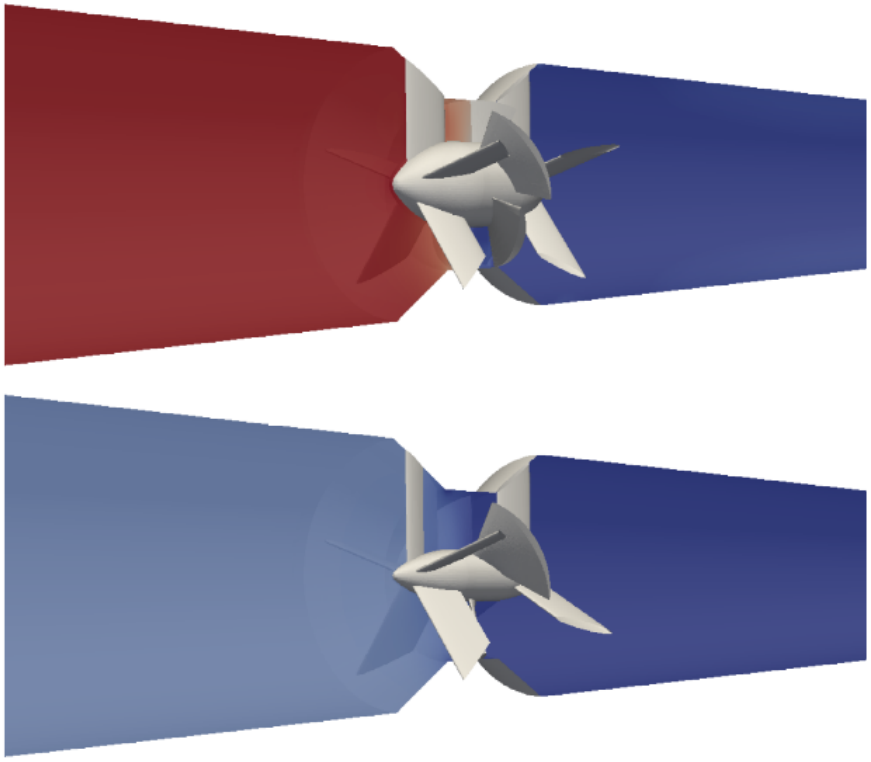
CHALMERS
UNIVERSITY OF TECHNOLOGY



Automated CFD Optimisation of a Small Hydro Turbine for Water Distribution Networks

Master's thesis in Applied Mechanics

Alexander Ohm and Hafþór Örn Pétursson

Department of Applied Mechanics
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2017

Master's thesis 2017:38

Automated CFD Optimisation of a Small Hydro Turbine for Water Distribution Networks

ALEXANDER OHM AND HAFÞÓR ÖRN PÉTURSSON

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Applied Mechanics
Division of Fluid Mechanics
Chalmers University of Technology
Gothenburg, Sweden 2017

Automated CFD Optimisation of a Small Hydro Turbine for Water Distribution Networks
ALEXANDER OHM AND HAFÞÓR ÖRN PÉTURSSON

Supervisor: Håkan Nilsson, Chalmers University, Department of Applied Mechanics
Supervisor: Martin Holm, Undisclosed company
Examiner: Håkan Nilsson, Chalmers University, Department of Applied Mechanics

Master's Thesis 2017:38
Department of Applied Mechanics
Division of Fluid Mechanics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Pilot design and final design after optimisation, in a mean kinematic pressure field spanning $0 \leq P/\rho \leq 2.25$.

Typeset in LaTeX
Gothenburg, Sweden 2017

Automated CFD Optimisation of a Small Hydro Turbine for Water Distribution Networks

ALEXANDER OHM and HAFÞÓR ÖRN PÉTURSSON
Applied Mechanics
Chalmers University of Technology

Abstract

Leakages in water distribution networks is a global problem, with leakage rates up to 50% of the flow [1]. One way to find leakages is to adopt more sensors in the network. Such sensors need electric power that is commonly not distributed with the network. An option is to generate the electricity where it is needed, using small hydro turbines that extract energy from the flowing water and produces just enough electric power for the sensor at each site. The present work provides an automated optimisation procedure for the design of such turbines, and employs it to optimise a turbine with respect to efficiency and kinematic pressure drop at a desired operating point.

The automated optimisation loop handles the entire procedure of geometry creation, meshing, simulation, post-processing and optimisation using open source software. The geometry generation and solution procedure is based on a the work by Bergqvist [2], who implemented a Ruby code to generate a parameterised turbine geometry, generated the mesh with cfMesh, and used OpenFOAM to simulate the flow. The present work extends the method by using Dakota to create an automated loop for optimisation, and applies it for water network turbines.

The optimisation is done by varying the pitch, number of rotor and stator blades, as well as the hub radius. A steady-state frozen-rotor approach with multiple reference frames (MRF) is used in the simulations. The turbulence is modelled with the $k-\omega$ SST turbulence model. After convergence studies of the pilot design, the simulations are carried out on unstructured grids with $2-3$ million cells for the varying geometries.

To reduce the computational burden that direct optimisation with CFD simulations implies, a surrogate modelling technique is implemented using the Efficient Global Optimisation (EGO) algorithm [3] [4]. For this, Kringing modelling is used to create a response surface, and division of rectangles (DIRECT) to optimise it and find infill points for further evaluation. 51 sample points, generated with latin hypercube sampling, are simulated before the first abstraction of the objective function is created. After the learning period, 61 additional design iterations are simulated before the algorithm reaches the stopping criteria. In between each of the 61 optimisation designs, a surrogate optimisation is carried out to select the next infill point.

The completed optimisation provides a new design that performs at 4.4 percentage points higher efficiency and with 72% lower kinematic pressure drop. The new design features a bigger flow through area and a lower work output that is more

adapted to the application with a reduction of 70%.

Keywords: Optimization, Turbomachinery, Hydro Turbine, OpenFOAM, Dakota, cfMesh, Ruby.

Acknowledgements

This project has been a valuable lesson and at certain times it proved to be tough. First and foremost, we would like to thank our supervisors Håkan and Martin for their input that made this project possible. We would also like to thank our better halves for their equally important support through all our ups and downs. Thank you also to Niklas and the senior staff at the company, for pushing us towards our goal and for showing interest in our progress. The very helpful PhD students at the Department of Applied Mechanics also deserve credit for their important tips and feedback and for putting up with our questions. Last but not least, we would like to thank Marcus, Karl and Fredrik for brightening up our days at the office.

Thank you!

Alexander Ohm, Hafþór Örn Pétursson Gothenburg, June 2017

Nomenclature

Abbreviation

CFD	Computational Fluid Dynamics
EGO	Efficient Global Optimisation
EIF	Expected Improvement Function
MRF	Multiple Reference Frames
RANS	Reynolds Averaged Navier Stokes
URANS	Unsteady Reynolds Averaged Navier Stokes

Greek Symbols

α	Absolute flow angle
β	Relative flow angle
ϵ_{ijk}	Permutation tensor
η	Efficiency
γ	Stagger angle
ν	Kinematic viscosity
ν_t	Turbulent kinematic viscosity
Ω	Angular velocity
ω	Specific dissipation
ϕ	Angle of attack
ρ	Density
ε	Dissipation

Latin Symbols

$\overline{u}$	Mean velocity
h_b	Hub axial length from leading edge
H_E	Hydraulic head
L_b	Blade chord length
N_{blades}	Number of blades
p	Instantaneous static pressure
P_H	Hydraulic power
P_t	Power by turbine
u	Instantaneous velocity
u'	Fluctuating velocity
u'_{RMS}	root-mean-square of the turbulent velocity fluctuation
U_∞	Free stream velocity
u_{Ai}	Absolute velocity in stationary reference frame
u_{Ri}	Relative velocity in rotational reference frame
w	Weights
w_b	Tangential chord length

Z	Zweifel number
b	Axial chord length
c	Absolute velocity
g	Gravitational acceleration
hr	Hub radius
I	Turbulence intensity
k	Turbulent kinetic energy
l	Turbulent length scale
LB	Lam-Bremhorst
LS	Launder-Sharma
M	Moment
Q	Volumetric flow rate
r	Radius
rb	Number of rotor blades
Re	Reynolds number
rp	Rotor pitch
s	Blades spacing
sb	Number of pre-swirl stator blades
sp	Pre-swirl stator pitch
U	Blade speed
w	Relative velocity

Other Symbols

i, j & k	Coordinate directions in space
x & θ	Representing axial and tangential directions respectively

Contents

1
Introduction

Fresh water is part of people's everyday life and in many parts of the world its availability in the tap is taken for granted. As global development continues, water demand increases. At the same time increasing water depletion is predicted [5]. This means that water should be used wisely; leakages in the water distribution systems is however, a global problem with leakage rates between 3%-50% of the flow [1].

There are several methods to detect leakages in the distribution pipe networks. Eyuboglu et al. [6] divided the procedures of detecting leakages into three categories; biological, hardware and software based methods. Biological methods detect leakages via on-checks, using human and animal senses, such as sight, hearing or sense of smell. Hardware methods uses tools, such as sensors, to measure any changes occurring in or around the leaking pipe, such as unwanted presence of water. Lastly, software methods analyse flow differences in the pipe system, such as mass equilibrium, which may indicate leakages.

One way to improve leakage detection is to introduce more sensors into the systems. These devices need energy and common solutions today are to provide energy with batteries or cables connected to the grid. Replacing batteries in remote locations or digging for cables in city environments is expensive and thereby limits the affordable number of sensors in the network. To make energy more available, a hydro turbine that provides the necessary energy directly from the water flow at the site is proposed by an undisclosed company. If many installations of such turbines are introduced into a connected network the downstream pressure influences from each unit adds up which could affect the function of the system. Contrary to most turbine applications, it is therefore important to limit the static pressure drop over the turbine for this kind of application. Thus, many established design principles do not apply and the development of the turbine is therefore an exploratory process, characterised by trial and error. Such processes often benefit from replacing physical tests with simulations. Furthermore, optimisation can be added to the development process to aid with finding the right turbine geometry suitable for fresh water pipelines.

Computational fluid dynamic (CFD) simulations has been the subject of much research to better understand turbine flows [7][8][9]. There are several different simulation methods to simulate rotor stator interactions with CFD, each with its own drawbacks and advantages. Three popular approaches are evaluated here; the frozen rotor-, mixing plane - and sliding grid approach.

Frozen rotor. The frozen rotor approach is a steady state method where the rotor does not rotate physically. It is kept fixed and the rotation is applied to the fluid through a source term in the momentum equation. This is done by splitting the domain into multiple reference frames or MRF regions. A rotating reference frame for the rotating part of the fluid and a stationary reference frame is used for the rest of the domain. Interactions between rotating and stationary regions which cause unstable flows do not take place as no real rotation is present. Thus, as it is a steady state method, the results only show the flow at an instant of the rotation and do not accurately represent the real unsteady flow. However, this method is not computational expensive and gives reasonable estimates of the flow behaviour. These traits have made the frozen rotor approach popular in early design phases of hydro turbines [8].

Mixing Plane. The mixing plane method is another steady state method where the flow is divided into different sections. Averaging is utilised on the interfaces of these sections to update the boundary conditions between the stationary and rotating sections. This method has the downside of not being able to predict the wakes generated by the blades due to the averaging. It is however more physical than the frozen rotor approach and can yield better results depending on the parameters of interest [8].

Sliding Grid. The sliding grid method is an unsteady method where the rotor moves in the domain. The real rotation of the flow is simulated by moving the mesh grid at the rotating region. The interaction which is not accounted for in the previous methods, is thus better predicted. As such it is more physical than the previously mentioned methods, but also computationally expensive [8].

Kaufmann and Bertram [7] compared the frozen rotor with a sliding grid approach and found good correlation for propeller efficiency when simulating surging forward motion of ships. Dubas [9] optimised a rim driven thruster and states that steady state based approaches show a good trade-off between computational cost and result accuracy. Because of its ability to handle varying geometries and its cheap computational cost, the frozen rotor seems to be a viable option.

In recent studies the so called evolutionary algorithms such as Genetic algorithms, non-dominant sorting genetic algorithm II and multi-objective genetic algorithm, have been used in combination with CFD [10][11][12][13][9]. All these methods explore the entire design space, to find a global optimum to the problems. Mohamed et al. [11] optimised an airfoil shape of a Wells turbine, using an evolutionary algorithm. His method managed to increase the work output by the blades by 11.3% as well as increase the efficiency by 1%. Using eleven variables describing the airfoil shape, it took 615 evaluations to reach an optimum. Öksüz and Akmandor [13] did an aerodynamic optimisation study on gas turbine blades. They used a modified multi-objective genetic algorithm combined with surrogate modelling to speed up the convergence towards an optimum. They concluded that the modified algorithm in association with surrogate model is a promising tool. Dubas [9] did an

optimisation study on a rim driven thruster, where he, like Öksüz and Akmandor, used a genetic algorithm in combination with a surrogate model to optimise the design. Generally, evolutionary algorithms are fast in finding prominent areas, their convergence to an optimum can however be slow [14]. The combination of a global optimisation tool with a surrogate model is therefore a favourable option when computational resources are scarce.

1.1 Aim of the project

The present work is carried out at an undisclosed company where a pilot design of a turbine for the drinking water network had been developed, see pilot design in Figure 1.1, where main components are shown. The goal is to develop an automated optimisation method that requires minimal manual input, and to apply it to improve the turbine designed by the undisclosed company. This is done by modifying the rotor blades, stator blades and the hub of the turbine, marked in Figure 1.1. Preferably the method should be cheap, both in terms of cost and computational resources. An optimisation approach based on open source software that can run and complete on a single workstation computer within a practical amount of time is therefore sought. The method should also be flexible such that it can be used in the future to carry out similar optimisation of new problems.

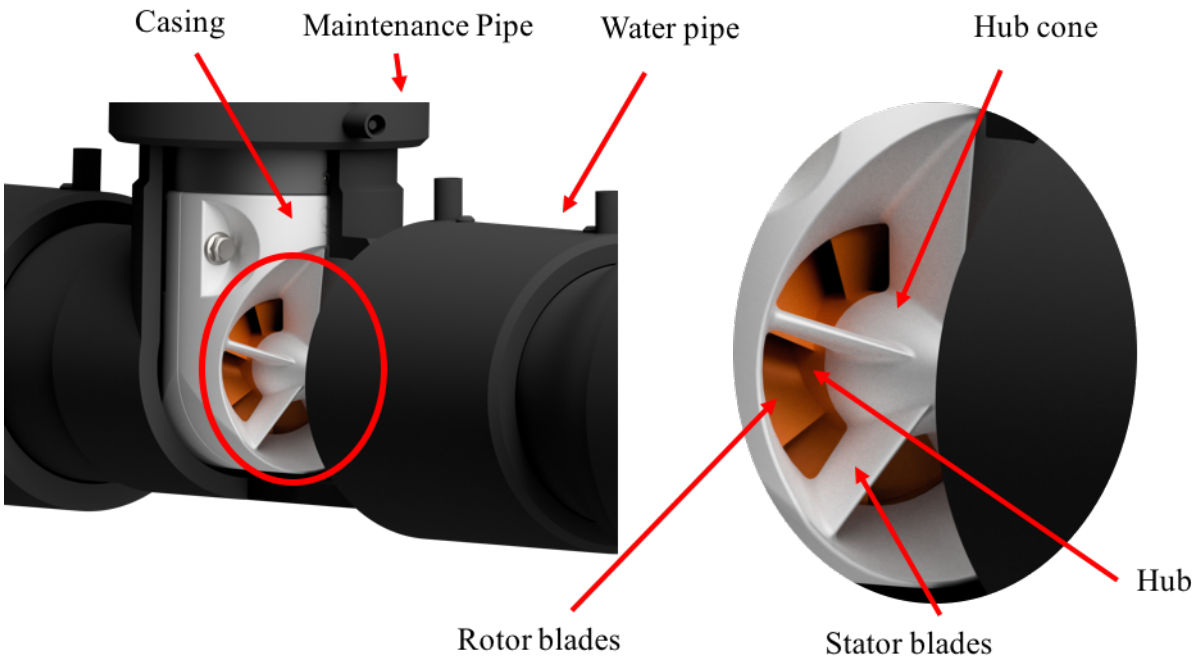


Figure 1.1: The pilot design, with markings of important components. Artist: Martin Holm [15]

1.2 Method Outline

The optimisation method is divided into four steps, illustrated in in Figure 1.2. First, geometry is created, then a mesh grid is made using the geometry from previous step. CFD simulations are carried out on the grid and lastly, the results are evaluated in an optimisation algorithm and a new design is proposed.

At the Gothenburg Region OpenFOAM User Group Meeting in 2015, Bergqvist [2] shared a tutorial case that creates and simulates a turbine from a script. He created geometries with the code language Ruby and used the CfMesh and OpenFOAM software to mesh and simulate the flow around them. He suggested that an optimisation software would be added to his loop to close the chain and form a fully automated loop.

In the present work, the mathematical functions from Bergqvist's [2] Ruby scripts are applied to create the geometry of the pilot turbine design from the company in a fully parameterised manner, with coded variables that can be used to modify the design during optimisation. The software CfMesh is then used to create a mesh to discretise the fluid volume. The OpenFOAM software is used to simulate the fluid flow with the frozen rotor approach because of its ability to handle varying geometries and its cheap computational cost. The software Dakota is used for the optimisation. With the goal of developing a computationally cheap method in mind, a surrogate optimisation technique is sought, similarly to the one used by [9]. The available derivative-free global method with surrogate modelling, the Efficient Global Optimisation (EGO) algorithm, is selected. It differs from the referenced papers in that it uses a sampling technique instead of an EA to optimise the response function of the surrogate model.

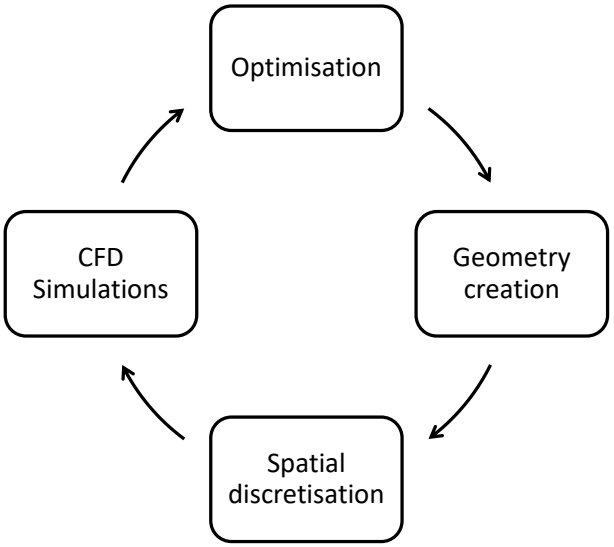


Figure 1.2: Overview of the optimisation process

1.3 Thesis Outline

This thesis is divided into 6 chapters. After this introduction, chapter two describes the theoretical background relevant for the present work. In chapter three, the workings of the developed loop are described. Chapter four presents the test that are carried out to verify the loop. Chapter five then show the results that are generated from applying the method to improve the pilot turbine. Lastly, chapter six ends with conclusions and suggestions for the future.

2
Theory

2.1 Turbine Performance

Theory surrounding turbomachinery is a wide topic with many different kind of applications. The turbine considered in this present work is a hydro turbine which converts flow of water into usable electric energy. Turbines are often evaluated based on several parameters, the ones of concern in the present work are shown and discussed in what follows.

An important quantity for most turbomachines is hydraulic efficiency. It quantifies the percentage of the available energy stored in the flow that can be subtracted and later turned into electrical energy. The hydraulic efficiency of a turbine is more precisely defined as the ratio between the power output of the turbine,P_t, and the hydraulic power of the flow, P_H, as [16],

$$\eta = \frac{P_t}{P_H}. \tag{2.1}$$

The power of the turbine is expressed as

$$P_t = \Omega \Sigma M, \tag{2.2}$$

and the hydrolic power of the flow as

$$P_h = Q\rho g(H_1 - H_2), \tag{2.3}$$

where Ω is the angular velocity, ΣM is the sum of all moments acting on the turbine around the rotational axis, Q is the volumetric flow rate, ρ is the fluid density, g is the gravitational constant and H_1 and H_2 is the hydraulic head in the flow before and after the turbine, respectively. The specific hydraulic energy term, gH, is the available energy in the flow. Dixon [17] describes this as the sum of pressure, kinetic and potential energy and Stoessel [16] shows that it can be expressed as

$$gH = \frac{p}{\rho} + \frac{u^2}{2} + gz, \tag{2.4}$$

where p is the static pressure u is the velocity magnitude and z an elevation between a reservoir and the inlet of the turbine.

Equation 2.4 can be rewritten as an expression of the total pressure p_0 as [18]

$$p_0 = p + \frac{1}{2}\rho u^2 + gz\rho = g\rho H. \tag{2.5}$$

By combining equations 2.1, 2.3 and 2.5, the hydraulic efficiency can be written in terms of the total pressure drop before and after the turbine as

$$\eta = \frac{\Omega \Sigma M}{Q \Delta p_0}. \tag{2.6}$$

In the present work, the total pressure is calculated at the inlet and outlet of the pipe enclosing the system.

Often, in turbomachinery analysis, it is desirable to rank or compare different turbines regardless of their sizes and configuration. This can be done by using dimensional analysis to create dimensionless quantities that describe the ratios between different relevant properties. One such number that is used in the present work is the specific speed, which can be used to evaluate what type of turbomachine is suitable for a given application. It is defined as [17]

$$\Omega_s = \frac{\Omega Q^{\frac{1}{2}}}{(gH^{\frac{3}{4}})}. \tag{2.7}$$

The turbine in this task can be identified as a propeller turbine, which according to Dixon [17] typically operates in the specific speed range of $\Omega_s \approx 2$ to 4.

2.2 Blade Design

At an early design phase, so called velocity triangles, shown in Figure 2.1, are commonly used to design the turbine blades. The analysis matches the flow angles and magnitudes to the geometry angles on the leading and trailing edges of all blades. Velocity triangles are drawn in the circumferential plane and cut through all blades at a constant radius. Analogue triangles can be drawn for every radius. By assuming a constant axial flow and that the flow occurs at a constant radius, the flow angles between the stations can be used to find the desired blade angles. The flow given by this analysis is not entirely representative since, in reality there is a radial flow and the flow does not perfectly follow the blade angles.

The work output in relation to the flow can be expressed with the Euler equation as [17]

$$\dot{W}_t = \dot{m}\Delta(Uc_\theta), \tag{2.8}$$

where $\dot{m}$ is the mass flow U is the blade speed calculated as $U = r\omega$, c is the absolute flow velocity and the index θ indicates the tangential direction. One way to increase the work output of a turbine is thus to increase the tangential redirection of the flow, Δc_θ. This can be accomplished by adding a pre-swirl in the flow by turning the first stator row. The stators themselves only introduce losses in the flow and do

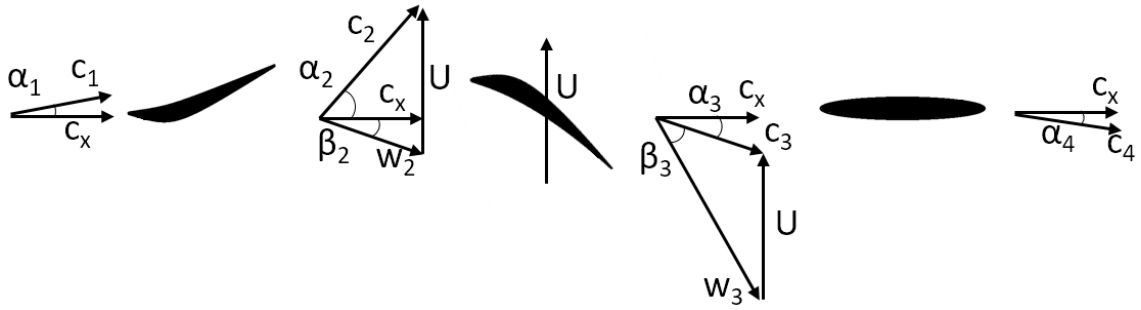


Figure 2.1: Velocity triangles for analysing the flow where c is the absolute velocity, w is the relative velocity, U is the blade speed, α is the absolute flow angle and β is the relative flow angle. Index 1 indicates the station upstream of the pre-swirl stator row, index 2 the station between the pre-swirl stators and the rotor, index 3 the station between the rotor and the third row, index 4 the station downstream of the third row and index x the axial direction.

not produce any work, but combined with a proper rotor design the total output can be increased. Amin and Xiao [19] managed to increase the overall efficiency of an open water horizontal axis turbine by 13% by introducing pre-swirl.

