NTNU
Norwegian University of
Science and Technology

The influence of the choice of propeller design tool on propeller performance

Edvard Knutsen Skåland

Marine Technology
Submission date: June 2016
Supervisor: Sverre Steen, IMT

Norwegian University of Science and Technology
Department of Marine Technology

NTNU

The Influence of the Choice of Propeller Design Tool on Propeller Performance

Edvard Knutsen Skåland

June 2016

MASTER THESIS

Department of Marine Technology

Norwegian University of Science and Technology

Supervisor: Sverre Steen

NTNU Trondheim
Norwegian University of Science and Technology
Department of Marine Technology

MASTER THESIS IN MARINE TECHNOLOGY

SPRING 2016

FOR

Edvard Knutsen Skåland
The influence of the choice of propeller design tool on propeller performance

There are many different propeller design and analysis methods, with different levels of complexity, like lifting line, vortex-lattice lifting surface, and panel methods. There are also different software where these theories have been implemented. The idea is to investigate the difference in obtained propeller design using different methods and softwares. The plan is for the student to make his own lifting line design program, and then to use OpenProp and AKPD in addition, and use these different softwares to design a limited number of different propellers, and compare the resulting optimum designs. Furthermore, the resulting designs shall be evaluated by the panel method AKPA, as well as using CFD. The objective is to give recommendations regarding what is suitable propeller design software.

For the project thesis the candidate shall, based on a thorough literature study, give an account of different propeller calculation methods. Furthermore the difference between propeller design programs and propeller analysis programs shall be explained, and different theories suitable for making propeller design programs shall be explained. Also, the propeller design process shall be outlined, explaining the propeller design requirements and typical criteria for evaluation. Together, these activities shall form the basis for the master thesis, as explained in the first paragraph.

The project report shall describe the theories applied, approximations and assumptions made, the implementation, the tuning and validation, and give recommendations for a continued development in the master thesis.

In the thesis the candidate shall present his personal contribution to the resolution of problem within the scope of the thesis work.

Theories and conclusions should be based on mathematical derivations and/or logic reasoning identifying the various steps in the deduction.

The candidate should utilize the existing possibilities for obtaining relevant literature, and is recommended to use the library actively.

The thesis should be organized in a rational manner to give a clear exposition of results, assessments, and conclusions. The text should be brief and to the point, with a clear language. Telegraphic language should be avoided.

The thesis shall contain the following elements: A text defining the scope, preface, list of contents, summary, main body of thesis, conclusions with recommendations for further work, list of symbols and acronyms, reference and (optional) appendices. All figures, tables and equations shall be numerated.

The original contribution of the candidate and material taken from other sources shall be clearly defined. Work from other sources shall be properly referenced using an acknowledged referencing system.

The thesis shall be submitted in two copies:

- Signed by the candidate
- The text defining the scope included, signed by the supervisor
- In bound volume(s)
- The bound volume shall be accompanied by a CD or DVD containing the written thesis in Word or PDF format. In case computer programs have been made as part of the thesis work, the source code shall be included. In case of experimental work, the experimental results shall be included in a suitable electronic format.

Supervisor	: Professor Sverre Steen
Start	: 15.01.2016
Deadline	: 10.06.2016

Trondheim, 25.08.2015

Sverre Steen
Supervisor

Preface

This work is the result of my master thesis at the Department of Marine Technology at the Norwegian University of Science and Technology (NTNU) in Trondheim during spring 2016. The thesis is the final part required for the degree of Master of Science.

The thesis was made in order to investigate differences in propeller design using different design tools. It aims to give some recommendations regarding which design tool is most suitable in the design process. The report assumes some preliminary knowledge of marine hydrodynamics and foil theory.

First, I would like to thank my supervisor Sverre Steen for his continuous support and guidance during the work with this thesis. His contributions in terms of sharing helpful opinions on the thesis and on relevant literature have been invaluable. Secondly, I will give my thanks to Senior Research Scientist at MARINTEK Vladimir Krasilnikov for his help and support with STAR-CCM+ and AKPA. I also give my thanks to PhD candidates Bushan Taskar and Øyvind Øksnes Dalheim for assisting me with the OpenProp software. Last I want to thank my family, friends and fellow students who have inspired me and kept my spirits up throughout these five years in Trondheim.

Trondheim, 2016-06-10

....

Edvard Knutsen Skåland

Summary

In this master thesis different propeller design and analysis methods are presented and compared in terms of the accuracy and computational efficiency of their theory. These methods include lifting line, vortex lattice lifting surface and panel methods. A propeller design program based on lifting line theory was developed by the author. This program has been used together with the propeller design programs OpenProp and AKPD to make six propeller designs. The designs are based on two sets of input data, making three designs for each set. Each propeller design has been analyzed for performance in the analysis software AKPA. Cavitation analyses have also been performed. An effort has been made to include a CFD (Computational Fluid Dynamics) analysis as was initially intended. Eventually this is not included due to time limitations and software issues. The objective of the thesis is to give recommendations regarding what is the most suitable propeller software.

The following conclusions could be drawn from the performed analysis on the two design programs utilized in the thesis:

- Based on the propellers designs analyzed in this thesis, OpenProp is able to produce the better designs. Both of the OpenProp propeller designs achieves the highest efficiency as well as showing the least cavitation.

- OpenProp has an advantage in time required to produce a design. It is able to design and run a performance analysis in a matter of seconds. AKPD requires several minutes to produce a full design if the number of unsteady calculation iterations are set to 5 or above (which is recommended by the author for convergence).

- AKPD is the only design tool of the two which is able to account for effects from skew and rake. Skew is often preferred in modern propeller design in order to reduce cavitation, noise and vibrations.

- AKPD is set up for a seamless transition to AKPA. If AKPA is the preferred analysis program, making the designs in AKPD may end up saving time in the design process.

- While both AKPD and OpenProp are restricted to circumferentially averaged inflow, AKPD can account for inflow in the radial as well as the axial and tangential direction. This might be of importance for propellers with high shaft angles or high rake.

Sammendrag

I denne masteroppgaven er flere design- og analysemetoder for propeller pesentert og sammenlignet. Sammenligningen er gjort med hensyn på nøyaktigheten og effektiviteten i teorien bak metodene. Metodene inkluderer løftelinje-metoden, løfteflate-metoden og panelmetoder. Et propelldesign-program basert på løftelinje teori er utviklet av forfatteren. Dette programmet er brukt sammen med designprogrammene AKPD og OpenProp til å generere seks forskjellige propelldesign. Designene er basert på to sett med input-data slik at kordelengde- og tykkelsesfordelingen er lik for halvparten av designene. Hver av de seks propellene er analysert i analyseprogrammet AKPA. Resultatene, i form av forventet ytelse og kavitasjon, er deretter sammenlignet. Det ble gjort et forsøk på å inkludere CFD-analyser (Computational Fluid Dynamics). Dette er ikke med i rapporten på grunn av tidsbegrensningene for oppgaven, i tillegg til flere problemer med programoppsettet. Målet med analysene som er gjort, er å gi anbefalinger for hvilket designprogram som er mest fordelsaktig.

De følgende konklusjonene kunne trekkes fra analyseresultatene:

- Basert på propelldesignene som er analysert i denne oppgaven viser OpenProp seg å være det programmet som gir det beste designet. Dette er både i forhold til propellytelse og minimal kavitasjon.

- OpenProp har en fordel i at det krever langt mindre kjøretid enn AKPD. Programmet kan designe en propell og utføre ytelsesanalyser i løpet av sekunder. AKPD bruker flere minutter på å generere et fullverdig design.

- AKPD er det eneste designprogrammet av de to som tar hensyn til *skew* (buet propellblad) og *rake* (vinkel på bladene i aksiell retning) på propellen. Moderne propeller er ofte designet med *skew* for å redusere kavitasjon, støy og vibrasjoner.

- Både AKPD og OpenProp er begrenset til et innstrømningsfelt basert på gjennomsnittsverdier over omkretsene i propellflaten. I motsetning til OpenProp lar AKPD beregningene ta hensyn til hastighetskomponenten i radiell retning. Dette kan for eksempel ha betydning for propeller med høy rake eller høy vinkel på propellakselen.

Contents

Preface i

Summary ii

Sammendrag iv

1 Introduction **1**

1.1 Motivation and background 1

1.2 Problem formulation 2

1.3 Literature study 2

1.4 State of the art 3

2 Propeller Calculation Methods **5**

2.1 Empirical methods 5

2.1.1 Open water model test 5

2.1.2 Propeller series 6

2.2 Numerical methods 7

2.2.1 Momentum theory 7

2.2.2 Blade element theory 7

2.2.3 Blade element momentum theory 7

2.2.4 Lifting line method 8

2.2.5 Lifting surface method 9

2.2.6 Boundary element method 10
2.2.7 Reynold Averaged Navier-Stokes 11

3 Software and Set-up 12

3.1 Software 12
3.1.1 MATLAB-script 12
3.1.2 AKPD 14
3.1.3 AKPA 14
3.1.4 OpenProp 15
3.1.5 STAR-CCM+ 15
3.2 Set-up for Comparison 16
3.2.1 Design inputs 17
3.2.2 Wake fields 17

4 Results 18

4.1 Validation 18
4.1.1 Performance analysis 19
4.2 Results 21
4.2.1 MATLAB-program design 21
4.2.2 OpenProp design 25
4.2.3 AKPD design 28
4.2.4 Geometry Comparison 31
4.2.5 Performance comparison 33

5 Discussion 37

5.1 Validation 37
5.2 Comparison of Designed Geometry 38

5.3 Comparison of Predicted Performance 39
5.4 Comparison of Cavitation Analyses 40
5.5 Comment on the Lack of CFD Simulation Results 40

