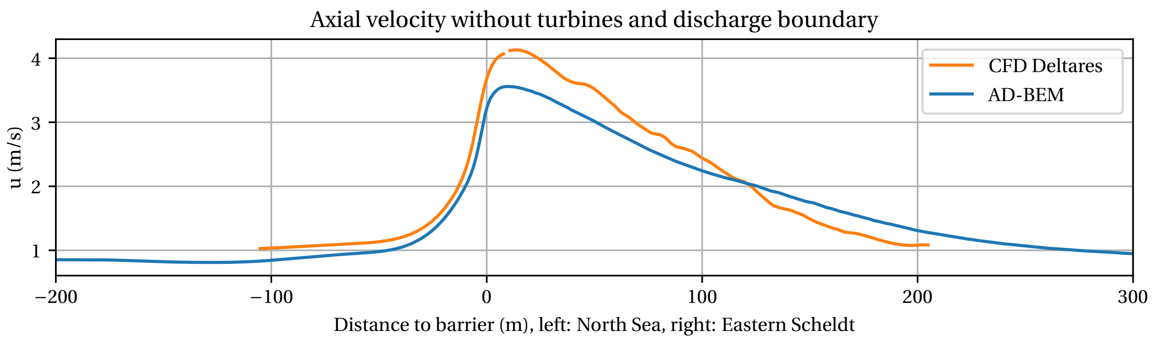
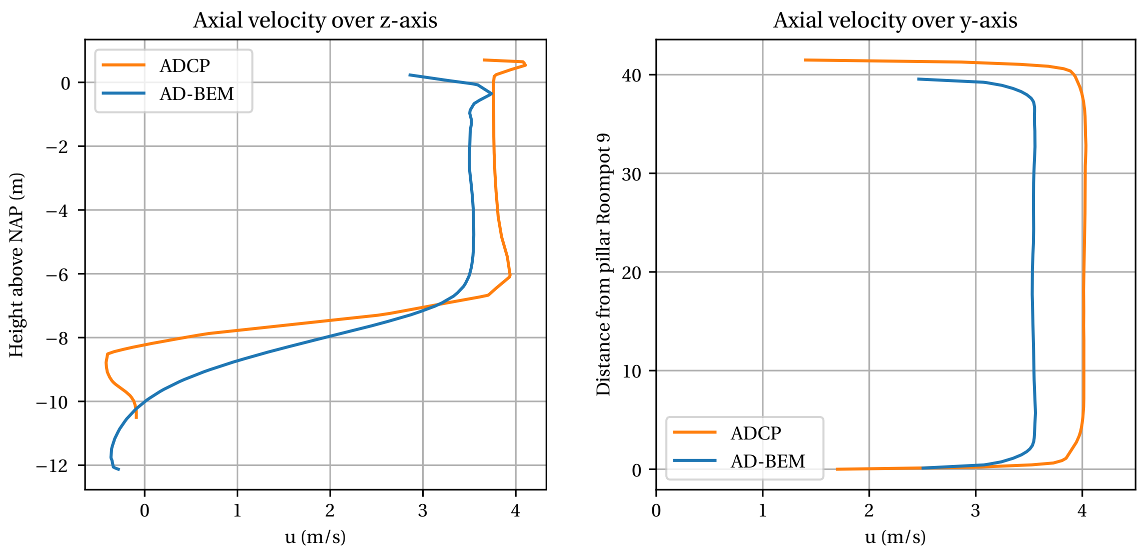


Figure 5.4: Axial velocity throughout flow domain in the situation without turbines and an upstream discharge boundary



(a) Axial flow velocity over the height of the water column at 9.2 m from pillar Roompot 9 (towards pillar Roompot 8)

(b) Axial flow velocity over the width of port Roompot 8 at the height of the turbine axis

Figure 5.5: Comparison of ADCP measurements with the AD-BEM model at 3.3 m downstream of the end of the weir

5.4. Conclusion

The Eastern Scheldt simulations in this chapter show the practical applicability of the AD-BEM model. Accurate estimations of the thrust and power of the turbines are made when the local flow velocity is modelled accurately. These results are reached with only a fraction of the computational cost of the blade resolved model by Deltares (O'Mahoney et al., 2020). This is a promising results as it shows that the model could, e.g., be used in future work to find the barrier gates at which the most power could be extracted. The simulations also show that flow fields from the Eastern Scheldt simulations as generated in this thesis are not accurate yet. The barrier geometry seems to be too much simplified within the mesh. This leads to a too low friction within the model which makes it impossible to determine accurate discharge coefficients. It does not seem possible to improve the mesh much further with method used in this thesis. A full three-dimensional model as used by Deltares (O'Mahoney et al., 2020) seems the only appropriate solution. The inflow profile as used by Deltares (O'Mahoney et al., 2020) is also expected to improve the simulation results. The priority should however be on the improvement of the mesh. At last, the simulations show that the adjusted k-ϵ model as introduced in chapter 4.2 is not needed due to irregular Eastern Scheldt geometry which leads to the merging of the wakes.

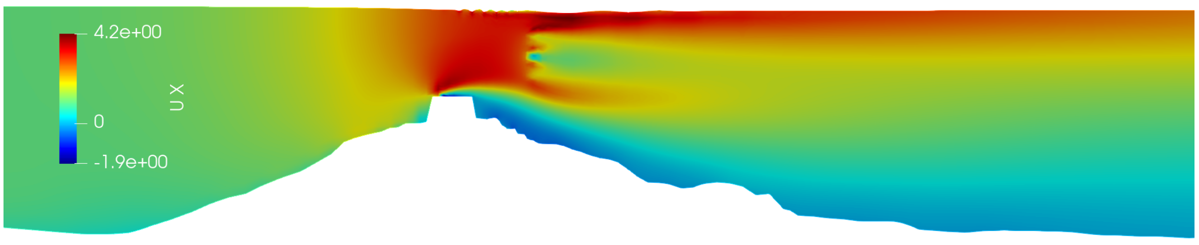


Figure 5.6: Side view of axial flow velocity plotted on plane through centre turbine

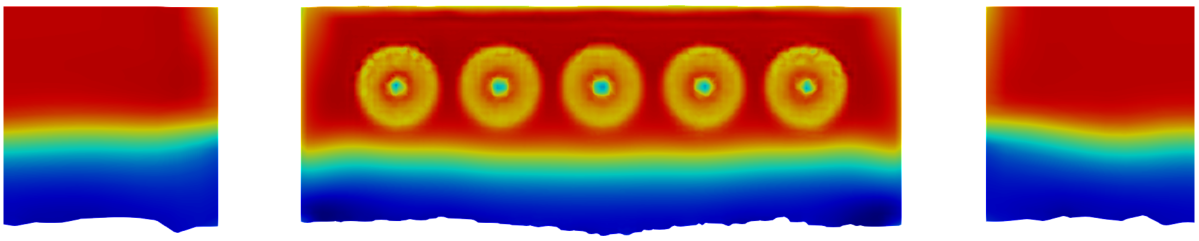


Figure 5.7: Axial view of axial flow velocity plotted on plane through the turbines

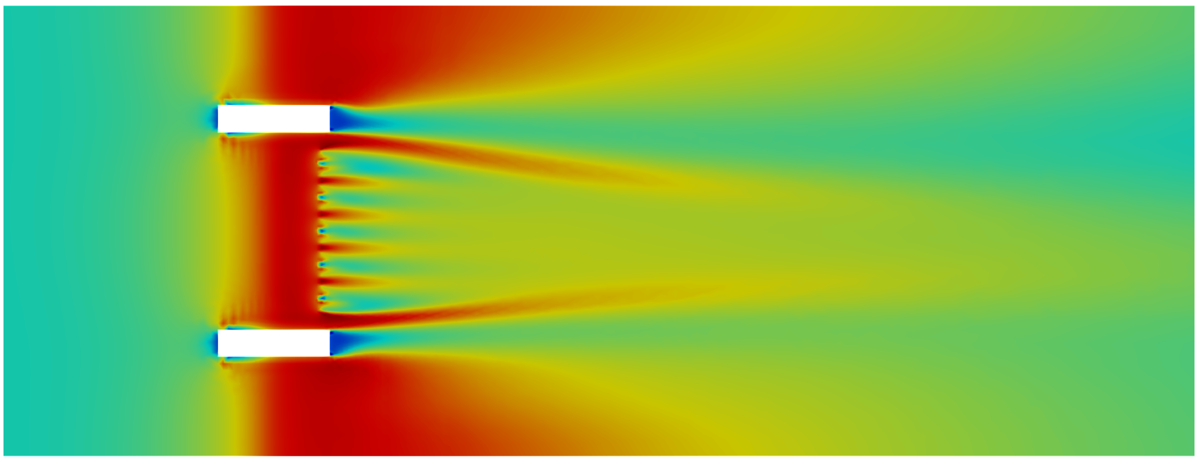


Figure 5.8: Top view of axial flow velocity plotted on plane through the turbines

6 Discussion

All the simulations in this thesis have been executed with either the uniform AD, AD-BEM or AL-BEM turbine parameterization. In this chapter, the results of the simulations with these methods and the flaws and strengths of each of these methods are discussed. Furthermore, applied simplifications to the parameterizations are discussed which could lead to improvement of the parameterizations in future work.

6.1. Thrust and power

In this thesis, thrust and power are used to indicate the performance of a turbine. The AD-BEM and AL-BEM are both able to make estimations of the generated thrust and power by the turbines, the uniform AD method can not. When interpreting the results of the simulations with one of the BEM models, it is important to notice that the thrust and power generation of both methods strongly depend on the local flow velocity. The produced thrust is quadratically dependent on the local flow velocity while the produced power even has a cubed dependency on the local flow velocity. This implies that an accurate validation of the flow situation without turbines is always required before implementing the turbines into a model.

In this thesis, all measured power data of the Tocardo turbines is corrected for 10% mechanical loss. This correction is an estimation for the mechanical loss in the full-scale turbine as provided by Tocardo. The mechanical loss in the scale model is estimated to be the same. Furthermore, all generated thrust data by the BEM models include the thrust on the nacelle. It is difficult to validate if the thrust generated by the nacelle model is accurate as no separate thrust measurements of the blades and nacelle are available. The thrust on the support structure is completely disregarded in the generated thrust data by the simulations while this thrust is taken into account in the measurements by Verbeek et al. (2021) and Stallard et al. (2013). In the Eastern Scheldt measurements by Tocardo (de Fockert & Bijlsma, 2018) it is not clear if the thrust on the support structure is added to the measurements .

6.1.1. AD-BEM

To put the accuracy of the AD-BEM model in perspective, it is interesting to compare the results of the simulations in this thesis with earlier conducted simulations in literature. The AD-BEM method can, e.g., be compared with the studies by Malki et al. (2013) and Edmunds et al. (2017). It must be noticed Malki et al. (2013) did not apply a tip-correction, Edmunds et al. (2017) of course did. In both studies, the nacelle is embedded in the mesh. Malki et al. (2013) found convergence on mesh dependency at 52 elements over the diameter of the turbine, Edmunds et al. (2017) started at 80 elements over the diameter. In this thesis, the mesh is no further refined than 40 elements over the diameters. This coarser mesh could, of course, influence the results. The overall thrust and power generation in the study by Malki et al. (2013) seems to correspond well with the results found in this thesis. A slight overestimation of the thrust and power coefficients in the optimum TSR range is found. The thrust and power estimations by Edmunds et al. (2017) correspond better with the experiments than the results found in this thesis.

The better performance of the AD-BEM model by Edmunds et al. (2017) could be due to the combination of the applied tip-correction with the used turbine. In the tip-correction by Edmunds et al. (2017), the applied momentum source on the flow is multiplied with a factor that is elliptically increasing towards the tip. This is a good approximation for the downwash produced by a tapered blade. Later, Edmunds et al. (2020) introduced a new blade-specific downwash distribution function based on the lifting line theory by Prandtl. With this new method, Edmunds et al. (2020) show that the blade-dependent tip-correction of the turbine as applied in the earlier study corresponds very well with an elliptical profile, which leads to the good results. The tip of the Tocardo T2 turbine is quite blunt, which could lead to a downwash distribution function which does not correspond well with the elliptical function. In addition, for both the turbine performance and the flow fields, the accuracy of the simulations with the Tocardo T2 using the Prandtl and Edmunds et al. tip-correction do not differ a lot. The choice to do all simulations with the Edmunds et al. tip-correction after one set of simulations could be stated as premature and it is recommended to implement the blade-specific tip-correction by Edmunds et al. (2020) in future work.

6.1.2. AL-BEM

In this thesis, only a start is made with the AL-BEM model. The first results of the simulations with the method can, nevertheless, be compared with literature, e.g., with the studies by Wimshurst and Willden (2018) and Nathan (2018). The results of both these studies show better correspondence with the respective used experimental data sets than the results of the AL-BEM method in this thesis. In future work, the AL-BEM method of this thesis could be improved to achieve a similar accuracy.

The velocity sampling method in AL-BEM simulations has always been subject to discussion. In this thesis, the velocity sampling method as introduced by Nathan (2018) is used. In this method, the velocity is sampled at the centre chord line of the blade in the previous time-step. Other velocity sampling methods exist where the velocity is, e.g., sampled at an upstream location (Shen et al., 2009) or by using a Lagrangian method following the movement of the blades (Yang & Sotiropoulos, 2018).

In this thesis, a method is proposed to uniformly distribute the blade force according to the shape of the blade by using the Fibonacci circle. In reality, the highest lift forces occur at the thickest point on the blade. Therefore, usually, a Gaussian distribution function is applied to distribute the forces. It must be noted that the generated thrust and power coefficient by the AL-BEM model are generally too high in this thesis. When applying the Gaussian distribution, the force would mainly be applied near the centre chord line. This would decrease the velocity near this line and therefore also decrease the forces on the blade. With the Gaussian distribution applied, a new and probably lower balance would be formed between the forces on the flow and the forces on the blades. A combination of the current sampling method with the implementation of a force distribution function seems to be the main improvement for the AL-BEM method in future work.

A smaller improvement could be made by correcting for the spanwise flow over the blade. As the AL-BEM model combined with the LES turbulence model resolves for the tip-vortices, no tip-correction is applied to the model. The spanwise velocity as caused by the pressure difference over the blade is, however, still ignored. The velocity sampling method in the AL integration points only considers the axial and tangential flow velocities. Wimshurst and Willden (2018) uses a correction function to correct for the influence of this spanwise flow.

No mesh-sensitivity study is performed for the AL-BEM method yet. The finest mesh of the AD-BEM method with 40 elements across the diameter of the turbine is used. The mesh is, however, refined in the axial direction to be able to capture the downstream trailing tip-vortices. It is expected that the AL-BEM method is more sensitive to the mesh than the AD-BEM method. The simulations in the study by Wimshurst and Willden (2018) are, e.g., performed on a mesh with 69 elements across the turbine radius, making the mesh about 3.5 times as fine as the mesh used in this thesis. A time-step sensitivity study also lacks in this thesis. As the blade spin, the time-step determines the rotation interval at which the BEM equations are solved which influences the flow velocity at the velocity sampling points. Therefore, it is expected that for the AL-BEM method the time-step does influence the performance of the model. In this thesis, one rotation is divided into 40 time-steps. Wimshurst and Willden (2018) use, e.g., 900 time-steps per rotation. Both these studies could be executed as future work.

6.2. Flow fields

The goal of this thesis is to produce a relatively cheap numerical model of a turbine which is not only able to estimate the performance of the turbine but also able to generate accurate flow fields. Each of the three parameterization methods generates a completely different flow field downstream of the turbine. The uniform AD method only causes an axial velocity deficit in the wake. The AD-BEM method adds a sharp velocity deficit of the nacelle and the rotation of the near wake to this axial velocity deficit. In the AL-BEM simulations, the blade rotation is resolved which leads to the addition of the transient features of the near wake to the flow field.

6.2.1. Wake recovery

Throughout the uniform AD and AD-BEM simulations, the wake of the turbine recovers too slow. The wake recovers faster and more accurate in the AD-BEM simulations than in the uniform AD simulations. The extra velocity shear in the near wake of the AD-BEM simulations caused by the boundary layer around the nacelle and the rotation of the near wake generates a higher TKE. The larger TKE could lead to a faster mixing of the flow and thus to the faster wake recovery. In the AD-BEM simulations with the weir the wake seems to recover a lot more accurate than in the simulations without the weir. In these simulations, a similar principle arises. The flow over the weir causes a large velocity shear in the recirculation zone and thus a large TKE which again leads to the faster and more accurate wake recovery.