2.2.1 Zweifel Number

The most suitable number of rotor or stator blades for a certain application is not an obvious choice. The more blades there are, the better the alignment with the flow becomes since it is guided more precisely. However more blades also means more surface area which generates friction losses. The so called Zweifel criteraion is commonly used to evaluate this trade off and by assuming an incompressible flow and a constant axial velocity, can be expresseed as

$$Z = 2\frac{s}{b}cos^2\alpha_2(tan\alpha_1 + tan\alpha_2), \tag{2.9}$$

where s is the spacing between the blades and b is the axial cord length. Through experiments it has been found that a value of $Z \approx 0.8$ is optimal and hence an optimal value for s/b can be found given the flow angles [17].

2.3 Governing Equations

The famous Navier-Stokes equation, here given for an incompressible fluid, govern the momentum of viscous fluids as

$$\frac{Du_i}{Dt} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu\frac{\partial^2 u_i}{\partial x_j \partial x_j} + f_i, \tag{2.10}$$

where p stands for static pressure, u for velocity and f is the body force. It is derived from the conservation of mass, momentum and energy and together with the continuity equation

$$\frac{\partial u_i}{\partial x_i} = 0, \tag{2.11}$$

the equation set can describe the motions of fluid flows.

By decomposing the velocity into a mean flow component and an instantaneous fluctuation, $u = \overline{u} + u'$, and then applying a time average, the Reynolds averaged Navier-Stokes (RANS) equations are obtained as, derivation shown by Davidson [20],

$$\frac{\partial \overline{u}_i \overline{u}_j}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \overline{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left[\nu \frac{\partial \bar{u}_i}{\partial x_j} - \overline{u_i' u_j'}\right] + f_i. \tag{2.12}$$

In this process a new unknown term, the Reynolds stress tensor $\overline{u_i' u_j'}$, appears. The Reynolds stress tensor, makes the equation set unsolvable without introducing further assumptions or modelling equations for closure. One way of modelling the Reynolds stress tensor is by applying the Boussinesq assumption [20],

$$\overline{u_i' u_j'} = -\nu_t \left(\frac{\partial \overline{u}_i}{\partial x_j} + \frac{\partial \overline{u}_j}{\partial x_i}\right) + \frac{1}{3}\delta_{ij}\overline{u_k' u_k'}, \tag{2.13}$$

where ν_t is the turbulent viscosity and $\overline{u_k' u_k'} = 2k$.

In the present work, a turbine is solved using the frozen rotor approach, as introduced in Chapter [**?**]. The formulation of Petit et al. [21] showed the governing RANS equations implemented in the MRF approach. The equations do not have the Reynolds stress term, and thus only apply for steady flows as

Continuity equation

$$\frac{\partial u_{Ai}}{\partial x_i} = 0 \tag{2.14}$$

Momentum equation

$$\frac{\partial u_{Ri} u_{Aj}}{\partial x_i} + \Omega_i u_{Aj} \epsilon_{ijk} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu \frac{\partial^2 u_{Ai}}{\partial x_j \partial x_j}, \tag{2.15}$$

where the ϵ_{ijk} is the permutation tensor and u_{Ai} is the absolute velocity in the stationary frame and u_{Ri} is the relative velocity, formulated as

$$u_{Ri} = u_{Ai} - \Omega_i r_j \epsilon_{ijk}. \tag{2.16}$$

If the turbulence is included, using the Boussinesq assumption and using same derivation assumptions as used in the development of MRF [22], the momentum equation 2.17 can be formulated as

$$\frac{\partial \overline{u}_{Ri} \overline{u}_{Aj}}{\partial x_i} + \Omega_i \overline{u}_{Aj} \epsilon_{ijk} = -\frac{1}{\rho}\frac{\partial \overline{p}}{\partial x_i} + \frac{\partial}{\partial x_j}\left[(\nu + \nu_t)\left(\frac{\partial \overline{u}_{Ai}}{\partial x_j} + \frac{\partial \overline{u}_{Aj}}{\partial x_i}\right)\right]. \tag{2.17}$$

2.4 Turbulence Models

It is generally known that most flows in nature are turbulent. Turbulent flows are seemingly chaotic since they are full of disturbances in the form of swirling structures. These structures are called turbulent eddies and exist at a wide variety of length scales. The biggest scales are of the same size as the biggest features of the flow. This is where energy is extracted from the mean flow by the turbulence. It is then transported through what is known as the cascade process to smaller and smaller scales, ultimately down to the smallest scale where the energy is dissipated as heat [23]. Turbulent flows are characterised by high Reynolds numbers, which are dimensionless numbers that describes the ratio between inertial and viscous forces in the flow. An example is that turbulence starts to build up in a pipe for Re $\geq$ 2000, and it undergoes a transition from laminar to turbulent until Re $= 10^5$, above that limit the flow is considered to be fully turbulent [24]. For the application in the present work, the Reynolds number in the pipe is about 3.5×10^4.

Flow structures smaller than the spatial discretisation, the mesh, cannot be resolved and thereby not simulated. Since the dissipative turbulent scales are extremely small compared to most applications relevant for engineers, properly simulating turbulence is often prohibitively difficult as it poses extreme requirements on the resolution. A different approach to deal with turbulence in simulations is to model it numerically. There are many differnt types of turbulence models [23], each of them are appropriate for different flow situations and come at different computational costs.

The model most widely used in industry is the is the $k - \varepsilon$ model. A reason for its popularity is its ability to model a wide range of flows [24]. It can however be inaccurate when simulating adverse pressure gradient flows [20] or flows which rotate, have curved boundaries or swirl [24]. The $k - \varepsilon$ model is a high Reynolds number model and as such it is meant to be used with high Reynolds number wall functions, meaning that the near wall flow is modelled by typical behaviour. Modifications to the model has been made in order to further account for the effects of walls. These models are so called low Reynolds formulations for the $k - \varepsilon$ model. The $k - \varepsilon$ models are discussed by Davidson [23] and Moradnia [25].

Another popular model is the shear stress transport model, the $k - \omega$ SST model. This model implements the $k - \varepsilon$ model in regions far from the wall and the $k - \omega$ model, discussed by Davidson [23], close to the wall. The $k - \omega$ model is known to be more accurate than the $k - \varepsilon$ model in the near wall region [26] and is better at solving adverse pressure gradient flows [20], which occurs in a flow around wing profiles. The ε and ω equations are combined into one transport equation by the expression $\varepsilon = \beta^* k\omega$.

The $k-\omega$ SST model shown by Menter et al. [26] is mainly used in the present work. Some of the coefficients are expressed with the same notation as in OpenFOAM, shown by Pirooz [25], but with a correction to the pressure term according to NASA [27]. The transport equations for k and ω are formulated as

$$\frac{\partial k}{\partial t} + \frac{\partial \bar{u}_i k}{\partial x_i} = min(P_k, 10\beta^* k\omega) - \beta^* k\omega + \frac{\partial}{\partial x_j}\left[(\nu + \alpha_\omega \nu_t)\frac{\partial k}{\partial x_j}\right] \tag{2.18}$$

$$\frac{\partial \omega}{\partial t} + \frac{\partial \bar{u}_i \omega}{\partial x_i} = \frac{\gamma}{\nu_t} min(P_k, 10\beta^* k\omega) - \beta\omega^2 + \frac{\partial}{\partial x_i}\left[(\nu + \alpha_\omega \nu_t)\frac{\partial \omega}{\partial x_i}\right] + 2(1 - F_1)\alpha_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i}, \tag{2.19}$$

where the production term P_k , which appears in both the k (2.18) and ω (2.19) equations, is calculated as

$$P_k = \nu_t\left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i}\right)\frac{\partial \bar{u}_i}{\partial x_j}, \tag{2.20}$$

where the turbulent viscosity ν_t is given by

$$\nu_t = \frac{a_1 k}{max(a_1\omega, SF_2)}. \tag{2.21}$$

F_1 in Equation 2.19 and F_2 in Equation 2.21 are blending functions written as

$$F_1 = tanh\left\{\left\{min\left[max\left(\frac{\sqrt{k}}{\beta^*\omega y}, \frac{500\nu}{y^2\omega}\right), \frac{4\rho\alpha_{\omega 2}k}{CD_{k\omega}y^2}\right]\right\}^4\right\} \tag{2.22}$$

$$F2 = tanh\left[\left[max\left(\frac{2\sqrt{k}}{\beta^*\omega y}, \frac{500\nu}{y^2\omega}\right)\right]^2\right]. \tag{2.23}$$

CD_{kw} in Equation 2.22 is defined as

$$CD_{kw} = max\left(2\rho\alpha_{w2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i}, 10^{-10}\right), \tag{2.24}$$

note the similarities with the fourth term in the ω equation (2.19). The equation constants are given by

$$\begin{array}{llll} \alpha_{\omega 1} = 0.5 & \alpha_{\omega 2} = 0.856 & \alpha_{k1} = 0.85 & \alpha_{k2} = 1 \\ \beta^* = 0.09 & \beta_1 = 0.075 & \beta_2 = 0.0828 & \\ \gamma_1 = 0.55\bar{5} & \alpha_2 = 0.44 & & \end{array}$$

Blending takes place between $k - \omega$ and $k - \varepsilon$ through the quantity F_1, which according to Menter et al. [26] is zero away from the wall where $k - \varepsilon$ model is used and one in the boundary layer where $k - \omega$ is used. The constants are blended as

$$\psi = \psi_1 F_1 + \psi_2(1 - F_1), \tag{2.25}$$

where ψ is an arbitrary constant.

2.5 Turbulence Quantities

The turbulence intensity, I, of a flow is defined by Russo and Basse [28] as

$$I = \frac{u'_{RMS}}{U_\infty}, \tag{2.26}$$

where $u'_{RMS} = \sqrt{\frac{1}{3}\overline{u'_i u'_i}}$ is the root-mean-square of the turbulent velocity fluctuations and U_∞ is the free stream velocity.

The scaling of turbulence intensity with Reynolds number is investigated by Russo and Basse [28] for compressible and incompressible flows in smooth pipes using both measurements and simulations. The results are fitted to the power-law expression

$$I = aRe^b, \tag{2.27}$$

where different values for the constants a and b are recommended for different flows.

The turbulent kinetic energy is defined as [20]

$$k = \frac{1}{2}\overline{u'_i u'_i}. \tag{2.28}$$

By combining equations 2.26 and 2.28 the turbulent kinetic energy can be expressed as

$$k = \frac{3}{2}(IU_\infty)^2. \tag{2.29}$$

The dissipation, ε, can then be expressed as

$$\varepsilon = \frac{k^{\frac{3}{2}}}{l}, \tag{2.30}$$

where l is the turbulent length scale [23].

The specific dissipation, ω, is then estimated from [20]

$$\omega = \frac{\varepsilon}{C_\mu k}, \tag{2.31}$$

where the value of the constant $C_\mu = 0.09$.

The Boussinesq assumption allows the Reynolds stresses to be modelled with a turbulent viscosity. It has the same dimensions as the laminar viscosity, and from dimensional analysis the turbulent viscosity can be expressed as [20]

$$\nu_t = C_\mu \frac{k^2}{\epsilon}. \tag{2.32}$$

2.6 Wall Treatment

The height of the first cell in the wall-normal direction needs to be considered when using turbulence models. The best way to capture the wall effect accurately would be to simulate the whole boundary using small cells close to the wall. This is not always practical with respect to computational time. Wall functions are thus introduced. They estimate the flow close to the wall by assuming that it behaves like a fully developed turbulent boundary layer. Using this formulation, a coarser grid close to the wall can be used, and as a result, less computationally expensive simulations are achieved [23].

A suitable wall mesh grid refinement is found by analysing the dimensionless quantity, y^+. This quantity is used to describe the mesh resolution in the vicinity of the wall [29]. The boundary layer can be divided in three main regions; viscous region, buffer region and logarithmic region. According to Davidson [20], the viscous region spans $y^+ \leq 5$ and the logarithmic region $y^+ \geq 30$. In between those two regions is the so called buffer region ($5 \leq y^+ \leq 30$).

According to literature, when resolving the boundary, the most appropriate value is $y^+ \approx 1$, while when applying wall functions it should be around $y^+ \approx 30$ for high Reynolds models, [29], and below $y^+ = 5$ for low Reynolds models [30].
The y^+ of the cell closest to the wall is calculated as

$$y^+ = \frac{y u_\tau}{\nu}, \tag{2.33}$$

where y is the distance from the wall and u_τ is the friction velocity [20].
The friction velocity can be expressed as

$$u_\tau = \left(\frac{\tau_w}{\rho}\right)^{1/2}, \tag{2.34}$$

where τ_w is the wall shear stress. The general expression for the wall shear stress is

$$\tau_w = \mu \left.\frac{\partial \bar{u}}{\partial y}\right|_w, \tag{2.35}$$

where the subscript w denotes the wall. The y^+ can only be evaluated in the presence of a flow field at a post processing stage. It is however good practice to have an initial estimate of the cell height y suitable for the flow. It is estimated by calculating the y^+, using a formula for the wall shear stress. The wall shear stress is then expressed as [31]

$$\tau_w = \frac{1}{2} C_f \rho U_\infty^2, \tag{2.36}$$

where C_f is the skin friction. There are several ways of estimating the walls skin friction, one is to use the Schlichting relation for a turbulent boundary layer over a flat plate where [32]

$$C_f = \frac{0.455}{(log_{10}Re)}. \tag{2.37}$$

As this is only an estimate, a good mesh grid convergence study is needed to find the appropriate cell height at the wall for each application.

2.7 Optimisation

The goal with the deliberate design of an application is often to maximise or minimise some relevant aspect of it, which can be called the objective. Alterations can be applied to certain aspects of the design, hereafter called variables, in order to change the performance or functionality of the application with regards to the objective. The objective can thus be described as a function of the variables, an objective function. Often, several aspects of the application are relevant and an objective function that includes all the desired aspects then has to be formulated as a combination of multiple individual objectives. These objectives are often conflicting, like low cost and high performance, and sophisticated methods are often needed to evaluate the impact of any change in the variables on all the objectives. One such method that facilitates complicated design is to apply mathematical optimisation of the objective function. Additionally, there is seldom complete design freedom, as the variables and objectives are constrained by limits, e.g. maximum size, minimum performance and maximum cost.

There are several ways of formulating multi-objective optimisation problems. One way is to use a weighted sum approach. When implementing this method, the objective function is constructed from a combination of weighted individual objectives. The weight determines whether or not one of the objectives should be dominant during the optimisation [14]. If written on the negative-null form, which indicates that the constraints are fulfilled when their function value is negative, the problem can be formulated as

$$\begin{aligned} \text{minimize}: \quad & f(\mathbf{x}) = w_1 f_1(\mathbf{x}) + w_2 f_2(\mathbf{x}) \\ \text{Subject to}: \quad & g_1(x_1, x_2) = f_2(\mathbf{x}) - f_{2,max}(\mathbf{x}) \leq 0 \\ & g_2(x_1) = x_1 - x_{1,max} \leq 0 \\ & g_3(x_1) = -x_1 + x_{1,min} \leq 0, \end{aligned} \tag{2.38}$$

where f is the combined objective function of the individual objectives f_1 and f_2, w_1 and w_2 are the weights, $\mathbf{x}$ is the state vector of the variables $\mathbf{x} = [x_1, x_2, ..., x_n]$ and g_1 to g_3 are the constraints.

For many applications there is not only one optimum, there are often multiple local optimums complementing the global optimum. The maximum or the minimum objective of all local optimum in a design space represents the global optimum, which is further explained in Papalambros and Wilde, [33]. In the present work the global optimum is the one of interest.

As mentioned in Chatper 1, so called evolutionary algorithms are popular in CFD related optimisations. These type of algorithms mimic natural evolution by evaluating a population of points and then combines the most prominent points to create the next generation. They are good at finding truly global optimal values. The most optimal value of an objective function that is constructed from multiple objectives can often be reached by several state configurations, there is a trade-off between the objectives. Since evolutionary algorithms works with entire generations, and not one single point at a time, these algorithms are good for finding the trade-off between objectives. For the same reason, evolutionary algorithms require many function evaluations before converging to the optimal solution and are computationally expensive [34]. A less expensive alternative is to use a surrogate model for the objective function. The true objective function is then approximated with a model function, which is optimised instead. The Efficient Global Optimization (EGO) algorithm is such a method, and is used in the present work.

2.7.1 Efficient Global Optimisation Algorithm

The EGO algorithm was first developed by Jones et. al. [3] and is a derivative-free, global, space filling algorithm that effectively evaluates the design space by refining the division of the variables in areas with high expected improvement and high uncertainty. First, the objective function is evaluated in a number of initial points. Kringing modelling is then applied to approximate the response function based on evaluated points, and its variance based on the spacing of the points. The objective function is modelled by the second degree polynomial

$$f(\mathbf{x}) \approx G(\mathbf{x}) = \beta_0 + \beta_{\mathbf{1}}\mathbf{x} + \beta_{\mathbf{2}}\mathbf{x}^2 + Z(\mathbf{x}), \tag{2.39}$$

where the constants of the vectors β_i are taken as the mean of the responses of previous function evaluations and Z is a Gaussian process that describes the model deviation from the true function [4]. An expected improvement function (EIF) is then formulated based on the current best design $\mathbf{x}^*$ and the predicted values from the Kringing model as

$$EIF(\mathbf{x}) = max(G(\mathbf{x}^*) - G(\mathbf{x}), 0). \tag{2.40}$$

The point which maximises the value of the EIF is selected as the next infill point at which to evaluate the objective function. The Kringing model is then updated with this point and the process restarts. Once the EIF reaches a low enough value, the process stops [4].

2.8 Cavitation

Cavitation is a phenomenon that occurs when the static pressure in a liquid is reduced below the vaporisation pressure. The liquid then vaporises and transforms into gas bubbles. In hydraulic turbines this mainly occurs in vortices or close to the blades, in zones where the flow is accelerated. This can cause severe structural and performance problems [17]. When the gas bubble is transported out of the accelerated zone, it collapses due to the surrounding static pressure. This can cause liquid micro jets and shock waves which excert large forces on nearby solid surfaces resulting in pitting and erosive damage. The bubbles also constrict the liquid flow and thereby affect the performance. Turbines can be designed to take advantage of cavitation, so called super-cavitating turbines, but otherwise the phenomenon should be avoided [17]. In a highly pressurised system, like in the present work, the static pressure does not drop below the vaporisation pressure and cavitation does not occur.

3
Methods

As previously explained in Chapter 1, the main goal of the project is to develop an automated optimisation procedure by coupling together several open source software and to use the method to improve the design of a turbine in fresh water pipelines. A study is carried out in the early stages of the project to determine what software are useful for the task. The method generated in the present work is based on a Ruby code to generate the geometry, CfMesh to generate the mesh grid, OpenFOAM to simulate the flow and Dakota to create a loop and perform optimisation. A schematic representation of the optimisation procedure is shown in Figure 3.1. The loop is initiated by Dakota in the first box where the structure for the rest of the procedure is defined. The algorithm then continues by creating the geometry of the first design with the Ruby scripts. The geometry is then sent to CfMesh that creates a mesh grid of the fluid domain around the current design. OpenFOAM then simulates the flow in the system and quantifies the performance of the turbine. If the optimisation is in the learning phase, this process repeats until the optimisation period is reached. If the learning process is completed the results from the simulations are used to update the response surface which models the system. The surrogate model of the response function is then optimised to find a new infill point. If the performance of the new point is predicted to be an improvement the process repeats, if not, the optimisation is complete. For further information about how the communication between the software is handled, see the folder structure in appendix A.

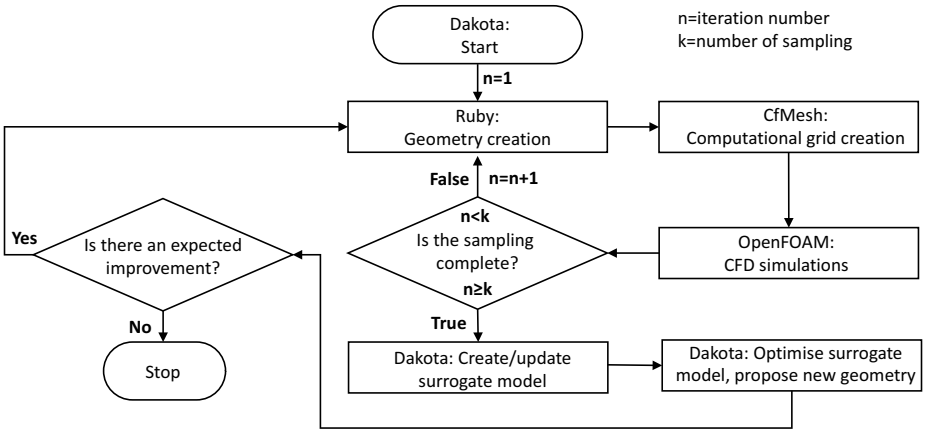


Figure 3.1: Schematic reprentisation of the optimisation loop.

In this chapter, a detailed description of how the task is approached is given, including rigorous discussions of settings and design estimates for each part of the loop.

3.1 Geometry

Studies carried out by Berqvist [2] are used as a guideline and as an initial start to creating the geometry. He published a Ruby code that generate a simple turbine with only a hub and a rotor, as well as CfMesh and OpenFOAM scripts to mesh and simulate the case. The geometry in the present work is created using Ruby scripts where `.stl` files of the surfaces are generated. Scripting the geometry, and thus parametrising it, gives the automated optimisation great control of different variables within the design. Berqvist's [2] Ruby scripts are modified to generate the additional geometries relevant for the current application, where drastic simplifications are made to some geometrical features, compared to the real system. All dimensions of the geometries are provided by the undisclosed company and a model of the pilot design is created. In this section, each step of the creation of the geometry is briefly explained for .

Axi-symmetric objects, seen in Figure 1.1 as water pipe, casing, hub and hub cone, are generated by rotating the objects profile curve around the centre axis. Figure 3.2 shows the modelled axi-symmetric geometrical objects in the current application, and the names of the corresponding patches.

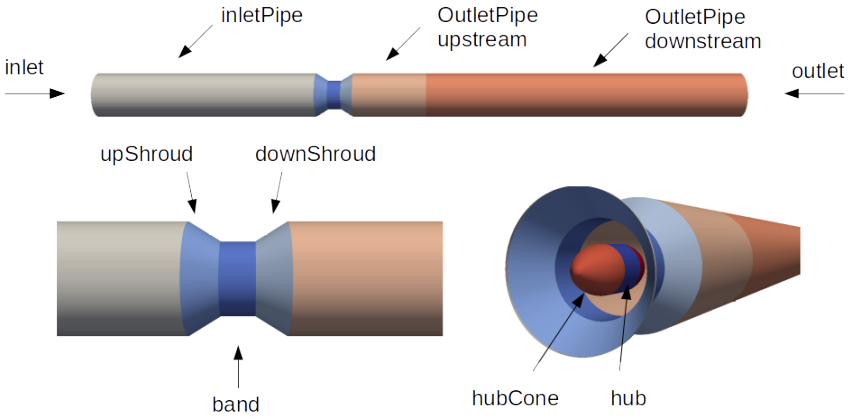


Figure 3.2: Axi-symmetric objects of the geometry.

Note that there are two `hubCone` patches. Both are attached to the hub, one is pointing in the upstream direction, while the other one, not visible in Figure 3.2, points in the downstream direction. Hereafter, these patches will be identified as `upHubCone` and `downHubCone`.

The pilot design features a total of ten stator blades, see Figure 3.3 (left), five in front of the rotor in blade row one, and five behind it in blade row three. Row one and three are identical in the pilot design. The blades act primarily as supports for the hub cones, which carry the rotor axis. The profile of the stator blades is a simple ellipse shown in Figure 3.4a. These blades are created by connecting the blade profiles at the patches, `upHubCone` and `upShroud` for row one and `downHubCone` and `downShroud` for blade row three. Hereafter, these patches will be identified as `upStatorBlades` and `downStatorBlades`.