6 Conclusion and Recommendations for Further Work **42**
6.1 Conclusion 42
6.2 Recommendations for Further Work 43

Bibliography **45**

A MATLAB code **49**
A.0.1 MAIN.m 50
A.0.2 Propeller.m 52
A.0.3 radialWake_model.m 63
A.0.4 radialWake.m 63
A.0.5 optimizeChordAndCirc.m 64
A.0.6 optimizeCirculation.m 68
A.0.7 goldsteinFactor.m 72
A.0.8 Thickness.m 73
A.0.9 dragCoef.m 74
A.0.10 chordLength.m 75

Nomenclature

Abbreviations

CFD	Computational Fluid Dynamics
BEMT	Blade Element Momentum Theory
BEM	Boundary Element Method
RANS	Reynold-Avereged Navier-Stokes
RPM	Pevolutions Per Minute

List of Symbols

Greek

Acronyms	Units	Description
η_0	$-$	propeller efficiency
σ	$-$	Cavitation number
Γ	$\frac{\mathrm{m}^2}{s}$	Circulation

Lowercase

Acronyms	Units	Description
b	m	Chord length

$f0$	m	Maximum camber
$e0$	m	Maximum section thickness
n	$\frac{1}{s}$	Revolutions per second

Uppercase

Acronyms	**Units**	**Description**
K_T	$-$	Thrust coefficient
K_Q	$-$	Torque coefficient
J_S	$-$	Advance number based on ship speed
J_A	$-$	Advance number based on speed of advance
Z	$-$	Number of blades
P	m	Pitch
D	m	Propeller diameter
A_E	m^2	Expanded blade area
A_0	m^2	Area of propeller disk
V_S	$\frac{\mathrm{m}}{s^2}$	Ship speed
V_A	$\frac{\mathrm{m}}{s^2}$	Speed of advance
C_{Pmin}	$-$	minimum pressure coefficient

Chapter 1

Introduction

The propeller design process has been in steady development since the beginning of the 20th century. Fast development in computational power and the need for higher accuracy models, have led to the implementation of increasingly complex methods in propeller design tools. New software arrives and others are outdated. To find the optimal propeller design one needs the optimal tool. This thesis will provide recommendations regarding what to use, based on an analysis of two existing software.

1.1 Motivation and background

A propeller design complexity vary heavily depending on the application and working condition of the propeller, from standardized merchant vessels to state-of-the-art naval ships. Finding an optimum propeller for each case often proves to be a time consuming process. With numerous approaches available it can be beneficial to harbor some knowledge in terms of their strengths and weaknesses, their assumptions and efficiency. The design process has evolved in pace with the need for, and availability of, higher accuracy methods. The evaluation of the existing design tools may lead to higher efficiency and better decisions on which methods and software applied in future designs.

1.2 Problem formulation

There are many different propeller design and analysis methods, with different levels of complexity, like lifting line, vortex-lattice lifting surface, and panel methods. There are also different software where these theories have been implemented. The idea is to investigate the difference in obtained propeller design using different methods and softwares. The plan is for the student to make his own lifting line design program, and then to use OpenProp and AKPD in addition, and use these different softwares to design a limited number of different propellers, and compare the resulting optimum designs. Furthermore, the resulting designs shall be evaluated by the panel method AKPA, as well as using CFD. The objective is to give recommendations regarding what is suitable propeller design software.

1.3 Literature study

In a project thesis written during the fall of 2015, the author conducted a literature study on propeller calculation methods (Skåland, 2015). The propeller theory and calculation methods presented in this master thesis are based mainly on the literature study carried out in the project thesis.

A huge amount of effort has been put into the research of propeller design and analysis over the years. Since the screw propeller was introduced as a propulsion device on ships in the 18th century numerous researchers and engineers have contributed in the work of finding the optimal propeller design procedure. Rankine (1865) and Froude (1889) made a first attempt with the *actuator disk*. This was an idealization of the propeller as a permeable disk introducing a pressure jump on the fluid flow passing through it. In its simplest form, this can be considered as the limiting case of a propeller with infinite blades with zero hub radius and uniform radial distribution of circulation on the blades. Prantl (1921) used an approximate method to find a solution for the circulation distribution that satisfied what is now knows as the "Betz condition". The condition states that the relationship between the blade pitch angle of the undisturbed inflow and the induced inflow be radially constant for optimum circulation distribution. Goldstein (1929) later found an exact solution to the problem. Prantl and Tietjens (1934) introduced a lifting line method to determine the lift generated by foils with high aspect ratio. The first vortex lattice method was developed by Falkner (1947) which decided the blade chord into equally spaced panels. Glauert (1947) presented a method to find the induced velocities on a lifting line by a 2D vortex distribution. Lerbs (1952) extended this method to develop a lifting line theory for a propeller in radially nonuniform inflow and arbitrary circulation distribution. With his work followed the "Lerbs criterion" which related the pitch angle of the undisturbed inflow and the wake adapted inflow. Years later, Wrench (1957) improved the accuracy and efficiency of this method. In 1978 Kerwin and Lee (1978) proposed a vortex lattice lifting surface method to predict the steady and unsteady performance of marine propellers. Greeley and Kerwin (1982) further improved this method and described its application in design and analysis of propellers in steady flow. Following the works of Hess and Smith (1964), which described the steady potential flow around arbitrary three-dimensional bodies, a number of panel methods appeared. These method, called velocity methods, were based on solving the unknown source singularity strength by applying the boundary condition of zero normal velocity at the control points on each panel. A panel method for lifting bodies in potential flow was introduced by Morino and Kuo (1974) in which the unknown was the potential strength. This is called the potential method. The first panel method developed for the application of analysis of marine propellers was developed by Hess and Valarezo (1985). A panel method for ducted propellers was described by Kerwin et al. (1987) and another method for propellers in steady flow was presented in Hoshino (1989).

1.4 State of the art

The present state of propeller design methods is based upon the extensive knowledge and research on the subject. The knowledge it not only gathered from the field of marine hydrodynamics. A considerable amount can be traced back to research and experience in aerodynamics. A continuous effort is made to increase the accuracy and efficiency of existing methods as well as to implement new methods. The introduction of parallel computing and fast development of computational capacity have made CFD relevant for propeller design. The less complex calculation methods still find their application during earlier stages of the design process.

Several programs are developed for application in propeller design and analysis. OpenProp is a free open-source program that performs design and analysis of marine propellers based on lifting line theory. OpenProp is designed to be a fast parametric design tool for use by engineers with little training in propeller design (D'Epagnier et al., 2007). AKPD and AKPA are marine propeller design and analysis programs. AKPD features a propeller design method based on a combination of lifting line and lifting surface method. The propeller analysis software AKPA applies a boundary element method. Both are developed by MARINTEK in cooperation with State Marine Technical University of St. Petersburg, Russia.

Chapter 2

Propeller Calculation Methods

The present state propeller design process offers a variety of different approaches for simulation and analysis of propellers. Since the development of the field of aerofoil theory took place in the early 1900s, the methods have come a long way in describing the complex problem of flow around a marine propeller. While today's available computational power have made the most complex methods both time efficient and reliable, the more simple analysis methods still find their application in the early part of the propeller design procedure.

In this section the most common methods will be presented along with their role and applicability in the process of design and analysis of a marine propeller.

2.1 Empirical methods

2.1.1 Open water model test

In open water model tests a propeller model is tested for performance in a cavitation tank. The models are rigged without the ship hull and is rotating in a fixed position in an uniform inflow. Thrust and torque are then measured over a range of flow velocities and revolution speeds. If available, open water model test data may be used in the initial steps of propeller design to determine the main dimensions. Polynomial curve fits of the test data are simple and efficient tools to evaluate the performance of a propeller based on diameter, mean pitch and blade area. Open water model tests may be both time consuming and costly. They are therefore not often used directly in the process of designing a single propeller.

2.1.2 Propeller series

Propeller series are based on extensive open water model tests. By systematically testing propellers with a range of pitch values, blade areas and number of blades the performance data can curve-fitted and compared. The results forms the basis for propeller series diagrams such as the Wageningen B-Screw series from MARIN (Lammeren et al., 1969). These series were developed for to design propellers for specific ship types. The results from most of these series have been synthesized and incorporated into computer routines for use in propeller design and analysis (Kerwin and Hadler, 2010).

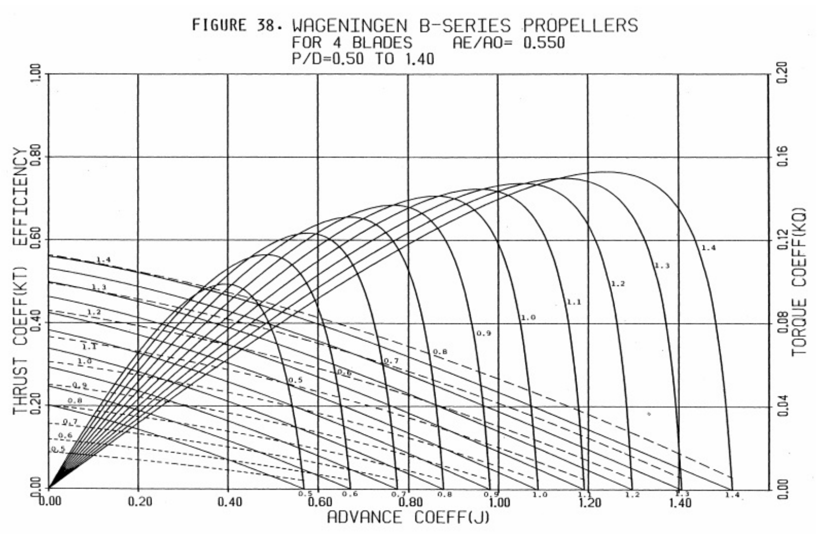

Figure 2.1: Example of Wageningen B propeller series (Bernitsas et al., 1981)

2.2 Numerical methods

2.2.1 Momentum theory

The simplest possible idealization of a propeller is the actuator disk. This idealization was introduced by Rankine (1865) and Froude (1889) and replaces the physical propeller with a permeable disk of vanishing thickness. The disk introduces a uniform jump in total pressure of the fluid passing through the disk, which tends to accelerate the fluid in the positive axial direction and thus results in a thrust force in the negative x-direction (Kerwin and Hadler, 2010). While these calculations are a relatively quick and easy way of estimating the total thrust, torque and delivered power of the propeller, a lot of the details around the physical attributes are lost. The information about the sectionwise distribution of thrust and torque on the propeller blades remains unknown. This is a pure 1-dimensional analysis and requires the assumption of homogeneous inflow with an ideal fluid.