The cause of the too small TKE in the near wake of the uniform AD and AD-BEM methods seems to be the time-averaged nature of both the AD and RANS methods. The transient near wake features are averaged out over time by the methods. The simulations with the AL-BEM method confirm that the transient features of the near wake influence the wake recovery rate. The wake in the AL-BEM method, which does resolve these features, recovers faster and more accurate than the wakes in the uniform AD and AD-BEM simulations.

In literature, several suggestions exist to, apart from only introducing a source in the momentum equation, also introduce a TKE source term into the k-ϵ model. Edmunds et al. (2017) do however, state that this should not be needed with the application of their tip-correction on the AD-BEM model. The tip-correction by Edmunds et al. (2017) increases the applied momentum source term near the tip and should therefore also increase the TKE near the tip. In the simulations in this thesis the application of the tip-correction by Edmunds et al. (2017) does however, not lead to a more accurate wake recovery in comparison with the application of the Prandtl tip-correction. Therefore, in future work, it could be useful to further investigate the tip-correction or the addition of a TKE source term in the AD methods.

The simulations with the AD-BEM model of the experiments by Stallard et al. (2013) also show a too slow wake recovery. However, for these simulations it can be argued that the high turbulence intensities in the experiments lead to a faster wake recovery. The experiments with five side-by-side placed turbines by Stallard et al. (2013) show that the wakes of the individual turbines merge within a distance of 5 to 8 diameters downstream. In the simulations, the wakes do not merge at all within the flow domain. The fast merging of the wakes in the experiments could also be due to the high turbulence intensities. However, the fact that the individual wakes in the simulations do not merge at all seems incorrect. In the k-ϵ model, the length scale of the dissipation term is limited by the small water depth in these simulations, while in reality, the horizontal eddies could grow up to the horizontal domain size. Therefore, in these simulations, the horizontal viscosity is multiplied with a factor 100. As shown, the results improve and wakes merge. The Eastern Scheldt simulations do, however, show that this adjusted k-ϵ model only applies on the merging of wakes in flat flows. In the Eastern Scheldt simulations the wakes merge without the k-ϵ model being adjusted. It is however not clear if this occurs because of the increasing water depth downstream of the turbines, or because of the developing horizontal mixing layer downstream of the pillars. In future work, more simulations with side-by-side placed turbines could be executed to give a stronger conclusion on when to apply the adjusted k-ϵ model.

6.2.2. Nacelle

In this thesis, a small AD is introduced to represent the nacelle of the turbine in the AD-BEM and AL-BEM methods. In most coupled CFD-BEM models in literature, the nacelle is embedded in the mesh. The benefit of the AD method is that a turbine can be placed anywhere in the mesh by just changing the input settings. The flow field around the nacelle as produced by this method can be compared with, e.g., the results of the simulations by Churchfield et al. (2015), who used an actuator line to represent the nacelle. In all runs as executed in this thesis the axial velocity in the nacelle centre varies between -0.1 m/s and 0.4 m/s. The highest nacelle velocity occurs in the Eastern Scheldt simulation where the approach flow is the highest too. In all simulations, the flow clearly deflects around the nacelle. In the study by Churchfield et al. (2015), this deflection was not very well noticed in the results. However, by modelling the nacelle as an actuator disc, the three-dimensional geometry of the nacelle is disregarded. This three-dimensional geometry could potentially be better captured by applying multiple AD's along the axial axis of the nacelle or by the AL method by Churchfield et al. (2015). The support structure is also modeled by an AL by Churchfield et al. (2015). In this thesis the effect of the support structure on the flow is completely disregarded. The support structure could be simply added to future work introducing this AL into the model. Improving the nacelle and adding the support structure into the model would lead to more velocity shear in the near wake and could therefore potentially lead to a faster and more realistic wake recovery.

In this thesis, the nacelle is modeled such that the axial velocity of one of the four nodes of the element in which the centre of the nacelle is located is brought to zero. Most of the time, this will of course be the most downstream node. However, many other choices could have been made to model nacelle. A large drag coefficient could, e.g., be applied to the local velocity. This, however, would lead to very large drag coefficients when the flow approaches zero, which subsequently increases the chance of instabilities. Another option could be to bring the average velocity in the elements in which the nacelle integration points are located to zero. This is, however, hard to manage as most of the time, this would lead to an positive axial velocity in the most upstream node of the element and a negative velocity in the most downstream point of the element. If the momentum source, needed to reach this zero average velocity, is then equally divided over all four nodes, this would lead to an even more negative velocity further downstream of the element.

6.2.3. Recirculation zone

Several simulations with flows over a weir are executed to validate the AD-BEM model for the situation in the Eastern Scheldt barrier. This thesis, however, mainly focuses on the validation of the turbine models. The accuracy of the flow separation produced in the simulations containing these weirs is not investigated throughout. E.g., in all simulations, a standard Nikuradse roughness is used. A sensitivity analysis to this Nikuradse roughness could have increased the accuracy of the velocity profile and therewith the accuracy of the flow separation. Furthermore, in all simulations which contain a weir, the k-ϵ turbulence model is used. As concluded in section 2.2.3 of the literature study, the k-ϵ tends to overestimate the mixing in the recirculation zone.

All simulations containing a weir, however, do show a negative flow velocity near the bed. The AD-BEM simulations of the experiments by Verbeek et al. (2021) also shows that the recirculation flow is suppressed differently with different turbine-weir geometries. The turbine-weir geometries in which the recirculation flows are most suppressed are also the turbine-weir geometries in which the maximum power is harvested. This hints to the confirmation of the conclusion by Verbeek et al. (2021) that a turbine-weir geometry exists in which the power harvesting is optimum and environmental impact limited. However, in this thesis this claim is only based on three simulations. In future work, the simulations containing the weir could be extended and further investigated to confirm this claim.

6.2.4. Free-surface

In all the simulations of the flume experiments by Verbeek et al. (2021) and Stallard et al. (2013) no free-surface is applied. In these simulations, the bypass flow causes a large flow velocity near the free-surface, leading to instabilities in the flow. The free-surface is replaced by a symmetry boundary in these simulations. With this simplification, the influence of a moving free-surface on the performance and wake recovery of the turbine remains unknown and could be investigated further in future work.

6.3. Field case: Eastern Scheldt

This thesis is of course mainly initiated by the pilot project with the five Tocardo turbines in the Eastern Scheldt barrier. The implemented turbine parameterization in FinLab can be used to do further research to the environmental impact of the pilot project and possible future extensions of the project within the barrier. In this thesis, a start is made with executing simulations with the turbine parameterization in the barrier geometry. The first results of the simulations with the AD-BEM model seem promising, but do need more extensive research to be able to compete with the accuracy of the blade resolved model by Deltares (O'Mahoney et al., 2020).

The main point of discussion in the Eastern Scheldt barrier field case simulations is the applied mesh. The applied mesh contains a very simplified barrier geometry. In the simulations, it appears that the simplified geometry does not provide enough resistance to the flow. Consequently, the overall discharge through the flow domain gets too high when two water level boundaries are applied. More accurate results are obtained in this thesis by applying a discharge boundary. However, with this boundary condition it is impossible to determine the discharge coefficients over the barrier gates. These coefficients are, nevertheless, one of the mayor points of interest when considering the environmental impact of the turbines in the barrier. Furthermore, even with the discharge boundary, some inaccuracies in the flow profiles around the barrier arise which can be assigned to the inaccuracy of the mesh. Conclusively, the development of a more accurate mesh should be the first step when considering further application of the turbine parameterizations from this thesis to the Eastern Scheldt pilot project.

Thereafter, the inflow velocity profile and roughness of the bed seem to be important parameters to further improve the accuracy of the simulations. In the mesh, the barrier is located about 300 m downstream of the upstream boundary. With a water depth of about 35 m, this does not seem sufficient for the flow to fully develop. Deltares (O'Mahoney et al., 2020) used an inflow velocity profile to overcome this problem. This inflow profile is derived from the bed roughness in the Eastern Scheldt.

6.4. Practical applicability

The simulations in the Eastern Scheldt domain show the practical relevance of the AD-BEM turbine parameterization. In future work, the uniform AD method can be used in large scale models to initiate the formation of a wake of a turbine. In smaller scale models, the AD-BEM and AL-BEM methods can be used to estimate the performance and the impact on the flow of a turbine. Both methods show to be able to make good rough estimates of the performance of the turbine when the local flow speed is validated accurately. E.g., in the simulations of the experiments by Verbeek et al. (2021), the model shows to be able to make accurate estimations of the location of optimum power extraction. As already discussed in section 6.2.1, flow fields of the wakes of the turbines resulting from the simulations should be interpreted with more care.

It should also be possible to implement the turbine module into another non-hydrostatic flow solver with, of course, slight modifications to the code depending on the chosen solver. The turbine module uses clear in- and output variables and is not completely intertwined with the other FinLab modules. Before the start of the iterations, the turbine module only reads in the mesh. During the iterations, the mesh and the flow velocity in the mesh elements are used as input, while the determined source term is the only output to the other FinLab modules. That means that, if implementing the module into another solver, only these in- and output variables need to be connected with the code of the solver.

When thinking out of the box, other practical applications of the turbine parameterizations in this thesis could also be imagined. All three turbine implementations could, e.g., be reversed to simulate the propellers or bow thrusters of a vessel.

6.5. Code

The turbine parameterizations are implemented into FinLab by the code which can be found in appendix C. Of course, several discussion points can be made about the method of the code. At the start of the FinLab iterations, the code prints the number of found turbine integration point (and AL points) to the Linux terminal. It is important to note if these agree specified number of turbine integration point in the input as there are some flaws in the subroutine '`element_search`'. In small scale simulations, the floats describing the positions of the nodes within the mesh should be written in a high enough accuracy (enough decimals), otherwise, a turbine integration point could be found multiple times in the mesh. The subroutine counts an integration point to be in the element if it is on the element boundary, consequently, with a too low accuracy, an integration point could be found twice. Furthermore, in a two-dimensional mesh an integration point that is exactly on a zero boundary could lead to a determinant of infinity (dividing by zero). The subroutine '`element_search`' could be improved in future work to overcome these flaws.

Furthermore, the subroutine '`element_search`' is not programmed in a computationally efficient way at the moment. It uses a nested loop to find the elements in which the turbine integration points are located. Computationally this is not a very effective method as it needs [number of turbine integration point * number of elements in partition] computational steps per partition. E.g., to find the 5024 points as in the AL-BEM simulation in this thesis, about 10 to 20 minutes is needed depending on the mesh and number of partitions. On simulations that take several hours to complete, this is acceptable and therefore not improved. It should however, be possible by performing matrix calculations to decrease the computational cost of this subroutine. This could be executed as future work but should not be seen as a priority.

A limitation of the code is that it is impossible to place the axial axis of the turbine under an angle with respect to the mesh axes. The turbine can only be placed with its axis on either the x-, y- or z-axis. It was simply not necessary in this thesis to place the turbine under an angle, this should however, not be too hard to implement in the code if it is needed in future work.

A minor point of discussion is how the code reads the velocity from the mesh in the '`get_velocity`' subroutine. The flow velocity in the turbine integration points is now determined by taking the average flow velocity of the four nodes which form the element in which the integration point is located. The subroutine could be changed such that the velocity is interpolated to the exact location of the integration point. The effect on the results is, however, expected to be minor.

7 Conclusion and recommendations

7.1. Conclusion

The goal of this thesis is to validate a relatively cheap yet accurate numerical model of a tidal turbine in the vicinity of a weir. The turbine in the numerical model is placed in the vicinity of a weir as the model is applied to the turbine implementation in the Eastern Scheldt barrier. In this barrier five turbines are attached to the geometry of the barrier and the flow passing these turbines is contracted by this geometry and a weir. Accordingly, the following main research question is set up:

How can horizontal axis tidal turbines in the vicinity of a weir be represented in a relatively cheap three-dimensional numerical model?

Three different turbine parameterizations are implemented into the finite element model FinLab to answer this main research question. These parameterizations are a uniform AD model, an AD-BEM model and an AL-BEM model. All three models add an extra source term to the Navier-Stokes equations which represents the forces exerted by the turbine on the flow. All three methods use an, in the mesh integrated, AD to exert the forces on the right location. This AD is formed by turbine integration points which are placed according to the pattern of a Fibonacci circle. The uniform AD model uniformly distributes the time-averaged axial force over the AD. The AD-BEM model determines the forces in the AD integration points according to the local flow in combination with experimental lift and drag. This leads to an integration point specific axial and tangential force. These forces are determined averaged over one full rotation. The AL-BEM model uses the same method but only exerts the axial and tangential forces at the time-step specific location of the blades. The three models are validated by performing simulations of flume experiments with and without a weir and performing simulations in the Eastern Scheldt domain. Conclusions of the performance of the three models can be drawn by answering the three sub-questions according to the results of these simulations.

How accurate are the main characteristics of the turbine represented in the numerical model with the turbine parameterization?

The uniform AD, AD-BEM and AL-BEM all show significantly different representations of the by the turbine generated thrust and power. In the uniform AD method, the thrust is an input value while no power is determined, this method is therefore further disregarded when it comes to the main turbine characteristics. The results of the AD-BEM simulations show a relative error of within 20% when compared to the power measurements of the experiments by Verbeek et al. (2021), the accuracy of the thrust coefficients fluctuates more. The accuracy depends on the element size, number of integration points, TSR, tip-correction, application of a free-surface and the choice of nacelle representation. In the simulations of the experiments with the weir by Verbeek et al. (2021), the AD-BEM model shows to correctly estimate the position of maximum power extraction in a combined turbine-weir geometry. In the Eastern Scheldt simulations, the AD-BEM reaches a relative error of 3% and 12% for respectively the thrust and power coefficient. This is slightly less accurate than the results of the blade resolved simulations in the Eastern Scheldt domain by Deltares (O'Mahoney et al., 2020). Both these sets of simulations show the practical applicability of the model.

The AL-BEM should be able to produce at least similar accuracy to the AD-BEM method when it comes to power and thrust. The AL-BEM method in this thesis is, however, not yet able to do so. In the AD-BEM and AL-BEM methods, the nacelle is represented by a small AD. This AD brings the flow in the nacelle to zero by applying a thrust force. Overall, the nacelle module works well and does not exert an extraordinary large force on the flow. The support structure of the turbine is not represented in the models in this thesis.

How accurate are the main characteristics of the flow around the turbine and the weir represented in the numerical model?

Each of the three methods generates a completely different flow field downstream of the turbine. The uniform AD method only causes an axial velocity deficit in the wake. The uniform AD method shows to be a useful turbine representation in large scale models. In the AD-BEM method the nacelle module and the tangential forces are added. The added tangential forces lead to an accurate time-averaged representation of the rotation of the near wake in the AD-BEM method. The time-averaging nature of the AD and RANS method filter out the transient near wake features. The nacelle module functions well, it brings the axial flow velocity close to zero and deflect the streamlines around the nacelle. The AL-BEM method adds transient features such as the downstream trailing tip-vortices to the flow.

The lack of the representation of the transient near wake features in both the AD models lead to an under-estimation of the velocity shear in the near wake. The uniform AD under-estimates the velocity shear even further as the representation of the nacelle and near wake rotation lack. The lower velocity shear leads to an under-estimation of the TKE in the k-ϵ in the near wake. This under-estimation of the TKE in the near wake leads to too little mixing of the flow and thus a too low wake recovery rate. In the simulations containing the weir, the wake recovery is more accurate due to the extra velocity shear as added in the recirculation zone. Resolving the transient features of the near wake in the AL-BEM method improves the accuracy of the wake recovery.

In the simulations with the weir, downstream of the weir a recirculation zone forms. This recirculation flow is suppressed if the turbine is placed in the range between the downstream end of the weir and several turbine diameters further downstream. These are also the turbine-weir geometries in which the maximum power is harvested. The combination of these two hint to confirmation of the conclusion by Verbeek et al. (2021) that a turbine-weir geometry exists in which the harvested power is optimum while the environmental impact of the turbines is limited.

In the Eastern Scheldt simulations, the flow is not very well represented yet. The mesh used in this thesis is not accurate enough which leads to an under-estimation of the friction over the barrier. Therefore, the model from this thesis is not yet able to compete with the blade resolved model by Deltares (O'Mahoney et al., 2020) when it comes to estimating the environmental impact of the turbines in the estuary.

How do the computational costs of the numerical model with the turbine parameterization compare to the computational costs of a blade resolved turbine model?