The rotor of the pilot design, seen in Figure 3.3 (right), consists of five blades that are connected to the patches `hub` and the `band`. The rotor blade profile has a six series NACA airfoil profile named NACA64A410. Similarly to the creation of the stator blades, the rotor blade surfaces are created by connecting the blade profiles between the hub and band. The rotor blade profile can be seen in Figure 3.4b.

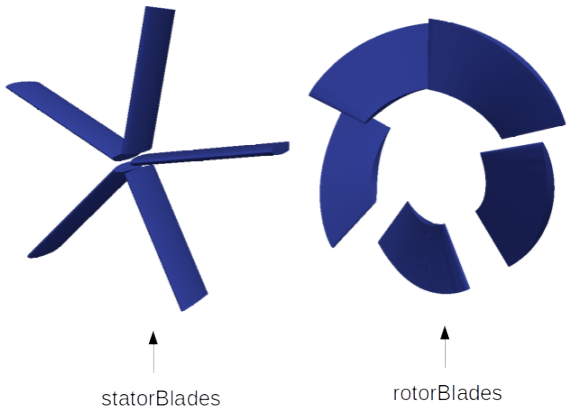
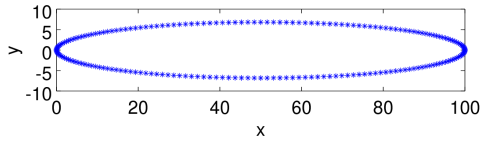
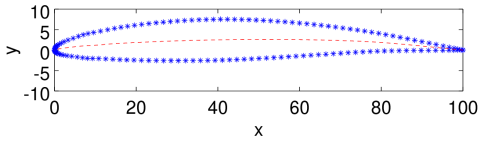


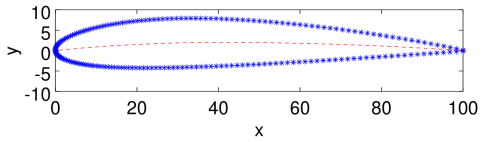
Figure 3.3: Stator blades (left) and rotor blades (right)



(a) Stator blade profile



(b) Rotor blade profile



(c) Pre-swirl blade profile

Figure 3.4: Blade profiles.

Certain modifications to the pilot design are proposed before the optimisation is conducted. As discussed in section 2.2 the efficiency of the turbine can be increased by adding pre-swirl to the flow. Therefore, the first stator row in the pilot design

is modified by swapping the elliptical profile, Figure 3.4a, out for an airfoil profile, Figure 3.4c. The profile is selected based on the work of Gish et al [35]. Out of several blades which they analysed, the NACA 2412 airfoil is selected for the present work, as it handles a high angle of attack before separation, which allows for high pitch angles and thus more pre-swirl. Additionally to the changes to the upstream stator blades, the number of downstream stator blades or supports is reduced. The reason for their existence is only to serve as a structural support. By reducing the number from five to three for all designs the frictional loss due to the profiles is likely to be reduced.

3.1.1 Geometrical Simplifications

The real pilot design features a large amount of details, all of which are not replicated in the model, the real pilot design is shown in Figure 1.1. The real pilot design features a casing, seen in Figure 3.5, which is circular in the vertical direction to allow for installation. This makes a non-axisymmetric casing, which forms narrow passages on the sides between the circular casing and the straight pipe. In these passages, illustrated by Figure 3.6 and also highlighted in Figure 3.5, there is a risk of leakage. Additionally, the stator blades chord length varies in the real pilot design, as the leading edge of four of the blades is tapered.

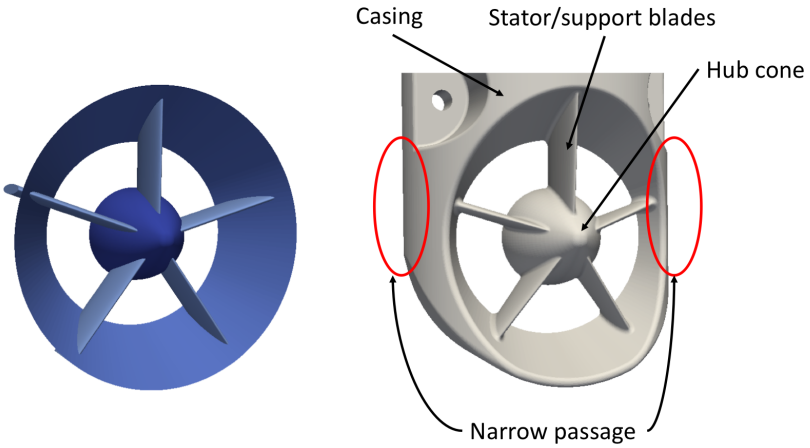


Figure 3.5: Comparison between the modelled pilot design casing (left) and the real pilot design casing (right).

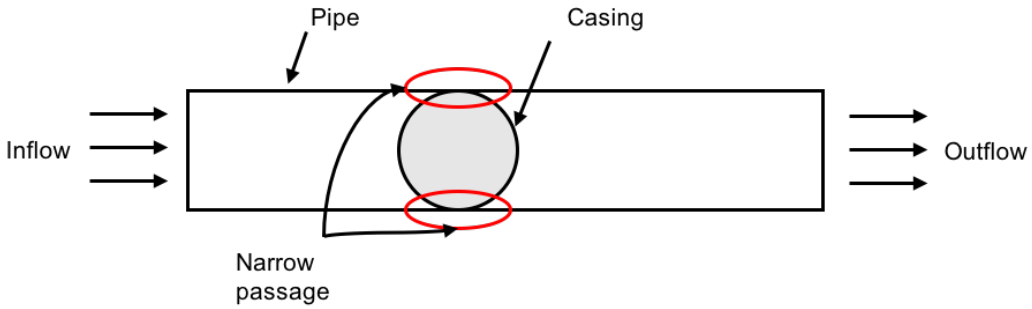


Figure 3.6: Schematic illustration of the real casing in the pipe, seen from above.

In the modelled pilot design, seen in Figure 3.5, the circular shape in the vertical direction is omitted and the casing is thus axi-symmetric. It covers the entire pipe and therefore there are no leakage passages present. The stator blades are all identical and the chord length is set such that the total area of the stator blades is equal to that of the real casing.

The rotor, see Figure 3.7, is replicated without any major differences between the real geometry and the model. The leakage between the shroud and the casing is omitted, and thus only the band of the rotor is modelled. Likewise, the central space for the shaft is not modelled, as no leakage is accounted for.

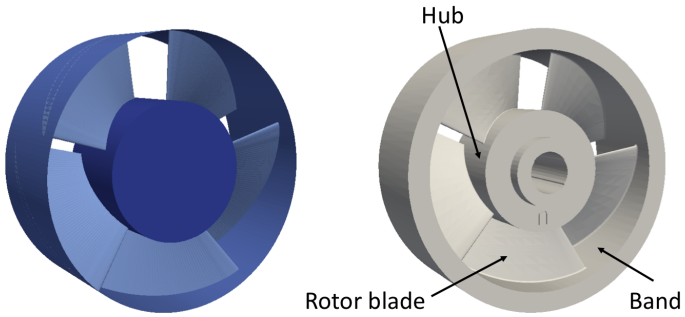


Figure 3.7: Comparison between the modelled pilot design rotor (left) and the real pilot design rotor (right).

3.1.2 Geometrical Variables for Optimisation

After optimisation the design is only more optimal with respect to some variables for which the optimisation is carried out. In the present work, five geometrical features are selected and allowed to vary for optimisation

- Radius of the hub, which also impacts the radius of the hub cones
- Number of pre-swirl stator blades
- Pitch of pre-swirl stator blades
- Number of rotor blades
- Pitch of rotor blades

These variables are chosen as they are the most likely to impact the flow, and have dramatic effects on the engineering quantities. The Ruby geometry creation is made so that it is easy to manipulate these quantities. Each parameter will be discussed further in the remainder of this chapter.

3.1.2.1 Hub

The limitations of the changes to the hub, illustrated in Figure 3.8, are chosen based on the radii of the bearings inside the hub cones. The hub is allowed to vary with 40% of the pilot design hub radius to ensure structural stability is maintained and

that the bearings fit. Any changes to the radius of the hub is also applied to the hub cones. Only the radius is varied, and the length of both the hub and the hub cones is kept constant for all designs.

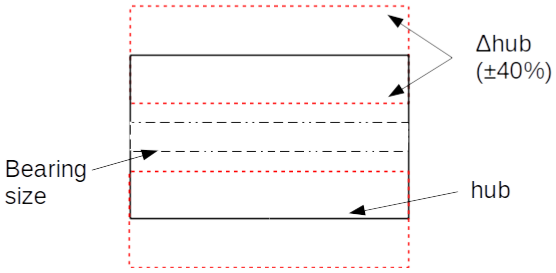


Figure 3.8: Schematic representation of the design space of the hub.

3.1.2.2 Pre-Swirl

As mentioned in section 3.1, the front stator blades are exchanged for pre-swirl blades. The stall angle for the selected pre-swirl blade, NACA2412 airfoil, is 10 degrees and thus the range of the pitch for the new pre-swirl blades is set from $\phi = 0°$ angle of attack, where there is no or little pre-swirl, to the stall angle of $\phi = 10°$. The range for the number of blades is found with the Zweifel criteria based on flow angle estimates. The calculations indicated between two and six blades, but to ensure structural integrity a minimum of three blades is fixed and the interval is set from three to six.

3.1.2.3 Rotor blades

The rotor blade design space is more difficult to define due to both manufacturing constraints and geometrical constraints. The pitch of the rotor blades needs to match the angle of the incoming flow. As other variables change, so does the incoming flow and the rotor pitch is therefore implemented as a pitch modifier. First, the incoming axial velocity is calculated based on the hub radius for the specific design, as it greatly alters the annular area of the flow through the turbine, and the mass flow. An estimate of the rotor pitch is thereafter calculated from velocity triangles using the new axial velocity and the rotational speed. Finally, the optimisation algorithm is allowed to modify the pitch by ±10°.

Due to geometrical constraints, illustrated in Figure 3.9, the blades are not allowed to extend further than the hub length minus a certain margin, which is calculated based on the pilot design. To make sure that the blades fulfilled these requirements, the chord length of the blades is adjusted to the pitch as

$$L_{b1} = \frac{h_b - \Delta_h}{sin(\gamma)}, \tag{3.1}$$

where L_{b1} is the blade chord length, h_b is the distance from the leading edge to the end of the hub, Δ_h is the marginal space which needs to be left clear and γ is

the stagger angle of the blade, either at the hub or the band. The reason for the geometrical constraints are discussed in more detail in chapter 3.3.2.

Due to manufacturing constraints, the blades are not allowed to overlap when viewed from the axial direction, again see Figure 3.9. This is handled by calculating the chord lengths based on number of blades, circumference and the pitch according to

$$L_{b2} = \frac{w_b}{cos(\gamma)} 0.97, \tag{3.2}$$

where L_{b2} is the blade chord length, w_b is the tangential chord and γ is the stagger angle. Note that the 3% clearance is used based on the company's requirements. Since the blades are not allowed to overlap when viewed from the axial direction, the maximum tangential space is calculated as

$$w_b = \frac{2r\pi}{N_{blades}}, \tag{3.3}$$

where N_{blades} is the number of blades. The cord length for each design is taken as the minimum of L_{b1} and L_{b2}, in order to fulfil both constraints.

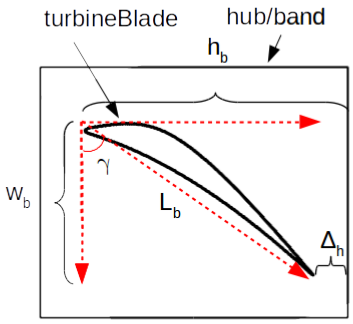


Figure 3.9: Schematic representation of the designspace for the rotor blade.

As with the stator blades, the range for the allowed number of blades is based on estimates of the Zweifel criterion, which indicated about three blades. The range is set to be three-six blades, which also included the pilot design.

3.2 Mesh Grid

Before CFD simulations can be carried out, a mesh where the fluid volume is discretised by a finite number of points, has to be generated. Figure 3.10 shows the turbine enclosed in the pipe system which defines the fluid volume. In the present work, an automated procedure is sought and the software CfMesh is used. As with most automated mesh software, CfMesh produces an unstructured mesh. The software is good at creating simple meshes in a robust way and is able to mesh poorly defined surfaces, such as `.stl` surfaces which are not directly connected node to node, as the blades which intersect axi-symmetric objects in the present work. However, good mesh quality can be difficult to obtain around unconnected surfaces. Attempts were

made to close these surfaces, but no robust method was found. The generated mesh is dominated by hexahedral cells, which ensures good mesh quality in regions that are easy to discretise. In more difficult regions, around complex features such as blades, the mesh quality can suffer.

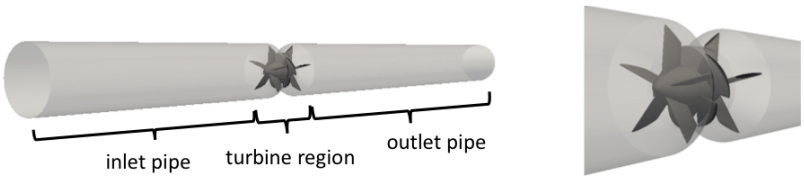


Figure 3.10: Turbine enclosed in the domain (left), with markings showing refinement areas and zoomed in on the turbine (right).

The mesh is adjusted in various regions in the domain by defining desired cell sizes. Both regions defined by spacial coordinates, as well as regions defined by different surface patches are used to guide the automatic mesh process. The domain can be divided into three main parts; the inlet pipe, the turbine region and the outlet pipe, marked in Figure 3.10. In the inlet pipe, the flow is aligned with the pipe and is yet undisturbed by the turbine. This is the coarsest region of the mesh. In the turbine region, the flow interacts with the turbine geometry and undergoes drastic changes. This is the most detailed region of the mesh. In the outlet pipe, the effect of the turbine is still present, as swirl and other flow patters are convected downstream with the flow. The mesh in the outlet pipe is of medium resolution. Each of the three main mesh regions are in turn refined additionally in different sub regions.

Figure 3.11 shows a part of the mesh in the turbine region, where several sub regions are visible. The mesh around the front stator blade is shown on the left in the figure and the mesh around the leading and trailing edges of the rotor blade on the right.

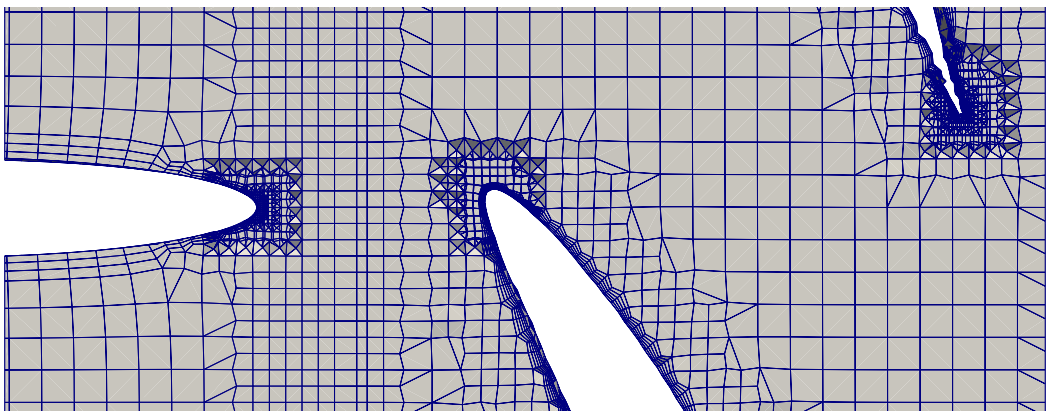


Figure 3.11: Turbine region of the mesh, with several sub regions. The front stator blade of the pilot design is visible on the left, and a leading and trailing edge of the rotor blade on the right.

One of the refinements in Figure 3.11, is a mesh region defined by coordinates in space, between the stator and rotor blades, where a desired cell size is fixed. This is the interface between the rotating and stationary parts of the turbine. Furthermore, several refinement regions are visible around the surfaces of the geometry. Generally, fluid flows close to walls are characterised by high gradients [20]. Therefore, the cells adjacent to the walls, the boundary layer cells, are slender and short in the wall normal direction to resolve these changes. These cells are defined by a desired number of layers, a growth ratio and the maximum first cell height. For surfaces where the flow passes at a high speed, the wall effects are sharper and an additional refinement region is needed between the smaller boundary layer cells and the internal mesh. This refinement can be seen on the rotor blades and is defined by a desired cell size and a refinement thickness away from the wall. Lastly, special treatment is needed to resolve the small and sharp geometry of the blade edges. This refinement is defined by a desired cell size. For the trailing edges, where the cells are particularly small, a refinement thickness is also set. The CfMesh software offers many settings to control how the mesh is generated and the ability to control the cell sizes in different regions is the main tool. These sizes are however closely interconnected, so without proper thought the tools existing in CfMesh can be confusing, as inputs which are not matching the surrounding mesh will be ignored.

In every CFD simulation, it is important to validate that the solution is independent of the mesh. Meaning that it has to be ensured that the mesh adequately resolves the scales in the simulations. A good way of determining this is to run simulations and evaluate some parameter, which is relevant for the application, while gradually refining the mesh until the parameter stops changing. This is called a mesh convergence study. In the present work the static pressure difference between the outlet and the inlet is the evaluated criteria, along with the efficiency of the turbine. Instead of globally refining the cell sizes, the convergence study is carried out in two steps due to the available refinement tools in the software. First, a suitable internal mesh resolution is found by sequentially adding new refinement regions in areas of the flow that are indicated to be interesting in preliminary simulations. Those areas are; the wake behind the last blade row, the rotor blades and the external pipes. After an internal mesh resolution is established, a similar study is undertaken for the boundary layer cells only. This is done since the refinement that is needed close to the walls depends on the flow speed past the patch, for which a good estimate is not known before the first test is completed. The result from these tests, and the resulting mesh, is presented in the method validation in section 4.2.

3.3 CFD setup

In the present work the OpenFOAM software is used to simulate the flow. All simulations are solved as steady state, incompressible and turbulent flows, using the SIMPLE algorithm to couple pressure to velocity [24] with the `simpleFoam` solver. In this section, each of the settings and assumptions regarding OpenFOAM a re outlined and discussed.

3.3.1 Flow Conditions

The conditions of the simulations, shown in Table 3.1, are set based on previous data and desired behaviour.

Table 3.1: Flow Condtions in OpenFOAM

	Pilot design	Optimisation designs
Flow speed [m/s]	0.3	0.3
Rotational speed [RPM]	466.5	496.6
Density $[kg/m^3]$	999.7	999.7
Kinematic viscosity $[kg/(s \cdot m)]$	$1.3e-6$	$1.3e-6$

The free stream velocity is based on the most frequently occurring flow speed and the density and kinematic viscosity are picked based on a typical water temperature of $10°C$. The original rotational speed is matched to previous experiments and the one for optimisation is selected based on desired behaviour. Note that the pilot design is simulated at the lower rotational speed since a comparison with the experimental data was initially planned. This data did however prove to be too unreliable and no validation of the model could be carried out. The desired RPM for the optimisation is 500 RPM, but was due to slight calculations errors accidentally set at 496.6 RPM.

3.3.2 Rotational Zone

In order to account for rotation with the frozen rotor approach, the flow is treated in two different ways, with a rotating or stationary reference frame. The domain is thus divided in three sections. Section one extends from the inlet to the hub and is in the stationary frame, section two stretches over the entire hub length and is in the rotational frame of reference and finally section three is from the hub to the outlet and is solved in the stationary reference frame. Figure 3.12 shows where the rotational zone is located.

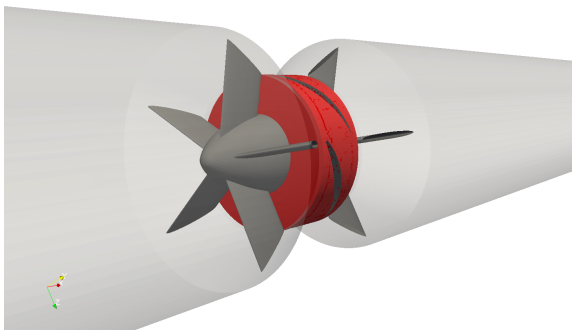


Figure 3.12: Geometry of the turbine with rotational zone coloured red.

When using the frozen rotor simulation approach it is important that the interfaces of the rotational zones are axi-symmetric [36]. For meshes generated with an automated mesh generator like CfMesh, that can be hard to achieve. The problem

is solved by moving the points in the mesh closest to the interface, exactly to the interface. By slightly stretching or compressing the cells by the interface in the axial direction, a perfectly flat plane of cell faces that align with the rotational zone in Figure 3.12 is achieved.

3.3.3 Boundaries and Wallfunctions

Boundaries often present the most interesting regions in fluid simulations, yet they are challenging to simulate accurately as they are often characterised by high gradients. It can also be hard to control the height of mesh cells on the boundaries of complex geometries, especially with an automatic mesh generator. As a result, the height of wall adjacent cells can vary through the domain and span all boundary layer sub regions, in which the flows behave very differently.

Since a short computational time is sought, a coarse mesh with boundary cell y^+ values suitable for high Reynolds number models with wallfunctions, see section 2.4, is desired. There are several wall functions that can be used with RANS models, most of which model quantities either in the log-law region or in the viscous region [30], but not explicitly in the intermediate buffer region. Salim and Cheah [29] therefore recommends that placing the first cell in the buffer layer should be avoided. Due to the combination of low free stream velocity and small geometrical features in the current application, such as trailing edges, it is difficult to obtain the desired high cell heights for the boundary cells, while still resolving the geometrical shapes. See Appendix B for further details. Instead, a low Reynolds number model is applied in combination with adaptable wall functions to handle the boundary regions at a low cost.

The $k-\omega$ SST model is chosen for the remainder of the present work, mainly for its stability during tests, see section 4.4. The model is used along with low Reynolds number wall functions and the mesh is refined as much as possible in important areas without increasing the cell count too much. Liu [37] states that the wall adjecent cells should be placed in the viscous layer, but ideally even at $y^+ \leq 1$. The lower recommendation is not fulfilled due to the effects on simulation time, but even the higher limit is difficult to achieve. Kalitzin et al [38] studied the effects of wall functions on different meshes in RANS simulations of flow over a flat plate and concluded that the largest errors are generated in region around $5 \leq y^+ \leq 11$ which is the lower part of the buffer layer. In order to evaluate the possible errors due to this violation, a fine boundary layer mesh is made as a benchmark to compare to the mesh used in the optimisation loop, see section 4.2.2.

The wallfunctions used with the $k-\omega$ SST model are the `omegaWallFunction`, `kLowReynoldsWallFunction` and the `nutLowReWallFunction`. In the source code of OpenFOAM, the `omegaWallFunction` is calculated differently in different boundary sub layers according to the following equations

$$\text{viscous: } \omega_{visc} = \frac{6\nu}{\beta_1(y)} \qquad \text{logarithmic: } \omega_{log} = \frac{\sqrt{(}k)}{\kappa C_\mu^{1/4} y}. \tag{3.4}$$

And the blending between the regions is calculated as

$$\omega = \mathrm{w}\sqrt{(\omega_{visc})^2 + (\omega_{log})^2}, \tag{3.5}$$

where the constants $C_\mu = 0.09$, $\kappa = 0.41$ and $\beta_1 = 0.075$ and w is a weight factor for the faces.

The `kLowReynoldsWallFunction` is expressed in the source code as

$$\text{viscous: } k = \frac{2400}{\sqrt{C_{eps2}}}\left[\frac{1}{\sqrt{y^+ + C}} + \frac{2y^+}{C^3} - \frac{1}{\sqrt{C}}\right] \qquad \text{logarithmic } k = \frac{C_k}{\kappa log(y^+)} + B_k, \tag{3.6}$$

where $C_{eps2} = 1.9$, $C = 11$, $C_k = -0.416$ and $B_k = 8.366$.