2.2.2 Blade element theory

The forces and moments acting on the blade are derived from a number of independent slices represented as two-dimensional aerofoils at an angle of attack to the fluid flow (Amini, 2011). The thrust and torque of each two-dimensional section can be calculated if the lift and drag coefficients of the propeller blade are already known. Lift and drag can easily be calculated using linearized hydrofoil theory.

2.2.3 Blade element momentum theory

BEMT combines the two dimensional action of the blade from blade element theory with momentum changes in the fluid from momentum theory to determine the effective angle of attack of each section and hence thrust and torque components of each section (Phillips et al., 2009). The momentum change in the fluid from momentum theory determines the average induced velocities on the propeller. This is combined with a sectionwise blade element analysis to find the thrust and torque along with the optimal angle of attack for each section. The advantage of BEMT over more advanced methods such as panel methods and Vortex Lattice methods is that it allows for the lift and drag properties of the two-dimensional sections representing the blade to include viscous effects such as stall and the effect of laminar separation at low Reynolds number by using empirically based lift and drag curves for the blade sections (Amini, 2011). In addition, the simplicity of the BEMT-model makes the calculations require far less computational effort than the more advanced methods.

2.2.4 Lifting line method

The lifting line method represents the propeller blades as lifting lines with an arbitrary distribution of circulation in the spanwise direction. This carries the same advantage as the BEMT method as it simplifies the three dimensional propeller blade into a set of two dimensional foil sections. While the BEMT method only solves the problem for each section independently, the lifting line method accounts for the influence from all sections simultaneously.

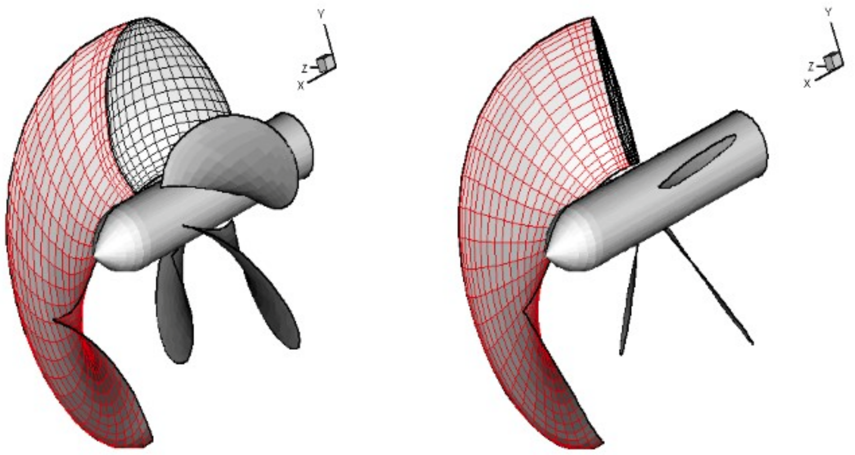

Figure 2.2: Illustration of the lifting line representation of the propeller blades as a limit of vanishing chord length. The free vortex lines convects downstream with constant radius and constant pitch. (Kerwin and Hadler, 2010)

The lifting line method is a simplification of the lifting surface problem. The propeller blades are represented as lifting lines and can be considered as the case of blades with vanishing chord lengths as seen in 2.2. The accuracy is considered within acceptable limits and the low computation time makes it especially applicable in the early stages of propeller design. As proposed by Praefke (2011), the lifting line method may be applied to determine the radial distributions of pitch and blade section cambers and to update the predicted efficiency. The lifting line method does not account for viscous inflow and may only be used in the case of light or moderately loaded propellers. However, lifting line theory offers several advantages over turbine blade element momentum theory (BEM), such as a more accurate relationship between the induction velocities and the radial circulation distribution (Epps and Kimball, 2013).

The Lerbs method (Lerbs, 1952)is still the universally accepted procedure for establishing at the early design stage the radial distribution of circulation and the resulting thrust, power and efficiency of a propeller. Kerwin and Hadler (2010). However, one drawback is that Lerbs based his calculations on the same assumption as Prantl and Tietjens (1934), namely high aspect ratio blades. Marine propellers often have low aspect ratio blades for reasons such as cavitation. In more recent studies by Epps and Kimball (2013), significant improvements have been introduced to the lifting line method for propeller design. Instead on interpolating the wake pitch from the pitch computed at the control points, the new wake model assumes that each vortex panel has a constant pitch and that the pitch angle of the trailing vortices are analytically related to the pitch of the control points. This assumption makes the influence functions analytically consistent with the pitch at the control points, and thus, it greatly improves the numerical stability and robustness of the rotor lifting line model (Epps and Kimball, 2013).

Lifting line theory assumes inviscid and incompressible flow in a circumferentially averaged flow field. The trailing helix of free vortex lines has a constant pitch.

2.2.5 Lifting surface method

The lifting surface method approaches the three dimensional propeller problem directly, but simplifies it by representing the blades as infinitely thin surfaces placed on the mean camber line.

In the lifting surface method presented by Kerwin and Lee (1978), the propeller blades are considered to be a set of symmetrically arranged thin blades of arbitrary form. The assumption of thin blades lets them be represented in the fluid flow by a distribution of sources,sinks and vortices placed on the mean camber surface of each blade and a distribution of shed vortices in the wake. Another assumption is that the spatial distribution of sources can be determined by sectional application of thin wing theory. As a result, the source distribution is known and remaining unknowns are the vortex distributions which will be resolved into spanwise and chordwise components. The corresponding components in the representation of the wake are termed "shed" vorticity and "trailing" vorticity. The unknown distributions are determined from the boundary conditions of the problem.

In the design case where the geometry of the blade is the unknown, an estimation of the radial loading is calculated by lifting line theory. As such, the same assumptions as in lifting line method are made.

2.2.6 Boundary element method

Boundary element methods, or panel methods, are similar to the lifting surface method in that they represent the propeller in three dimensions. The propeller is discretized into panels and, depending on the method, a distribution of a combination of vortices, sources, or dipoles is placed on each panel. Again depending on the method, the propeller forces are found by solving for the velocity or the velocity potential. Due to the inclusion of thickness and hub, the panel methods are generally more accurate and closer to the physical model. The cost is of course longer computation times. The use of panel method for a single propeller in open water conditions is known to predict the propeller torque and thrust with good accuracy, at least close to the design condition (Amini, 2011). The BEM approach is seldom used in the blade design calculations which are mostly based on the lifting surface algorithms. The exception is given by the algorithms that attempt to design a blade with prescribed pressure distribution (Krasilnikov, 2015).

The propeller blades are often discretized into panels with nodes distributed with cosine spacing. Control points are placed at the midpoint of each panel. This assures a denser distribution of control points in the proximity of the tip, hub, leading and trailing edge. The wake is discretized with the same cosine spacing in the radial direction as the body and uniform spacing in the axial direction. The wake geometry model may be described in advance as a helical surface with fixed pitch and radius, or as a deformed wake model based on empirical knowledge and physical reasoning. It can also be modeled using a wake alignment model, which allows direct satisfaction of the no-force condition on the free vortex lines in the wake. Politis (2004) describes the Wake Relaxation Method where the wake is prescribed with an initial geometry, and is then deformed through an iterative scheme.

The boundary element methods are limited to the case of potential flow and viscous effects can only be accounted for by artificial corrections.

2.2.7 Reynold Averaged Navier-Stokes

The last decades of exponential development of computational power have made CFD methods increasingly relevant in propeller analysis. Reynold Averaged Navier-Stokes (RANS) methods have matured to the point where they may be a steady part of the propeller design process. The RANS method offers several advantages over the potential-flow methods. Without the assumptions of inviscid and irrotational flow, RANS may allow for the modeling of tip-vortex roll up and flow separation. Based on either final element or final volume approach, the RANS solver calculates the average flow field by modeling the fully viscous flow field. Steady RANS methods can predict the propeller's powering performance, possibly including propeller-hull interaction effects, as well as the scale effects to be expected in model tests (Praefke, 2011). The obvious downside is the long computation times and the required effort to model and prepare the problem for calculation.

Chapter 3

Software and Set-up

This chapter describes the different programs used in the design and analysis process as well as how the analyses of the design programs are set up.

3.1 Software

3.1.1 MATLAB-script

A part of the thesis was to develop a propeller design program in MATLAB. The program was based on the lifting line method and applies Goldstein factors (Goldstein, 1929) to correct for finite number of blades. The program structure is based on the method described by Steen (2014). All program code is included in appendix 1. The program set up to find the optimum propeller geometry based on a given propeller diameter, hub diameter, RPM, number of blades, shaft immersion, ship speed, and circumferentially averaged radial wake. Two form function with two parameters described the chord and circulation distributions. By calculating propeller efficiency η_0 for a range of chord and circulation distributions, the program finds the optimum non-cavitating propeller geometry. Hub effects are not directly calculated, but the circulation distribution is set to zero at the hub center to allow for non-zero ciculation at the blade root. The NACA a = 0.8 mean line was used to calculate the camber distributions on the blade, and NACA 66 for thickness calculations. All data was gathered from the Theory of Wing Sections (Abbott and Doenhoff, 1959).