The computational costs of the three methods significantly differ. The uniform AD has no mesh resolution requirement. The AD-BEM requires at least about 20 elements over the diameter of the turbine. The AL-BEM method is expected to require an even higher mesh resolution, but this is not yet validated. In addition, the AL-BEM method requires a significantly smaller time-step than the two AD methods due to the blade rotation. Overall the computational times of the executed simulations differ from several hours up to one day on 20 processors. The computational times of the Eastern Scheldt field case simulations are only a fraction of the earlier executed blade resolved simulations by Deltares (O'Mahoney et al., 2020).

7.2. Recommendations

The main recommendations for future work are discussed here. The recommendations follow straight up from the conclusion and the discussed items in chapter 6.

7.2.1. Uniform AD

In the simulations with the uniform AD method, the wake of the turbine recovers too slow due to the too low velocity shear and TKE in the near wake. Therefore, the main recommendation for the uniform AD method would be to further investigate the TKE in the near wake. This could be done by adding a TKE source term to the uniform AD method.

7.2.2. AD-BEM

In the simulations with the AD-BEM method the wake also recovers too slow due to the too low velocity shear and TKE in the near wake. The wake does, however, recover faster and more accurate in comparison to the uniform AD method. The main recommendation for the AD-BEM method would therefore also be to further investigate the TKE in the near wake. This could be done by either adding a TKE source term or by investigating the applied tip-correction further. For the tip-correction it is recommended to implement the aerofoil specific downwash distribution function application as proposed by Edmunds et al. (2020).

7.2.3. AL-BEM

In the simulations with the AL-BEM method the wake of the turbine recovers more accurate and the focus should be to improve the accuracy of the thrust and power estimates. This could be done by applying a more realistic force distribution function. To do so, the Gaussian force distribution function could be combined with the Fibonacci distribution as applied in this thesis. Thereafter, the AL-BEM method could be further improved by implementing the spanwise flow correction by Wimshurst and Willden (2018). This correction method could be calibrated for the Tocardo turbines by using the data from the blade resolved simulations by Tralli et al. (2015). After improvement of the method, it is recommended to perform a sensitivity analysis of the AL-BEM method to mesh resolution and time-step size.

7.2.4. Nacelle

For both the AD-BEM and AL-BEM methods, it is recommended to further develop the nacelle model as this could lead to a better representation of the velocity shear in the near wake. This again could lead a more accurate wake recovery in the simulations. The nacelle subroutine could be made three-dimensional by applying more AD's or AL's along the axial axis of the nacelle. The support structure can be implemented into the model by applying an AL with an axial thrust force.

7.2.5. Field case: Eastern Scheldt

In the Eastern Scheldt simulations, the barrier geometry does not provide enough resistance to the flow. This happens due to the simplified mesh which is used in this thesis and therefore the priority should be to improve this mesh. It is recommended to use a fully three-dimensional mesh and not use an extended 2DH mesh as is done in this thesis. A fully three-dimensional mesh should, furthermore, prevent the formation of bad quality mesh elements. After the formation of an improved mesh, the inflow velocity profile as used by Deltares (O'Mahoney et al., 2020) can be used to overcome the short length of the flow domain.

7.2.6. Code

The main improvements in the code can be achieved in the '`element_subroutine`'. In the code provided with this thesis, the code uses a nested loop to find the turbine integration points in the mesh. The nested loop can be replaced by matrix calculation to decrease the computational cost. The two-dimensional version of this subroutine has the flaw that it is not able to find integration points which are located on a exact zero coordinate. It is therefore recommended to improve this in future work. Furthermore, in the code as provided with this thesis, it is not yet possible to place the turbines in a direction independent of the mesh axes. The feature to do so could be added to the code.

Bibliography

Abdulrahim, A. (2014). *Experimental investigation of the effect of tip-injection on the aerodynamic loads and wake characteristics of a model horizontal axis wind turbine rotor* (Doctoral dissertation). https://doi.org/10.13140/RG.2.1.1793.8405

Afgan, I., Mcnaughton, J., Rolfo, S., Apsley, D., Stallard, T., & Stansby, P. (2013). Turbulent flow and loading on a tidal stream turbine by les and rans. *International Journal of Heat and Fluid Flow, in press*. https://doi.org/10.1016/j.ijheatfluidflow.2013.03.010

Anwar-ul-Haque, F., Yamada, S., & Chaudhry, S. (2007). Assessment of turbulence models for turbulent flow over backward facing step. *Proceedings of the World Congress on Engineering*, *2*, 2–7.

Apsley, D., Stallard, T., & Stansby, P. (2018). Actuator-line cfd modelling of tidal-stream turbines in arrays. *Journal of Ocean Engineering and Marine Energy*, *4*(4), 259–271.

Baratchi, F., Jeans, T., & Gerber, A. (2017). Actuator line simulation of a tidal turbine in straight and yawed flows. *International Journal of Marine Energy*, *19*. https://doi.org/10.1016/j.ijome.2017.08.003

Baratchi, F., Jeans, T., & Gerber, A. (2019). A modified implementation of actuator line method for simulating ducted tidal turbines. *Ocean Engineering*, *193*. https://doi.org/10.1016/j.oceaneng.2019.106586

Betz, A. (1920). Das maximum der theoretisch moglichen ausnutzung des windesdurch windmotoren, zeitschrift fur das gesamte turbinenwesen. *Heft 26*.

Bijlsma, A., Tralli, A., Verbruggen, W., & Haas, P. (2017). Detailed hydrodynamics of the eastern scheldt storm surge barrier: Validation of a cfd approach. *Proc. of the 4th International Symposium of Shallow Flows, 2017*.

Borthwick, A. G. (2016). Marine renewable energy seascape. *Engineering*, *2*(1), 69–78. https://doi.org/https://doi.org/10.1016/J.ENG.2016.01.011

Burton, T., Sharpe, D., Jenkins, N., & Bossanyi, E. (2002). Wind energy handbook. https://doi.org/10.1002/0470846062.ch9

Cheng, L., & Changhong, H. (2019). An actuator line - immersed boundary method for simulation of multiple tidal turbines. *Renewable Energy*, *136*, 473–490. https://doi.org/https://doi.org/10.1016/j.renene.2019.01.019

Chow, F., & Moin, P. (2003). A further study of numerical errors in large-eddy simulations. *Journal of Computational Physics*, *184*(2), 366–380. https://doi.org/https://doi.org/10.1016/S0021-9991(02)00020-7

Churchfield, M., Lee, S., Schmitz, S., & Wang, Z. (2015). Modeling wind turbine tower and nacelle effects within an actuator line model. https://doi.org/10.2514/6.2015-0214

Churchfield, M., Li, Y., & Moriarty, P. (2013). A large-eddy simulation study of wake propagation and power production in an array of tidal-current turbines. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *371*(1985), 20120421. https://doi.org/10.1098/rsta.2012.0421

de Fockert, A., & Bijlsma, A. (2018). Environmental impact of tidal power in the eastern scheldt storm surge barrier. *Deltares Project*, 11200119–000.

Edmunds, M., Williams, A., Masters, I., Banerjee, A., & VanZwieten, J. (2020). A spatially nonlinear generalised actuator disk model for the simulation of horizontal axis wind and tidal turbines. *Energy*, *194*, 116803. https://doi.org/https://doi.org/10.1016/j.energy.2019.116803

Edmunds, M., Williams, A., Masters, I., & Croft, T. (2017). An enhanced disk averaged cfd model for the simulation of horizontal axis tidal turbines. *Renewable Energy*, *101*, 67–81. https://doi.org/https://doi.org/10.1016/j.renene.2016.08.007

Ehrich, S., Schwarz, C., Rahimi, H., Stoevesandt, B., & Peinke, J. (2018). Comparison of the blade element momentum theory with computational fluid dynamics for wind turbine simulations in turbulent inflow. *Applied Sciences*, *8*, 2513. https://doi.org/10.3390/app8122513

Ferreira, C. (2020). Ae4135 rotor/wake aerodynamics. *Delft University of Technology*.

Garrett, C., & Cummins, P. (2005). The power potential of tidal currents in channels. *Proceedings of The Royal Society A: Mathematical, Physical and Engineering Sciences*, *461*, 2563–2572. https://doi.org/10.1098/rspa.2005.1494

Garrett, C., & Cummins, P. (2007). The efficiency of a turbine in a tidal channel. *Journal of Fluid Mechanics*, *588*, 243–251. https://doi.org/10.1017/S0022112007007781

Griffiths, R., & Woollard, M. (1978). Performance of the optimal wind turbine. *Applied Energy*, *4*(4), 261–272. https://doi.org/https://doi.org/10.1016/0306-2619(78)90025-9

Guijt, K. (2018). *Impact of tidal energy extraction in the eastern scheldt storm surge barrier on basin hydrodynamics and morphology* (Master's thesis). Delft University of Technology. http://resolver.tudelft.nl/uuid:d81e9586-a3a9-4000-a4d8-6a6ea22f4022

Hollasch, S. (1994). *Point in tetrahedron*. http://steve.hollasch.net/cgindex/geometry/ptintet.html (accessed: 10.10.2020)

Jehad, D., Hashim, G., Zarzoor, A., & Azwadi, C. (2015). Numerical study of turbulent flow over backward-facing step with different turbulence models. *Journal of Advanced Research Design*, *4*(1), 20–27.

Labeur, R. (2009). *Finite element modelling of transport and non-hydrostatic flow in environmental fluid mechanics* (Doctoral dissertation). Delft University of Technology. http://resolver.tudelft.nl/uuid:7b2d7144-4ea8-4bba-9a05-14ec761b43c3

Lanchester, F. (1915). A contribution to the theory of propulsion and the screw propeller. *Journal of the American Society for Naval Engineers*, *27*(2), 509–510. https://doi.org/https://doi.org/10.1111/j.1559-3584.1915.tb00408.x

Leopold, M., & Scholl, M. (2019). *Monitoring getijdenturbines oosterscheldekering: Jaarrapportage 2018* (tech. rep.). Wageningen Marine Research.

Malki, R., Williams, A., Croft, N., Togneri, M., & Masters, I. (2013). A coupled blade element momentum – computational fluid dynamics model for evaluating tidal stream turbine performance. *Applied Mathematical Modelling*, *37*, 3006–3020. https://doi.org/10.1016/j.apm.2012.07.025

Martinez, L., Leonardi, S., Churchfield, M., & Moriarty, P. (2012). A comparison of actuator disk and actuator line wind turbine models and best practices for their use. *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, 900. https://doi.org/https://doi.org/10.2514/6.2012-900

Martínez, L., Churchfield, M., & Leonardi, S. (2014). Large eddy simulations of the flow past wind turbines: Actuator line and disk modeling: Les of the flow past wind turbines: Actuator line and disk modeling. *Wind Energy*, *18*. https://doi.org/10.1002/we.1747

Masters, I., Malki, R., Williams, A., & Croft, T. (2013). The influence of flow acceleration on tidal stream turbine wake dynamics: A numerical study using a coupled bem–cfd model. *Applied Mathematical Modelling*, *37*(16-17), 7905–7918. https://doi.org/https://doi.org/10.1016/j.apm.2013.06.004

Ministerie van Economische Zaken en Klimaat. (2020). Klimaatplan 2021-2030.

Myers, L., & Bahaj, A. (2009). Near wake properties of horizontal axis marine current turbines. *Proceedings of the 8th European wave and tidal energy conference*, 558–565.

Nakagawa, H., & Nezu, I. (1987). Experimental investigation on turbulent structure of backward-facing step flow in an open channel. *Journal of Hydraulic Research*, *25*(1), 67–88. https://doi.org/10.1080/00221688709499289

Nathan, J. (2018). *Application of actuator surface concept in les simulations of the near wake of wind turbines* (Doctoral dissertation). École de technologie supérieure.

Olczak, A., Stallard, T., Feng, T., & Stansby, P. (2016). Comparison of a rans blade element model for tidal turbine arrays with laboratory scale measurements of wake velocity and rotor thrust. *Journal of Fluids and Structures*, *64*, 87–106. https://doi.org/https://doi.org/10.1016/j.jfluidstructs.2016.04.001

O'Mahoney, T., de Fockert, A., Bijlsma, A., & de Haas, P. (2020). Hydrodynamic impact and power production of tidal turbines in a storm surge barrier. *International Marine Energy Journal*, *3*(3), 127–136. https://doi.org/10.36688/imej.3.127-136

Ouro, P., Ramírez, L., & Harrold, M. (2019). Analysis of array spacing on tidal stream turbine farm performance using large-eddy simulation. *Journal of Fluids and Structures*, *91*, 102732. https://doi.org/https://doi.org/10.1016/j.jfluidstructs.2019.102732

Ouro, P., & Stoesser, T. (2019). Impact of environmental turbulence on the performance and loadings of a tidal stream turbine. *Flow, Turbulence and Combustion*, *102*. https://doi.org/10.1007/s10494-018-9975-6

Sanderse, B., van der Pijl, S., & Koren, B. (2011). Review of computational fluid dynamics for wind turbine wake aerodynamics. *Wind Energy*, *14*(7), 799–819. https://doi.org/10.1002/we.458

Scheijgrond, P. (2015). Potentials of energy from water. *Energising Deltas*. http://www.energisingdeltas.com/wp-content/uploads/Potentials-of-Energy-from-Water.pdf

Schepers, G. (2016). Wake (farm) aerodynamics. *Energy research Centre of the Netherlands*.

Schiereck, G., & Verhagen, H. (2016). *Introduction to bed, bank and shore protection* (Vol. 2). Delft Academic Press / VSSD.

Shen, W. Z., Zhang, J. H., & Sørensen, J. N. (2009). The actuator surface model: A new navier–stokes based model for rotor computations. *Journal of solar energy engineering, 131*(1). https://doi.org/https://doi.org/10.1115/1.3027502

Shives, M., & Crawford, C. (2014). Turbulence modelling for accurate wake prediction in tidal turbine arrays. *5th International Conference on Ocean Energy, 4*.

Sorensen, J., & Shen, W. (2002). Numerical modeling of wind turbine wakes. *J. Fluids Eng.*, *124*(2), 393–399. https://doi.org/https://doi.org/10.1115/1.1471361

Stallard, T., Collings, R., Feng, T., & Whelan, J. (2013). Interactions between tidal turbine wakes: Experimental study of a group of three-bladed rotors. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, *371*(1985), 20120159. https://doi.org/https://doi.org/10.1098/rsta.2012.0159

Stallard, T., Feng, T., & Stansby, P. (2015). Experimental study of the mean wake of a tidal stream rotor in a shallow turbulent flow. *Journal of Fluids and Structures*, *54*, 235–246. https://doi.org/https://doi.org/10.1016/j.jfluidstructs.2014.10.017

Talstra, H. (2017). Numerical flow model finel3d: General intro and application to roompot 6-10. *Svasek Hydraulics*.

Tralli, A., Bijlsma, A., te Velde, W., & de Haas, P. (2015). Cfd study on free-surface influence on tidal turbines in hydraulic structures. *Proceedings of the ASME 2015 34th International Conference on Ocean, Offshore and Arctic Engineering, St. John's, NL, Canada*, *31*. https://doi.org/10.1115/OMAE2015-41187

Uijttewaal, W. (2020). Turbulence in hydraulics. *Delft University of Technology*.

van Bussel, G. (2011). Windmill and wind turbine aerodynamics. *Delft University of Technology*.

Verbeek, M., Labeur, R., & Uijttewaal, W. (2020a). Estimating the stability of a bed protection of a weir-mounted tidal turbine. *International Marine Energy Journal*, *3*(1), 21–24. https://doi.org/https://doi.org/10.36688/imej.3.21-24

Verbeek, M., Labeur, R., & Uijttewaal, W. (2020b). The performance of a weir-mounted tidal turbine: Field observations and theoretical modelling. *Renewable Energy*, *153*. https://doi.org/https://doi.org/10.1016/j.renene.2020.02.005

Verbeek, M., Labeur, R., & Uijttewaal, W. (2021). The performance of a weir-mounted tidal turbine: An experimental investigation. *Renewable Energy*, *168*, 64–75. https://doi.org/https://doi.org/10.1016/j.renene.2020.12.013

Vogel, C., Houlsby, G., & Willden, R. (2016). Effect of free surface deformation on the extractable power of a finite width turbine array. *Renewable Energy*, *88*, 317–324. https://doi.org/10.1016/j.renene.2015.11.050

Whelan, J., Graham, J., & Peiro, J. (2009). A free-surface and blockage correction for tidal turbines. *Journal of Fluid Mechanics*, *624*, 281–291. https://doi.org/10.1017/S0022112009005916

Wimshurst, A. (2018). How do aerofoils generate lift? https://www.youtube.com/watch?v=WQU9AUPHR90

Wimshurst, A., & Willden, R. (2018). Spanwise flow corrections for tidal turbines. *International Marine Energy Journal*, *1*(2 (Nov)), 111–121. https://doi.org/10.36688/imej.1.111-121

Yang, X., & Sotiropoulos, F. (2018). A new class of actuator surface models for wind turbines. *Wind Energy*, *21*(5), 285–302. https://doi.org/https://doi.org/10.1002/we.2162

A

AD model by Betz

In this appendix an extensive derivation of the actuator disc (AD) model by Betz (1920) is given, these derivations come from the Rotor Aerodynamics course of the TU Delft (Ferreira, 2020).