No blending takes place for k between the regions, and the use of each equation is controlled by the location of the first cell. The transition between the viscous and the logarithmic region is controlled by the y^+ value and takes place at $y^+ = 11.53$. The expression for k at the wall is derived from the v^2-f model [38], which might not be ideal for use with the $k - \omega$ SST model. Out of the available options for wall functions in the OpenFOAM software, this is assumed to be the most suitable one for the task, as the location of the cells at the walls varied and an adaptive wallfunction is needed.

The `nutLowReWallFunction` is designed to be used with low Reynolds number models and simply sets ν_t to zero at the wall. The main reason for using it is that it allows y^+ to be calculated, and it can be used both in the viscous and logarithmic layers [30]. It is debatable whether or not another type of wall function should be used, such as `nutUSpaldingWallFunction`. Instead it gives a profile of ν_t all the way to the wall and can be used in all boundary layer sub regions [37][30].

3.3.4 Boundary Conditions

Boundary conditions are set on the edges of the domain and initial guesses for the turbulent boundary values are estimated using the equations shown in section 2.5. The initial value for the turbulence intensity is calculated using equation 2.27 with $a = 0.14$ and $b = -0.079$ following the recommendations of Russo and Basse [28]. The inlet value for the turbulent kinetic energy is then calculated from equation 2.29.

Following the recommendations of Kaul [39] the turbulent length scale is taken as $l = 0.07d_h$ where the characteristic length d_h is the hydraulic diameter. For circular pipes the hydraulic diameter is equal to the pipe diameter. The remaining turbulent properties at the inlet are then calculated form equations 2.31 and 2.32 respectively.

The boundary conditions are shown in tables 3.2 and 3.3.

Table 3.2: Boundary conditions for velocity and pressure.

	Velocity	Pressure
Inlet	`fixedValue` uniform (0.3,0,0)	`zeroGradient`
Outlet	`zeroGradient`	`fixedValue` uniform (0)
Walls	`fixedValue` uniform (0,0,0)	`zeroGradient`

Table 3.3: Boundary conditions for turbulent quantities using k-ω SST.

	k & ω	ν_t
Inlet	`fixedValue` uniform	`nutLowWallFunction` uniform
Outlet	`zeroGradient`	`zeroGradient`
Walls	ω - `OnegaWallFunction` uniform	`nutLowWallFunction` uniform
Walls	k - `kLowWallFunction` uniform	-

The uniformal values used as an initial guess for k, ω and ν_t are 0.011007, 192.72 and 0.0000562 respectively.

3.3.5 Solvers

There are a number of linear solvers that can be used in OpenFOAM to solve the discretised governing equations. Tests are carried out with different linear solvers to determine which one gives results the fastest. Based on the results, the `GAMG` (generalised geometric-algebraic mutli-grid) solver, using a `GaussSeidel` smoother is selected for the kinematic pressure. For velocity, k and ω the solver `smoothSolver` is picked, using a `GaussSeidel` smoother and two `nSweeps`. The tolerance for all quantities, is picked to be small, as the simulations are controlled by number of iterations rather than a convergence criteria. The relative tolerance is varied to investigate its impact and lowering it only increases the simulation time, not the quality of the solution. The solver settings used in the present work are summarised in Table 3.4.

As the flow is not convected through the domain per se in a steady state simulation the flow field can vary a lot between iterations, especially in the beginning of every simulation. Restricting this behaviour can increase the stability of the simulations and is common for steady state simulations [24]. It is done by defining under relaxation factors that are used to weigh values from the previous iteration with the ones calculated for the current iteration to limit changes. They should be selected to be small enough to obtain stability but big enough for the solution to propagate forward [40]. The under-relaxation factors used in the present work are shown in Table 3.5.

Table 3.4: Solver Settings

p	Solver	`GAMG`
	Smoother	`GaussSeidel`
	tolerance	10^{-8}
	relative tolerance	0.01
	`minIter` & `maxIter`	1 & 1000
U, k and ω	Solver	`smoothSolver`
	Smoother	`GaussSeidel`
	tolerance	10^{-8}
	relative tolerance	0.1
	`minIter` & `maxIter`	1 & 1000

Table 3.5: Under-relaxation factors

p	0.3
U, k and ω	0.7

3.3.6 Schemes

Just as for the solvers, there are several numerical schemes that can be used with OpenFOAM. In this section, the schemes that are used in the present work are listed and discussed. It is important to keep mind that incorrectly chosen discretisation schemes can cause numerical errors in the solution. The schemes, therefore, need to be chosen with care.

Time Scheme. The *steadyState* time scheme is used as the RANS equations are simulated and no time dependency is considered.

Gradient Schemes. The default and most used gradient scheme in OpenFOAM is the *Gauss linear* scheme. It is however recommended for simulations with poor quality meshes to use schemes that overwrite certain gradient terms. This is done by the *cellLimited Gauss linear* scheme [40] to improve stability. In this task an auto mesh generator is used and a proper mesh quality can not be guaranteed for all geometries. The *cellLimited Gauss linear* scheme is therefore selected.

Divergence Schemes. At least second order accurate schemes are desired for the evaluation of all flow quantities, since the general transport equation is of the second order [41]. For velocity, the second order accurate *linear upwind* is used successfully. For the turbulent quantities however, second order accuracy can not be achieved due to stability issues. The *upwind* scheme, which is first order accurate, must be used instead. For the effective viscosity, the second order accurate *linear* scheme is selected.

Surface Normal Gradient Scheme. It is recommended to use the *corrected* scheme for meshes which have non-orthogonality above 50°, but in cases when the non-orthogonality reaches above 70°, the correction causes the solution to become unstable. The maximum non-orthogonality for many of the meshes generated reached as high as 70°, thus the *limited corrected 0.5* scheme is selected [40]. This scheme increase the simulations stability but affect the accuracy of the solution.

Interpolation scheme. The default *linear* interpolation scheme is chosen.

Laplacian Scheme. Based on the selection for the surface normal gradient and the interpolation scheme, the *Gauss linear limited corrected 0.5* laplacian scheme is selected.

Table 3.6 summarises the information about all schemes used in the present work.

Table 3.6: Discretisation schemes.

Time scheme	
	`default steadyState;`
Gradient scheme	
	`default Gauss linear;`
	`grad(U) cellLimited Gauss linear 1;`
Divergence schemes	
	`div(phi,U) bounded Gauss linearUpwind grad(U);`
	`div(phi,k) bounded Gauss upwind;`
	`div(phi,epsilon) bounded Gauss upwind;`
	`div((nuEff*dev2(T(grad(U))))) Gauss linear;`
laplacian scheme	
	`default Gauss linear limited corrected 0.5;`
interpolation scheme	
	`default linear;`
snGradSchemes	
	`default limited corrected 0.5;`

3.4 Post Processing

In steady state simulations, the solution ideally converges to be steady where the results do no longer change. For such a case the residuals between two subsequent iterations of the simulation have a very low value that can be used as a marker for when to stop the simulations. Results can then be extracted from the last data point as it is representative of the steady state. This is however not always the case.

In the present work, it is found that the simulations converge to fluctuating solutions and no true steady state can be achieved. There can be several reasons for this, such as numerical or mesh issues. Since an automatic meshing tool is used it is hard to control the mesh quality at certain regions, which can cause numerical errors [41]. It may also be due to a physical phenomenon in the flow, as the flow in a rotating turbine is inherently unsteady. When flows separate from bodies, a phenomenon called vortex shedding arises. This happens for a wide range of Reynolds numbers and is an unsteady phenomenon where the wake behind the object oscillates back and forth [18]. Unsteady simulations that resolve these features come at a considerably higher computational cost that is often impractical for optimisation studies that require many simulations. In the presented work a robust method to evaluate the steady simulations is implemented, one which can handle the different flow fields

in the varying designs.

The tool used to overcome these fluctuations is averaging. In the scenarios with small fluctuations in the solution, an average can be taken over one period of the fluctuation to get results which resemble a steady solution. In order to handle variations in the fluctuations between different designs, a long averaging is instead implemented in the present work so that variations in the period of the fluctuation do not change the result.

The post processing technique is found by analysing the moment on the rotor blades, see Figure 3.13, and static pressure fluctuation in a point, see Figure 3.14, from simulations of the pilot design. To ensure trustworthy results the simulations run for a long time. Averages for the interesting quantities are then calculated, starting with only one iteration and sequentially including more values to find a suitable sample size that results in a steady mean. These results are extracted after the residuals stopped developing and show a fluctuating behaviour, which occurs at iteration 500.

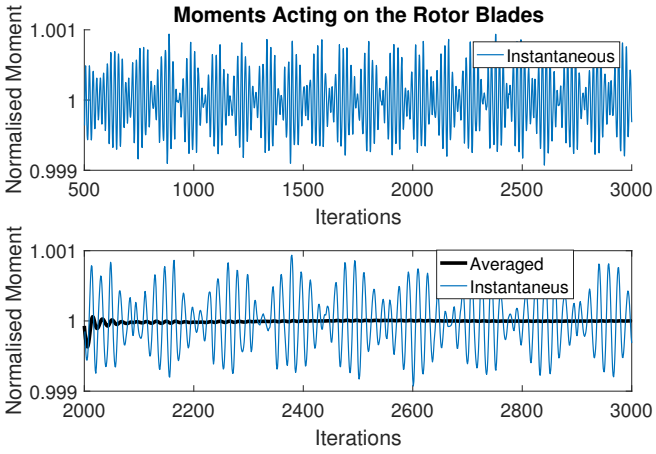


Figure 3.13: Blade momentum. Instantaneous momentum for span of 500-3000 iterations (above) and instantaneous momentum for span of 2000-3000 and plot of averaged quantities (below).

As can be seen in Figure 3.13, averaging over one period 175 iterations, of the regularly fluctuating moment is enough to get an almost stationary mean value. Contrary to the total moment, the static pressure is for this test only analysed in a single monitored point and thus, as expected, varies more. Therefore, the pressure results are used to find the limiting number of iterations for averaging.

Though it varies more, the static pressure results are still periodic, as seen in Figure 3.14. 1000 iterations are concluded to be the minimum sample size that is thought to give consistent means for fluctuations of varying character throughout the optimisation. Since the fluctuations are showing regular, and in some sense smooth, behaviour without apparent spikes, the results are judged to not be noise, rather signs of a physical phenomenon, such as vortex shedding.

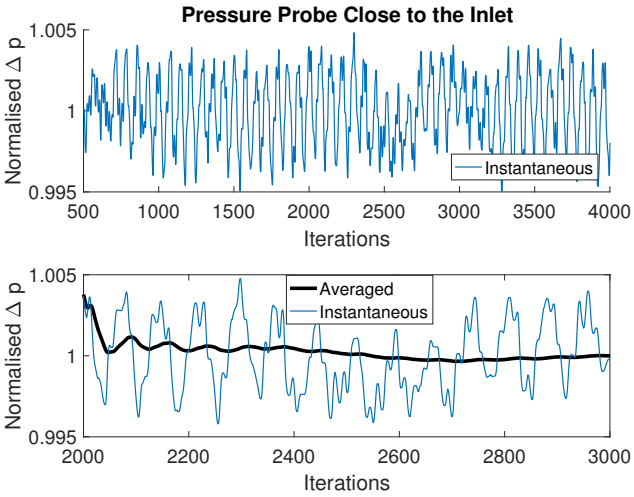


Figure 3.14: Inlet pipe static pressure probe. Instantaneous pressure for span of 500-4000 iterations (above) and instantaneous momentum for span of 2000-3000 and plot of averaged quantities (below).

Once the sample size is determined, the starting iteration number for the average is sequentially reduced to find an appropriate total simulation time. Figure 3.15 shows how the mean value of the static pressure drop change with first sampling iteration number.

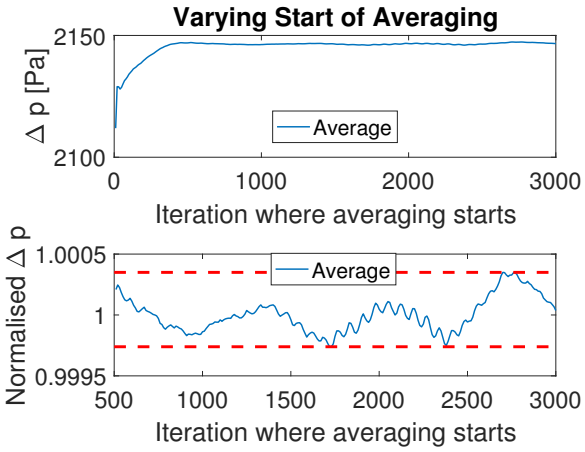


Figure 3.15: Averaging starting iteration for the inlet pipe static pressure with averaging over 1000 iterations (above) and zoomed normalised static pressure showing maximum and minimum markings (below).

Figure 3.15 shows that starting the average approximately after 500 iterations should be sufficient to obtain a steady mean, however some margin needs to be added so that the method works for all designs. 2000 iterations are considered necessary before sampling can start. The average varies, depending on what iteration window it is carried out for. However, the percentage difference between the minimum and

maximum, indicated by the red lines in Figure 3.15, proves to be 0.061%, which is concluded to be acceptably low. All later simulations are therefore decided to run for 3000 iterations, where the results are extracted from the last 1000.

3.5 Optimisation Algorithm

The iterative process of optimisation is handled by the software Dakota where the multi-objective optimisation is handled by constructing the objective function that is minimised from a weighted sum of the different objectives. The goal is to minimise the sum of the inverse efficiency and the kinematic pressure drop in the system, where normalisation and weights are added to each objective in the objective function.

The optimisation problem is formulated on the negative null form as

$$
\begin{aligned}
\underset{\mathbf{x}=rb,sb,hr,rp,sp}{\text{minimise}} \quad & f(\mathbf{x}) = w_1\left(\frac{\hat{1}}{\eta}\right) + w_2\left(\frac{\hat{\Delta P}}{\rho}\right) \\
\text{subject to} \quad & g_1(rb) = -rb + 3 \leq 0 \\
& g_2(rb) = rb - 6 \leq 0 \\
& g_3(sb) = -sb + 3 \leq 0 \\
& g_4(sb) = sb - 6 \leq 0 \\
& g_5(hr) = -hr + 9mm \leq 0 \\
& g_6(hr) = hr - 21mm \leq 0 \\
& g_7(rp) = -rp - 10^\circ \leq 0 \\
& g_8(rp) = rp - 10^\circ \leq 0 \\
& g_9(sp) = -sp + 0^\circ \leq 0 \\
& g_{10}(sp) = sp - 10^\circ \leq 0 \\
& g_{11}(\mathbf{x}) = -\eta + \eta_{min} \leq 0 \\
& g_{12}(\mathbf{x}) = \frac{\Delta P}{\rho} - \frac{\Delta P}{\rho}_{max} \leq 0 \\
& g_{13}(\mathbf{x}) = -\Omega(M_{pre} + M_{vis}) + W_{min} \leq 0,
\end{aligned}
\tag{3.7}
$$

where the objective function, f, is formulated as a function of the state vector, $\mathbf{x}$, made up of the number of rotor blades rb, number of pre-swirl stator blades sb, hub radius hr, rotor pitch rp and pre-swirl stator pitch sp. The objectives in the objective function are normalised with the constraint values and are formulated as $\frac{\hat{1}}{\eta} = \frac{1}{\eta}/\frac{1}{\eta_{min}}$ and $\frac{\hat{\Delta P}}{\rho} = \frac{\Delta P}{\rho}/\frac{\Delta P}{\rho}_{max}$. Furthermore, W is the work done by the turbine, Ω is the angular velocity and M_{pre} and M_{vis} are the moments on the blade due to pressure and viscous forces respectively.

Due to external factors, the kinematic pressure drop is valued three times higher for the desired behaviour of the current application with the weights being $w_1 = 1$

and $w_2 = 3$. This has a great impact on the progression of the optimisation. To define the design space, each variable is limited by a pair of constraints, $g_1 - g_{10}$. The objective function is formulated to evaluate the performance of each design, but the individual objectives are also important on their own. Constraint g_{11} and g_{12} are therefore set on the objectives, where the limits are set to $\eta_{min} = 45\%$ and $\frac{\Delta P}{\rho}_{max} = 3m^2/s^2$ based on the performance of the pilot design. The amount of work produced is also constrained to keep above a desired level 0.27 Watts, which is the minimum requirement at the operation point in the optimisation.

The Efficient Global Optimization (EGO) algorithm, see section 2.7.1, is implemented in the present work to solve the optimisation problem. Based on the recommendations of Jones et al. [3] the initial sample is set following the 10x rule, where x is the dimensions of the design space. In the present work, 5 variables are investigated, so the sample should be at least 50 function evaluations. In order to get an even inter-point spacing [3]

$$\frac{1}{n-1} = \frac{1}{50} = 0.02, \tag{3.8}$$

$n = 51$ points are chosen.

No way of defining the optimisation variables as integers can be found for Dakota, thus Dakota only outputs floating point numbers. The number of blades needs to be an integer number while other quantities do not. The output for number of blades is therefore rounded down to the closest integer. The effect on the response equation due to this manipulation is unknown. This method was tested by doing similar rounding of the Rosenbrock optimisation benchmark function, which proved successful.

4
Method Validation

Although the design of the turbine could be improved after the present work, the most valuable outcome is the method procedure for optimisation that is developed. A method which easily allows for additional evaluation of future concepts and that can be used in studies with different limitations. In order to ensure a robust method for future use, each major assumption and decision on which it builds, have to be verified. This is done mainly by running different tests which are presented in this chapter.

4.1 Mesh Quality

Good quality meshes can be hard to accomplish when using automated meshing tools. In the OpenFOAM package there is a utility called `checkMesh`, which reports the quality of a mesh based on some predefined criteria. A number of mesh failures can be reported, where some are more critical than others. During the 7th OpenFOAM Workshop, Engys [42] presented a guide, where the importance of the different checks are discussed in detail. Most of the errors occuring in the present work can be related to complex shapes and sharp corners in the geometry as well as the usage of an automated meshing tool. The majority of the mesh errors can be resolved by refining troublesome areas. However, in this work not all of the mesh errors are solved.

The types of mesh problems within the geometry are listed below, and their severity is evaluated according to Engys guidelines [42]. Figure 4.1 and 4.2 show the location of the errors in the turbine region for the pilot design.

Low Volume Ratio Faces. Faces reporting this kind of error are bad, and can cause problems during simulations. A high number of these cells are present around edges of the blades. In order to lower their number refinements along both leading and trailing edges are needed. An extension to the geometry creation in Ruby is written, which captures the edges. CfMesh uses them to apply certain refinements, which in return improves the mesh in the areas around. It is noted that the edge refinement thickness, surprisingly, has a huge impact on the mesh quality.
Low Weight Faces. These are faces with low interpolation weights. They are bad and may cause stability issues. However, only a small amount of faces with this problem are present in the domain.
Concave Cells/Faces. These cells and faces are not as bad as the ones mentioned

above; however they should be limited as much as possible. There is a large number of these cells in the domain, which are mostly located at refinement regions with sharp reduction in cell sizes and near axi-symmetric intersections, such as where the hub meets the hub cone.
Low Quality Tet Faces. These are cells with low quality or negative volume decomposition. Their severity is ranked low, similar to the concave. The locations of the cells are not bounded to any specific feature in the domain but they are scatter throughout the refinement region around the turbine.
Other Errors/Problems. Additional types of mesh problems are present, but they varied between different designs. Cells with high *non-orthogonality* are present and reached above 70°, some meshes are close to 80°. Cells with high skewness, above 4, are present in a few cells for some meshes. Also, the mesh error *Warped faces* are present.

Figure 4.1 shows the mesh errors in the turbine area.

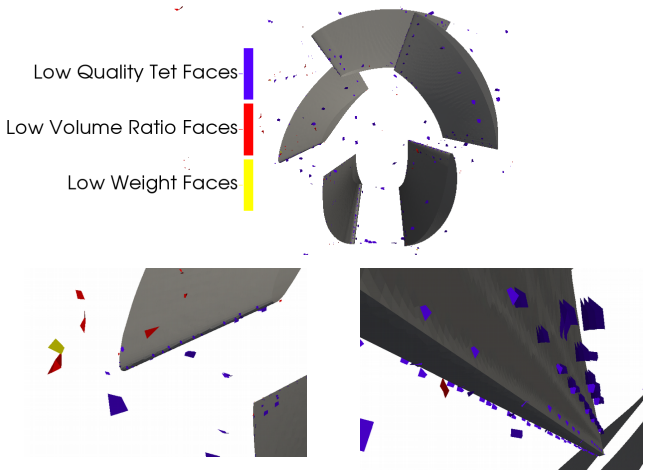


Figure 4.1: Mesh errors - Around the rotor blades (top), zoomed in on leading edge (bottom left) and zoomed in on trailing edge (bottom right).

Figure 4.1 shows that the majority of the faces with poor quality are *Low Quality Tet Faces* and that they are located in the entire refinement region around the turbine. Many of them are clustered around the trailing edge of the rotor blades. *Low Volume Ratio Faces* also existed everywhere around the turbine, although in smaller quantities. Finally the *Low Weight Faces* are barely visible as the number of those cells is limited.

The Concave cells and faces are shown separately in Figure 4.2

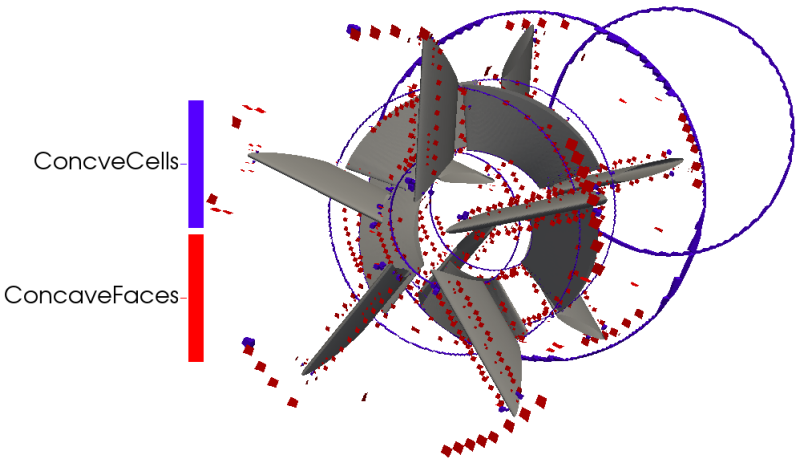


Figure 4.2: Mesh errors - Concave cells and faces around turbine and statorblades.

As seen in Figure 4.2 the concave cells are around intersection areas where the axi-symmetric objects meet and the faces are located in the refinement transition regions.

4.2 Mesh Grid Independence

To evaluate the importance of the mesh issues and to find an appropriate mesh resolution, with respect to the static pressure drop and efficiency that is sought in the present work a mesh independence study is carried out, as described in section 3.2. All simulations in this section are carried out by using the $k-\omega$ SST model. The static pressure drop is evaluated by taking the spatial average pressure on the inlet patch averaged over the last 1000 iterations. This is done due to unsteadiness in the simulations and is implemented with the `fieldAverage` utility in OpenFOAM. The efficiency is evaluated as described in section 2.1 where the moment is found by making use of the `forces` function object in OpenFOAM. The moment is calculated about the center axis, averaged over 1000 iteration steps. The convergence study is carried out by constructing a more accurate benchmark mesh with a resolution unaffordable for optimisation, and then trying to find an affordable resolution where the resulting engineering quantities does not vary more than 1% from the benchmark.

4.2.1 Internal Mesh Comparison

The first step of the independence evaluation is to refine the internal mesh resolution. This is done by comparing simulations on three different meshes, with increasing

resolution in interesting regions of the domain. Table 4.1 summarises the statistics for the different meshes.

Table 4.1: Convergence Meshes Statistics

Mesh Resolution	A1	A2	A3
Cell Count	1 596 000	2 132 000	4 360 000
Additional Refinement Region	-	Wake, Rotor Blades	Inlet Pipe, Outlet Pipe

Figure 4.3 shows the resulting static pressure drop and efficiency from the three meshes and show that the most competitive mesh is the medium mesh, resolution A2. The efficiency difference per cell is roughly 8 times bigger between resolution one and two than between resolution two and three, which can be seen from the slope of the line in Figure 4.3. For the static pressure drop, the first refinement results in 10 times bigger difference per additional cell than the second refinement. The argument can be made that even the coarse mesh shows good results compared to the benchmark as the differences are about 2%, the medium mesh resolution, which deviates about 1% for both efficiency and static pressure drop, is chosen for the rest of the work.