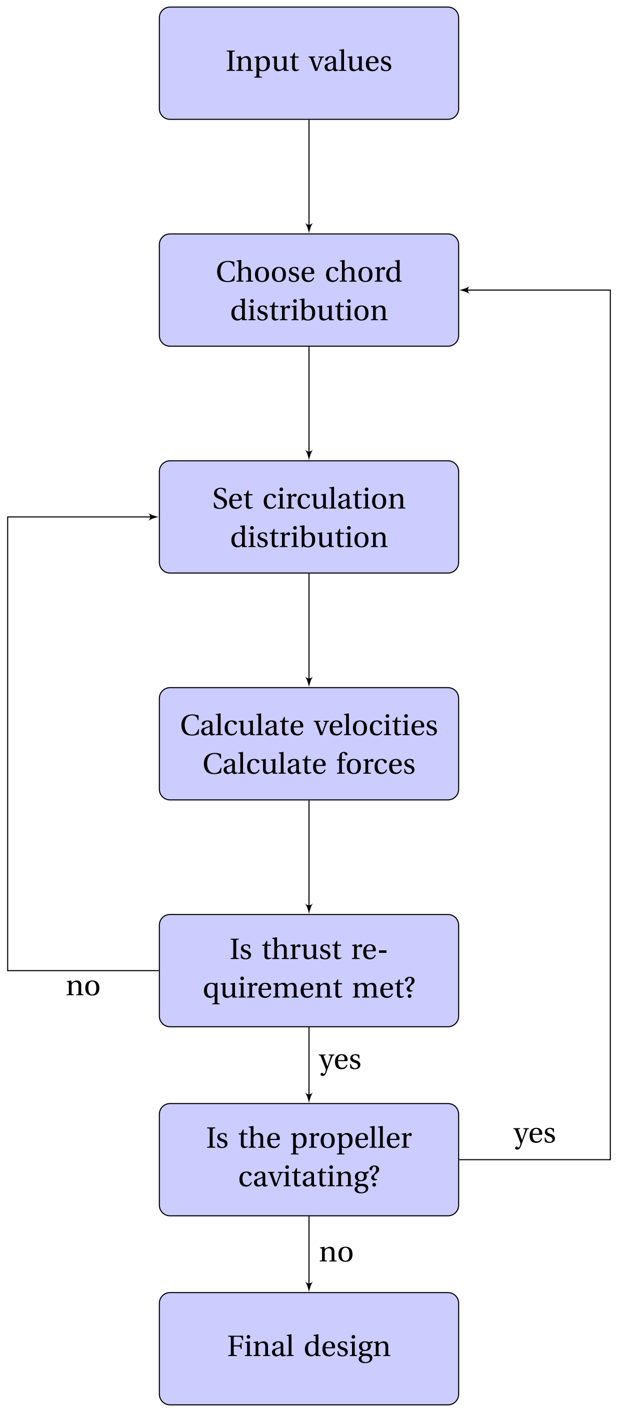
Input values
Choose chord distribution
Set circulation distribution
Calculate velocities
Calculate forces
Is thrust requirement met?
no
yes
Is the propeller cavitating?
yes
no
Final design

Figure 3.1: Flowchart of MATLAB program design process

3.1.2 AKPD

AKPD is a propeller design program, and part of a propeller design/analysis system developed by MARINTEK as part of a long-time cooperation with State Marine Technical University of St Petersburg, Russia. It allows numerical simulation of conventional open propellers (fixed and controllable pitch), propeller/rudder systems, shaft and pod two-staged arrangements. Achkinadze et al. (2003). The propeller design algorithm is based on a lifting line method (GLM) REF and non-linear lifting surface theory described in REF. The lifting line is used to calculate the optimum circulation distribution along the propeller radius. Pitch and camber distributions are determined through lifting surface calculations based on camber, thickness and circulation distribution.

3.1.3 AKPA

The AKPA (AK - trade mark, Propulsor Analysis) program is intended for the hydrodynamic analysis of complex marine propulsion systems, such as open and ducted, fixed pitch (FPP) and controllable pitch (CPP) propellers, podded propellers, propeller/rudder systems and tunnel propellers, using a velocity based boundary element method (BEM), or panel method. (Achkinadze and Krasilnikov, 2012). The BEM is described in detail in citepVBBEM. One of the reasons AKPA applies a velocity based BEM is to avoid the necessity of iterations to satisfy the non-linear Kutta-Jowkovski condition on the trailing edge. Velocity based methods define velocities directly while potential based methods require numerical differentiation. This reduces the computational effort needed in the calculations. AKPA offers both steady and quasi-steady analysis of the propeller. In the course of the years, the propulsor analysis algorithm has been extended through semi-empirical models to account for the effects of viscosity, which were derived on the basis of correlations with systematic viscous flow computations and measurements. More recently, the program has incorporated the two Euler equation solvers - axisymmetric and fully 3D - that allow for the estimation of effective inflow on propeller at specified design point. (Krasilnikov and Sileo, 2012)

3.1.4 OpenProp

OpenProp is an open-source code suite that can be used for the design, analysis, and fabrication of optimized propellers and horizontal-axis turbines (Epps, 2010). OpenProp runs in MATLAB and uses a lifting line method in both analysis and design of propeller blades. In OpenProp, the induced velocities are calculated from the method presented by D'Epagnier et al. (2007) based on the formulas developed by Lerbs (1952) and Wrench (1957). These formulas are high accuracy approximations to Lerbs (1952) lifting line solution using asymptotic formulas for the modified Bessel function. OpenProp is based on moderately-loaded lifting line theory, in which a propeller blade is represented by a lifting line, with trailing vorticity aligned to the local flow velocity. The induced velocities are computed using a vortex lattice, with helical trailing vortex filaments shed at discrete stations along the blade. The blade itself is modeled as discrete sections, having 2D section properties at each radius. Loads are computed by integrating the 2D section loads over the span of the blade. Currently the analysis is limited to rotors without skew and rake.

3.1.5 STAR-CCM+

STAR-CCM+ is a CFD solver developed by CD-Adapco. The software is used as an engineering tool to solve problems involving fluid flows, heat transfer or stress. STAR-CCM+ is well suited for flow simulations of propeller designs and may be used to highlight viscous effects, realistic full scale performance and cavitation problems. The RANS solver is able to mesh and simulate a propeller geometry within a reasonable time limit even on an average laptop computer. The author was provided with templates for geometry preparation and suitable meshing settings for propeller simulations. However, the template was limited for open water analysis and could not, to the authors knowledge, be easily adjusted to simulate a wake field.

3.2 Set-up for Comparison

The comparison of the propeller design programs was set up as follows:

1. Two propellers were designed by the MATLAB program based on different values for diameter, hub size, required thrust, ship speed, RPM, number of blades and shaft immersion. Two different radial wake fields were also prescribed. The resulting chord and thickness distributions provided the basis for designs in OpenProp and AKPD.
2. The two initial propeller designs were imported in AKPD and OpenProp wich calculated optimal pitch and camber values based on the input geometry and inflow. The hub were in both cases modeled as a cylinder. The calculations resulted in four different propeller designs.
3. All four propeller designs were analyzed in AKPA to compare the predicted performance. A cavitation analysis was also conducted as the amount of cavitation is an important feature in a propeller design.

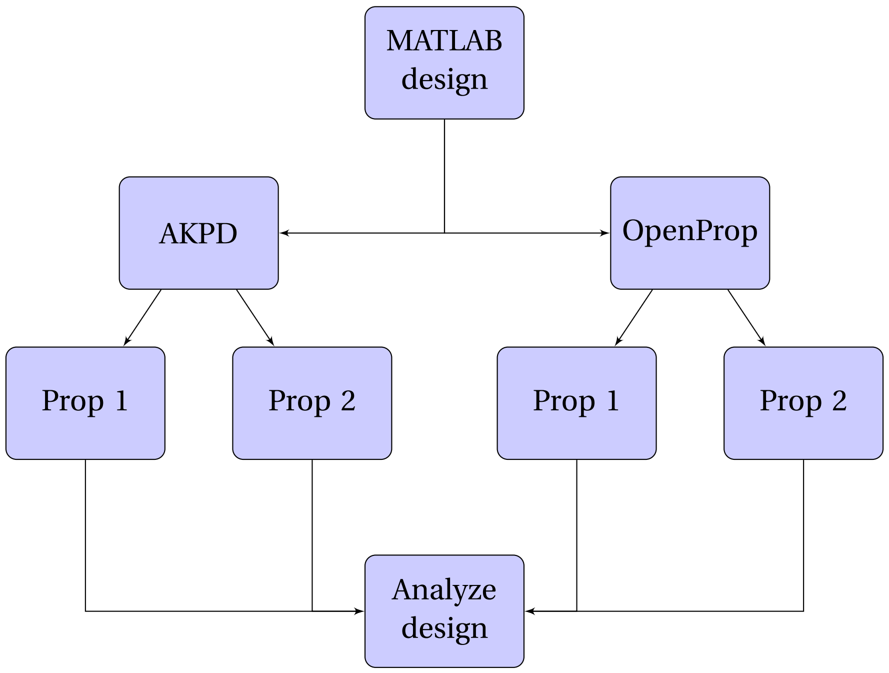

Figure 3.2: Flowchart of comparison process

3.2.1 Design inputs

The values presented in table 3.1 are the input values used to produce the two propeller designs from the MATLAB program.