As turbines spin a disc is drawn which is called the AD. When an upstream flow approaches the AD an one-dimensional streamtube can be drawn, considering only the parts that flow through the AD. This one-dimensional streamtube is shown in figure A.1. On the AD in the streamtube an axial force is applied by the flow on the turbine (F_a) and a opposed force is applied by the turbine on the flow. The domain is considered in four sections, the upstream undisturbed area (A_1), the area just upstream of the turbine (A_2), the area just downstream of the turbine (A_3) and the fully developed area far downstream of the turbine (A_4). With areas A_2 and A_3 being of the size of the blade swept area of the turbine or rotor and therefore further mentioned as A_R. Furthermore, pressure and velocity are defined in every section. An extra velocity (u_∞) is stated in the undisturbed region far upstream.

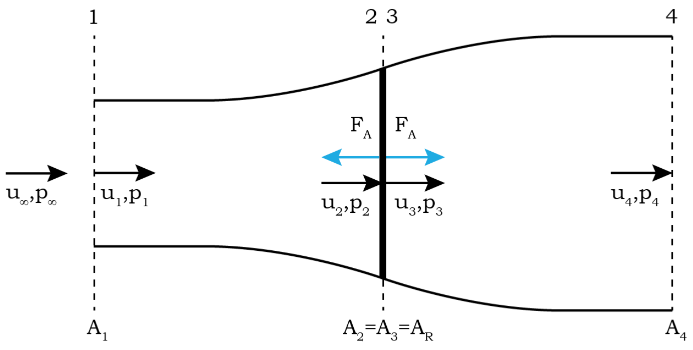


Figure A.1: AD model by Betz, adjusted from (Ferreira, 2020)

Continuity, momentum and energy equations can now be set up. To simplify this process, the flow is assumed steady, inviscid and incompressible. In addition constant internal energy is assumed while the Bernoulli equation is applied as the energy equation. This leads to only two remaining variables, the earlier defined pressure and velocity.

The continuity equation can be defined in four sections, the total streamtube (section 1 to 4), the section upstream of the turbine (section 1 to 2), the section downstream of the turbine (section 3 to 4) and the section across the turbine (section 2 to 3). Their respective continuity equations are given in equations A.1, A.2, A.3 and A.4.

$$u_1 A_1 - u_4 A_4 = 0 \rightarrow u_1 = \frac{u_4 A_4}{A_1} \tag{A.1}$$

$$u_1 A_1 - u_2 A_2 = 0 \rightarrow u_1 = \frac{u_2 A_2}{A_1} \tag{A.2}$$

$$u_3 A_3 - u_4 A_4 = 0 \rightarrow u_3 = \frac{u_4 A_4}{A_3} \tag{A.3}$$

$$u_2 A_2 - u_3 A_3 = 0 \rightarrow u_2 = u_3 \tag{A.4}$$

Momentum equation in x-direction. The momentum equation can again be defined over the complete streamtube. Assuming $p_1 = p_4$ (straightforward if points are infinitely up- and downstream) and the total contribution of the pressure being zero leads to the momentum equation as shown in equation A.5.

$$\rho u_1^2 A_1 - \rho u_4^2 A_4 = -F_a \tag{A.5}$$

The momentum equation at the AD can be defined as shown in equation A.6.

$$\rho u_2^2 A_2 - \rho u_3^2 A_3 = -p_2 A_2 + p_3 A_3 - F_a \tag{A.6}$$

Using $u_2 = u_3$ and $A_2 = A_3 = A_R$ leads to equation A.7.

$$(p_2 + p_3) A_R = -F_a \tag{A.7}$$

The last step is to define the Bernoulli equation, as energy is extracted by the turbine it can't be set across the turbine and only in section up- and downstream. The general Bernoulli equation is given in A.8, the equation applied on the up- and downstream part are given in equations A.9 and A.10 respectively.

$$p + \frac{1}{2}\rho u^2 = constant \tag{A.8}$$

$$p_1 + \frac{1}{2}\rho u_1^2 = p_2 + \frac{1}{2}\rho u_2^2 \tag{A.9}$$

$$p_3 + \frac{1}{2}\rho u_3^2 = p_4 + \frac{1}{2}\rho u_4^2 \tag{A.10}$$

With the continuity, momentum and energy equations defined, the equation for force and power can be set. In first the axial velocity at the AD must be known. This velocity can be taken as the average from the up- and downstream velocity as is shown in equation A.11.

$$u_a = \frac{1}{2}(u_1 + u_4) \tag{A.11}$$

The thrust at the AD can now be defined by using equations A.6 and A.11. After a few algebraic steps this leads to equation A.12.

$$F_a = 2\rho(u_1 - u_a) u_a A_R \tag{A.12}$$

The power at the AD can now simply be defined as the thrust at the AD multiplied by the velocity at the AD as is shown in equation A.13.

$$P = F_a u_a = 2\rho(u_1 - u_a) u_a^2 A_R \tag{A.13}$$

The power is now defined as a function of the density of the flow, the upstream velocity, the velocity at the turbine and the area of the turbine.

The next step is to research the performance of the AD. From up- to downstream of the turbine, the streamtube expands, the flow area increases. Continuity states that the flow speed decreases at the same rate. Pressure increases towards the turbine (decreasing flow velocity), at the actuator due to the force balance the pressure decreases in a jump, afterwards the pressure increases again due to the still decreasing velocity. Introducing the induction factor a, a determines the relation between the velocity at the AD and the unpertubed velocity.

$$u_a = (1-a)u_\infty \tag{A.14}$$

Rewriting equations A.11, A.12 and A.13 in terms of a leads to equations A.15, A.16 and A.17 respectively.

$$\frac{u_a}{u_\infty} = (1-a) \tag{A.15}$$

$$\frac{F_a}{\frac{1}{2}\rho u_\infty^2 A_R} = 4a(1-a) = C_T \tag{A.16}$$

$$\frac{F_a}{\frac{1}{2}\rho u_\infty^3 A_R} = 4a(1-a)^2 = C_p \tag{A.17}$$

With C_T and C_P being the power and thrust coefficients respectively. The Betz limit can easily be derived by taking the derivative of the power coefficient over a.

$$\frac{dC_p}{da} = 4(1-a)(1-3a) = 0 \rightarrow a = \frac{1}{3} \tag{A.18}$$

$$C_{p,max} = \frac{16}{27} = 0.593 \tag{A.19}$$

B
FinLab numerical model

The computational fluid dynamics software package FinLab is used to execute the simulations in this thesis. The model is developed by Labeur (2009) at the Delft University of Technology. FinLab solves the non-hydrostatic three-dimensional incompressible Navies-Stokes equations in an unstructured mesh with an optional moving free surface. In other words, the vertical and horizontal flow are treated equally in a mesh that can be locally refined. This makes the model suitable for flow around complex three-dimensional geometries. The equations in this chapter are based on the Turbulence in Hydraulics lecture notes (Uijttewaal, 2020).

B.1. Solution algorithm

Several methods exist to solve a discretized system. Common methods are the finite difference method (FDM), finite volume method (FVM) and finite element method (FEM). FDM replaces the derivative with a difference, it solves the variables at the mesh nodes only. FVM solves for the average (volume) of the variables in the mesh elements, while not solving the variables at the mesh nodes. While FDM and FVM only at one position, FEM approximates the function with a polynomial. The benefit of FEM compared to FDM and FVM lies in the possibilities to reach for higher-order solutions in unstructured meshes. FDM and FVM can only reach higher-order solutions when higher-order stencils are used, which are hard to apply in unstructured meshes. FinLab is designed to approximate three-dimensional flow in complex geometries and therefore uses an unstructured mesh with a FEM algorithm.

FEM uses classes of functions to approximate the exact solution. To choose the best approximation in an element, the FEM model needs constraints and a definition of the best approximation. A constraint can, e.g., be such that the solution should be continuous (variables of two elements should be equal on their joint node), or only a certain class of function could be used for the approximation. A piece-wise linear function would, e.g., lead to the most simple approximation. Fewer constraints could lead to a better approximation of the exact solution, but can also drastically increase the computational cost as more degrees of freedom arise. Two common approaches in FEM are the continuous Galerkin (CG) and discontinuous Galerkin (DG). In the Galerkin method, the integral of the function that is solved for is multiplied by a weighting function (θ) and forced to zero. This weighting function is of the same form as the approximated function. When comparing CG and DG, DG has better stabilizing properties than CG. However, DG leads to enormous systems due to the extra degrees of freedom, making it computationally expensive to solve for implicit problems.

In FinLab, the benefits of both the CG and DG are combined into one method, the Galerkin Interface Stabilization (GIS) method. In GIS, the size of the element matrices to be solved are equal of the size in the CG method. Within each element DG is applied as it is more stable. However, as applying DG leads to too many degrees of freedom, the values on the adjacent element boundaries are averaged to create one single value. This leads to a system with a number of degrees of freedom equal to CG, making it feasible to implicitly solve.

B.2. Discretization

FEM can numerically solve discretized systems. Therefore, to model a real-life flow problem numerically, the problem must be discretized in space and time.

B.2.1. Mesh partitioning

To discretize the flow problem in space, the geometry of the problem must be divided into elements. In FinLab, a three-dimensional FEM mesh constructed from unstructured tetrahedrons can be used. The unstructured meshes used in this thesis are made in 2DV or 2DH in Sepran. Thereafter, in FinLab, these 2DV or 2DH meshes are extended in the remaining axis to form a three-dimensional mesh.

B.2.2. Time-stepping

In order to solve for a flow solution which evolves over time or generate a steady state solution with multiple iterations, the model must be discretized into discrete time levels. In FinLab, the time interval I can be parted in N sub-intervals: I = (t_0, t_1, ... , t_{n-1}, t_n) with time interval ∂ t. The solution at time step t_n is obtained from the previous time step t_{n-1}. The θ-scheme and the Fractional Step scheme are the implemented time-stepping schemes in FinLab. In the θ-method, θ must be defined as a value between zero and one. For $\theta = 0$ and $\theta = 1$ the scheme is first-order accurate. For $\theta = 0.5$ is second-order accurate but weakly stable. The θ-schemes with $\theta = 0$, $\theta = 0.5$ and $\theta = 1$ are also referred to as backward Euler, Crank-Nicolson and forward Euler respectively. The fractional step method is both second-order accurate and strongly stable. In computational modelling, accuracy is defined as the order of the truncation error of the scheme. Stability states that a solution is bounded and therefore will not reach either infinity or minus infinity. Together they secure convergence, the numerical solution approaches the exact solution. The fractional time step splits every time step in three sub time steps: $t_n \rightarrow t_{n+\alpha} \rightarrow t_{n+1-\alpha} \rightarrow t_{n+1}$. In all steps a one step θ-scheme is applied, in the first and last step with variable θ in the second step with variable $1-\theta$.

The backward Euler scheme has firm damping characteristics which lead to a faster convergence to a steady-state than in the fractional-step method. Therefore, in the steady AD and AD-BEM simulations in this thesis, the backward Euler method is used. The Crank-Nicolson scheme does not have any amplitude damping mechanism, meaning that any small perturbation will not be smoothed. The fractional time step scheme leads to the most detailed solutions of time-dependent problems and is therefore applied to the AL-BEM simulations.

B.3. Governing equations

Conservation laws govern the flow of water. FinLab uses the conservation of mass and momentum in the form of the incompressible Navier-Stokes equations to describe the flow. Incompressibility assumes a constant density of the fluid, which simplifies the equations. Conservation of mass states that in a control volume, the rate of change of mass is equal to the net flux of mass through its faces. In vector form, this can be written as equation B.1.

$$\frac{\partial \rho}{\partial t} + \nabla_i \rho u_i = 0 \tag{B.1}$$

FinLab solves for incompressible flow, as commonly the density variations in bodies of water are negligible with respect to the absolute density of the water (Boussinesq-approximation). Therefore, the continuity equation can be written as stated in equation B.2.

$$\nabla_i u_i = 0 \tag{B.2}$$

According to Newton's second law, the rate of change of momentum on a control volume equals the sum of the forces applied to the control volume. In a three-dimensional geometry, this leads to three components of the momentum equation, which in vector form are shown in equation B.3 (with $i = 1,2,3$) =. The source term (S) can include forces such as gravity, Coriolis and the turbine forces. The three equations are known as the Navier-Stokes equation and form together with the continuity equation a complete set to describe the motion of a flow.

$$\frac{\partial}{\partial t}(\rho u_j) + \nabla_i(\rho u_i u_j) = -\nabla_j p + \eta \nabla_i^2 u_j + S \tag{B.3}$$

B.4. Turbulence modelling

With the Navier-Stokes equations known since 1845, it is in principle possible to solve all flow problems once the initial and boundary conditions are known. To do so, the smallest and the largest length scales of the flow should be represented, which means that the computational domain should be split into elements of the order of the Kolmogorov length scale (the smallest turbulence scale). A very simplified estimation of the Kolmogorov length scale can be made by using the large scale dissipation rate. Applying equation B.4 on the flume experiments leads to an estimate of the length scale of about 10 μm, which is of course completely unfeasible to apply in a mesh.

$$L_k = \left(\frac{\nu^3}{\epsilon}\right)^{\frac{1}{4}} \tag{B.4}$$

In order to make realistic simulations with feasible computational times, Reynolds Averaged Navier-Stokes (RANS) equations, as shown in equation B.6, can be used in FinLab. The RANS equations are an time-averaged version of the Navier Stokes equations. In the equations the velocity is decomposed into an average velocity and a fluctuating velocity, see equation B.5. By time averaging, the time-varying component disappears in most terms as $\overline{u'} = 0$. However, as can be seen in the third term of equation B.6, a stress term (Reynolds stresses) on the length scale of the small fluctuations arises. Due to this term, the RANS equations are not a closed set of equations and additional equations should be introduced to represent the smallest scale turbulent motions.

$$u_i = \bar{u}_i + u'_i \tag{B.5}$$

$$\frac{\partial}{\partial t}(\bar{u}_j) + \nabla_i(\bar{u}_i\bar{u}_j) + \nabla_i(\overline{u'_i u'_j}) = -\frac{1}{\rho}\nabla_j p + \nu\nabla_i^2\bar{u}_j + S \tag{B.6}$$

An exact representation for the Reynolds stresses is still to be found. Therefore, until the equations are solved, turbulence models can serve as approximations. Most turbulence models employ a turbulent viscosity (μ_t) to approximate the Reynolds' stresses. In FinLab, the turbulent viscosity can be determined by either a constant eddy viscosity method, the Bakmhetev mixing length model or the k-ϵ model.

B.4.1. Constant eddy viscosity

The constant eddy viscosity model is the simplest turbulence model. It states that the eddy viscosity is equal to a length scale multiplied by a time scale, which is physically sound when these are the length and timescales of the large energy transporting eddies. In large civil engineering domains with enormous computational meshes, the constant eddy viscosity can be a good option due to its simplicity and workability with depth-averaged models. However in this thesis, various turbulence length scales play a role, making this model inaccurate.

B.4.2. Bakmhetev mixing length model

In mixing length models, the turbulent viscosity is again represented by a length scale multiplied by a velocity scale. The velocity scale is represented by the product of the length scale and the velocity gradient over the vertical, see equation B.7. The length scale is proportional to the distance from the wall, which means that the turbulent viscosity will be too, see equation B.8. Further away from the wall larger eddies can form leading to a higher viscosity, while at the free surface and the bed the mixing length and the turbulent viscosity are zero. In the wall region, this leads to the characteristic logarithmic profile, while in the outer region the mixing length can be taken as a constant proportional to the width of the boundary layer.

$$u_t = l_m\left|\frac{\partial\bar{u}}{\partial y}\right| \tag{B.7}$$

$$\mu_t = l_m^2\left|\frac{\partial\bar{u}}{\partial y}\right| \tag{B.8}$$

Mixing length models are widely used due to their simplicity. However, as stated before, various turbulence length scales play a role in this thesis, making this model also inaccurate.