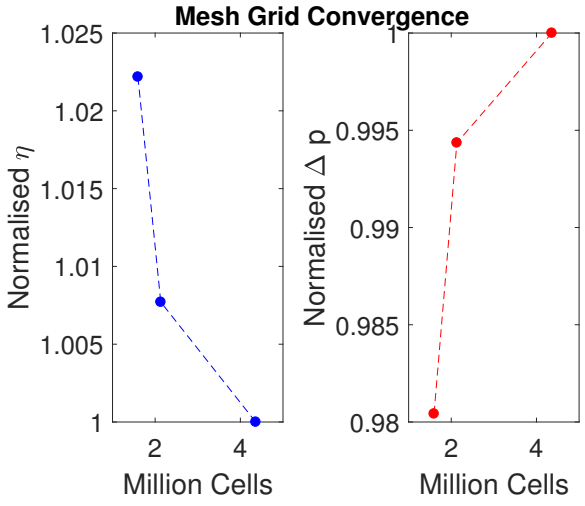


Figure 4.3: Results comparison between the mesh resolutions. Efficiency on the left and static pressure drop on the right.

4.2.2 Boundary Layer Resolution

After results from the internal grid convergence are obtained the estimates of y^+ can be evaluated. The results shows that the actual values for y^+ spans all the different boundary layer regions. In order to investigate the effect of this, a boundary layer convergence study is carried out. The main goal is to keep the average y^+ within the viscous layer, without excessive cell count. Additionally, the y^+ values on important

surfaces, such as the blades, are kept low.

The mesh A2 is refined by adding more boundary layers in three steps. The setup for the boundary layer evaluation is shown in Table 4.2, including the mesh statistics and the resulting average values of y^+ from the `yPlus` function object in OpenFOAM, recall the names of the patches in section 3.1. A comparison of efficiency and static pressure drop for all meshes is shown in Figure 4.4.

Table 4.2: Boundary layer statistics

Boundary Layer Resolution	A2	B2	B3	B4
Cell Count	2 132 000	2 533 000	2 687 000	6 114 000
Average y^+ per patch				
inletPipe	9.9	4.5	4.5	2.7
upShroud	7.6	4.7	4.7	3.0
upStatorBlades	3.6	1.1	1.1	0.6
upHubCone	7.4	3.7	3.7	2.1
band	4.5	3.2	3.1	2.3
turbineBlades	2.5	2.5	1.4	1.4
hub	4.1	4.1	3.9	2.0
downShroud	3.8	3.7	3.6	1.7
downStatorBlades	2.5	2.5	2.5	1.2
downHubCone	8.3	5.1	5.2	3.0
upOutletPipe	8.8	4.4	4.4	1.8
downOutletPipe	14.7	5.2	5.2	1.3

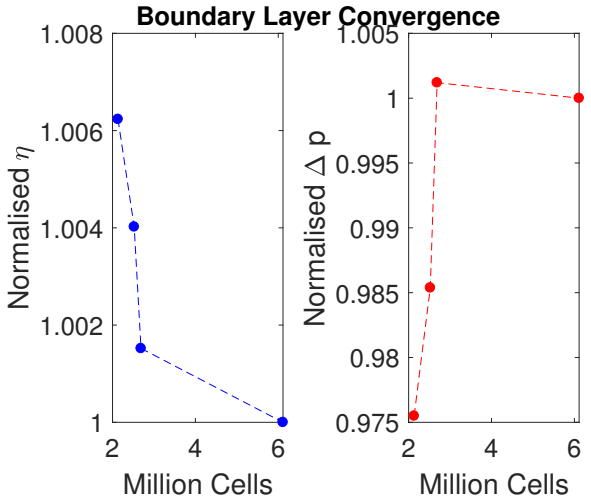
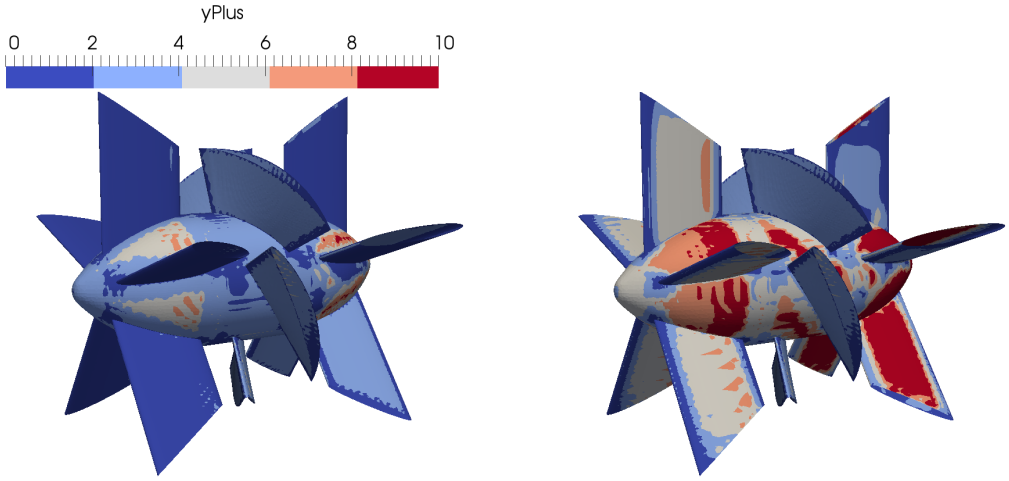


Figure 4.4: Results comparison between the boundary layer approaches. Efficiency on the left and static pressure drop on the right.

One thing to keep in mind when reading the values in Table 4.2 is that the average is based on cells, thus small surfaces impact the average just as much as big ones.

To get an idea of the difference between cell average and surface average, compare Table 4.2, which shows cell average, to Figure 4.5 and 4.6 which show the surface average distribution around the turbine and in the channel. The results from the simulations in Table 4.2 and Figure 4.4, shows that refining the boundary layer to obtain an average y^+ that is mostly in the viscous region ($y^+ < 5$) is not that costly, but has a big impact on the efficiency and static pressure drop. The difference between the mesh B3 and B4 in terms of efficiency and static pressure drop is 0.15% and 0.12% respectively. Boundary layer resolution B3 is therefore used for the rest of the work.

Though the cell averages in Table 4.2 show acceptable values, they do not reflect the results accurately. Figure 4.5 show the y^+ around the turbine for mesh B4 and mesh B3 respectively and visualises the surface average y^+ values.



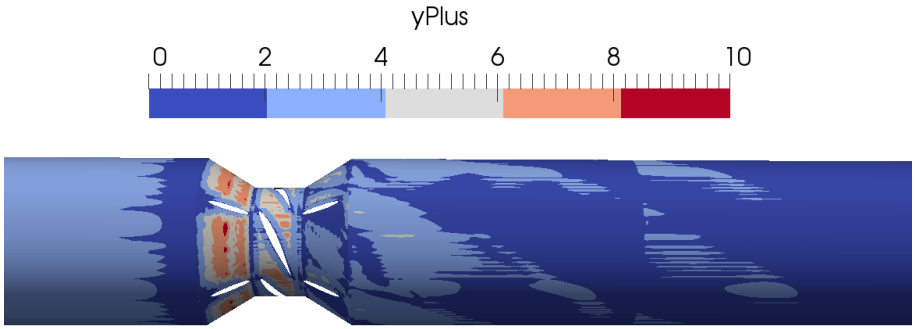
(a) y^+ values around the turbine for the benchmark mesh grid, mesh B4.

(b) y^+ values around the turbine for the mesh grid used during the optimisation process, mesh B3.

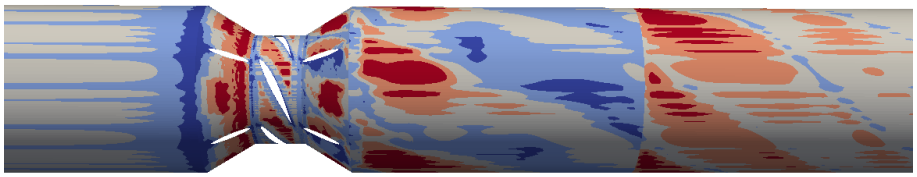
Figure 4.5: y^+ values on the wall surface of the turbine.

It may be seen from Figure 4.5a that more or less all the blades are below or around $y^+ = 5$. However, other components, such as the hub and the hub cones show slightly higher values in isolated areas. Even a location where the y^+ is above ten is noted on the downstream hub cone. If examining the mesh grid B3, which is used for the remainder of the work, in Figure 4.5b, completely different y^+ values are observed. Blade row one and two, are around $y^+ \approx 5$, which is acceptable, though spots with higher values are noted. At blade row three, the flow is impinging on the blade on one side which gives a large increment to y^+. The hub and the hub cones vary in y^+ and high values are noted. According to theory, many of these values are undesirable.

It is necessary to also analyse the channel in a similar manner as previously in terms of y^+ values. Figures 4.6a and 4.6b show the y^+ for mesh B4 and mesh B3 respectively in the channel.



(a) y^+ values in the channel for the benchmark mesh grid, mesh B4.



(b) y^+ values in the channel for the mesh grid used during the optimisation process, mesh B3.

Figure 4.6: y^+ values on the wall surface of the channels.

Figure 4.6 show similar behaviours as seen in Figure 4.5, mesh B4 fulfils the theory acceptably, though high y^+ regions still persist, whereas mesh B3 on the other hand show high y^+ values nearly everywhere in the channel.

Though this analysis show undesired results the importance of a grid which is affordable for multiple simulations during the optimisation needs to be underlined, as well as the results in Figure 4.4, which justify the selection of boundary layer resolution B3.

Figure 4.7 shows the mesh resolution for the mesh grid B3 which is used in the present work. The meshes A1 and A3 can be seen in appendix C.

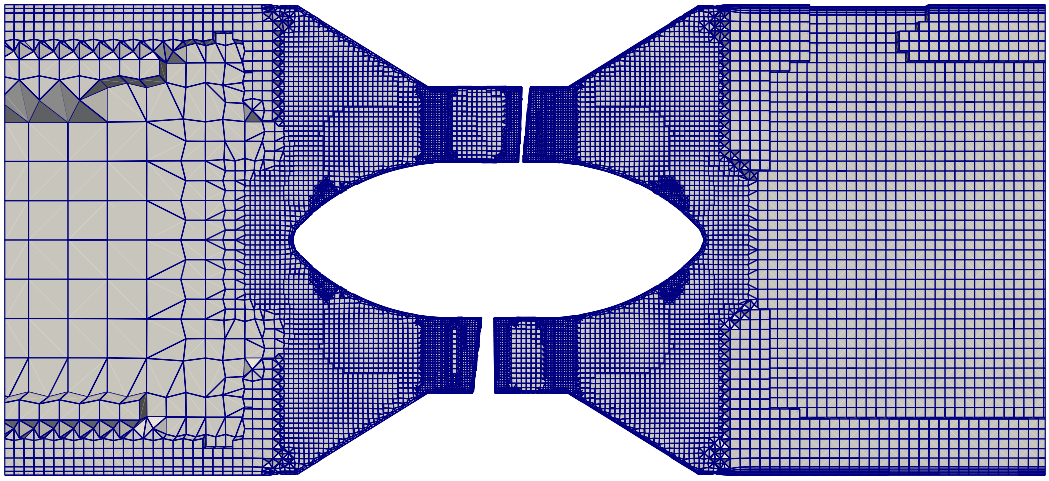


Figure 4.7: Cross sectional view of the mesh grid, mesh B3.

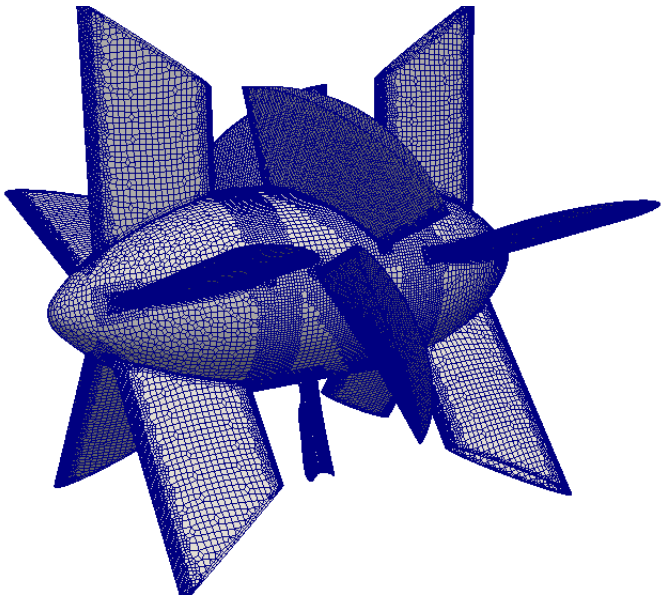


Figure 4.8: Surface mesh of the turbine, mesh B3.

In Figure 4.7, the refinement area in the vicinity of the turbine is located where the `upShroud` starts and where the `downShroud` ends. Around the blades, as well as the rotating region, an additional mesh grid refinement is made. The refinements around the rotating regions are made in order to avoid errors when manipulating the mesh and get a completely axi-symmetric interfaces at the rotatinal zone, as described in section 3.3.2. The surface resolution of the turbine is shown in Figure 4.8. There are refinement zones which can be seen more clearly, the rotational zone and the different surface refinement densities.

4.2.3 Domain Size

In order to evaluate the impact of the boundary conditions on the flow around the turbine, the lengths of the inlet and outlet pipes are varied and simulated. The pipes are varied individually and the results are shown in Figure 4.9. It is mportant to note that when the inlet is varied, the outlet is kept fixed and visa versa. The more precise the estimates on the boundaries the smaller the domain can be. Since the boundary conditions at the inlet and outlet are imprecise, as shown in Table 3.2 and 3.3, the domain needs to be adjusted. A fully developed boundary is desired in the inlet pipe, which results in a long pipe. Additionally, a sufficiently long outlet pipe is needed in order to eliminate any turbine effects at the outlet. Those effects can alter the results regarding both static pressure drop and efficency.

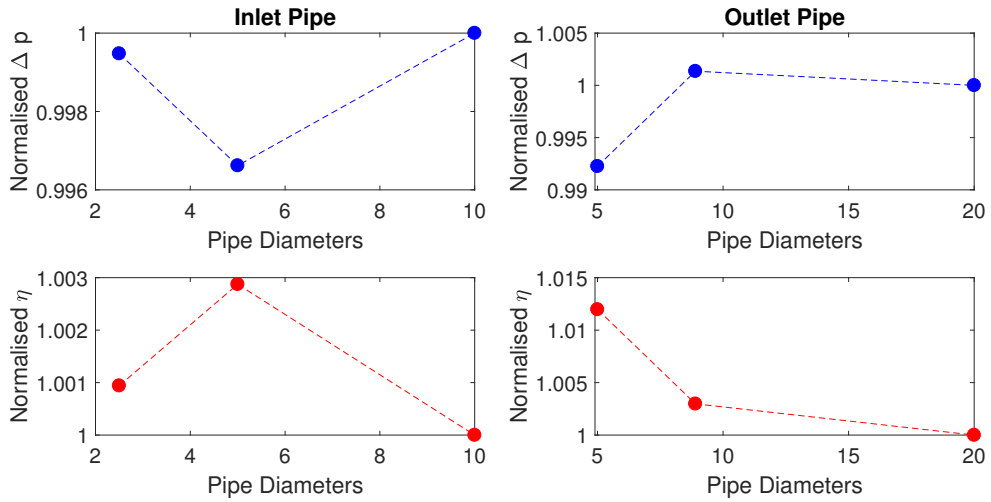


Figure 4.9: Performance variation for different domain sizes. Inlet pipe (left) and outlet pipe (right). Static pressure drop (above) and efficiency (below).

Surprisingly, as shown in Figure 4.9, the differences between the domains seem to be small for all domain sizes. Theoretically, the entrance length for a pipe after which a fully developed boundary layer is obtained, is long. The convergent-divergent casing that surrounds the turbine in the present work does however influence this, as it disturbs or destroys the established boundary layer in the inlet pipe. As the pipe lengths proves to be of little importance the middle sized pipes, for which there is already a finished model, is selected for the remainder of the present work with 5 pipe diameters between the inlet and the turbine, and 8.9 pipe diameters to the outlet.

4.3 Rotor Position for the Pilot Design

As described in Chapter 1, the rotor is fixed in space when implementing the frozen rotor approach. The resulting flow therefore only resembles the flow at an instant in the real system, where the rotor is rotating. The flow varies as the rotor moves so to allow comparison to the real turbine, several frozen positions are simulated. Figure

4.10 shows the efficiency and static pressure drop results for the different positions. Note that the pilot design has five rotor blades and hence one blade every $\frac{360}{5} = 72°$. The space between the blades is divided into five positions, so the test covers the entire revolution in steps of $\frac{72}{5+1} = 12°$.

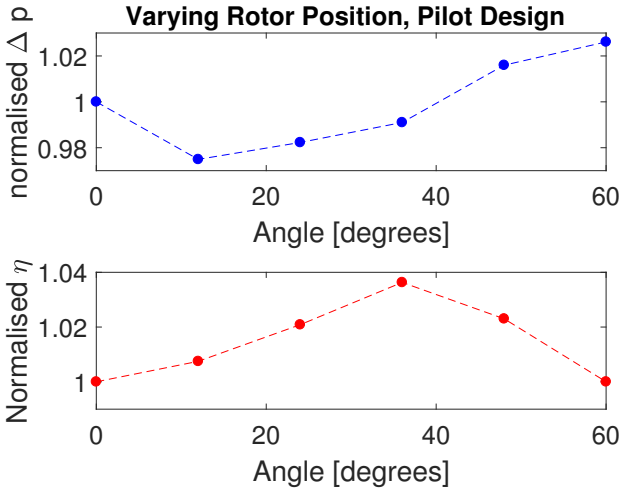


Figure 4.10: Results comparison with varying rotor orientation. Static pressure drop (above) and efficiency (below).

Due to the unphysical frozen rotor approach, the results vary over the rotation, as seen in Figure 4.10. The static pressure drop varies with ±2% whereas the efficiency varies with up to 4%. The maximum efficiency is 49.7% and is obtained at 36°, when the rotor blade is positioned tangentially between stator/support blades in row one and three.

The pilot design that is simulated in this test has the same number of stator blades as rotor blades. The relative spacing between the stator and rotor blades is therefore the same for all blades. However, when varying the number of blades this is not the case and as a result the wake of blade row one and two hits blade row two and three differently. In the geometry creation, the leading edge of the first rotor blade is always placed at the same rotational position and the rest are then spread evenly around the hub. In order to make sure that the first rotor blade is not always created directly behind the first stator blade, the rotor is created with an offset of 36°.

4.4 Turbulence Model Comparison

As discussed in section 2.4 there is a multitude of models for turbulence. Three different low Reynolds number turbulence models are tested in the present work, the $k-\omega$ SST model, the Launder-Sharma (LS) $k-\varepsilon$ model and the Lam-Bremhorst (LB) $k-\varepsilon$ model. Great stability is achieved with the $k-\omega$ SST model. A converged solution however proves hard to achieve with the other models. Attempts to ramp the under relaxation factors for a more stable progression are made successfully for

the LS $k-\varepsilon$ while the solution for LB $k-\varepsilon$ still diverges.

Figure 4.11 shows the comparison between the $k-\omega$ SST and the LS $k-\varepsilon$ models, where the static pressure and velocity profiles in the spannwise direction are presented at two axial positions, behind the rotor blades and behind the downstream support stator blades in row three.

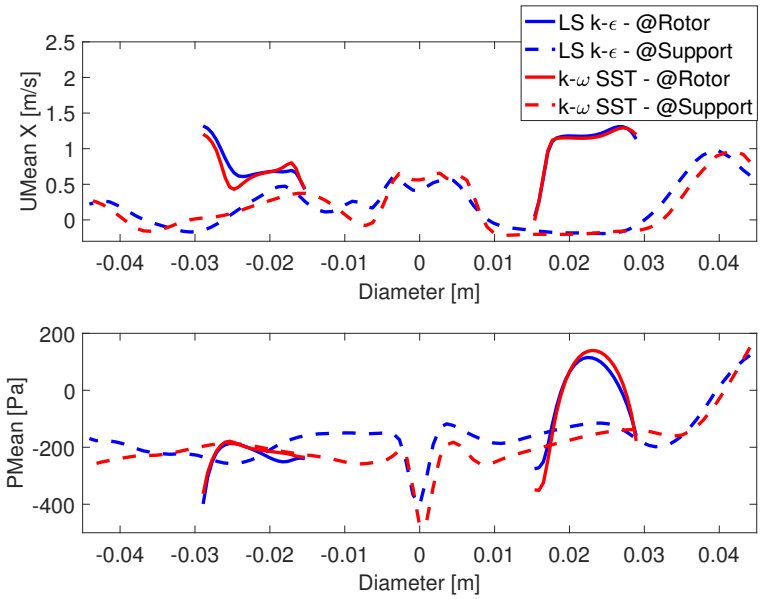


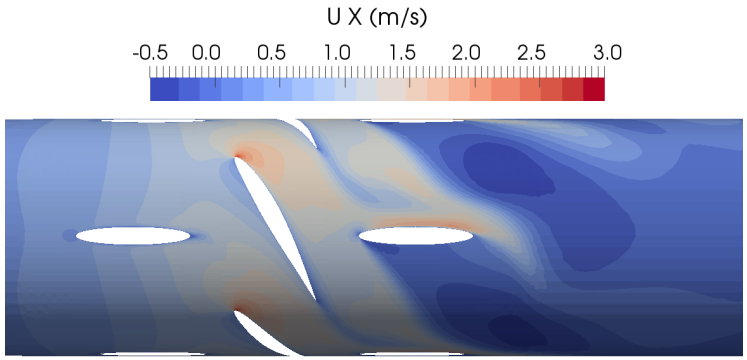
Figure 4.11: Results comparison between $k-\omega$ SST and LS $k-\varepsilon$, mean velocity in axial direction (above) and mean static pressure (below).

As can be seen from Figure 4.11, both the pressure and velocity fields show similar behaviours for both models. Since the optimisation is to be carried out for a great variety of designs, stability is important and the $k-\omega$ SST model is therefore used throughout the present work.

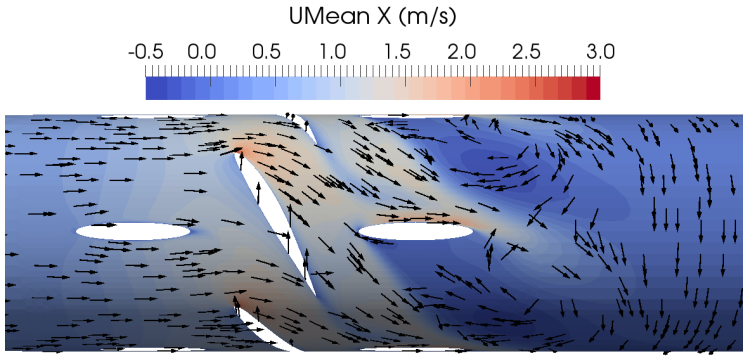
4.5 Flow Fields of the Pilot Design

The velocity and pressure fields of the pilot design are analysed to see the flow behaviour across the turbine. Also, the source of the fluctuation is pinpointed to evaluate if it is physical, if it can be linked to poor mesh grid quality or numerical issues.

Figure 4.12 shows both the instantaneous and average axial velocity, in a cylindrical plane half way between the hub and the band in the radial direction.



(a) Instantaneous axial velocity filed.



(b) Averaged axial velocity field over 1000 iterations (contour) and directional velocity vectors (arrows).

Figure 4.12: Axial velocity field.

When analysing Figure 4.12, the acceleration due to the convergent divergent casing can be seen, as well as how the flow is guided by and separates from the blades. As can be seen, the flow impinges on the blades in row three which then shields the flow and creates a recirculation region in its wake. The differences between the instantaneous and average velocities are mostly limited to the wake region behind the supports in row three. The difference is a fluctuation present in and around the shear layer between the main flow and the recirculation behind the supports in row three.