	Propeller 1		Propeller 2	
Propeller diameter	3.5	m	3.7	m
Number of blades	4		4	
Hub diameter	1.020	m	0.9953	m
RPM	150		151.35	
Ship speed	15.5	$knots$	15.9	$knots$
Shaft immersion	4.5	m	4.7	m
Required thrust	400900	N	454866	N

Table 3.1: Input values for the MATLAB design program for propeller 1 and 2

3.2.2 Wake fields

The wake fields prescribed to propeller 1 and propeller 2 are presented in figure 3.3 and 3.4. Only the axial component of the wake was included.

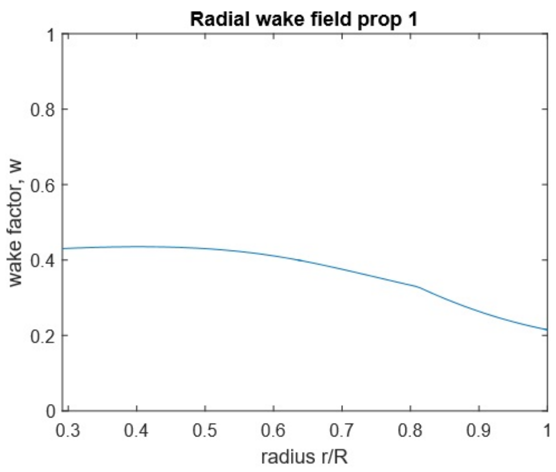

Figure 3.3: Radial wake field in the axial direction for propeller 1

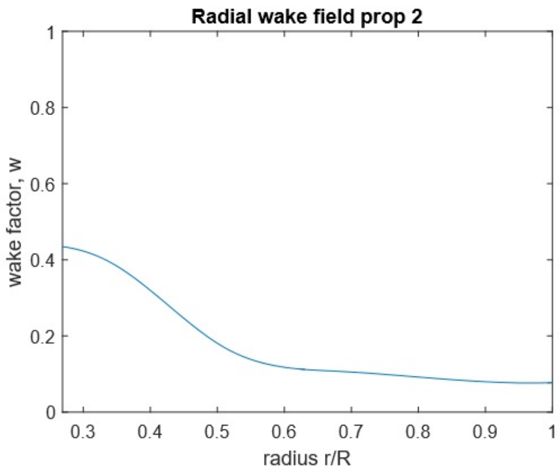

Figure 3.4: Radial wake field in the axial direction for propeller 2

Chapter 4

Results

4.1 Validation

The KVLCC2 propeller was used to validate analysis results. A visualisation of the propeller from both OpenProp and AKPA can be seen in figure 4.1 and 4.2. OpenProp and AKPD were tested in how close their design would be to the real propeller given the chord, thickness and wake distributions. The result resulting pitch and camber distributions can be seen in table 4.1.

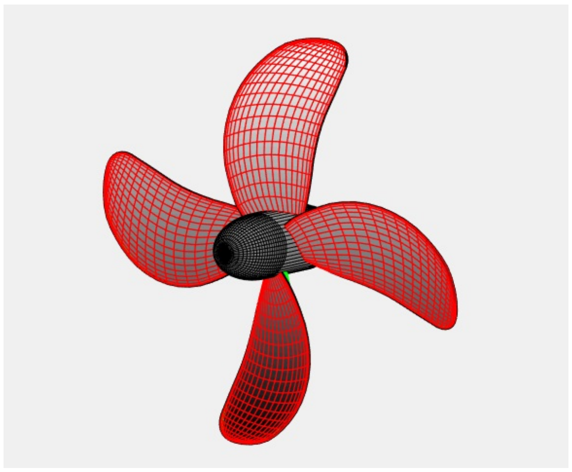

Figure 4.1: KVLCC2 propeller visualized in OpenProp

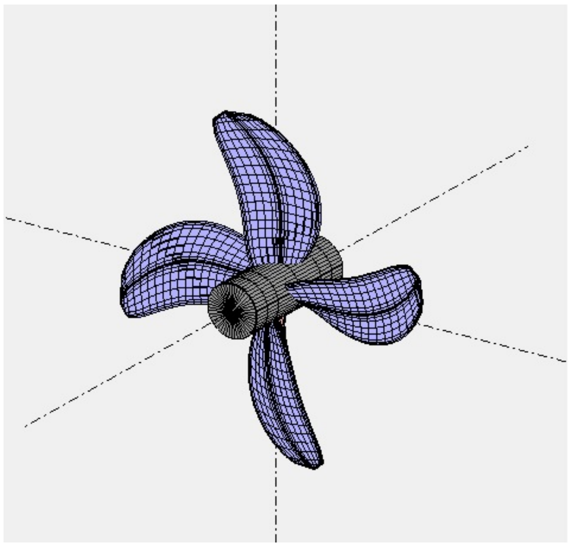

Figure 4.2: KVLCC2 propeller visualized in AKPA

	KVLCC2	Geometry			OpenProp		AKPD	
r/R	b/R	e/R	P/D	f0/R	P/D	f0/R	P/D	f0/R
0.155	0.2998	0.09399	0.5743	0.0093	0.6464	0.00227	0.691	0.00433
0.16	0.303	0.09362	0.5765	0.00948	0.6482	0.00181	0.689	0.00432
0.25	0.3554	0.08434	0.6130	0.01237	0.6481	0.00159	0.652	0.00416
0.3	0.3784	0.07704	0.631	0.01347	0.6481	0.00122	0.644	0.00407
0.4	0.4186	0.06404	0.663	0.01415	0.6481	0.00072	0.645	0.00388
0.5	0.4494	0.05204	0.691	0.01317	0.6481	0.00049	0.644	0.00368
0.6	0.467	0.0411	0.712	0.01168	0.6481	0.00033	0.641	0.00352
0.7	0.4676	0.0312	0.721	0.01024	0.6481	0.00002	0.634	0.00342
0.8	0.4384	0.0221	0.716	0.00868	0.6481	0.00002	0.624	0.00334
0.9	0.3616	0.014	0.693	0.00582	0.6481	0.00002	0.605	0.00306
0.95	0.2884	0.00944	0.675	000364	0.6482	0.00002	0.590	0.00284
1.0	0.02	0.008	0.651	0.00006	0.6483	0.00001	0.571	0.00258

Table 4.1: KVLCC2 Propeller geometries. The original to the left, and OpenProp and AKPD designs to thr right.

4.1.1 Performance analysis

Performance curves from AKPA, OpenProp and experimental measurements are shown in figure 4.3, 4.4 and 4.5.

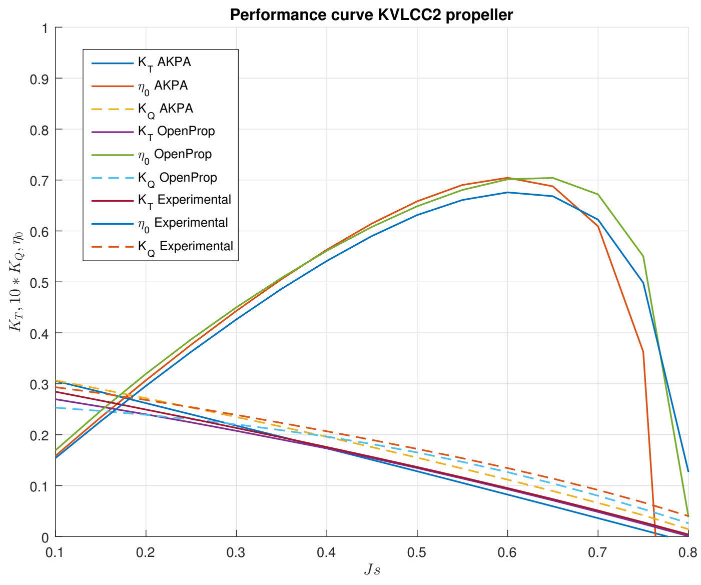

Figure 4.3: Performance curves for KVLCC2 propeller

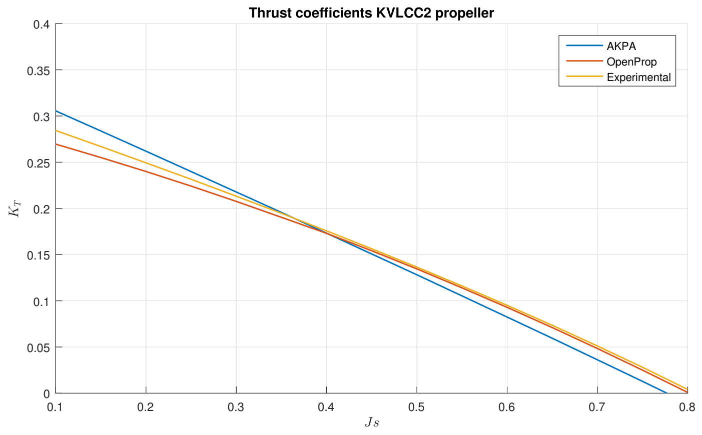

Figure 4.4: Thrust coefficient analysis results for KVLCC2 propeller

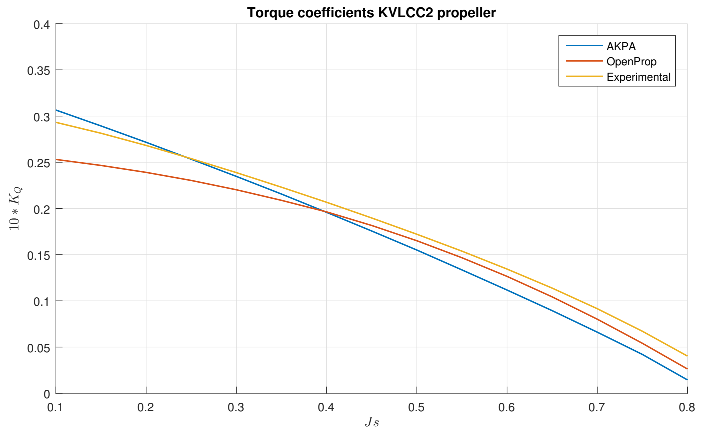

Figure 4.5: Torque coefficient analysis results for KVLCC2 propeller

4.2 Results

4.2.1 MATLAB-program design

The MATLAB program designed two propellers which were intended as basis geometries for the designs made in AKPD and OpenProp. The MATLAB geometry was still subjected to analysis.