B.4.3. k-ϵ model

To better represent the length and velocity scales, transport equations can be set up. To find a more physically relevant velocity scale, it can be represented by the square root of the turbulent kinetic energy. A physically more relevant length scale can be obtained by setting up a transport equation for the dissipation rate. The dissipation is seen as the amount of energy dissipated per unit of time, subsequently, equation B.9 follows for the turbulent viscosity. The turbulent kinetic energy itself can be estimated by setting up a transport equation (see equation B.10) consisting of a diffusion, production and dissipation term. The transport equation of the dissipation can be seen in equation B.11, it again consists of the diffusion, production and dissipation terms. A dissipation term of the dissipation rate, of course, is physically not sound. Still, the k-ϵ model is proven to be a useful turbulence model and is still widely used.

$$\mu_t = C_\mu \frac{k^2}{\epsilon} \tag{B.9}$$

$$\frac{\partial k}{\partial t} = \frac{C_\mu}{\sigma_k}\nabla_i\left(\frac{k^2}{\epsilon}\nabla_i k\right) + C_\mu\frac{k^2}{\epsilon}(\nabla_i\bar{u}_j(\nabla_i\bar{u}_j + \nabla_j\bar{u}_i)) - \epsilon \tag{B.10}$$

$$\frac{\partial \epsilon}{\partial t} = \frac{C_\mu}{\sigma_\epsilon}\nabla_i\left(\frac{k^2}{\epsilon}\nabla_i k\right) + C_1 k^2(\nabla_i\bar{u}_j(\nabla_i\bar{u}_j + \nabla_j\bar{u}_i)) - C_{d\epsilon}\frac{\epsilon^2}{k} \tag{B.11}$$

Table B.1: Constants in the k-ϵ turbulence model in FinLab (Uijttewaal, 2020)

c_1	$c_{p\epsilon}$	$c_{d\epsilon}$	σ_k	σ_ϵ
0.09	0.13	1.92	1.0	1.3

In this thesis, the k-ϵ is used in all the AD simulations. The time-averaging character with the solved level of detail in both makes a good combination.

B.4.4. LES

The RANS equations are introduced to average out the smallest length scales in time and make a computational feasible numerical model. Another possibility is to average the Navier-Stokes equations in space, which is done in LES. In LES the turbulence is resolved on all scales as no Reynolds-decomposition is applied. The restricting variable for the resolved turbulence now becomes the mesh element size. All turbulent structures larger than the element size are solved. The turbulent structure smaller than the element size are again represented by a turbulence model. This turbulence model is very comparable with the mixing-length model, but then on the scale of the element size.

The turbulent macro scales show a large anisotropy and contain the greater part of the turbulent energy. These large structures are broken down to the smallest scales by the energy cascade and get more isotropic and less influenced by the geometry. Resolving the larger anisotropic eddies on the mesh often leads to a smaller error of the model compared to RANS models. The filtered Navier-Stokes equation as used in LES is shown in equation B.12.

$$\frac{\partial}{\partial t}(\bar{u}_j) + \nabla_i(\bar{u}_i\bar{u}_j) = -\frac{1}{\rho}\nabla_j p + \nu\nabla_i^2\bar{u}_i + \nabla_i\tau_{ij} \tag{B.12}$$

As can be seen in the last term of equation B.12, a sub-mesh stress is to model the small scale turbulence. The sub-mesh stress is determined by equation B.13.

$$\tau_{ij} = -\mu_t\left(\frac{\partial\bar{u}_i}{\partial x_j} + \frac{\partial\bar{u}_j}{\partial x_i}\right) \tag{B.13}$$

The isotropy of the filtered out turbulent stresses allows it to define a sub-mesh viscosity based on the length scale of the mesh elements (Δ_e) and the sub-mesh deformation (S_{ij}). This sub-mesh viscosity is determined according to equation B.14 with the sub-mesh deformation give in equation B.15. The Smagorinsky constant (C_s), which determines the length scale, is set to its standard value in FinLab of 0.125.

$$\mu_t = (C_s\Delta_e)^2\sqrt{2S_{ij}S_{ij}} \tag{B.14}$$

$$S_{ij} = \frac{1}{2}\left(\frac{\partial \overline{u}_i}{\partial x_j} + \frac{\partial \overline{u}_j}{\partial x_i}\right) \tag{B.15}$$

In this thesis, the LES model is applied to the AL-BEM simulations. The LES model is able to solve for the transient near wake features as caused by the AL-BEM method. In comparison to the AD-BEM simulations the mesh is adjusted to allow for LES simulations. In the AD-BEM simulations the mesh is strongly refined in the vicinity of the turbine, with large elements in the far wake. In the AL-BEM fine mesh throughout the complete domain is used, which makes it possible to use the LES model.

B.5. Initial conditions

Initial conditions must be given to the model to provide a starting point for the calculations. An initial condition must be given for each element and must be in accordance with the Navier-Stokes equations. In this thesis, the initial condition of flow velocity and pressure are simply set to zero and the water level respectively.

B.6. Boundary conditions

The boundary conditions define the flow conditions at the boundaries of the mesh. The boundary conditions initiate the flow in the domain. To prevent instabilities in the flow, the boundary conditions linearly increase from zero to the input value during a given damping time. This same damping time is added to the turbine code to prevent instabilities in the flow in the vicinity of the turbine.

Open and closed boundaries can be formed. Sidewalls of a flume can be stated as closed boundaries while the in and outlet of the flume can be stated as open boundaries. The free-surface is treated as an open and closed boundary in this thesis. In all simulations, except for the Eastern Scheldt simulations, the free surface is represented by a non-moving symmetry boundary. The symmetry boundary can be seen as a closed boundary without friction. In the Eastern Scheldt simulations, a moving free-surface boundary is used. All the open upstream boundaries are modeled by applying a discharge or velocity boundary to initiate the flow. The open downstream boundaries are modeled by applying a fixed water level.

The sides and bed of the flumes in the flume experiments and the bed and geometry of the barrier in the Eastern Scheldt case are treated as closed boundaries. Flow through the closed boundaries is not allowed, therefore, the flow normal to these boundaries is set to zero by the boundary. In addition, closed boundaries apply friction on the flow, according to a set Nikuradse bed roughness (k_s). This roughness pushes the flow to form a logarithmic profile according to equation B.16. In the simulations of the flume experiments, a Nikuardse roughness of $1 \cdot 10^{-4}$ m is applied. In the Eastern Scheldt field case simulations, a much larger roughness of 0.1 m is applied.

$$u(z) = \overline{u}\frac{\ln \frac{z}{z_0}}{\ln \frac{0.4h}{z_0}} \tag{B.16}$$

$$z_0 = k_s/30 \tag{B.17}$$

C
FinLab code

The computational fluid dynamics software FinLab is coded in Fortran. The code consists of various modules which are controlled and connected by the main program. The code uses MPI to be able to split the mesh and run parallel on a selected number of processors on one CPU node. In this thesis, a turbine module is added to the FinLab package. The added code is elaborated in this appendix. For each subroutine, an elaboration of its function is given here. The function of the individual lines of code is mainly explained in the comments of the code itself.

C.1. Main module

In the main file of FinLab the user can set all parameters of the flow. During this thesis, the parameters below * TURBINE * were added to this file. The main turbine parameters should always be given, including the turbine method. Note that 'turbine_method = 1' equals the uniform AD method, 'turbine_method = 2' in combination with 'integration_method = 1' equals the AD-BEM method and 'turbine_method = 2' in combination with 'integration_method = 2' equals the AL-BEM method. The AD-BEM and AL-BEM methods both only work in a three-dimensional mesh. The time-step is automatically set if the AL-BEM method is selected. The turbine parameters of the selected method should be given. The turbine is finally added to the mesh by calling the subroutine 'add_turbine', together with the coordinates of the centre of the turbine. Multiple turbines can be added by calling the 'add_turbine' subroutine multiple times. Thereafter, the 'init_turbine' subroutine must be called to initiate various turbine parameters. During the time iterations, the 'add_turbine_force' subroutine is called to add the turbine forces to the source term of the momentum equation.

```


! * TURBINE *

! main turbine parameters
R = 0.2       ! radius of turbine(s)                [m]
Nt = 1000       ! turbine integration points per turbine    [-]
rho_w = 1000.0      ! fluid density                [kg/m3]
turbine_method = 2  ! turbine integration method: 1 = uniform actuator disc, 2
    = blade element method (AD or AL)

! uniform actuator disc parameters
F_thrust = 35.5590  ! total thrust force per turbine         [N]

! blade element method parameters
integration_method = 1  ! integration method: 1 = actuator disc, 2 = actuator
    line
nacelle_method = 1  ! nacelle method: 0 = no nacelle, 1 = flow nacelle centre
    to zero
```

```
tip_correction = 0  ! tip-correction method: 0 = no correction, 1 = Prandtl, 2
    = Edmunds et al.
turbine_scale = 13.17   ! scale factor compared to BEM files         [-]
TSR = 4.0        ! rotational speed                [rad/s]
N_blades = 2         ! number of blades per turbine      [-]
lift_drag_file = 'T2LiftDrag.csv'
geo_file = 'T2Geometry.csv'

! actuator line parameters
AL_nlines = 40       ! number of actuator lines
AL_npoints = 20 ! number of sampling points per actuator line

! add a turbine: add_turbine(([x_axis,y_axis,z_axis])
call add_turbine([4.01,0.385,-0.35])
! initiate turbine parameters
call init_turbines
```

C.2. Turbine module

C.2.1. Parameters

The top of the turbine module introduces all used parameters within the code. These are arranged according to the subroutine in which they are used. The parameters are, as much as possible, in correspondence with the parameters used in this thesis and are therefore not shown here. Within the parameters, a turbine type is introduced to be able to add multiple turbines to a flow domain. This type saves the geometrical characteristics of all the turbine integration (and AL) points of a turbine in a turbine specific matrix.

C.2.2. Add turbine subroutine

The 'add_turbine' subroutine adds a turbine to the flow domain each time it is called in the main file. The subroutine has the defined coordinates of the centre point of the turbine as input. Each time the subroutine is called in the main file, one is added to the number of turbines. To be able to choose the right configuration of the turbine integration points, the subroutine checks if the mesh is two- or three-dimensional. Thereafter, it allocates lists for the turbine and integration point specific parameters which are used throughout the code.

```

contains
!!! TURBINE SUBROUTINES BEFORE START ITERATIONS !!!

! subroutine to introduce integration points of turbine
subroutine add_turbine(t_axis)
  real, intent(in) :: t_axis(:)
  integer :: cel

  ! add new turbine
  nturb = nturb + 1
  j = nturb

  ! determine if 2D or 3D
  if (size(x(1,:))==2) then
    ax = 1
    ay = 2
    ndim = 2
  else if (size(x(1,:))==3) then
    ax = 1
    ay = 2
    az = 3
    ndim = 3
  end if

  ! allocate turbine specific list
  allocate(turb(j)%element_list(Nt))
  allocate(turb(j)%dThrust(Nt),turb(j)%dTorque(Nt),turb(j)%dPower(Nt))
  allocate(turb(j)%xt(Nt,ndim),turb(j)%rad(Nt),turb(j)%theta(Nt))
```

The BEM files, containing the blade geometry and lift and drag coefficients, are read before the turbine integration points are introduced. This is executed here as the radius of the nacelle needs to be known before introducing the AL sampling points. Introducing the turbine integration point is done by either introducing a straight line of turbine integration points in a 2DV case or position the turbine integration points according to a Fibonacci circle in the three-dimensional case. The AL-BEM sampling points are introduced subsequently if the AL-BEM method is selected.

```

  ! read BEM files
  if (turbine_method==2.and.j==1) call read_bem_files

  ! introduce turbine integration points on a straight line for a 2D situation
  if (ndim==2) then
    do n = 1, Nt
      dy = R*2/Nt
      turb(j)%xt(n,1) = t_axis(1)
      turb(j)%xt(n,2) = t_axis(2) + R - dy*n
      turb(j)%rad(n) = abs(turb(j)%xt(n,2)-t_axis(2))
    end do

    if (turbine_method==2) then
      if (ROOT) write(*,*) 'BEM methods not available on a 2D mesh'
      stop
    end if

  ! introduce turbine integation points in a fibonacci circle for a 3D
   situation
  else if (ndim==3) then
    phi = (sqrt(5.0)-1.0)/2.0
    ! introduce fibonacci points
    do n = 1,Nt
      turb(j)%rad(n) = R*sqrt(REAL(n))/sqrt(REAL(Nt))
      turb(j)%theta(n) = n*phi*2*PI
      turb(j)%xt(n,1) = t_axis(1)
      turb(j)%xt(n,2) = turb(j)%rad(n)*cos(turb(j)%theta(n)) + t_axis(2)
      turb(j)%xt(n,3) = turb(j)%rad(n)*sin(turb(j)%theta(n)) + t_axis(3)
    end do
    ! introduce actuator line points
    if (turbine_method==2.and.integration_method==2) then
      allocate(turb(j)%AL_points(AL_nlines,AL_npoints,ndim),turb(j)%AL_velocity
   (N_blades,AL_npoints,2),AL_rad(AL_npoints))
      allocate(AL_theta(AL_nlines),AL_activated(N_blades),turb(j)%
   AL_velocity_sent(N_blades,AL_npoints,2))
      do n = 1,AL_nlines
        AL_theta(n) = 2*PI/AL_nlines*n
        do k = 1,AL_npoints
          AL_rad(k) = (R_nac+(R-R_nac)/AL_npoints*k)
          turb(j)%AL_points(n,k,1) = t_axis(1)
          turb(j)%AL_points(n,k,2) =  cos(AL_theta(n))*(R_nac+(R-R_nac)/
   AL_npoints*k) + t_axis(2)
          turb(j)%AL_points(n,k,3) =  sin(AL_theta(n))*(R_nac+(R-R_nac)/
   AL_npoints*k) + t_axis(3)
        end do
      end do
    end if
  end if

  if (turbine_method==2.and.nacelle_method==1) turb(j)%C_nac = 1.0


end subroutine add_turbine
```

C.2.3. Initiate turbines subroutine

The 'initiate_turbines' subroutine is used to find corresponding mesh elements for all turbine integration points of all turbines and to introduce several parameters. To find the elements, the subroutine loops over all turbines while activating the 'turbine_elements' subroutine. This is the point where the turbine module writes the number of found integration points per processor to the Linux terminal. It should be checked if the total number of points corresponds with the input number. When the AL-BEM method is selected, the same process is executed for the AL sampling points by activating the 'AL_elements' subroutine.

```


subroutine init_turbines

  ! allocate non turbine specific lists
  allocate(matx(ndim+1,ndim),vi(ndim+1),det(ndim+2))
  allocate(matA(ndim+1,ndim+1),matB(ndim+1,ndim+1),matC(ndim+1,ndim+1),matD(
    ndim+1,ndim+1),matE(ndim+1,ndim+1))
  allocate(nx(ndim), ny(ndim),nz(ndim), ntan(ndim),pt(ndim),ui(ndim))


  ! find elements in which turbine integration points are located
  do j=1,nturb
    call turbine_elements(j)
  end do
  write(*,*) count_Nt, 'turbine integration points found in partition ', PROC

  ! find element in which actuator line points are located
  if (turbine_method==2.and.integration_method==2) then
    count_Nt = 0
    do j=1,nturb
      call AL_elements(j)
    end do
    write(*,*) count_Nt, 'actuator line points found in partition ', PROC
  end if
```

When the elements have been found, the 'initiate_turbines' subroutine introduces several method-specific parameters. Normal vectors and an axis number for each axis (x,y,z) are introduced first for either the two- or the three-dimensional case. These can be adjusted if the axes of a mesh file differ from standard. Furthermore, the output files per processor are created and if the uniform AD force method is selected, the partial force is introduced. If one of the BEM methods is selected, the size represented by one integration point is determined.

```


  ! introduce normal vectors
  if (ndim==2) then
    nx = [1.,0.]
    ny = [0.,1.]
  else if (ndim==3) then
    nx = [1.,0.,0.]
    ny = [0.,1.,0.]
```

```fortran
    nz = [0.,0.,1.]
  end if

  ! iniate uniform disc if selected
  if (turbine_method==1) dF = F_thrust/Nt/rho_w/(ndim+1)

  ! initiate BEM if selected
  if (turbine_method==2) then
    ! determine area represented by one integration point
    A_i = R**2*pi/Nt
    ! create output files
    if (PROC < 10) write (x1,'(I2)') PROC
    if (PROC >= 10) write (x1, '(I2)') PROC
    x1 = adjustl(x1)
    bem_results='bem_results'//trim(x1)//'.csv'
    bem_radial='bem_radial_results'//trim(x1)//'.csv'
    open(unit=13,file=bem_results)
      write(13,'(a, 8(",", a))') 'iteration', 'turbine', 'T_total', 'T_blades',
   'T_nacelle', 'torque', 'power', 'omega', 'u_ref'
    close(13)
  end if

end subroutine init_turbines
```