In order to locate the fluctuations in the flow, the tool *UPrime2Mean* is used. It calculates the mean square field of fluctuations about an average mean, $\overline{u_i' u_j'}$. As the name steady state solution suggest, there should not be any fluctuations in the flow. The tool is thus used qualitatively to locate fluctuating regions. Figure 4.13 shows the *Uprime2Mean XX* in a cylindrical view, which highlights the zones where differences between the instantaneous and the average velocities are present.

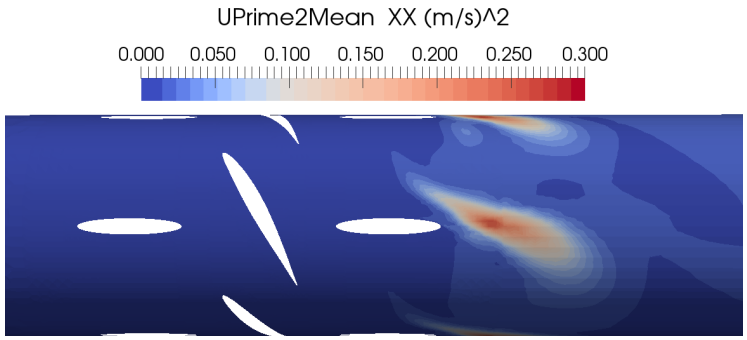


Figure 4.13: *Uprime2Mean XX* in downstream direction, over total of 1000 iterations.

The region in the wake of the supports, highlighted in Figure 4.13, is a typical region where shedding could be expected, especially due to the misalignment between the supports in row three and the incoming flow. This further strengthens the conclusion that the fluctuations in the simulations are of a physical nature.

Figure 4.14 shows the average static pressure field across the turbine. Here it is interesting to see where the majority of the static pressure drop occurs and if some parts of the turbine causes a higher static pressure drop than others.

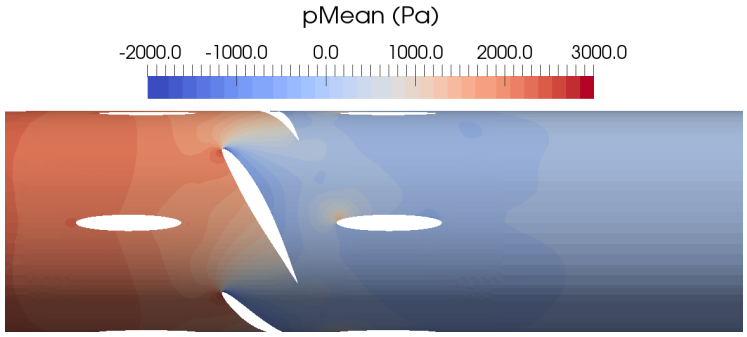
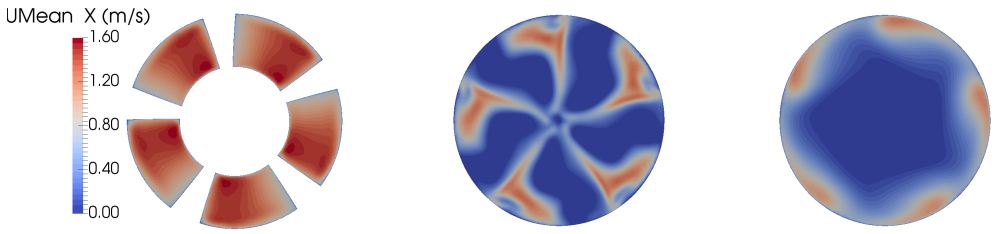


Figure 4.14: Averaged static pressure field over 1000 iterations.

From Figure 4.14 it may be seen that the static pressure starts to drop at the first blade row but that the majority of the static pressure drop takes place across the rotor in row two. The third row does not have any significant impact on the static pressure, though a small pressure increment is present on the leading edge. The static pressure drop between the inlet and the outlet is $2189.0Pa$ for the pilot design.

The swirl in the wake of the turbine, created when using the frozen rotor method is unphysical, as mentioned in chapter 1, it is however of interest to see the cross sectional flow patterns at different locations in the pipe. Figure 4.15 shows the axial flow field in three interesting cross sections.

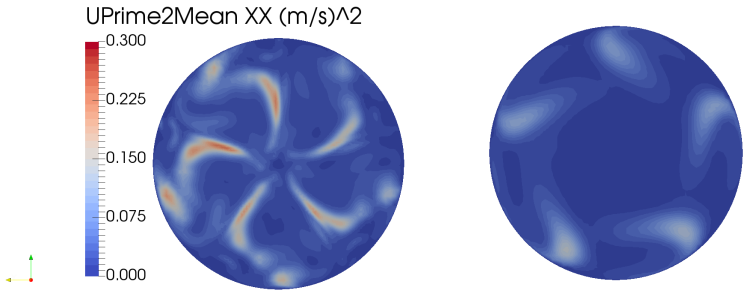


(a) At blade row two. **(b)** Behind blade row three. **(c)** Downstream.

Figure 4.15: Mean axial velocity at three different cross sections.

It may be seen from Figure 4.15a that, as expected, the velocity is the highest at the suction side of the blades, and lower on the pressure side, with a maximum of approximately 1.6 m/s located close to the hub. Figures 4.15b and 4.15c, show the swirl behind row three and further downstream. These swirls are unrealistic, as mentioned, but do however show that complex flow structures persist downstream of the turbine, which may causes fluctuation.

It is interesting to analyse the wake and the fluctuations further. Figures 4.16a and 4.16b show the *UPrime2Mean XX* for two cross sections.



(a) Behind blade row three. **(b)** Downstream.

Figure 4.16: *UPrime2Mean XX* at two different cross sections.

By comparing Figure 4.16 to Figure 4.15 it can clearly be seen that the fluctuations follow the swirl patterns generated by the turbine. There seem to be two kinds of phenomenon, fluctuations due to shedding from blade row three and fluctuations due to swirl vortices from the rotor blades. The latter one propagates far downstream.

A qualitative visualisation of how the fluctuation extend downstream can be seen in Figure 4.17.

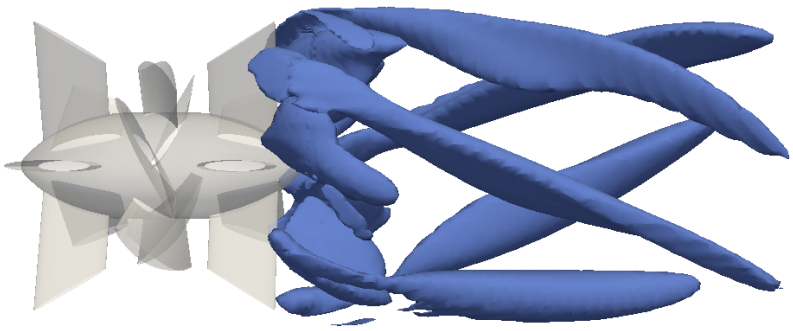


Figure 4.17: Isosurface of *UPrime2Mean*= $0.05m^2/s^2$ in the trail of the turbine.

What is interesting to see in Figure 4.17, is that the majority of the fluctuation do not appear by the blades, where the poor mesh cells are located; rather it is placed in the wake behind the turbine. However, there are fluctuations in the turbine region too, but they are a magnitude lower; thus they are not visible in the figure.

4.5.1 Casing Simplification Validation

The impact of the geometrical simplifications is evaluated by comparing the static pressure drop obtained from the different casings. Figure 4.18 shows the fluid domain for the modelled pilot casing (left), the fluid domain for the realistic modelled pilot casing, which is simplified by sealing the sides (center) and the inside of the realistic modelled pilot casing, which shows the wetted area (right). Figure 4.19 shows the axial velocity for both casings. The casings are simulated with the $k-\omega$ SST turbulence model on low Reynolds number meshes using low Reynolds number wall functions. No rotation or additional parts of the turbine are included.

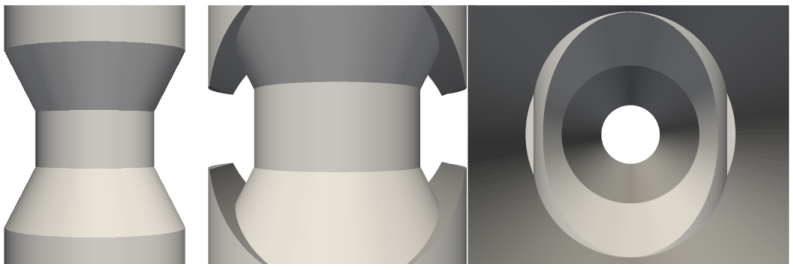


Figure 4.18: Computational domain for modelled pilot casing from above (left), realistic modelled pilot casing with sealed side leakage from above (center) and its inside (right).

The realistic model experiences a 3.78% higher static pressure drop than the modelled pilot casing. A higher static pressure drop is expected for the realistic casing with additional geometries that introduce losses. Apart from the areas where the geometry differs, the casings show similar flow fields, see Figure 4.19 and the simpler modelled pilot casing is adopted for the remaining work.

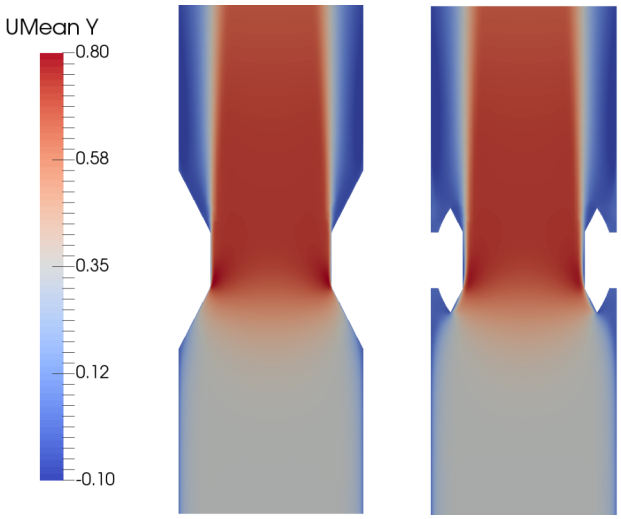


Figure 4.19: Axial mean flow field of the two casing models. Modelled pilot casing (left) and realistic modelled pilot casing (right).

One of the main differences in the flow with the cone approach in the simpler casing is that the boundary layer from the pipe wall is carried in to the turbine, whereas in reality a free shear layer connects the pipe wall with the edge of the inlet cone. In both cases the boundary layer growth is suppressed by the acceleration caused by the converging inlet cone.

5
Results

In this chapter the results from the optimisation of the turbine are presented and discussed. The design suggested by the optimisation algorithm is further evaluated and compared to the pilot design. The important aspects of the results are also highlighted at the end of the chapter.

5.1 Optimisation

The optimisation has a great impact on the design of the turbine. The pilot design operates at a specific speed of $\Omega_s \approx 0.7$ which is in the typical range for Francis turbines. The final design after optimisation, however, operates at a specific speed of $\Omega_s \approx 1.9$, which is closer to the typical range for propeller turbines [17]. In this section, an overview of the optimisation progression is presented, followed by a more detailed analysis of the most prominent designs. For the optimisation overview, the efficiency is mistakenly calculated with the static pressure drop, instead of the total pressure drop. Revisiting the calculations, and correcting for the best ten designs, alters the efficiency by up to 0.5 percentage points and the objective function up to 5 ‰. This does not affect the ranking of the best designs and is considered not to influence the optimisation.

5.1.1 All Designs

Figure 5.1 shows an overview of the results from the optimisation (left) and a zoomed in view (right). The dots represent each evaluated turbine design and the numbers the evaluation order. Further information is given in the figure caption. As can be seen by comparing the performance of the original design, indicated by the green cross, to the final design, indicated by the blue plus, the characteristics of the turbine are altered notably. At the same time as the requirement on power output is fulfilled, the efficiency is increased by 4.4 percentage points and the static pressure drop is reduced by 72% compared to the pilot design. It is interesting to note that many of the sample designs are in the bottom left corner, among the good designs. As an example, iteration five is surprisingly close to the best point, iteration 102. However, the designs produced during the optimisation process are prominent among the best designs.

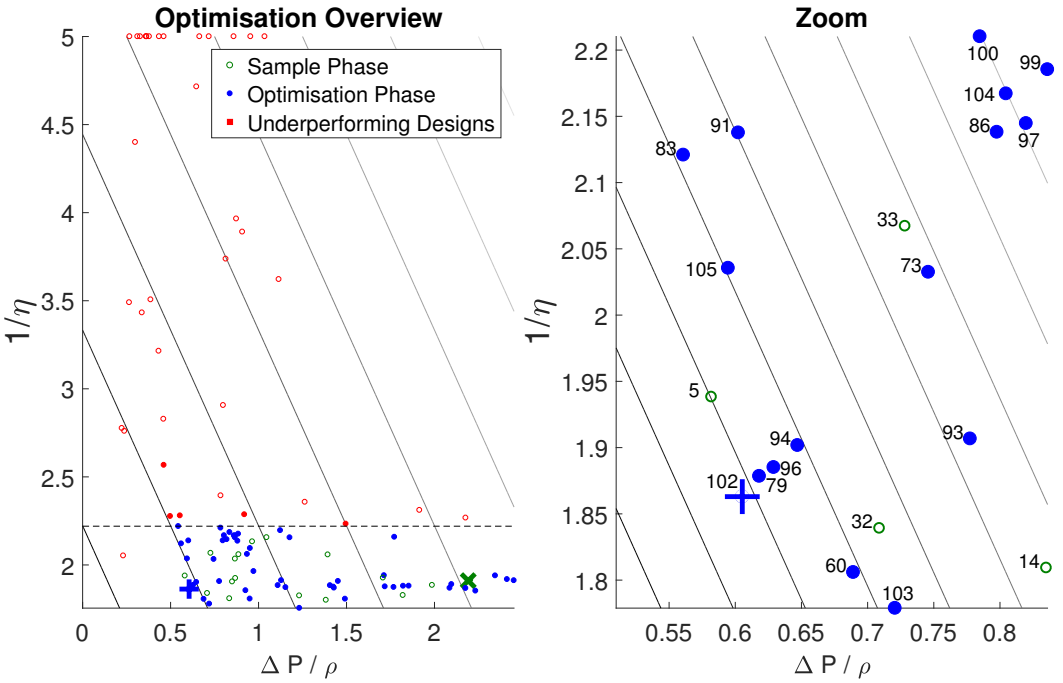


Figure 5.1: Each data point represents one evaluated turbine design. Red designs fail to fulfil one or more of the constraints on minimum efficiency (horizontal line), maximum kinematic pressure drop (outside picture) or minimum work output. The diagonal lines show the trade-off between the minimised normalised objectives kinematic pressure drop and efficiency, 3 to 1. The X highlights the pilot design and $+$ the final design. For visualisation, the efficiencies below 20% were cropped off.

After 112 iterations the optimisation stopped because the expected improvement reached its stopping criteria. The optimisation progression may have been slowed down due to the fact that the forces were acting on the blades in the opposite direction for some of the designs. This is not accounted for explicitly in the learning process, but is included during the optimisation phase. This mistake is likely to affect on the improvement function. However, these bad designs all produce poor efficiency. During the optimisation phase, no turbines with opposite forces are detected.

It can be seen from Figure 5.1 that the designs cluster in the bottom left corner. That is where the objective function gains its minimum value and is the desired location. When using evolutionary type algorithms for multi-objective optimisation, it is expected that the results line up along a trade-off line where several different designs perform equally well. The EGO algorithm, used in the present work, stops when no further improvement is expected and does therefore never show sharp trade-off behaviours.

Figure 5.2 visualises the coverage of the design space during the entire optimisation, complimented by Figure 5.3 for the integers. Important to note is that the variable values seen in the figure are the ones, which Dakota outputs. Thus the lines representing the rotor pitch do not show the actual pitch, rather they show any pitch

modification, see section 3.1.2.3.

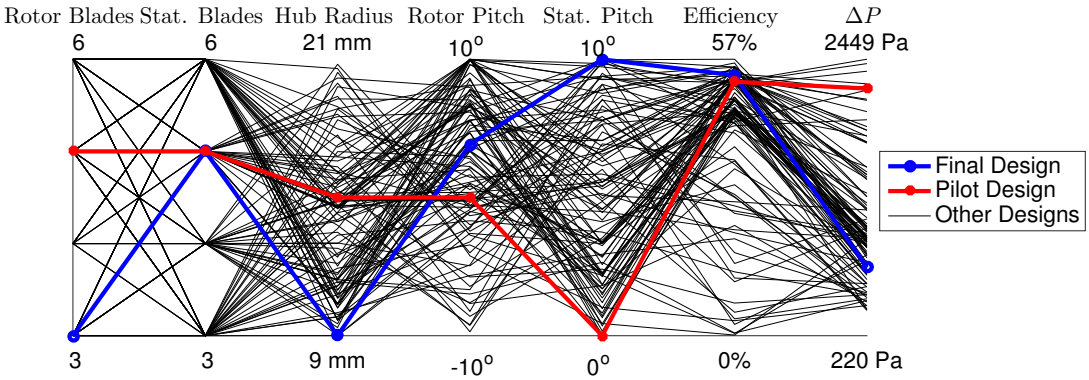


Figure 5.2: Variable spread for all designs in the design space. One line per design with one node per variable.

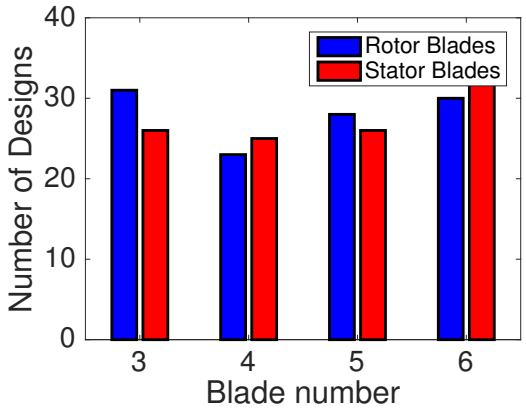


Figure 5.3: Variable spread of all designs in the design space for integer variables.

As can be seen in Figure 5.2, there is not a lot of white space for the floating design variables and the integers are well spread, as seen in Figure 5.3. The coverage is therefore in some sense global and spans the entire design space. A denser cluster, also seems to be emerging where the algorithm makes many similar tries, all with a small hub radius and a high rotor pitch. However, the pitch of the pre-swirl stator blades seems to be more spread out. The best evaluation point seems to be in the outskirt of this cluster.

If only considering the best ten designs, the images change character, see figures 5.4 and 5.5.

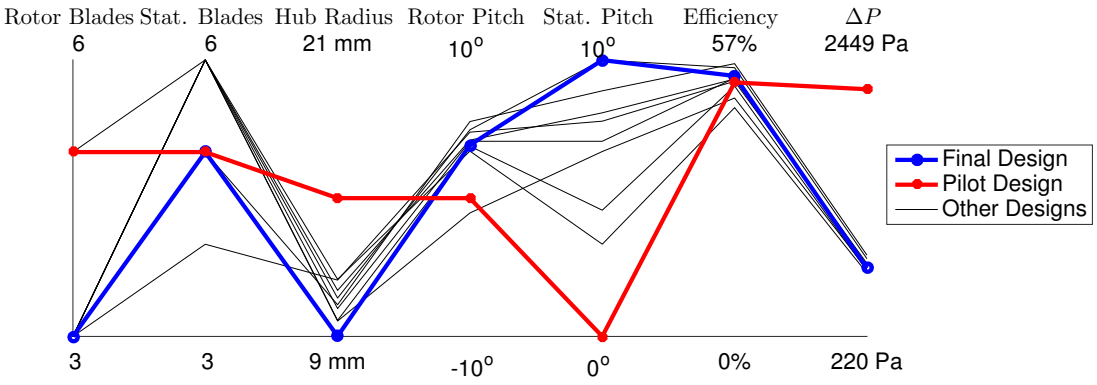


Figure 5.4: Variable spread for the best ten designs in the design space. One line per design with one node per variable.

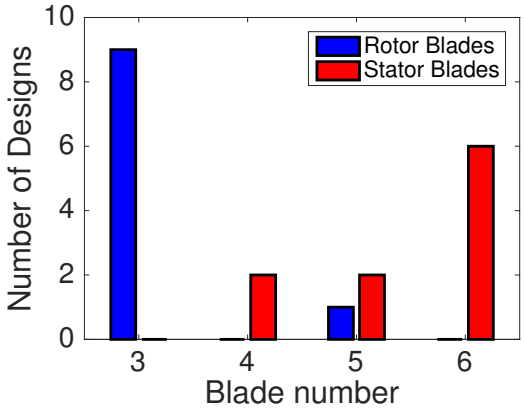


Figure 5.5: Variable spread for the best ten designs in the design space for integer variables.

Trends for good designs can be identified in Figure 5.4. As can be seen, most designs have rotors with three blades, while there is more uncertainty regarding the number of pre-swirl stator blades. A trend of a small hub radius, to minimise the obstacle for the flow and thus lower the static pressure drop, is clear. All designs show a slight rotor pitch modification and again there is inconsistency with respect to the pre-swirl stator pitch. It is surprising that the final design deviates slightly from the trend, especially regarding the pre-swirl stator. Nonetheless, it can be concluded that three rotor blades, a low hub radius and a slight increase to the pitch of the rotor are probable traits for a good design. The impact of the pre-swirl stator blades should, however, be further investigated both regarding number of blades and the pitch.

5.1.2 Best Designs

The results in terms of efficiency, static pressure drop, work output and objective function for the ten best designs are shown in Table 5.1. As mentioned in the introduction to this chapter, the efficiency of the turbines is calculated wrongly during

the optimisation. The table shows the corrected values for both efficiency and objective function within brackets. What can be noted is that the best designs do not have the highest efficiency or the lowest static pressure drop, but rather a good combination of both.

Table 5.1: Performance of the ten best designs, corrected values in brackets

Design Number	Efficiency [%]	Static Pressure Drop Δ P [Pa]	Work Output [W]	Objective Function [-]
102	53.7 (54.1)	605.1	0.62	1.44 (1.44)
5	51.6 (52.0)	581.7	0.57	1.45 (1.45)
79	53.2 (53.7)	618.0	0.63	1.46 (1.46)
96	53.0 (53.4)	628.9	0.64	1.48 (1.47)
60	55.4 (55.8)	689.0	0.73	1.50 (1.50)
94	52.6 (53.0)	646.9	0.65	1.50 (1.50)
105	49.1 (49.5)	594.5	0.56	1.51 (1.50)
83	47.2 (47.5)	560.6	0.50	1.52 (1.51)
103	56.2 (56.7)	720.5	0.77	1.52 (1.51)
32	54.4 (54.9)	708.7	0.74	1.54 (1.53)

The highest efficiency observed during the optimisation is 57%, which is three percentage points higher than the best design. Considering that the highest efficiency design did not make the top ten list, see Table 5.1, it is remarkable to note that the lowest one that did operates with an efficiency as much as ten percentage points lower than the highest one. The static pressure drop among the best ten designs varies similarly with +19% and −7% compared to design number 102. The work output of the best designs show interesting results as even though the kinematic pressure drop is minimised, the power of the good designs are all well above the minimum constraint value of $0.27W$. This indicates that the geometrical constraints of the design as well as other constraints prevent the work output from reducing further. The values can be compared to the work output of the pilot design which is just over $2W$. The results are very dependent on the formulation of the objective function. A different selection of the objective weights will likely result in a different best point. This can clearly be noted in the comparison between the best and the second best designs. Design number 5 produces lower static pressure drop than design number 102, but vice versa for the efficiency.

The percentage difference of the variables between the pilot design and the ten best ones are tabulated in tables 5.2 and 5.3. From this, the similarities and differences between the best designs can be analysed further.

Table 5.2: Differences between the pilot design and the ten best ones for the rotor.