Blade geometry

A visualisation plot of propeller 1 can be seen in figure 4.6 and 4.7, and of propeller 2 in 4.8 and 4.9. The values of the compete geometry of both propeller are presented in table 4.2

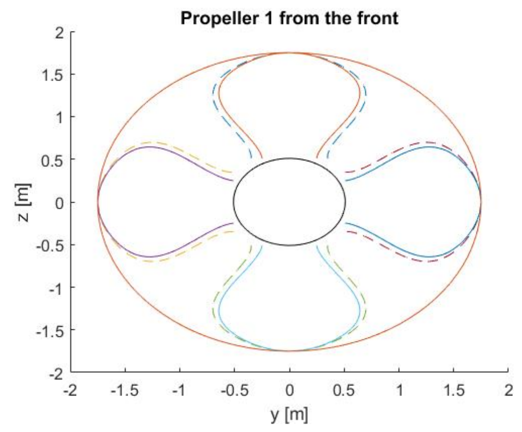

Figure 4.6: Visualization of MATLAB design of propeller 1 from the front

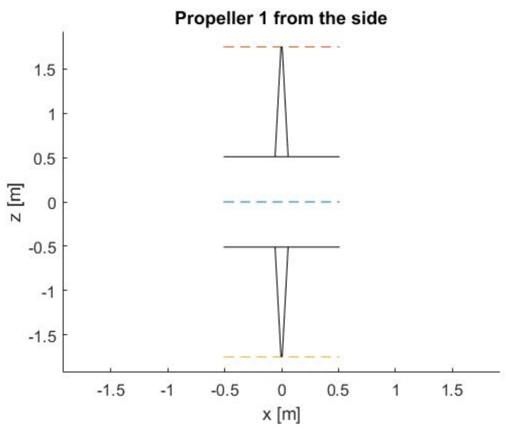

Figure 4.7: Visualization of MATLAB design of propeller 1 from the side

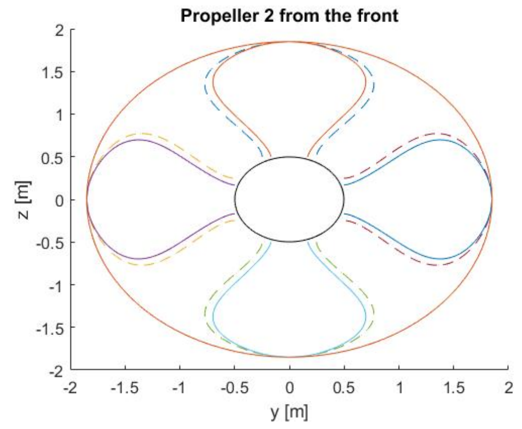

Figure 4.8: Visualization of MATLAB design of propeller 2 from the front

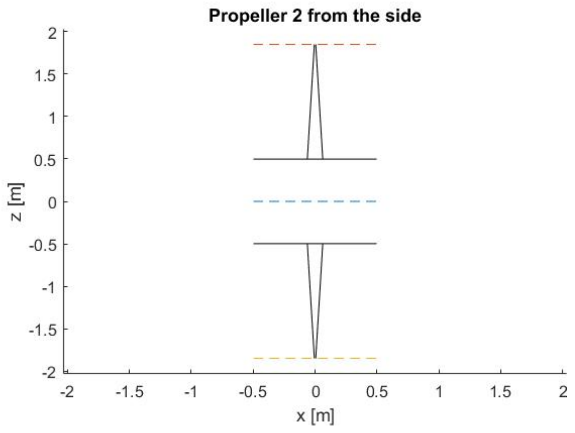

Figure 4.9: Visualization of MATLAB design of propeller 2 from the side

Propeller 1					Propeller 2				
r/R	b/R	e/R	P/D	f0/R	r/R	b/R	e/0	P/D	f0/R
0.2915	0.3999	0.06551	0.9023	0.0631	0.269	0.269	0.0674	0.9263	0.09429
0.3	0.3983	0.0648	0.9048	0.0642	0.3	0.2799	0.0648	0.9263	0.09026
0.4	0.4698	0.0564	0.9094	0.0617	0.4	0.4070	0.0564	0.9687	0.05809
0.5	0.6048	0.0480	0.9161	0.0485	0.5	0.5838	0.0480	1.0645	0.03658
0.6	0.7312	0.0396	0.9280	0.0380	0.6	0.7427	0.0396	1.0978	0.02630
0.7	0.7960	0.0312	0.9469	0.0318	0.7	0.8290	0.0312	1.0917	0.02169
0.8	0.7555	0.0228	0.9639	0.0288	0.8	0.7994	0.0228	1.0931	0.02041
0.9	0.5728	0.0144	0.9980	0.0278	0.9	0.6162	0.0144	1.1058	0.02211
0.95	0.4048	0.0102	1.0010	0.0274	0.95	0.4393	0.0102	1.1231	0.02481
1.0	0.004	0.006	0.7683	0.0229	1.0	0.004	0.006	0.8933	0.03449

Table 4.2: MATLAB design for propeller 1 and 2

Performance curves

The performance analysis of the MATLAB designs can be seen in figure 4.10 and 4.11

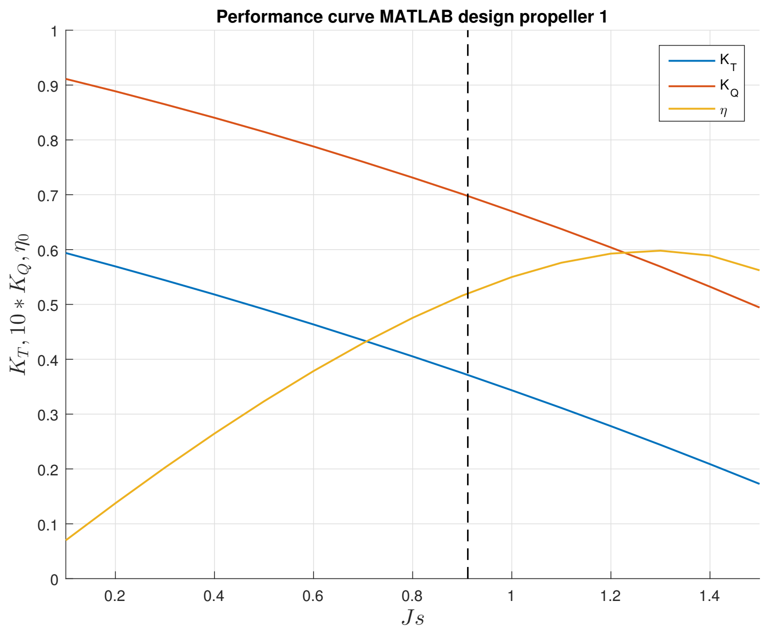

Figure 4.10: Performance curves for MATLAB design of propeller 1

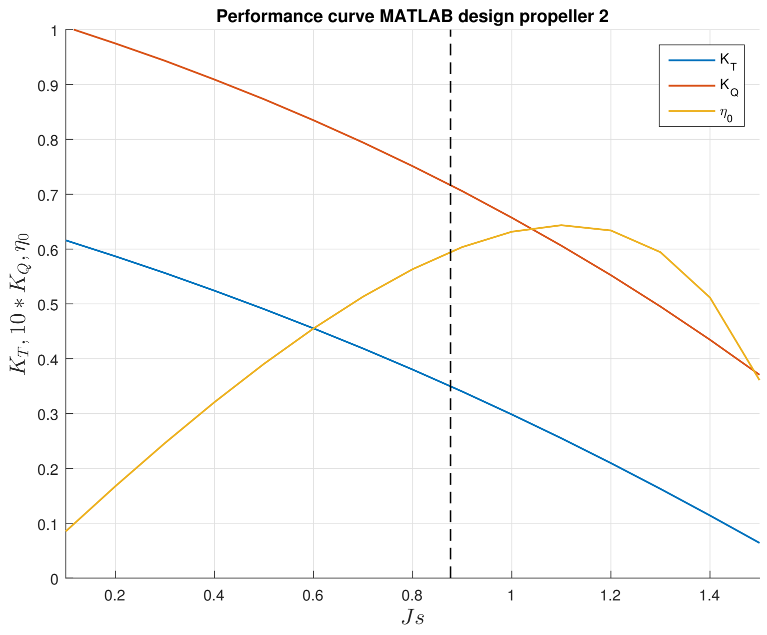

Figure 4.11: Performance curves for MATLAB design of propeller 2

Cavitation Analyses

The cavitation analysis is displayed as $-C_P/\sigma$ contour plots in 4.12 and 4.13.