C.2.4. Add turbine force subroutine

In each time step, the main module calls the subroutine 'add_turbine_force'. This subroutine controls the complete turbine module during the time iterations. The module starts with a check if any MPI communications need to be executed. If so, the 'MPI_turbine' subroutine is activated. Thereafter, not to sum up the new source terms to the source terms of the previous iteration, all source terms are set equal to zero. Thereafter, the subroutine starts looping over all the turbines in the domain. It checks if any of the turbine integration points of the selected turbine are located in the mesh partition, if not, it skips the remaining part and checks it again for the next turbine. At last, before the force application part starts, it is checked whether any points (integration or AL sampling) have changed elements. A point can change element if a free-surface with a deforming mesh is applied, therefore, the subroutine 'free_surface_check' is only activated if this condition is met.

```fortran


!!! TURBINE SUBROUTINE DURING ITERATIONS

! subroutine to execute turbine module during the iterations
subroutine add_turbine_force

  ! check if one of the conditions which needs MPI communication is met
  if (nacelle_method==1.or.move==1.or.(turbine_method==2.and.integration_method
   ==2)) call MPI_turbine

  ! set all source terms to zero at the start of the new iteration
  fe(:,:,:) = NUL

  !!! Start loop over all turbines !!!
  do j = 1,nturb

    ! skip the remaining of the subroutine if no turbine integration points are
     located in the partition
    if (any(turb(j)%element_list>0)) then

      ! execute the free-surface check if a free surface is applied
      if (move/=0) call free_surface_check(j)
```

In the following block of code, if one of the BEM methods is selected, the reference velocity is determined according to the settings in the main file. In the BEM methods, the reference velocity is used to determine the rotational speed (ω) and to apply the nacelle force. The velocity is based on the upstream velocity boundary. Both the rotational speed and nacelle could be easily adjusted to be independent of this reference velocity. However, in this thesis, it was the most logical choice to use this velocity. Furthermore, the BEM output lists are set to zero and the total angle of rotation (λ) is determined if the AL-BEM method is selected.

```

      !!! Force application part !!!
      ! set BEM parameters if method is selected
      if (turbine_method==2) then
        ! set output lists to zero
        turb(j)%T_nac = 0
        turb(j)%dThrust = 0
        turb(j)%dTorque = 0
        turb(j)%dPower = 0

        ! determine TSR
        u_ref = Fu(1)
        omega = TSR*u_ref/R

        ! Calculate total angle of rotation lambda if AL-BEM is applied
        if (integration_method == 2) then
          turb(j)%lambda = turb(j)%lambda + 2*PI/AL_nlines
        end if
      end if
```

The next step is to loop over all the turbine integration points which are positioned within the mesh partition. For each integration point, the force module is selected according to the settings in the main file and the corresponding subroutine is activated. The uniform disc method only takes the element number as input, the BEM modules also take the turbine and integration point number as input. Note that, for the BEM methods, the difference between blade integration points and nacelle integration points is made here.

```


      ! loop over the turbine integration points
      do n = 1,Nt
        i = turb(j)%element_list(n)
        if (i>nboun.and.i<nelem) then
          ! call subroutine of selected method
          if (turbine_method==1) then
            call uniform_disc(i)
          else if (turbine_method==2) then
            if (turb(j)%rad(n)*turbine_scale<r_list(1)) then
              if (nacelle_method == 1) call nacelle(j,i,n)
            else
              call BEM(j,i,n)
            end if
          end if
        end if
      end do
      !!! End force application part !!!
```

At the end of the subroutine, the output is written to the processor-specific output files. In the first file, the output per turbine integration point is written. Figures 4.3 and 4.4 are examples of figures which can be made with this output file. In the second output file, the bulk parameters of the turbines are saved. All the thrust and power values from this thesis are coming from this file. The file gives a good overview if the flow model is converged to a steady solution as the bulk parameters are saved each output time-step.

```


   !!! Output part !!!
   ! write output files
   if (mod(itijd,nwrit)==0.and.turbine_method==2) then
    ! radial output file
    open(unit=12,file=bem_radial,status='replace',position='append',action='write')
    write(12,'(a, 3(",", a))') 'r', 'F_a', 'F_t', 'u_a'
     do n = 1,Nt
      if (turb(j)%element_list(n)>0) then
       call get_velocity(turb(j)%element_list(n))
       u_a = dot_product(ui,nx)
       if (turbine_method==1) write(12,'(f16.8, 3(",", f16.8))') turb(j)%rad(n),dF,0.0,u_a
       if (turbine_method==2.and.turb(j)%rad(n)>R_nac) then
        write(12,'(f16.8, 3(",", f16.8))') turb(j)%rad(n),turb(j)%dThrust(n),turb(j)%dTorque(n),u_a
       end if
      end if
     end do
    close(12)
   end if

   ! write BEM output files
   if (mod(itijd,nwrit)==0.and.turbine_method==2) then
    ! iterations output file
    T_t = -sum(fe(:,:,ax))*rho_w
    T_b = sum(turb(j)%dThrust)
    open(unit=13,file=bem_results,status="old", position="append", action="write")
     write(13,'(I5,"," I2, 5(",", f16.8))')  itijd,j,T_t,T_b,turb(j)%T_nac,sum(turb(j)%dTorque),sum(turb(j)%dPower)
    close(13)
   end if
   !!! End output part !!!

  end if
 end do
 !!! End loop over all turbines !!!
end subroutine add_turbine_force
```

C.2.5. Find actuator disc integration points subroutine

To apply the force of the turbines on the right elements the 'turbine_elements' is activated for all added turbines in the 'init_turbine' subroutine. The subroutine consists of a nested loop. It loops over all the mesh elements in a partition with a loop over all the turbine integration point of one turbine. Within the nested loop the subroutine 'element_search' is activated. This subroutine gives the corresponding element number of a turbine integration point when it is found. The 'element_search' subroutine is set up as a separate subroutine as it is called in multiple occasions within the code. The parameter 'inelem=0' denotes that a point is not found yet, the parameter 'findAL=0' denotes that a turbine integration point is to be found.

```

!!! FIND SUBROUTINES !!!
! subroutines to find turbine integration points and AL points in mesh and get their corresponding velocity
```

```fortran

! subroutine to find elements in which the turbine integration points are
    located
subroutine turbine_elements(j)
  integer :: j
  do n = 1,Nt
    inelem = 0
    ! note that turbine point need to be found
    findAL = 0
    pt = turb(j)%xt(n,:)
    find: do i = 1+nboun,nelem
      call element_search(pt,i)
      if (inelem==1) exit find
    end do find
  end do
end subroutine turbine_elements
```

C.2.6. Find actuator line sampling points subroutine

The 'AL_elements' searches for all the actuator line sampling points. The subroutine is very similar to the 'turbine_elements' subroutine and is also activated in the 'init_turbine' subroutine. The subroutine uses an extra loop to first loop over all the actuator lines and thereafter loop over all the points per line. Parameter 'findAL=1', denotes here that an AL sampling point is to be found.

```fortran

! subroutine to find elements in which the actuator lines points are located
subroutine AL_elements(j)
  integer :: j
  allocate(turb(j)%AL_element_list(AL_nlines,AL_npoints))
  turb(j)%AL_element_list(:,:)=0
  do n = 1, AL_nlines
    do m = 1, AL_npoints
      inelem = 0
      ! note that AL point need to be found
      findAL = 1
      pt = turb(j)%AL_points(n,m,:)
      find: do i = 1+nboun,nelem
        call element_search(pt,i)
        if (inelem==1) exit find
      end do find
    end do
  end do
end subroutine AL_elements
```

C.2.7. Find points in mesh subroutine

In the subroutine 'element_search' the theory to find if a point is located within either a triangle (2DV) or tetrahedron (3D) as explained in section 3.1 is executed. When the mesh is three-dimensional, an extra matrix is added. When the turbine integration point is inside the element, the element number is added to the integration element list or the actuator line element list of the specific turbine. The corresponding list is selected by the parameter findAL.

```fortran

! Subroutine to check in which element, point xi is located
subroutine element_search(xi,i)
  integer :: i
  real(kind=DP) :: xi(ndim)

  ! introduce a pointer and form matrix A: [element coordinates, 1]
  p => elem(i)%p
  matx = x(p,:)
  do k = 1,ndim+1
    matA(k,:) = [matx(k,:),ONE]
  end do

  ! introduce the integration point vertice and form other matrices

  vi = [xi,ONE]
  matB = matA
  matC = matA
  matD = matA
  matB(1,:) = vi
  matC(2,:) = vi
  matD(3,:) = vi

  ! make list of all determinants of the matrices
  if (ndim==2) then
    det = (/.det.matA, .det.matB, .det.matC, .det.matD/)
  else if (ndim==3) then
    matE = matA
    matE(4,:) = vi
    det = (/.det.matA, .det.matB, .det.matC, .det.matD, .det.matE/)
  end if

  ! turbine integration points
  if (findAL==0) then
    ! if all determinants have the same sign, integration point is in element
   -> output subroutine
    if ((all(det<=NUL).or.all(det>=NUL))) then
      turb(j)%element_list(n)=i
      inelem=1
      count_Nt=count_Nt+1
    else
      if (inelem==0.or.turb(j)%element_list(n)<=nboun.or.turb(j)%element_list(n
   )>=nelem) then
        turb(j)%element_list(n) = 0
      end if
    end if
  end if

  ! actuator line points
  if (findAL==1) then
    ! if all determinants have the same sign, integration point is in element
   -> output subroutine
    if ((all(det<=NUL).or.all(det>=NUL))) then
      turb(j)%AL_element_list(n,m)=i
      inelem=1
      count_Nt=count_Nt+1
    else
      if (inelem==0.or.turb(j)%AL_element_list(n,m)<=nboun.or.turb(j)%
   AL_element_list(n,m)>=nelem) then
        turb(j)%AL_element_list(n,m) = 0
      end if
    end if
  end if

end subroutine element_search
```

C.2.8. Mesh deformation check subroutine

FinLab deforms the mesh during the time iterations when a moving free-surface is applied. The vertical column of elements is stretched out when the free surface level above the column rises. Therefore, if such a free-surface is applied, it could be that turbine integration points or AL sampling points switch element during the iterations. The subroutine 'free_surface_check' checks this. It is however, not feasible to simply run the subroutines 'turbine_elements' and 'AL_elements' completely in each iteration to perform this check. These subroutines loop over all elements in the partition for each point, which leads to enormous computational costs.

In the 'free_surface_check' subroutine the check is executed first for the turbine integration points. The subroutine loops over all turbine points and checks if they are positioned in the mesh partition. If so, it is checked with subroutine 'element_search' first if the point is still in the same element as in the previous iteration. The coordinates of the turbine integration point and the element in which the point was located in the previous equation are used as input. Afterwards, only if the turbine has left the element, the loop over all the elements in the partition is executed to find the new position of the integration point. This loop starts at the element-number of the element of the previous iteration to save on computational cost.

```fortran

! subroutine to check if turbine integration point or AL points have changed
   element or partition
subroutine free_surface_check(j)
  integer :: j

  ! check location of actuator disc turbine integration points within
   iterations if free surface is applied
  do n = 1,Nt
    findAL = 0
    pt = turb(j)%xt(n,:)
    ! check if turbine integration point is in the partition

    if (turb(j)%element_list(n)/=0) then
      inelem = 0
      l = turb(j)%element_list(n)
      ! check if turbine integration point is still in the same element

      call element_search(pt,turb(j)%element_list(n))
      if (inelem==0) then
        ! if the point is not found in the same element start a loop to search
   for the new element
        newposition_AD: do i = 1,nelem
          ! start the loop at the old element to minimize computational cost,
   exit loop if found
          m = l + i
          if (m<nelem) call element_search(pt,m)
          if (inelem==1) exit newposition_AD
          m = l - i
          if (m>1+nboun.and.m<nelem) call element_search(pt,m)

          if (inelem==1) exit newposition_AD
        end do newposition_AD
```

In the last line of the previous block, an if-statement indicates whether the turbine integration point is found or not. When the point is not found, it either got above the free-surface, or it changed partition. In the first case, the point is lost forever in the current code. For the second case, the turbine number and integration point number of the lost point are saved to send them to all partitions to look for the element. To do so, the number of lost turbine integration point is stored in the parameter 'AD_lost', thereafter the array 'AD_xlost' is allocated in which all the turbine and point numbers are stored of the points lost during the iteration. If more than one point is lost, the temporary array 'AD_xlost1' is used to extend the array 'AD_xlost'. The lost points are sent to the other partitions at the start of a new iteration in the 'MPI_turbine' subroutine.

```


     ! the point switched partition if it’s not found
     if (inelem/=1) then
      ! count the number of lost points in the partition
      AD_lost = AD_lost + 1
      ! add the turbine and point number of the first lost point to the array ’xlost’
      if (AD_lost==1) then
       allocate(AD_xlost(2))
       AD_xlost(1) = j
       AD_xlost(2) = n
      end if
      ! expand the ’xlost’ array if more points are lost during one iteration
      if (AD_lost>1) then
       allocate(AD_xlost1((AD_lost-1)*2))
       AD_xlost1 = AD_xlost
       deallocate(AD_xlost)
       allocate(AD_xlost(AD_lost*2))
       AD_xlost = [AD_xlost1,j,n]
      end if
      if (AD_lost>1) deallocate(AD_xlost1)
     end if
    end if
   end if
  end do
```

The exact same check is also performed for the AL sampling points if the AL-BEM method is selected in combination with a moving free-surface.

```
  ! check location of actuator line points within iterations if free surface is applied
  if (turbine_method==2.and.integration_method==2) then
   do n = 1, AL_nlines
    do m = 1, AL_npoints
     findAL = 1
     pt = turb(j)%AL_points(n,m,:)
     ! check if actuator line point is in the partition
     if (turb(j)%AL_element_list(n,m)/=0) then
      inelem = 0
      l = turb(j)%AL_element_list(n,m)
      ! check if actuator line point is still in the same element
      call element_search(pt,turb(j)%AL_element_list(n,m))
      if (inelem==0) then
       ! if the point is not found in the same element start a loop to search for the new element
       newposition_AL: do i = 1,nelem
        ! start the loop at the old element to minimize computational cost, exit loop if found
        q = l + i
        if (q<nelem) call element_search(pt,q)
        if (inelem==1) exit newposition_AL
        q = l - i
        if (q>1+nboun.and.q<nelem) call element_search(pt,q)
        if (inelem==1) exit newposition_AL
       end do newposition_AL
      ! the point switched partition if it's not found
      if (inelem/=1) then
       ! count the number of lost points in the partition
       AL_lost = AL_lost + 1
       ! add the turbine number, line number and point number of the first lost point to the array 'xlost'
       if (AL_lost==1) then
        allocate(AL_xlost(3))
        AL_xlost(1) = j
        AL_xlost(2) = n
        AL_xlost(3) = m
       end if
       ! expand the 'xlost' array if more points are lost during one iteration
       if (AL_lost>1) then
        allocate(AL_xlost1((AL_lost-1)*3))
        AL_xlost1 = AL_xlost
        deallocate(AL_xlost)
        allocate(AL_xlost(AL_lost*3))
        AL_xlost = [AL_xlost1,j,n,m]
       end if
       if (AL_lost>1) deallocate(AL_xlost1)
      end if
     end if
    end if
   end do
  end do
 end if

end subroutine free_surface_check
```

C.2.9. Get velocity from mesh subroutine

The subroutine 'get_velocity' is used to read the velocity in a given direction in a given element. It takes the average velocity of the four nodes as the velocity at the element. The subroutine could be extended if the option to add turbines under an angle with the axes is added. The subroutine could be made more exact by interpolating the velocities on the four nodes to the exact location of the point.