Design Number	Blades Rotor	Pitch Hub [%]	Pitch Band [%]	Hub Radius [%]
102	3	-29.7	+7.7	-39.9
5	3	-28.4	+5.7	-35.3
79	3	-27.1	+5.9	-31.9
96	3	-25.6	+4.0	-26.7
60	3	-25.4	+8.3	-28.9
94	3	-24.1	+5.1	-23.7
105	5	-32.5	-11.4	-35.6
83	3	-25.6	-0.5	-23.5
103	3	-25.3	+12.0	-30.8
32	3	-28.4	+14.3	-39.8

Table 5.2 clearly shows a trend for a low number of rotor blades. It is interesting however to see that the one exception, design number 105, is not in the bottom part of the list, as it is the fourth last. Since the hub radius directly alters the blade speed at the hub, as the blade speed is proportional to the radius, the hub reduction can also be seen in the modification of the rotor hub pitch. The smaller hub also reduces the speed of the flow, since the annular area becomes bigger, and thus it affects the alignment between the flow and the rotor blades at the band too.

In the pilot design, no pre-swirl blades are included, only supports to hold the turbine in place. As mentioned in section 3.1.2.2, a pre-swirl profile is included in the optimisation. Table 5.3 shows the settings for the pre-swirl blades in the ten best designs.

Table 5.3: Differences between the pilot design and the ten best ones for the pre-swirl stator.

Design Number	Blades Stator	Pitch Hub and Band [°]
102	5	10.0
5	6	4.6
79	6	7.1
96	6	8.1
60	6	10.0
94	4	7.8
105	6	6.7
83	6	3.3
103	5	8.9
32	4	7.1

Table 5.3 shows that six pre-swirl stator blades is the dominating setting for the best turbines. However, the results are inconclusive regarding the pitch of the blades. The big spread in pre-swirl stator pitch can mean that other variables influence the

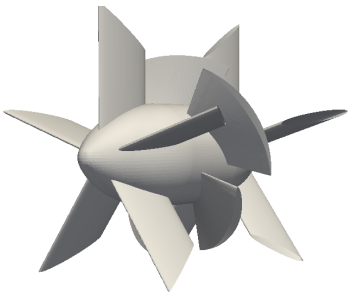
performance of the turbine to a greater degree, as the algorithm does not reach a consistent conclusion about the setting. The pre-swirl is introduced with the ambition to increase the efficiency of the turbine, but it does, however, also introduce losses which affect the static pressure drop. In the final design, the algorithm seems to have found a good trade-off, regardless of the large spread, by giving a high pre-swirl angle but with fewer blades to reduce surface losses.

5.2 Evaluation of Final Design

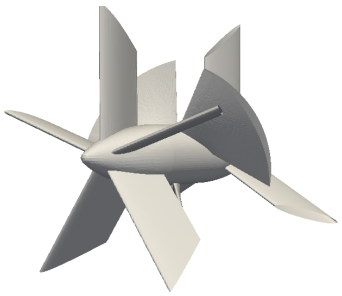
In order to find the truly best design, further analysis is needed. The simulations are only models of reality and and their validity still needs to be tested. For the case of thoroughness, such analysis should be carried out for several promising designs to validate the results of the algorithm. Due to lack of time, such analysis is however only carried out for the most promising design, number 102, which is presented in this section.

5.2.1 Geometry Comparison

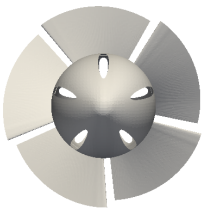
Figure 5.6 show a visual comparison between the pilot design and the final design.



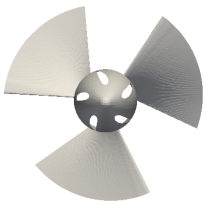
(a) Pilot design, side view, including stator, rotor and support blades.



(b) Best design, side view, including pre-swirl stator, rotor and support blades.



(c) Pilot design, front view, only showing rotor blades.



(d) Best design, front view, only showing rotor blades.

Figure 5.6: Visual comparison between the pilot and the final design.

Between figures 5.6a - 5.6b, the most obvious differences are the number of blades and the hub radius. If the difference is analysed thoroughly, from upstream to downstream of the turbine, some additional features can be identified. The pilot design has five straight ellipsoidal stator blades, whereas in the improved designs, the stator profiles are NACA airfoils. Dakota keeps the same number of blades, but adds a large pre-swirl. The hub radius is smaller in the final design, which increases the annular area. Since the length of the hub cone is constant, it becomes sharper when the radius is reduced. A small hub radius is assumed to be the biggest contributor to the low static pressure drop. The number of rotor blades is decreased from five to three. Since the blades were limited in the axial direction to not extend further than the hub, see section 3.1.2.3, the lower number of blades also comes with a bigger spacing between the blades. If the chord length is allowed to vary more freely, the final design may look different and thus with a bigger or smaller gap.

5.2.2 Rotor Position for the Final Design

As mentioned in section 4.3, the rotor was placed at the same rotational position for all designs. Depending on the number of rotor and stator blades and how they align with each other, this favours some designs more than others. Several rotor positions are therefore simulated for the final design, number 102.

Figure 5.7 shows the results for the different rotor positions. Just as with the pilot design, the space between the rotor blades is divided into five additional positions. For the three bladed rotor there is now rotational symmetry every 120°, ranging from the original 36° to 136° with 20° increments. As can be seen in Figure 5.7, there is some variation in the efficiency and the best objective function belongs to the position used in the optimisation. The variation in the objective function between the best and worst positions is however only 3% and the mean value would still place design number 102 as the second best compared to the other one-position results.
The efficiency drop in Figure 5.7 for rotor orientation of 76° is interesting, this is further evaluated in in Appendix D.

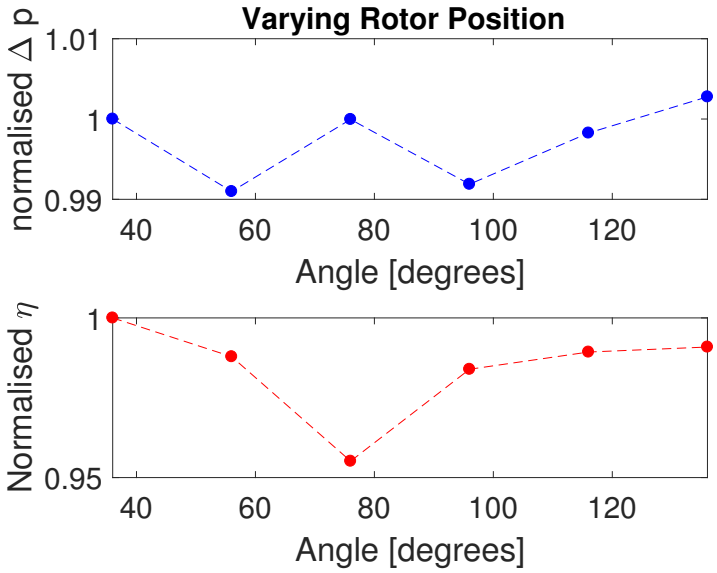


Figure 5.7: Encompassing the full rotation for a three bladed rotor in steps of 20°. Static pressure (above) and efficiency (below).

5.2.3 Flow Fields of the Final Design

The flow fields of the final design are analysed, similarly to the pilot design shown in section 4.5, and a comparison with the pilot design is made in this section. The fluctuations in the simulations are analysed further to see where they are located and how they propagate.

Figure 5.8 shows the pre-swirl generated by the added functionality of the first stator row. The flow is not uniform and two different inter blade fields with higher and lower swirl magnitudes, caused by upstream influences of the three bladed rotor, can be identified. The rotor, which induces a swirl opposite to the pre-swirl, is located behind the patches with lower tangential flow in Figure 5.8. The figure also clearly shows the radial flow that arises from the convergent casing shape.

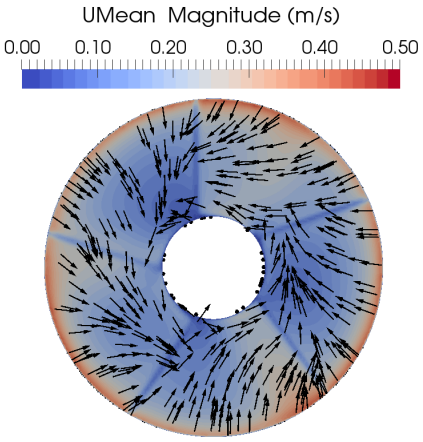


Figure 5.8: Radial and tangential flow in cross section behind first pre-swirl stator rown (contour) and directional vectors (arrows).

Figure 5.9 shows the averaged velocity flow field for the final design, where the arrow vectors show the velocity direction in the cylindrical plane. Note that Figure 5.9 does not have the same colour map as the corresponding pilot design in Figure 4.12b.

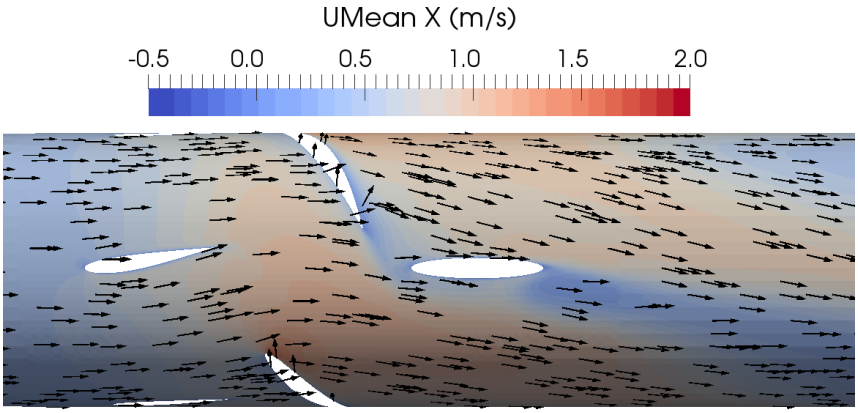


Figure 5.9: Averaged axial velocity field of the final design over 1000 iterations (contour) and directional vectors (arrows).

Since the final design has less rotor blades and a larger spacing between the blades, it guides the flow less than the pilot design. Combined with the introduced pre-swirl, this reduces the exit swirl downstream of the rotor. With a weaker swirl the flow is more aligned with the supports in the third blade row and the shielded wake of the pilot design that can be seen in Figure 4.12b is dramatically reduced. It can also be seen that the smaller hub size reduces the mean flow speed, as the annular area increases.

The unstable flow is analysed in the same manners as the pilot design in Figure 4.13. The phenomenon of interest is the unstable flow due to impingements on the supports in blade row three. Figure 5.10 shows the field of *UPrime2Mean XX* for the final design.

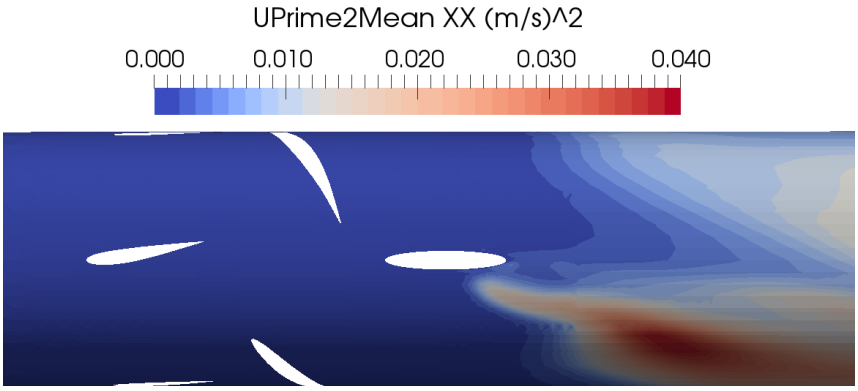


Figure 5.10: *Uprime2Mean XX* in downstream direction, over a total of 1000 iterations.

Though the wake is reduced, it does still exist and as for the pilot design the fluctuations in the simulation are mainly located within it. As can be seen in Figure 5.10, the fluctuation is positioned behind the supports in row three and is large. This is similar to what was shown for the pilot design in Figure 4.13, but about ten times smaller in magnitude. It is noted that the swirling motion extends far downstream.

The average static pressure field for the final design is shown in Figure 5.11.

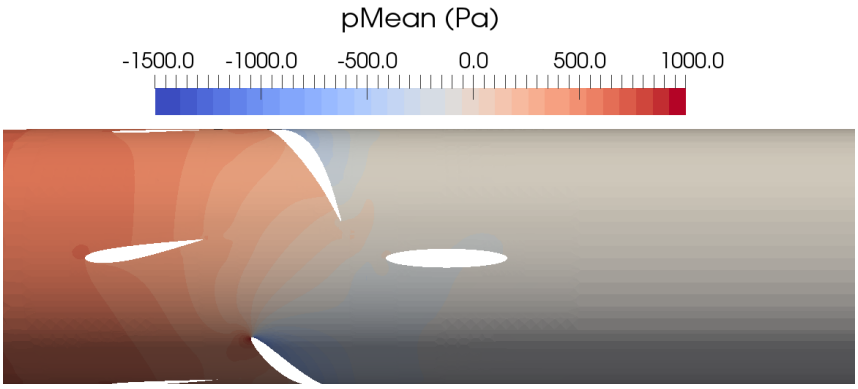


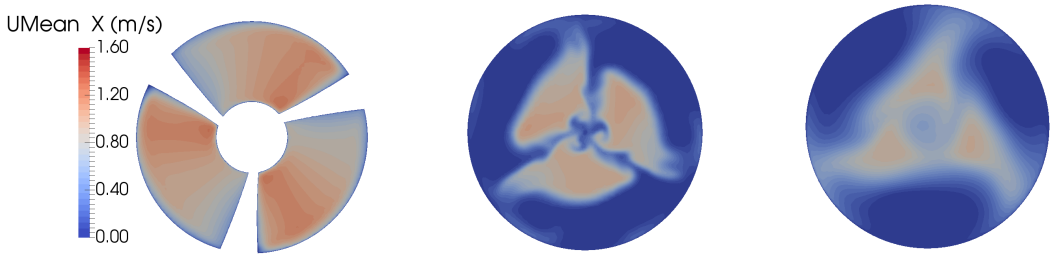
Figure 5.11: Averaged static pressure field over 1000 iteration.

By comparing the static pressure field of the final design, Figure 5.11, with that of the pilot design, Figure 4.14, three main differences are apparent. First and foremost, the static pressure drop is significantly reduced, from 2189.0Pa to 605.1Pa . Secondly, the static pressure drop over the first stator row is proportionally higher for the final design. This is where the pre-swirl was introduced and hence some of the internal pressure energy is converted to kinetic energy when the flow is redirected. Lastly, the high pressure zone at the impingement on the third row is reduced.

The reduction of the impingement shown here in Figure 5.11 is partially due to the rotor position that is used during the optimisation and is not entirely representative for the design. As mentioned in section 5.2.2, that rotor position is the most favourable one for this design. When the rotor position is altered, a higher impingement on the third row is observed, for more details see appendix D.

The favourable alignment between the rotor and the support is something that likely occurs for many three bladed designs with similar rotor pitch and indeed such cases dominate the list of best designs. The previous conclusion from section 5.2.2 that the rotational average still performs among the best designs, however, indicates that the alignment is not the reason why the optimisation turned out as it did.

Figure 5.12 shows the axial velocity at three different cross-sections of the final design. The same cross-sectional locations in the pipe are picked as for Figure 4.15. Again, its worth pointing out that the swirl created in the wake of the turbine is unphysical due to the simulation strategy. It is mainly interesting to compare the different flow patterns obtained from the final design and the pilot design.



(a) At blade row two. **(b)** Behind blade row three. **(c)** Downstream.

Figure 5.12: Mean axial velocity at three different cross sections.

The behaviour at the rotor blade location in Figure 5.12a is similar to what was seen for the pilot design in Figure 4.15a, were the highest speed is at the suction side of the blades. However, a flow speed reduction is noted, mainly due to increment in annular area. Figures 5.12b and 5.12c show the flow behind the turbine and in the downstream pipe, where a different flow behaviour is detected. The highest velocities, present in the swirl motion, are now located closer to the centre of the pipe, and not at the channel walls as for the pilot design, seen in figures 4.15b and 4.15c. This interesting shift in flow behaviour might be due to the lower swirl magnitude caused by the bigger space between the blades and the increased annular area. The centrifugal forces in the flow are thus lower and do not force the swirl towards the walls.

5.3 Propositions for and Discussion of Future Designs

The optimisation clearly shows that the hub radius should be kept small. The effect of a small hub is that the annular area in the turbine is increased. With a larger area, the flow does not accelerate as much, and less of the pressure energy gets converted into kinetic energy. This makes the hub radius variable the most influential one on the flow field. The blades need to be matched with the hub to align with the incoming flow. The optimisation also clearly shows that a low number of rotor blades is desirable, as the suggested amount is at the limit of the design space. When revisiting the Zweifel calculations, see section 2.2.1, on hindsight for the final design, it can prove beneficial to also evaluate two bladed designs. The impact of the pre-swirl blades is positive, it is however hard to conclude an optimum value as there is inconsistency in the results. The reason for this can be that the impact on the performance of the turbine due to other variables is much bigger.

Many variables, other than the ones investigated in the present work, affect the performance of the turbine. As suggested by Dubas [9] the distribution of the propeller pitch is an important parameter that should be suitable for further optimisation.

A cancelled optimisation of rotor twist and thickness distribution carried out on an erroneous design case indeed showed that there is still room for improvement as it increased the efficiency of the turbine by several percentage points.

Since there is inconsistency in the results regarding the pre-swirl stator configuration, it may be interesting to optimise only the pitch of the rotor and pre-swirl stator while keeping the other parameters fixed, to find a more optimal configuration. This can be done using the same objective function or by changing it so that the efficiency is the primary objective. In addition to varying the pitch, it is also interesting to include the chord length and camber of the pre-swirl stator as variables for such a study. If the flow is guided more carefully by a longer pre-swirl stator blade, perhaps a higher level of pre-swirl, which will further cancel the remaining swirl, can be achieved.

Another factor that can be interesting to evaluate regarding the rotor is its axial length. The hub length is fixed in the present work and as a result, all rotors with a high number of blades also have a large blade surface. Perhaps a greater number of small blades could achieve a different trade-off between work output at high efficiency and a low static pressure drop.

6
Conclusion

The main goal of the present work is to optimise a small hydro turbine used in the freshwater system. A fully automated process where multiple software are coupled together in an optimisation loop is developed. The software are all open source and the loop handles the entire process from geometry creation, computational grid creation, CFD simulation, post processing and optimisation. A total of 112 design iterations are evaluated before the optimisation algorithm comes to a stop and no further improvement is expected. The entire process takes about 3500 core hours on an Intel 2650v3 processor. Out of the 112 designs, 51 are sample designs and 61 optimisation designs. The best design, number 102, is analysed further. Compared to the pilot design, the final design performs at 4.4 percentage points higher efficiency and 72% lower static pressure drop, whereas the work output is adjusted to a more suitable level for the current application by a reduction of 70%.

The constructed automatic loop is considered successful and achieves the initial goals of being computationally cheap, based on open source software and flexible such that it can be used in the future for similar evaluations. The software selected for the loop all have individual advantages and disadvantages.

Ruby is a good tool when it comes to creating a fully parametrised geometry of the system. It allows for precise generation of the turbine, though its complexity increases rapidly when adding complex geometrical features and control.
CfMesh is a very simple and, at the same time, robust mesh generator which handles imprecise geometries. In this work, due to certain complex shapes, the mesh quality is hard to control at some locations. Though the initial mesh is set up very easily, this results in a great amount of time spent on fine tuning the settings to achieve higher quality.
OpenFOAM is a powerful tool for simulations of turbines and allows for a simple implementation when post processing results. The frozen rotor approach is implemented though it is an unrealistic way of modelling the unsteady flow. It is however assumed to give an acceptable resemblance of engineering quantities in order to find general design principles. It is highly recommended that a validation on the results is carried out in experimental tests later on. The $k-\omega$ SST turbulence model shows to be the most suitable model for the simulations, however unstable behaviour is detected in the flow. It is concluded to be of a physical nature and is treated with averaging.
Dakota is a versatile optimisation toolbox and has several different algorithms suitable for a variety of optimisation problems. Great documentation makes the software

easy to use. How the problem is constraint in the present work and how the variables are manipulated might have slowed the progression during the optimisation.

6.1 Future Work

Several aspects of the procedure can be done differently and the outcome of the present work is a result of the choices that the method builds on. Apart from changing the formulation of the problem to include more functionality, the entire procedure also has to be validated with experimental data. As discussed in section 3.3.1, no such comparison is made in the present work due to the poor quality of the available data.

As mentioned, a great amount of time is spent meshing the geometry, due to some errors and quality issues close to complex geometrical features. Partially, these are the result of how the different geometrical parts are constructed separately, without joining the surface nodes before meshing. In order to resolve this issue, a different approach to the geometry creation, likely with other software, is necessary.

The steady state simulations in the present work do not converge to a steady solution. A fluctuation is present, but it is concluded to be of a physical nature. Though the quasi unsteady methods presented in section 4.3 gives some insight in the unsteady behaviour, time dependant transient simulations would resolve additional features of the flow and are needed in order to better be able to predict the performance of the turbine. The computational cost of such methods make them unsuitable for most optimisation, but they can be used as a first step to validate the resutls presented here.

Another factor that is not touched upon in the present work is the casing design. The leakage through the narrow channels on the sides where the casing meets the pipe can have a significant impact on the performance of the turbine. Matching the casing to the the water pipe such that these channels are eliminated is likely favourable. The leakage around the runner is also a big source of losses, especially for small turbines where the leakage flow is often proportionally big compared to the total flow. In order to more accurately predict the performance of the turbine, additional capabilities to deal with leakages should be added to the method

The optimisation in the present work is only carried out at one single operating point. The water flow in the distribution network does however vary a lot. The ideal turbine therefore performs well at several operating points and a method where optimisation for several conditions can be carried out simultaneously can achieve a more optimal design for the real application. See Appendix E for more details on the off design performance of the pilot design in.

The optimisation approach implemented in the present work stops when no further improvement of the design is expected. In order to better understand the complex relation between the design variables, and how the trade off between them works,

a different optimisation approach is needed. Other, typically more expensive, algorithms, such as evolutionary methods, that finds several equally good designs, could be investigated for more comprehensive results.

Bibliography

[1] R. Puust, Z. Kapelan, D. A. Savic, and T. Koppel, "A review of methods for leakage management in pipe networks," *Urban Water Journal*, vol. 7, no. 1, pp. 25–45, 2010. doi:10.1080/15730621003610878.

[2] B. Berqvist, "A parametric turbine using openfoam+cfmesh+sciptsfor the geometry," 2015. Accessed 19.jan.2017 from `http://www.tfd.chalmers.se/~hani/OFGBG15/parametric_turbine_openfoam_meeting.pdf`.

[3] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global Optimization*, vol. 13, p. 455–492, 1998.

[4] B. M. Adams, M. S. Ebeida, M. S. Eldred, J. D. Jakeman, K. A. Maupin, J. A. Monschke, L. P. Swiler, J. A. Stephens, W. J. Bohnoff, K. Dalbey, J. Eddy, R. W. Hooper, K. T. Hu, P. Hough, E. Ridgway, and A. Rushdi, *Dakota, A Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and Sensitivity Analysis: Version 6.4 Theory Manual.* Albuquerque, New Mexico 87185: Sandia National Laboratories, 2016.

[5] N. Hanasaki, S. Fujimori, T. Yamamoto, S. Yoshikawa, Y. Masaki, Y. Hijioka, M. Kainuma, Y. Kanamori, T. Masui, K. Takahashi, and S. Kanae, "A global water scarcity assessment under shared socio-economic pathways – part 2: Water availability and scarcity," *Hydrol. Earth Syst. Sci.*, vol. 17, no. 7, pp. 2393–2413, 2013. doi:10.5194/hess-17-2393-2013.

[6] S. Eyuboglu, H. Mahdi, and H. Al-Shukri, "Detection of water leaks using ground penetrating radar," in *Proceedings of the Third International Conference on Applied Geophysics, Orlando-FL*, pp. 8–12, Department of Applied Science University of Arkansas at Little Rock, 2003.

[7] J. Kaufmann and V. Bertram, "Comparison of multi-reference frame and sliding-interface propeller models for ranse computations of ship-propeller interaction," in *Proceedings of the 14th Numerical Towing Tank Symposium*, University of Southampton, 2011.

[8] O. Petit, *Towards Full Predictions of the Unsteady Incompressible Flow in Rotating Machines, Using OpenFOAM.* Phd thesis, Chalmers University of Technology, SE-412 96 Gothenburg, 2015. ISSN 0346-718X.