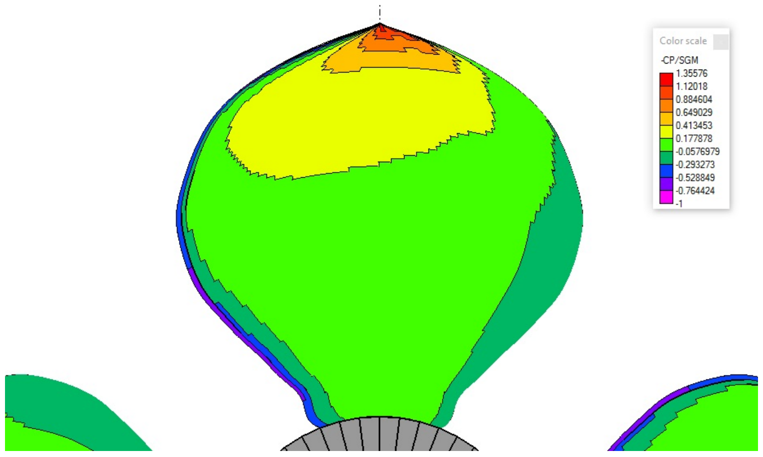

Figure 4.12: Cavitation analysis on MATLAB design for propeller 1

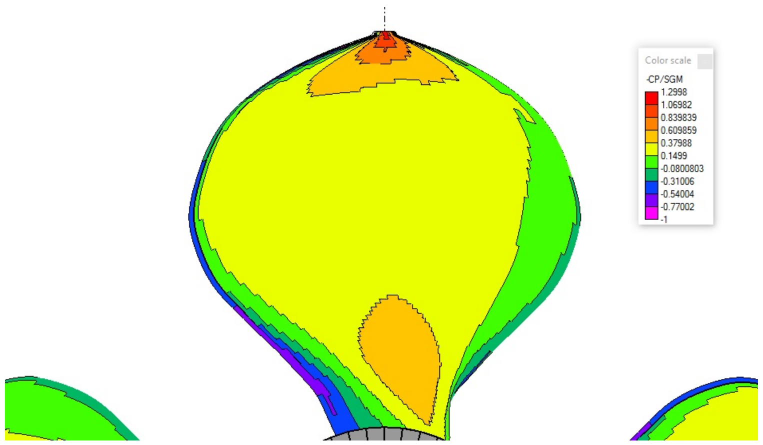

Figure 4.13: Cavitation analysis on MATLAB design for propeller 2

4.2.2 OpenProp design

OpenProp is able to visualize 3D geometries of the propeller including an arbitrary hub form. The OpenProp designs for propeller 1 and 2 are shown in figure 4.14 and 4.15. The pitch and camber values produced are presented in table 4.3.

Blade geometry

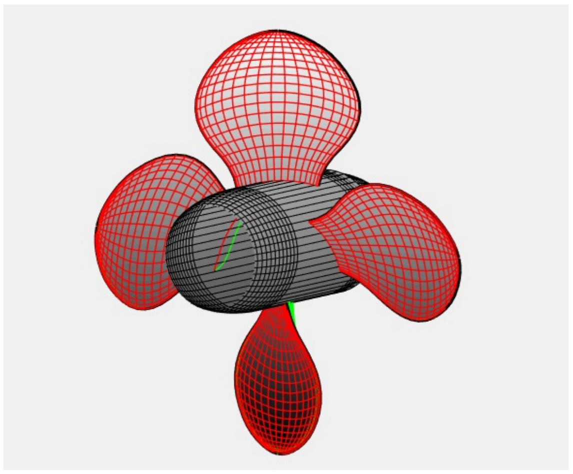

Figure 4.14: Visualization of OpenProp design for propeller 1

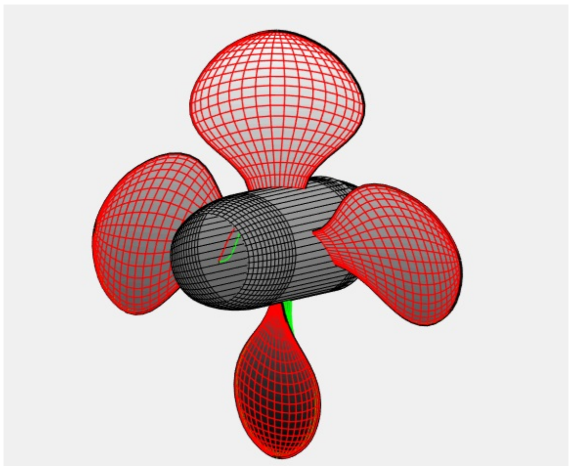

Figure 4.15: Visualization of OpenProp design for propeller 2

Propeller 1					Propeller 2				
r/R	b/R	e/R	P/D	f0/R	r/R	b/R	e/0	P/D	f0/R
0.2914	0.4002	0.0660	0.9338	0.0335	0.2690	0.2690	0.0674	0.9713	0.0318
0.3470	0.4248	0.0604	0.9913	0.0322	0.3264	0.3064	0.0626	1.0433	0.0298
0.4023	0.4722	0.0562	1.0322	0.0314	0.3834	0.3816	0.0578	1.0940	0.0287
0.5104	0.6196	0.0472	1.0728	0.0293	0.4949	0.5746	0.0484	1.1607	0.0263
0.6131	0.7440	0.0385	1.1023	0.0266	0.6009	0.7438	0.0395	1.2121	0.0234
0.7079	0.7972	0.0305	1.1304	0.0234	0.6987	0.8284	0.0313	1.2493	0.0203
0.7925	0.7634	0.0234	1.1580	0.0198	0.7859	0.8124	0.0240	1.2719	0.0171
0.8956	0.5838	0.0148	1.1961	0.0140	0.8923	0.6346	0.0150	1.2757	0.0124
0.9653	0.3294	0.0079	1.2257	0.0081	0.9642	0.3644	0.0081	1.2757	0.0074
1.0000	0.0040	0.0006	1.2431	0.0001	1.0000	0.0020	0.0006	1.2698	0.0001

Table 4.3: OpenProp design geometries for propeller 1 and 2

Performance curves

The performance curves for each of the OpenProp designs can be seen in figure 4.16 and 4.17.

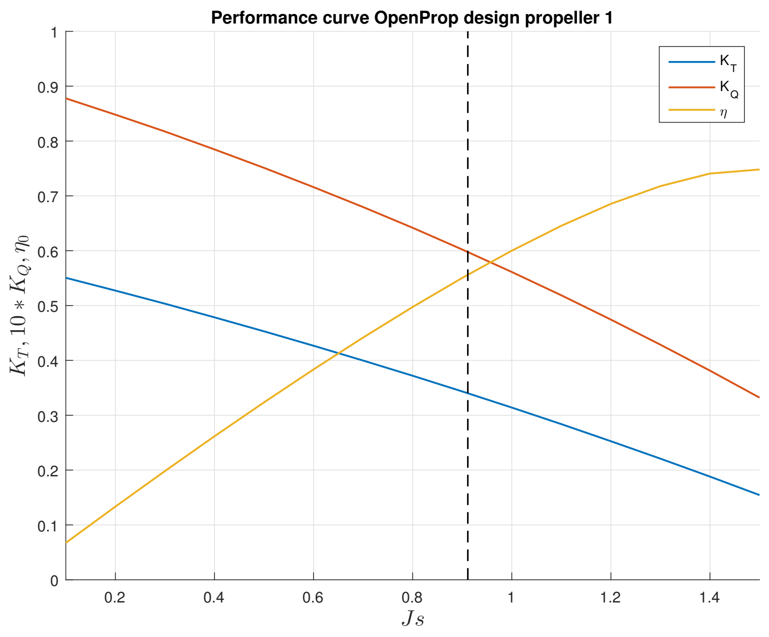

Figure 4.16: Performance curves for OpenProp design for propeller 1

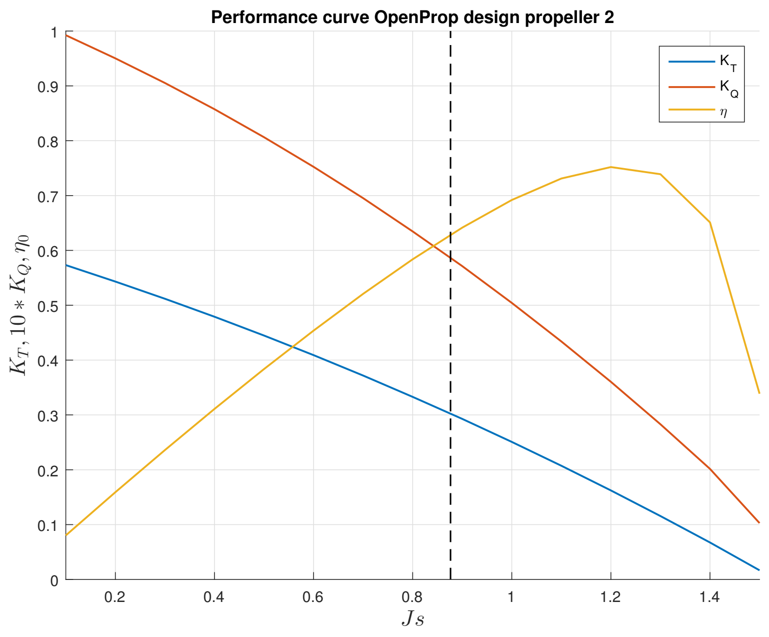

Figure 4.17: Performance curves for OpenProp design for propeller 2

Cavitation Analyses

The cavitation contour plots with $-Cp/\sigma$ ratios are included for both OpenProp designs 4.18, 4.19.