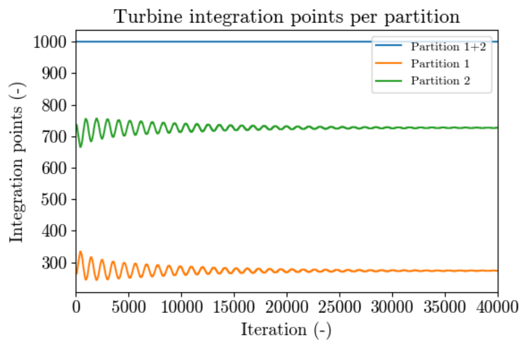
```

! subroutine to get average velocity in element
subroutine get_velocity(i)
 integer :: i
 p => elem(i)%p
 if (ndim == 2) then
  ui(1) = sum(u(p,1))/3
  ui(2) = sum(u(p,2))/3
 else if (ndim == 3) then
  ui(1) = sum(u(p,1))/4
  ui(2) = sum(u(p,2))/4
  ui(3) = sum(u(p,3))/4
 end if
end subroutine
```

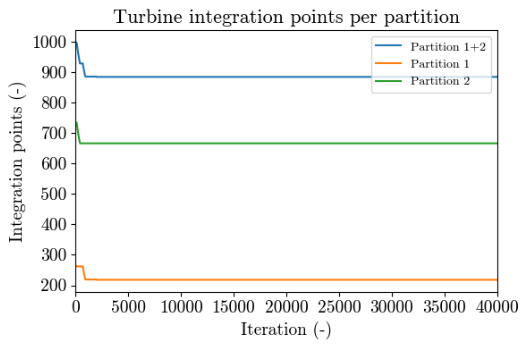
C.2.10. MPI communication subroutine

All the MPI communication between the different processors is done in the 'MPI_turbine' subroutine. This subroutine is activated at the start of the new iteration by the 'add_turbine_force' subroutine if any communication needs to be executed. The subroutine is activated at the start of the new iteration as the subroutine needs to block all processes. This could lead to an unnecessary delay of the computational time when executed in the middle of an iteration.

Figures C.1 and C.2 clearly show the effects of the communication of the partitions, to give an example of why communication is needed. These plots are coming from a 2DV uniform AD simulation with a moving free-surface. The AD is located in two partitions. In figure C.1 it can be seen that the total number of found turbine integration points decreases when the partitions do not communicate. In figure C.2 it can be seen that this leads to a gap in the uniform disc in which the flow can bypass.

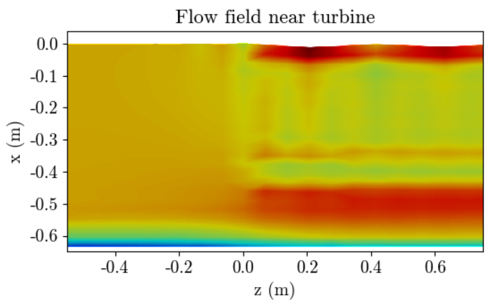


(a) With communicating partitions

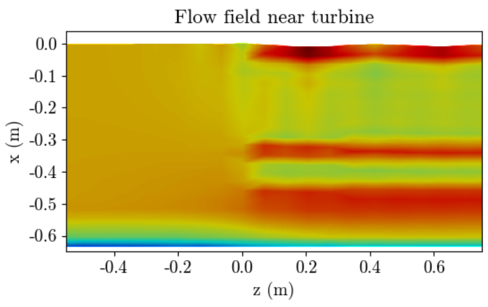


(b) Without communicating partitions

Figure C.1: The number of turbine integration points located in the partitions, with and without communicating partitions



(a) With communicating partitions



(b) Without communicating partitions

Figure C.2: Flow field for a simulation with and without communicating partition (red: high flow velocity, yellow: medium flow velocity, blue: low flow velocity

Apart from communicating if any turbine integration or AL sampling points have left the partition, the subroutine 'MPI_turbine' also optionally communicates the sampled AL velocities and the nacelle thrust coefficients. To do so, the subroutine is split in two major parts. In first, the processors send all information to each other. Thereafter, the process on all the processors is blocked until all processors have reached this point. Then in second, the processors receive and process all sent information.

```
!!! MPI COMMUNICATION SUBROUTINE
subroutine MPI_turbine
  !!! MPI communication part !!!
  !   Depending on the input settings the following are sent to all partitions
  !     If the nacelle subroutine is applied
  !     - Nacelle thrust coefficients: turb(j)%C_nac
  ! If a moving free surface is applied
  !     - Lost turbine integration points
  !     - Lost actuator line points (of course only if the actuator line is
    applied)
  ! If actuator line
  !     - Tangential and axial velocities on the actuator line points
```

```

call mpi_comm_rank(mpi_comm_world, rank, ierr)
```

The first information that is sent by the processors are the sampled AL velocities. Of course, only if the AL-BEM method is selected. The processors send the sampled AL velocities to each other as it is possible that the centre chord line of the blade and a part of the blade over which the force is distributed are positioned in different partitions. To read the velocities from the mesh, the actuator lines on which the blades are positioned in the particular time-step are activated. Each time-step the activated blades are switched one position. See 3.6 for an example of two activated actuator lines.

```

!!! Actuator line: send velocities !!!
! get velocity at actuator lines if actuator line method is applied
if (turbine_method==2.and.integration_method==2) then
  ! first step is to activate the right actuator lines according to the
 timestep
  do j = 1,nturb
    ! loop over actuator lines
    do n = 1, N_blades
      ! activate the actuator lines
      AL_activated(n) = mod(itijd + (n-1)*AL_nlines/N_blades,AL_nlines)
      if (AL_activated(n) == 0) AL_activated(n) = AL_nlines
    end do
```

The velocities can be read from the mesh once the actuator lines are activated. The velocity is read by using a nested loop over the number of blades and the AL points per activated line. The axial and tangential velocities are sampled if the AL sampling point is located in the mesh partition, otherwise, a value of zero is saved for both the velocities.

```

    ! next step is to read the velocities at the activated actuator lines
 from the mesh
    ! set the velocity in all actuator line points to zero before reading new
  velocities from the mesh
    turb(j)%AL_velocity = 0.0
    ! loop over the actuator line points
    do n = 1, N_blades
      do m = 1, AL_npoints
        ! if actuator line point is on activated blade read velocity in mesh
 element i
        if (turb(j)%AL_element_list(AL_activated(n),m)/=0) then
          i = turb(j)%AL_element_list(AL_activated(n),m)
          call get_velocity(i)
          ! add axial velocity to the AL_velocity list
          turb(j)%AL_velocity(n,m,1) = dot_product(ui,nx)
          ! add tangential velocity to the AL_velocity list
          if (ndim==3) then
            ntan(1) = 0
            ntan(2) = -sin(turb(j)%lambda+2*PI/N_blades*(n-1))
            ntan(3) = cos(turb(j)%lambda+2*PI/N_blades*(n-1))
            turb(j)%AL_velocity(n,m,2) = dot_product(ui,ntan)
          end if
        end if
      end do
    end do
```

The velocities are saved in the turbine specific 'AL_velocity_sent' array. The processor sends this list to all processors by looping over all processors and calling the 'MPI_send' function.

```

    ! at this point each partition has a list with velocities at the activated actuator line points
     ! the velocities are equal to zero if the actuator line point is not located in the partition
     ! the velocities equal to the average velocity in the mesh element if the actuator line point is located in the partition
    ! the velocities are transferred to a list which can be sent
    turb(j)%AL_velocity_sent = turb(j)%AL_velocity

    ! all partitions send all their velocities at the activated actuator line points to all partitions
    do l = 1,NPROC
      call mpi_send(turb(j)%AL_velocity_sent,N_blades*AL_npoints*2,MPI_DP,l-1,0,mpi_comm_world,ierr)
    end do

   ! end loop over turbines
   end do
  end if
  !!! End Actuator line: send velocities !!!
```

In the next block of code, the processors send the turbine specific nacelle thrust coefficients to all processors. In the 'nacelle' subroutine, each centre turbine integration point produces a turbine specific nacelle thrust coefficient which brings the axial flow velocity to zero in the centre of the nacelle. This thrust coefficient can be unknown for the other integration points which are located within the nacelle area if the nacelle is split over multiple partitions. In this subroutine, the processor first sends the number of nacelle centres which are located within the partition to all partitions, this is the parameter 'c_nac_count'. By doing this, the other partitions know how many thrust coefficient they can expect. Thereafter, if 'c_nac_count' is higher than zero, the processor sends the array 'c_nac_turb', to all processors. This array contains the turbine numbers of the turbines with the nacelle centres located in the partition. At last, the processor sends the array 'c_nac_temp' to all processors. This array contains the nacelle thrust coefficients corresponding to the sent turbine numbers.

```

  !!! Send lost turbine integration points and nacelle thrust coefficient !!!
  ! send how many integration points were lost in the partition during the previous iteration to all partitions
  ! loop over all partitions
  do l = 1,NPROC
   ! send nacelle thrust coefficients if nacelle method is applied
   if (nacelle_method==1) then
    ! first send how many nacelle centres are located in the partition -> the receiving partitions know how many values to expect
    call mpi_send(c_nac_count,1,MPI_int,l-1,0,mpi_comm_world,ierr)
    ! thereafter send the nacelle thrust coefficients that are located in the partition
    if (c_nac_count>0) then
     ! send the turbine numbers of turbines which centres are located in the partition
     call mpi_send(c_nac_turb,c_nac_count,MPI_int,l-1,0,mpi_comm_world,ierr)
     ! send the nacelle thrust coefficients of the turbines which centres are located in the partition
     call mpi_send(c_nac_temp,c_nac_count,MPI_DP,l-1,0,mpi_comm_world,ierr)
    end if
   end if
```

The last items to be sent are the turbine integration and AL sampling points, which switched positions. The number of lost points with the corresponding point numbers are saved in arrays in the 'free_surface_check' subroutine. In this subroutine, the processors first send the number of lost integration and AL points to all processors, these are parameters 'AD_lost' and 'AL_lost' respectively. This is again done to let the receiving processors know how many points can be expected. Thereafter, if the number of lost points is higher than zero, the corresponding list containing turbine number and points are sent. Apart from these two, the AL sampling points list also contains the line number.

```fortran


    ! send lost turbine integration points if free-surface is applied
    if (move==1) then
      ! first send how many integration points are lost in the partition -> the receiving partitions know how many values to expect
      call mpi_send(AD_lost,1,MPI_int,l-1,0,mpi_comm_world,ierr)
      ! if any points were lost, send the correspoding turbine and integration point numbers to all partitions
      if (AD_lost>0) call mpi_send(AD_xlost,2*AD_lost,MPI_int,l-1,0,mpi_comm_world,ierr)
      if (turbine_method==2.and.integration_method==2) then
        ! first send how many integration points are lost in the partition -> the receiving partitions know how many values to expect
        call mpi_send(AL_lost,1,MPI_int,l-1,0,mpi_comm_world,ierr)
        ! if any points were lost, send the correspoding turbine and integration point numbers to all partitions
        if (AL_lost>0) call mpi_send(AL_xlost,3*AL_lost,MPI_int,l-1,0,mpi_comm_world,ierr)
      end if
    end if
  end do
```

The arrays which are used to send the information are also used to receive the information, therefore, all arrays are deallocated. Thereafter, the sending part is done. The code is now blocked by calling the 'MPI_barrier' function.

```fortran

  ! deallocate temporary lists
  if (AD_lost>0) deallocate(AD_xlost)
  if (AL_lost>0) deallocate(AL_xlost)
  if (c_nac_count>0) deallocate(c_nac_turb,c_nac_temp)
  !!! End send lost turbine integration points and nacelle thrust coefficient !!!

  ! Transition from send to receive: wait untill all partitions have sent their data
  if (nacelle_method==1.or.move==1) call mpi_barrier(mpi_comm_world,ierr)
```

The receiving part starts once all processors have reached the 'MPI_barrier' function. The order of the receiving process is the same as the order of the sending process. Therefore, in first, the sampled AL velocities are received by all processors. To do so, a nested loop over all turbines and processors is used. The processors receive a set of all AL sampling points from each processor. Once a set is received a loop is started which filters out all nonzero velocities. Together all sets form the complete set of turbine specific sampled velocities in the array 'AL_velocity'.

```fortran

  !!! Actuator line: receive velocities !!!
  if (turbine_method==2.and.integration_method==2) then
    ! loop over turbines
    do j = 1,nturb
      ! loop over partitions
      do l = 1,NPROC
```

```
   ! set velocities of receiving list to zero to prevent mixing up with velocity values of the previous partition and iteration
   turb(j)%AL_velocity_sent = 0.0
   ! receive actuator line velocity values from other partitions
   call mpi_recv(turb(j)%AL_velocity_sent,N_blades*AL_npoints*2,MPI_DP,l-1,0,mpi_comm_world,mpi_status_ignore,ierr)
   ! loop over all values in the received velocity matrix
   do n = 1, N_blades
    do m = 1, AL_npoints
     ! the axial and tangential velocities are saved to the AL_velocity list if a value in the received matrix not zero
     if (turb(j)%AL_velocity_sent(n,m,1) /= 0.0) then
      turb(j)%AL_velocity(n,m,1) = turb(j)%AL_velocity_sent(n,m,1)
      turb(j)%AL_velocity(n,m,2) = turb(j)%AL_velocity_sent(n,m,2)
     end if
    end do
   end do
  end do
 end do
end if
!!! End actuator line: receive velocities !!!
```

The sent nacelle thrust coefficients are received next. Again, a loop over all processors is used to do so. First, from each processor the number of thrust coefficients is received. Accordingly, an array of the corresponding size is allocated, which is thereafter filled the received turbine numbers and thrust coefficients. At last, a loop is used to apply the received thrust coefficients to the correct turbines.

```


!!! Receive lost turbine integration points and nacelle thrust coefficient !!!
do l = 1,NPROC
 ! receive nacelle thrust coefficients if nacelle method is applied
 if (nacelle_method == 1) then
  ! receive how many nacelle centres are located in the sending partition
  call mpi_recv(c_nac_count,1,MPI_int,l-1,0,mpi_comm_world,mpi_status_ignore,ierr)
  if (c_nac_count>0) then
   ! allocate list to receive the right number of nacelle thrust coefficients
   allocate(c_nac_turb(c_nac_count),c_nac_temp(c_nac_count))
   ! receive the turbine numbers of the turbines which centres are located in the sending partition
   call mpi_recv(c_nac_turb,c_nac_count,MPI_int,l-1,0,mpi_comm_world,mpi_status_ignore,ierr)
   ! receive the nacelle thrust coefficients of the turbines which centres are located in the sending partition
   call mpi_recv(c_nac_temp,c_nac_count,MPI_DP,l-1,0,mpi_comm_world,mpi_status_ignore,ierr)
   ! apply the received nacelle thrust coefficient to the right turbine
   do m = 1,c_nac_count
    j = c_nac_turb(m)
    turb(j)%c_nac = c_nac_temp(m)
   end do
   deallocate(c_nac_turb,c_nac_temp)
  end if
 end if
```

At last, the lost turbine integration and AL sample points are received. The same process of first receiving how many points will be sent and thereafter actually receiving the points is used. The difference with the nacelle thrust coefficient is that here, a loop over all mesh elements is started each time a point is received. This loop is used to find the received point in the receiving partition by calling the '`element_search`' subroutine. The point will always be found by one of the partitions if the point is not positioned above the free-surface.

```


    ! receive lost turbine integration points if free-surface is applied
    if (move==1) then
      ! receive how many points were lost in the sending partition
      call mpi_recv(AD_lost,1,MPI_int,l-1,0,mpi_comm_world,mpi_status_ignore,ierr)
      if (AD_lost>0) then
        ! allocate a list of the size of the number of lost points in the sending partition
        allocate(AD_xlost(2*AD_lost))
        ! receive the list of the lost points in the sending partition [turbine number, integration point number]
        call mpi_recv(AD_xlost,2*AD_lost,MPI_int,l-1,0,mpi_comm_world,mpi_status_ignore,ierr)
        ! loop over all received point too check if they moved to the current partition
        do m = 1,AD_lost*2,2
          j=AD_xlost(m)
          n=AD_xlost(m+1)
          ! get coordinates of lost point
          pt = turb(j)%xt(n,:)
          inelem = 0
          findAL = 0
          ! loop over the elements of the partition to search for the lost point
          find_AD: do i = 1+nboun,nelem
            call element_search(pt,i)
            ! exit if the point is found
            if (inelem==1) then
              exit find_AD
            end if
          end do find_AD
        end do
        ! deallocate temporary list
        deallocate(AD_xlost)
      end if

      ! receive how many points were lost in the sending partition
      if (turbine_method==2.and.integration_method==2) then
        call mpi_recv(AL_lost,1,MPI_int,l-1,0,mpi_comm_world,mpi_status_ignore,ierr)

        if (AL_lost>0) then
          ! allocate a list of the size of the number of lost points in the sending partition
          allocate(AL_xlost(3*AL_lost))
          ! receive the list of the lost points in the sending partition [turbine number, integration point number]
          call mpi_recv(AL_xlost,3*AL_lost,MPI_int,l-1,0,mpi_comm_world,mpi_status_ignore,ierr)
          ! loop over all received point too check if they moved to the current partition
          do q = 1,AL_lost*3,3
            j=AL_xlost(q)
```

```
              n=AL_xlost(q+1)
              m=AL_xlost(q+2)
              ! get coordinates of lost point
              pt = turb(j)%xt(m,:)
              inelem = 0
              findAL = 1
              ! loop over the elements of the partition to search for the lost
    point
              find_AL: do i = 1+nboun,nelem
                call element_search(pt,i)
                ! exit if the point is found
                if (inelem==1) then
                  exit find_AL
                end if
              end do find_AL
            end do
            ! deallocate temporary list
            deallocate(AL_xlost)
          end if
        end if
      end if
    end do
```