[9] A. J. Dubas, *Robust Automated Computational Fluid Dynamics Analysis and Design Optimisation of Rim Driven Thrusters.* Phd thesis, University of Southampton, 2014.

[10] S. Lee and J. Lee, "Aerodynamic analysis and multi-objective optimization of wings in ground effect," *Ocean Engineering*, vol. 68, pp. 1–13, 2013.

[11] M. H. Mohamed, G. Janiga, E. Pap, and D. Thévenin, "Multi-objective optimization of the airfoil shape of wells turbine used for wave energy conversion," *Energy*, vol. 36, pp. 438–446, 2011.

[12] X. Liu and R. C. Bansal, "Integrating multi-objective optimization with computational fluid dynamics to optimize bouler combustion process of a coal fired power plant," *Applied Energy*, vol. 130, pp. 658–669, 2014.

[13] Ö. Öksu and İ. S. Akmandor, "Multi-objective aerodynamic optimization of axial turbine blades using a novel multilevel genetic algorithm," *Jornal of Turbomachinery*, vol. 132, 2010.

[14] B. M. Adams, M. S. Ebeida, M. S. Eldred, J. D. Jakeman, K. A. Maupin, J. A. Monschke, L. P. Swiler, J. A. Stephens, W. J. Bohnoff, K. Dalbey, J. Eddy, R. W. Hooper, K. T. Hu, P. Hough, E. Ridgway, and A. Rushdi, *Dakota, A Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and Sensitivity Analysis: Version 6.4 User's Manual*. Albuquerque, New Mexico 87185: Sandia National Laboratories, 2016.

[15] M. Holm, "Small hydro turbine in freshwater system (image)." CTO at Undisclosed Company, 2016.

[16] L. Stoessel, *Numerical simulations of the flow in the Francis-99 turbine, Steady and unsteady simulations at different operating points*. Master thesis, Chalmers University of Technology, 2014.

[17] S. Dixon and C. Hall, *Fluid Mechanics and Thermodynamics - Turbomachinery*. The Boulevard, Langford Lane, Kidlington, Oxford, OX5 1GB, UK and 225 Wyman Street, Waltham, MA 02451, USA: Butterworth-Heinemann is an imprint of Elsevier, 7 ed., 2014.

[18] F. White, *Fluid Mechanics*. 1221 Avenue of the Americas, New York, NY 10020: McGraw-Hill, 7th in si units ed., 2011.

[19] I. Amin and Q. Xiao, "Numerical simulation of a horizontal axis tidal turbine with a pre-swirl stator," *Developments in Maritime Transportation and Exploitation of Sea Resources*, pp. 863–869, 2014.

[20] L. Davidson, "Fluid mechanics, turbulent flow and turbulence modeling," lecture notes, Chalmers University of Technology, 2017.

[21] O. Petit, M. Page, M. Beaudoin, and H. Nilsson, "The ercoftac centrifugal pump openfoam case-study," in *3rd IAHR International Meeting of the Workgroup on Cavitation and Dynamic Problems in Hydraulic Machinery and Systems*, 2009.

[22] Unknown, "See the mrf development," 2016. Accessed 2017-06-03 from `http://openfoamwiki.net/index.php/See_the_MRF_development`.

[23] L. Davidson, "An introduction to turbulence models," lecture notes, Chalmers University of Technology, 2016.

[24] H. Versteeg and W. Malalasekera, *An Introduction to Computational Fluid Dynamics*. Pearson Education M.U.A, 2nd ed., 2007.

[25] P. Moradnia, *CFD of Air Flow in Hydro Power Generators*. Thesis for licentiate of engineering, Chalmers University of Technology, SE-412 96 Gothenburg, 2010. ISSN 1652-8565.

[26] F. R. Menter, M. Kuntz, and R. Langtry, "Ten years of industrial experience with the sst turbulence model," *Turbulence, Heat and Mass Transfer 4*, 2003.

[27] Unknown, "The menter shear stress transport turbulence model," 2015. Accessed 2017-05-09 from the Langley Research Center Turbulence Modeling Resource on 09 may 2017 at `https://turbmodels.larc.nasa.gov/sst.html`.

[28] F. Russo and N. T. Basse, "Scaling of turbulence intensity for low-speed flow in smooth pipes," *j. Flow Measurement and Instrumentation*, vol. 52, pp. 101–114, 2016. 10.1016/j.flowmeasinst.2016.09.012.

[29] S. Salim and S. Cheah, "Wall y^+ strategy for dealing with wall-bounded turbulent flows," in *"Proceedings of the International MultiConference of Engineers and Computer Scientists 2009"*, pp. 2165–2170, International Association of Engineers, Newswood Limited, 2009. ISBN: 978-988-17012-7-5.

[30] S. Liu, "Implementation of a complete wall function for the standard k-ε turbulence model in openfoam 4.0. in proceedings of cfd with opensource software," tech. rep., Chalmers University of Technology, 2016. Editied by H. Nilsson, `http://www.tfd.chalmers.se/~hani/kurser/OS_CFD_2016`.

[31] W. K. George, "Is there a universal log law for turbulent wall-bounded flows?," *Philosophical Transactions of the Royal Society*, 2007.

[32] S. Guðmundsson, *General Aviation Aircraft Design: Applied Methods and Procedures*. The Boulevard, Langfor Lane, Kidlington, Oxford OX5 1GB, UK: Butterworth-Heineman, 1st ed., 2014.

[33] P. Y. Papalambros and D. J. Wilde, *Optimization Models*, p. 1–43. Cambridge University Press, 2 ed., 2000.

[34] R. T. Haftka and Z. Gürdal, *Elements of Structural Optimization*. Springer Netherlands, 1992.

[35] L. A. Gish, A. Carandang, and G. Hawbaker, "Numerical optimization of preswirl stators for horizontal axis hydrokinetic turbines," tech. rep., Department of Naval Architecture and Ocean Engineering - United States Naval Academy, 2016.

[36] H. Jamshidi, H. Nilsson, and V. Chernoray, "Experimental and numerical investigation of hydro power generator ventilation," in *IOP Conference Series - Earth and Environmental Science: 27th IAHR Symposium on Hydraulic Machinery and Systems (IAHR)*, 2014.

[37] F. Liu, "A thorough description of how wall functions are implemented in openfoam," tech. rep., Chalmers University of Technology, 2016. Editied by H. Nilsson, `http://www.tfd.chalmers.se/~hani/kurser/OS_CFD_2016`.

[38] G. Kalitzin, G. Medic, G. Iaccarino, and P. Durbin, "Near-wall behabior of rans turbulence models and implications for wall functions," *Journal of Computational Physics*, vol. 204, pp. 265–291, 2005.

[39] U. Kaul, "Effect of inflow boundary conditions on the turbulence solution in internal flows," *AIAA j.*, vol. 49, no. 2, 2011. doi: 10.2514/1.J050532.

[40] C. Greenshields, "Openfoam user guide," June 2016. Accessed 2017-06-18 from `https://cfd.direct/openfoam/user-guide/`.

[41] H. Jasak, *Error Analysis and Estimation for the Finite Volume Method with Applications to Fluid Flows*. Phd thesis, Imperial Collage of Science, Technology and Medicine, 1996.

[42] Engys, "A comprehensive tour of snappyhexmesh." 7th OpenFOAM Workshop, June 2012.

A

Folder Structure

For a better understanding of the coupling between the software, a schematic representation of the folder structure is made and shown in Figure A.1. Dakota runs in the top most folder where the scripts that define the optimsation are located. All subfolders needed for Dakota to generate designs and running simulations are located in the `caseTemplate` folder. The folder `geometryCode` contains the scripts and data files needed for the Ruby geometry creation. The folders `0`, `constant` and `system`, contain the information to run OpenFOAM. The file `meshDict` located in system includes all settings for CfMesh to construct the mesh grid. Finally the post processing is carried out in several scripts.

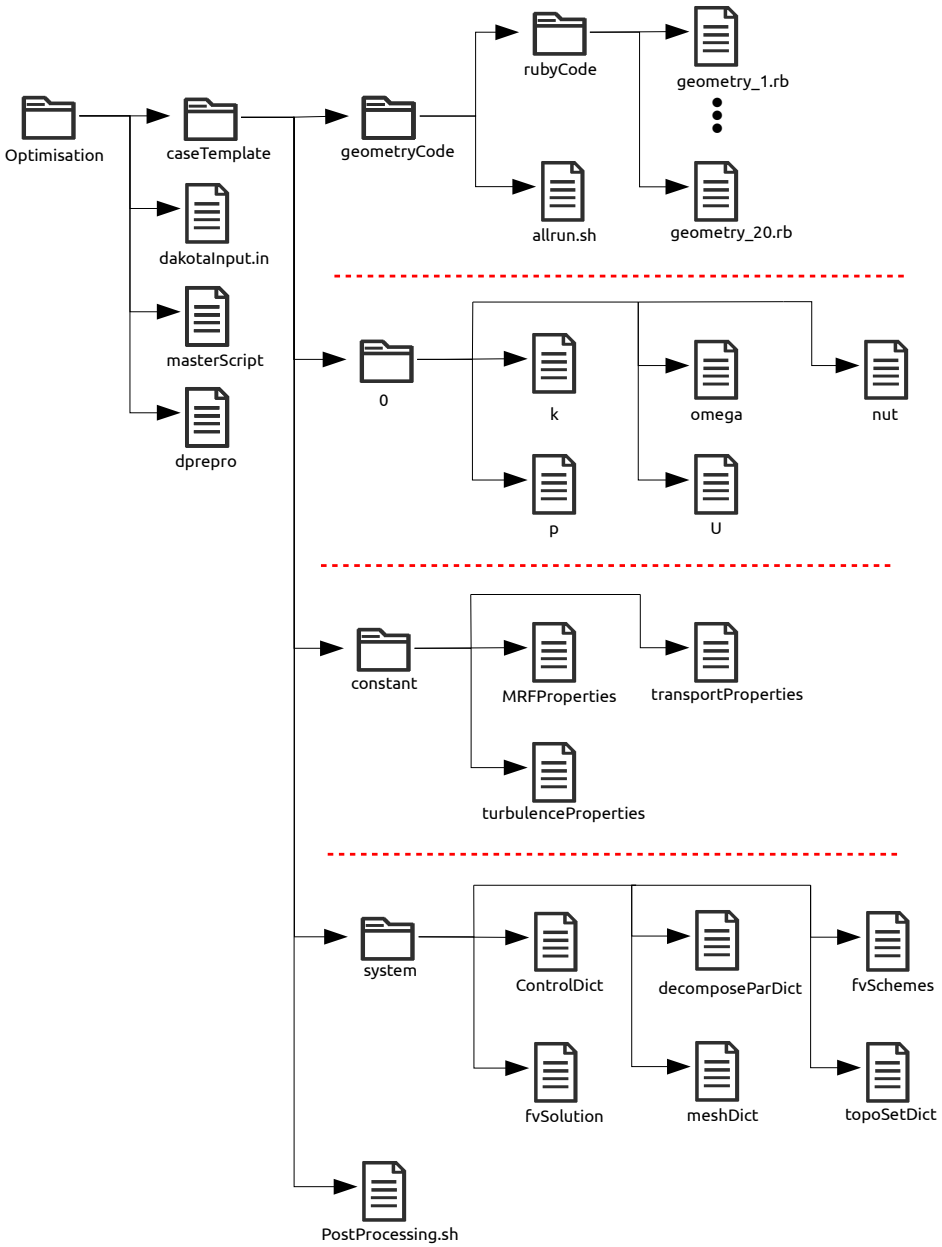
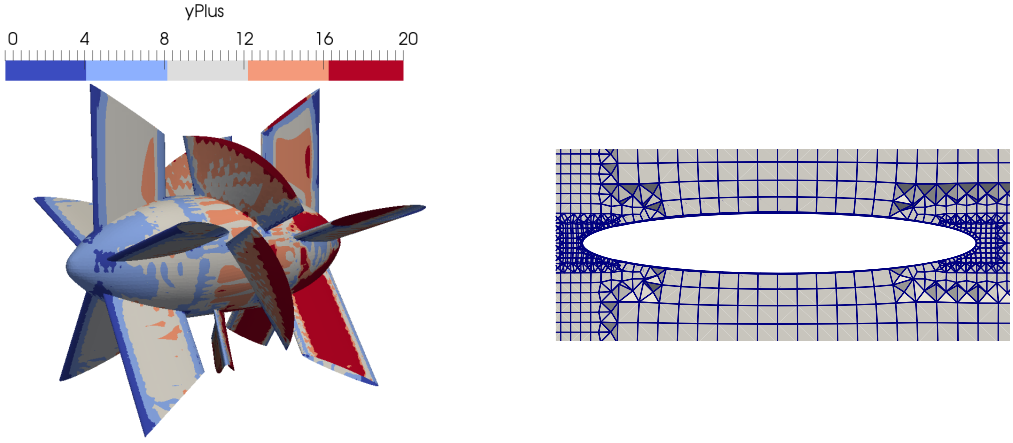


Figure A.1: Folder structure.

B

Preliminary Simulation Test Using $k - \varepsilon$

A computationally cheap simulation approach is sought in the present work and a high Reynolds number model is therefore desired in order to cheaply treat the wall regions. A mesh grid with coarse grid cells on the boundary is constructed and simulated. The high Reynolds number $k - \varepsilon$ turbulence model and high Reynolds number wall functions are used. Figure B.1b shows the mesh resolution at one of the stator blades where the edges are refined for proper geometrical representation. Figure B.1a show the y^+ for the coarse mesh. The *OpenFOAM* utility `yPlus` is used to find the range of the y^+ on the surfaces. .



(a) y^+ values for a high Reynolds number mesh.

(b) Refinement around a stator blade for a high Reynolds number mesh.

Figure B.1: Preliminary simulation.

As seen from the Figure B.1, the values do not fulfil $y^+ \geq 30$ even for a mesh which is coarse on the large surfaces. As the flow speed in the pipe is low, and the application itself is small, very large cells are needed to reach the logarithmic boundary layer above $y^+ = 30$. In order to properly resolve the geometry, a low Reynolds approach is therefore adopted.

C

Additional Mesh Resolutions

Figures C.1 and C.2 show mesh resolutions A1 and A3 respectively in the vertical mid plane. These meshes are used in the internal mesh comparison study, see section 4.2.1. In mesh A1, the refinement in the turbine region is visible. As is the refinement close to the walls in the down stream pipe, where there is swirl. A small bit of the refinement around the stator blades are also visible close to the tips of the hub cones. The stator blades themselves are however pointing out of the picture. Lastly a refinement around the interfaces of the MRF zone is visible. This had to be done in order to ensure that the interface surface turned out axi-symmetric, more on this in section 3.3.2.

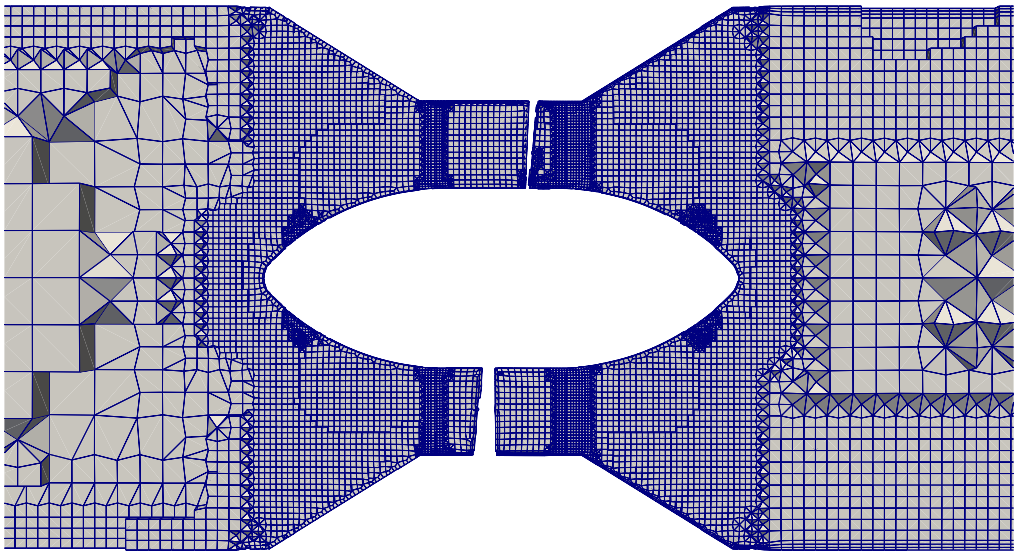


Figure C.1: Mesh A1, flow from left to right.

Mesh A3, seen in Figure C.2. This is the finest mesh used in the internal mesh comparison study, see section 4.2.1. It only has two different areas, the outer pipe refinement and the turbine region refinement. The refinements around the interfaces of the rotational region are no longer visible due to fine refinement in the entire turbine region.

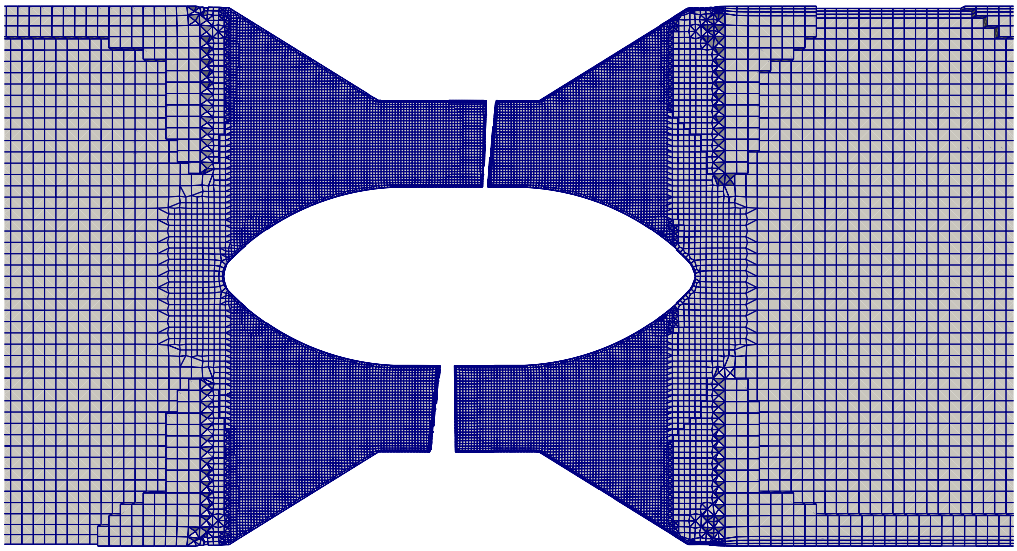


Figure C.2: Mesh A3, flow from left to right.

D

Flow Fields for Varying Rotor Position

The flow fields for the orientation of the rotor, 76°, which showed a reduction in efficiency, see Figure 5.7, are shown bellow. Figure D.1 shows the the velocity flow field in x direction and Figure D.2 shows the static pressure field.

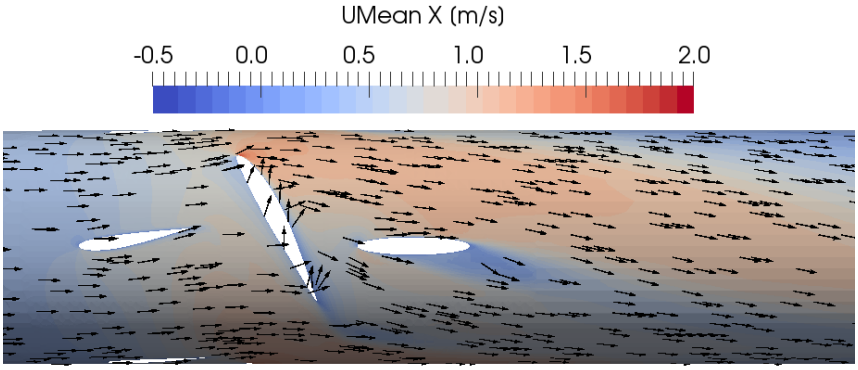


Figure D.1: Average axial velocity field of the final design over 1000 iterations (contour) and directional velocity vectors (arrows), with 76 degree rotation to the rotor.

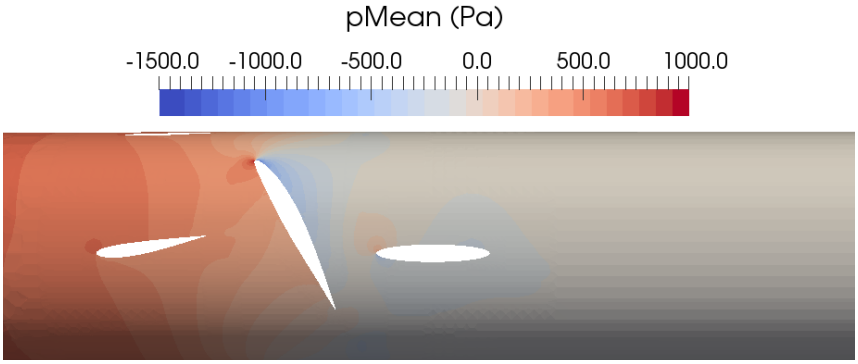


Figure D.2: Static pressure field of the final design after 1000 iterations, with 76 degree rotation to the rotor.

Similar behaviour around the blade is noted in Figure D.1 as shown in 5.8. However, it seems as the support blade in row three detaches the flow on the suction side of the rotor. This might be the reason for lower efficiency for this rotor orientation.

By analysing the static pressure in Figure D.2 and comparing it to 5.11, a pressure increase is noted on the support blade tip in row three, as the flow redirected due to the rotor impinges the support blade. This causes a pressure increment behind the rotor which also can alter the efficiency of the blade.

E

Off Design

As mentioned in section 5.3 it is interesting to not only optimise performance at one operating point, but rather optimise the design for several operational points simultaneously. If such functionality is added to the current optimisation approach, methods to determine the operational points have to be developed. In the present work the turbine is optimised to perform at a desired operating point. The simulations can therefore be over defined to specify exactly this operating point. This is done by setting both the volumetric flow rate and the rotational speed of the turbine and then optimising performance at this point. If other flow conditions are also considered, this method alone is not enough since every turbine design only operates along a certain working line. The relation between volumetric flow and rotational speed depends on the force the water exerts on the blades and the load on the turbine. This relation is unknown and no reliable way to predict the behaviour for a certain turbine geometry and include it in the optimisation is found in the present work.

For the pilot design, there is experimental data of the rotational speed that is used to set several realistic operating points, see flow conditions in Table E.1. Figure E.1 shows the results from those simulations.

Table E.1: Off design flow conditions

Flow speed [m/s]	0.1	0.2	0.3	0.4	0.5
Rotational Speed [RPM]	75.0	266.5	466.5	686.3	882.8

The turbine is designed to perform well at low flow speeds and the efficiency seems stable even for slightly higher flow rates than the nominal 0.3 m/s.

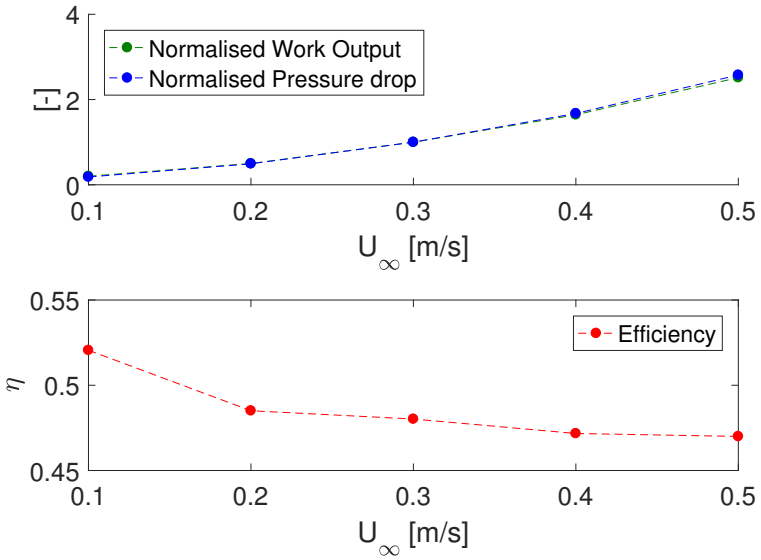


Figure E.1: Off design performane of the pilot design.