At the end of the 'MPI_turbine' subroutine all counters are set to zero.

```

  ! set all communicate counters to zero before the start of the new iteration
  AD_lost = 0
  AL_lost = 0
  c_nac_count = 0

end subroutine MPI_turbine
```

C.2.11. Uniform AD subroutine

The 'uniform_disc' subroutine is rather simple. It applies the force per turbine integration point, as determined in subroutine 'init_turbine', to the axial source term of the selected element.

```

!!! UNIFORM DISC SUBROUTINE
subroutine uniform_disc(i)
  integer :: i
  ! apply force to source term
  fe(i,:,ax) = fe(i,:,ax) - dF
end subroutine uniform_disc
```

C.2.12. BEM subroutine

The BEM equations are applied in the 'BEM' subroutine. The first step of the subroutine is to interpolate the parameters from the BEM files exactly to the location of the integration points. Thereafter some general parameters are determined, which are later used to determine the blade solidity.

```

!!! BEM SUBROUTINES

! subroutine blade element momentum
subroutine BEM(j,i,n)
```

```
integer :: j,i,n

! use radius of blade element to get output of geometry file
m = 38
k = index_used(turb(j)%rad(n)*turbine_scale, r_list, m)
ki = interpolate(turb(j)%rad(n)*turbine_scale, r_list, m, k)
chord = get_interpolate(c_list, m, k, ki)/turbine_scale
et = get_interpolate(e_list, m, k, ki)
Transit = get_interpolate(transit_list, m, k, ki)
NACA = get_interpolate(NACA_list, m, k, ki)

! determine cell parameters
Re = u_ref*chord*rho_w/visc0
dr = 2*sqrt(A_i/pi)
r1 = turb(j)%rad(n) - 0.5*dr
r2 = turb(j)%rad(n) + 0.5*dr
A_annulus = pi*(r2**2 - r1**2)
```

In the BEM subroutine the 'on_blade' parameter is introduced. This parameter indicates whether or not the current turbine integration point is located on one of the blades in the current time-step. The remaining of the subroutine is only executed if this parameter is equal to one. In the AD-BEM method, due to the time-averaging nature, all turbine integration points are of course always located on the blades. Therefore, 'on_blade=1' if the AD-BEM method is selected. If the AL-BEM method is selected the subroutine 'AL_on_blade' is activated to determine the value of the parameter. The 'AL_on_blade' subroutine forms, e.g., the figures 3.8a and 3.8b.

```

! a point should be on the blade if the AL-BEM method is applied
on_blade = 0
! the points are always on the blade if the AD-BEM is applied
if (integration_method == 1) on_blade = 1
! check if point is on blade if the AL-BEM method is applied
if (integration_method == 2) call AL_on_blade(j,i,n)
! only execute the BEM functions if the point is on the blade (note again
 that in AD-BEM the points are always on the blade)
if (on_blade==1) then
```

The next step is to read the local velocities from the mesh. The velocities are read in the element of the turbine integration point if the AD-BEM method is selected. The centre chord velocities are used if the AL-BEM method is selected. These centre chord velocities are read and sent to all partitions in the 'MPI_turbine' subroutine. Here, the velocities are interpolated according to the radius of the turbine integration point and AL sampling points.

```


  ! get velocities near blade for AD-BEM method
  if (integration_method==1) then
    call get_velocity(i)
    u_a = dot_product(ui,nx)
    ntan(1) = 0
    ntan(2) = -sin(turb(j)%theta(n))
    ntan(3) = cos(turb(j)%theta(n))
    u_t = dot_product(ui,ntan)
  ! get velocities at chord centre line of previous iteration for AL-BEM
 method
  else if (integration_method==2) then
    ! first find which actuator line point should be used
    k = index_used(turb(j)%rad(n), AL_rad, AL_npoints)
    ! use first point
```

```
      if (k == 0) then
        k = 1
        ki = 0
      ! use last point
      else if (k >= AL_npoints) then
        k = AL_npoints
        ki = 0
      ! interpolate if integration point is located betweeen first actuator
    line point and last actuator line point
      else
        ki = interpolate(turb(j)%rad(n), AL_rad, AL_npoints, k)
      end if
      ! use the interpolated value and blade number to read the velocity from
    the AL velocity list
      u_a = get_interpolate(turb(j)%AL_velocity(blade,:,1), AL_npoints, k, ki)
      u_t = get_interpolate(turb(j)%AL_velocity(blade,:,2), AL_npoints, k, ki)
    end if
```

In the next block of code, the resultant velocity with its inflow angle and angle of attack are determined according to equations 3.2, 3.3 and 3.4 respectively. With this angle of attack and the chord Reynolds number known, the lift and drag coefficients can be determined by interpolating over the lists coming from the BEM files.

```

    ! determine phi according to resultant velocity
    u_r2 = u_a**2+(omega*turb(j)%rad(n)-u_t)**2
    phi = atan((omega*turb(j)%rad(n)-u_t)/u_a)

    ! determine alpha
    alpha = 90 - et - phi*180/pi

    ! get output lift and drag file
    m = 148
    k = index_used(alpha, alpha_list, m)
    ki = interpolate(alpha, alpha_list, m, k)

    ! get columns with right Reynolds number
    if (Re <= 2000000.) then
      CL_column = NCL1
      CD_column = NCD1
    else if (Re > 2000000. .and. Re <= 3500000.) then
      CL_column = NCL2
      CD_column = NCD2
    else if (Re > 3500000. .and. Re <= 4500000.) then
      CL_column = NCL3
      CD_column = NCD3
    else if (Re > 4500000. .and. Re <= 5500000.) then
      CL_column = NCL4
      CD_column = NCD4
    else if (Re > 5500000.) then
      CL_column = NCL5
      CD_column = NCD5
    end if
    ! determine lift and drag coefficients
    C_L = Transit*get_interpolate(TCL, m, k, ki) + NACA*get_interpolate(
   CL_column, m, k, ki)
    C_D = Transit*get_interpolate(TCD, m, k, ki) + NACA*get_interpolate(
   CD_column, m, k, ki)
```

Subsequently, the lift, drag, axial and tangential forces of the blades are calculated with equations 3.5, 3.6, 3.7 and 3.8 respectively. The axial and tangential forces only include the chord solidity if the AD-BEM method is selected. A spin-up correction is applied to prevent instabilities from occurring during the damping time.

```

    ! run last BEM functions
    dF_L = 0.5*rho_w*u_r2*C_L*A_i
    dF_D = 0.5*rho_w*u_r2*C_D*A_i
    S_A = dF_L*sin(phi) + dF_D*cos(phi)
    S_T = dF_L*cos(phi) - dF_D*sin(phi)
    ! calculate chord solidity
    s = N_blades*chord*(r2-r1)/A_annulus
    ! apply chord solidity if AD-BEM method is applied
    if (integration_method == 1) then
      S_A = S_A*s
      S_T = S_T*s
    end if

    ! apply spin-up correction to prevent instabilities
    if (itijd*dt<Tdamp) then
       S_A = S_A*itijd*dt/Tdamp
       S_T = S_T*itijd*dt/Tdamp
     end if
```

In the next block of code, the tip-correction is applied according to the settings in the main file. The applied forces can be decreased by the Prandtl tip-correction. The Edmunds et al. tip-correction introduces an extra term according to an elliptical distribution function. This extra term is applied in the next block of code in which the forces are applied to the mesh. This elliptical distribution is set to zero if either no or the Prandtl tip-correction is applied.

```

    ! apply tip-correction if applied
    ! prandtl tip correction
    if (tip_correction == 1) then
      f_tip = (2.0/PI) * acos(exp(-N_blades*(R-turb(j)%rad(n))/(2*turb(j)%rad(n
   )*cos(phi))))
      S_A = S_A*f_tip
      S_T = S_T*f_tip
      theta0 = 0.0
      gamma0 = 0.0
    ! edmunds tip correction
    else if (tip_correction == 2) then
      theta0 = acos(turb(j)%rad(n)/(-R))
      gamma0 = 1-sin(theta0)
    else
      theta0 = 0.0
      gamma0 = 0.0
    end if
```

In the last block of code, the forces are applied to the mesh elements in the x-, y- and z-direction according to equations 3.13, 3.14 and 3.15 respectively. At last, the turbine- and integration point-specific data is saved in the output arrays.

```

    ! apply force to source term
    fe(i,:,ax) = fe(i,:,ax) - (1+gamma0)*(S_A/rho_w)/(ndim+1)
    fe(i,:,ay) = fe(i,:,ay) - (1+gamma0)*(S_T/rho_w)*sin(turb(j)%theta(n))/(
   ndim+1)
    fe(i,:,az) = fe(i,:,az) + (1+gamma0)*(S_T/rho_w)*cos(turb(j)%theta(n))/(
   ndim+1)
```

```fortran

    ! add bem results to export lists
    turb(j)%dThrust(n) = S_A
    turb(j)%dTorque(n) = S_T*turb(j)%rad(n)
    turb(j)%dPower(n) = S_T*turb(j)%rad(n)*omega
  end if

end subroutine BEM
```

C.2.13. BEM nacelle subroutine

In the 'nacelle' subroutine equation 3.24 is applied to set the flow velocity in the element to zero in the axial direction. To do so, the axial flow velocity in the element is read and the turbine specific C_{nac} is adjusted accordingly. Thereafter, an array containing all turbine numbers and nacelle thrust coefficients of the nacelles centres that are located in the mesh partition is prepared. The processor sends this array to all the other processors in the 'MPI_turbine' subroutine. At last, the force is applied to the element. The sum of the force of all the turbine integration points within the nacelle area is determined for the selected turbine.

```fortran

! Nacelle subroutine
subroutine nacelle(j,i,n)
  integer :: i,j,n

  ! adjust C_nac if necessary
  if (n==1) then
    p => elem(i)%p
    if (all(u(p,ax) >= 0.05)) turb(j)%C_nac = turb(j)%C_nac*1.001
    if (any(u(p,ax) <= -0.05)) turb(j)%C_nac = turb(j)%C_nac*.999
    print*, u(p,ax)

  ! make C_nac transferable to other partitions
    c_nac_count = c_nac_count + 1
    if (c_nac_count==1) then
      allocate(c_nac_temp(1),c_nac_turb(1))
      c_nac_temp = turb(j)%C_nac
      c_nac_turb = j
    end if
    if (c_nac_count>1) then
      allocate(c_nac_temp1(c_nac_count-1),c_nac_turb1(c_nac_count-1))
      c_nac_turb1 = c_nac_turb
      c_nac_temp1 = c_nac_temp
      deallocate(c_nac_temp,c_nac_turb)
      allocate(c_nac_temp(c_nac_count),c_nac_turb(c_nac_count))
      c_nac_temp = [c_nac_temp1,turb(j)%c_nac]
      c_nac_turb = [c_nac_turb1,j]
    end if
    if (c_nac_count>1) deallocate(c_nac_temp1,c_nac_turb1)
  end if

  ! apply force to source term
  theta0 = acos(turb(j)%rad(n)/(-R_nac))
  gamma0 = sin(theta0)
  fe(i,:,ax) = fe(i,:,ax) - 0.5*turb(j)%C_nac*gamma0*u_ref**2*A_i/(ndim+1)
  turb(j)%T_nac = turb(j)%T_nac + 0.5*turb(j)%C_nac*gamma0*u_ref**2*A_i*rho_w

end subroutine nacelle
```

C.2.14. AL subroutine

The 'AL_on_blade' subroutine forms determines the value of the 'on_blade' parameter. This parameter states whether or not the turbine integration points is located on one of the blades in the current time-step. Figures 3.8a and 3.8b are formed by coloring all the turbine integration points according to the value of the 'on_blade' parameter in the first time-step. The parameters λ_1 and λ_2 are determined for all blades according to figure 3.7. The 'on_blade' parameter is set to one if the turbine integration point specific angle θ is located between λ_1 and λ_2. The parameter 'blade' indicates on which blade the turbine integration point is located. This parameter is used to read the centre chord velocity from the correct blade.

```fortran

subroutine AL_on_blade(j,i,n)
  integer :: i,j,n

  ! Actuator line: check if turbine integration point is located on one of the
    blades
  if (integration_method == 2) then
    ! determine delta lambda
    dlambda = asin(0.5*chord/turb(j)%rad(n))
    ! loop over blades
    do m = 0,N_blades-1
      ! determine lambda1 and lambda2 of according to the position of the blade
      lambda = modulo((turb(j)%lambda + 2*PI/N_blades*m),2*PI)
      theta = modulo(turb(j)%theta(n),(2*PI))
      lambda1 = lambda+dlambda
      lambda2 = lambda-dlambda
      ! the point is on one of the blades if the fibonacci angle of the turbine
     integration point (theta) is between lambda1 and lambda2
      if (theta<lambda1.and.theta>lambda2) then
        ! register that point is on blade
        on_blade = 1
        ! register on which blade the point is located
        blade = m + 1
      end if
      ! check for negative angle to make sure no points are missed near lambda
    = 0.0
      if (lambda2 < 0.0) then
        if (theta<2*PI.and.theta>2*PI+lambda2) then
          on_blade = 1
          blade = m + 1
        end if
      end if
    end do
  end if

end subroutine AL_on_blade
```

C.2.15. Read BEM file subroutine

The blade element method files are read if the method is selected in the subroutine 'init_turbine' by calling the subroutine 'read_BEM_files'. The first file consists of radius dependent blade geometric data. The second file consists of the angle of attack and Reynolds number dependent lift and drag coefficients.

```fortran

! read bem files if bem method selected
subroutine read_bem_files
  !Read geometry file
  open(unit=10,file=geo_file,status='old')
  read(10,*) columns
  do i = 1,38
```

```fortran
      read(10,*)r_list(i), c_list(i), e_list(i), t_list(i), tc_list(i), Transit_list(i), NACA_list(i)
    end do
    R_nac = r_list(1)/turbine_scale

    !Read lift and drag file
    open(unit=11,file=lift_drag_file,status='old')
    read(11,*) columns
    do i = 1,148
      read(11,*) alpha_list(i),TCL(i),TCD(i),NCL1(i),NCD1(i),NCL2(i),NCD2(i),NCL3(i),NCD3(i),NCL4(i),NCD4(i),NCL5(i),NCD5(i)
    end do
end subroutine read_bem_files
```

C.2.16. Interpolation subroutines

The data in both the geometric and the coefficient files are given on discreet intervals of the radius and the angle of attack respectively. Therefore, three interpolation functions were used to get the interpolated values of the data in the lists. The function 'get_index' takes either the radius or the angle of attack as input and gives the lower index of the rows between which the value is located as an output. Thereafter, the function 'interpolate', determines at which point the input value is linearly interpolated between both the values in the list. At last, the function 'get_interpolated', gives the interpolated values of the parameters in the other columns as an output.

```fortran

! function to find closest value in a list to a given value
  ! function to find integer index
  function index_used(var_ref, var_list, m)
    real(kind=DP) :: var_list(m), var_ref, var_used
    integer :: z, m, index_used
    ! find where between which two values u_ref is located in the table array
    do z = 1, m-1
      if (var_ref >= var_list(z) .and. var_ref < var_list(z+1)) then
        index_used = z
      end if
    end do
    if (var_ref >= var_list(m)) index_used = m
    if (var_ref <= var_list(1)) index_used = 1
  end function index_used

  ! get position parameter of the given value between the two values in the list
  function interpolate(var_ref, var_list, m, i)
    real(kind=DP) :: interpolate, var_ref, var_list(m)
    integer :: m, i
    interpolate = (var_ref - var_list(i))/(var_list(i+1) - var_list(i))
  end function interpolate

  ! get interpolated value from lists
  function get_interpolate(var_list, m, i, ki)
    real(kind=DP) :: get_interpolate, var_ref, var_list(m), ki
    integer :: m, i
    if (i == m) then
      get_interpolate = var_list(i)
    else
      get_interpolate = var_list(i) + ki*(var_list(i+1)-var_list(i))
    end if
  end function get_interpolate
end module turbine
```