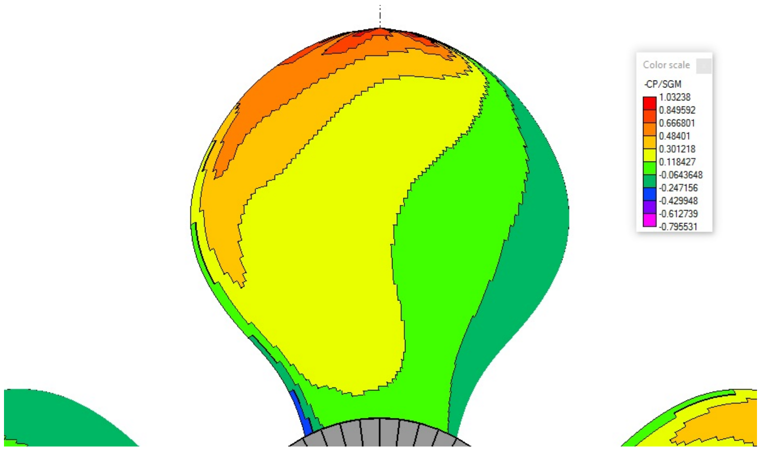

Figure 4.18: Cavitation analysis on OpenProp design for propeller 1

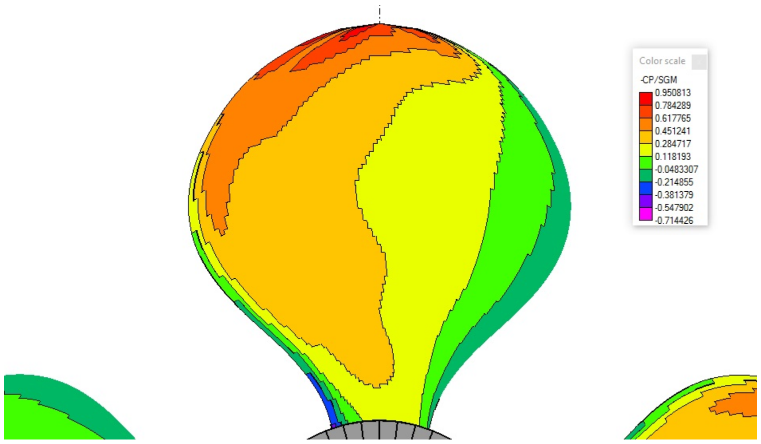

Figure 4.19: Cavitation analysis on OpenProp design for propeller 2

4.2.3 AKPD design

The visualisations of the AKPD designs are presented in figure 4.20 and 4.21, and the geometry values are shown in table 4.4.

Blade geometry

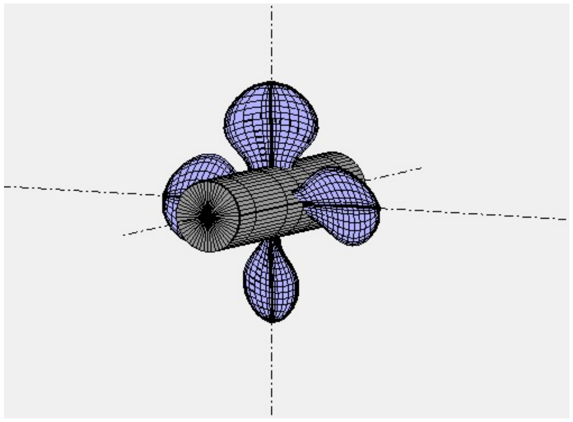
Figure 4.20: Visualization of AKPD design for propeller 1

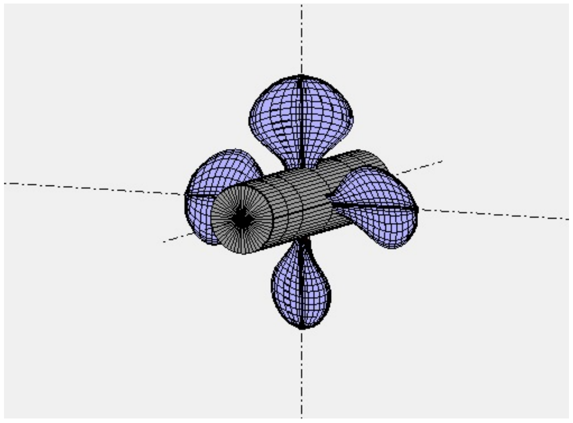
Figure 4.21: Visualization of AKPD design for propeller 2

Propeller 1					Propeller 2				
r/R	b/R	e/R	P/D	f0/R	r/R	b/R	e/0	P/D	f0/R
0.2915	0.4000	0.0655	1.1832	0.0321	0.269	0.269	0.0674	1.1123	0.0306
0.3	0.3983	0.0648	1.1761	0.0320	0.3	0.2799	0.0648	1.1132	0.0306
0.4	0.4698	0.0564	1.1172	0.0311	0.4	0.4070	0.0564	1.1682	0.0305
0.5	0.6048	0.0480	1.1003	0.0301	0.5	0.5838	0.0480	1.2739	0.0300
0.6	0.7312	0.0396	1.1092	0.0294	0.6	0.7427	0.0396	1.3234	0.0290
0.7	0.7960	0.0312	1.1376	0.0293	0.7	0.8290	0.0312	1.3296	0.0282
0.8	0.7555	0.0228	1.1766	0.0294	0.8	0.7994	0.0228	1.3359	0.0283
0.9	0.5728	0.0144	1.2379	0.0269	0.9	0.6162	0.0144	1.3384	0.0262
0.95	0.4048	0.0102	1.2527	0.0242	0.95	0.4393	0.0102	1.3251	0.0239
1.0	0.0200	0.0001	1.2502	0.0205	1.0	0.0200	0.0001	1.2989	0.0206

Table 4.4: AKPD design geometries for propeller 1 and 2

Performance curves

The performance curves for AKPD design are shown in figure 4.22 and 4.23.

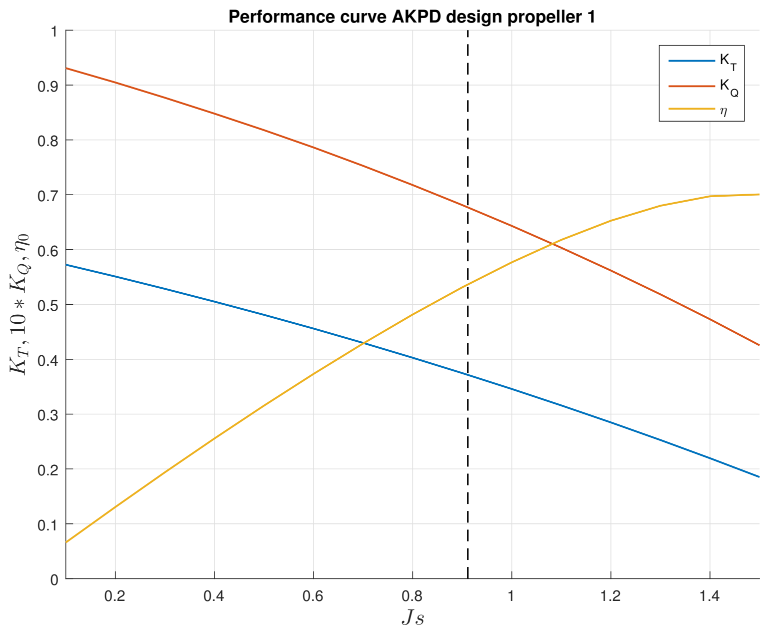

Figure 4.22: Performance curves for AKPD design for propeller 1

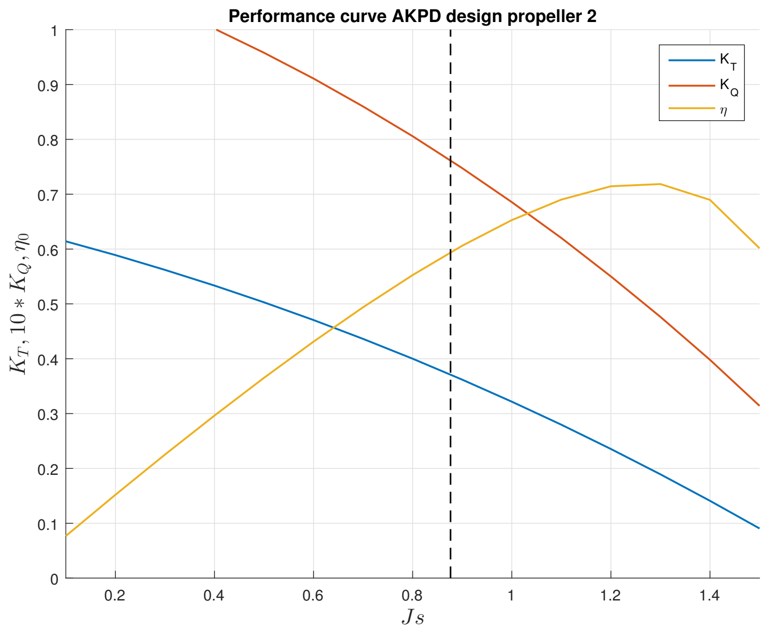

Figure 4.23: Performance curves for AKPD design for propeller 2

Cavitation Analyses

The cavitation results from AKPA analysis are presented in figure 4.24 and 4.25. The contour plots visualize how $-Cp/\sigma$ ratios are distributed over the blade.

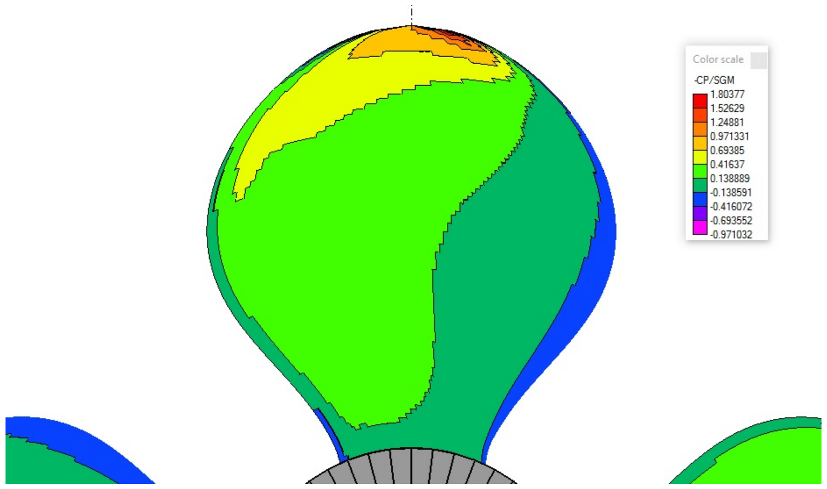

Figure 4.24: Cavitation analysis for AKPD design for propeller 1

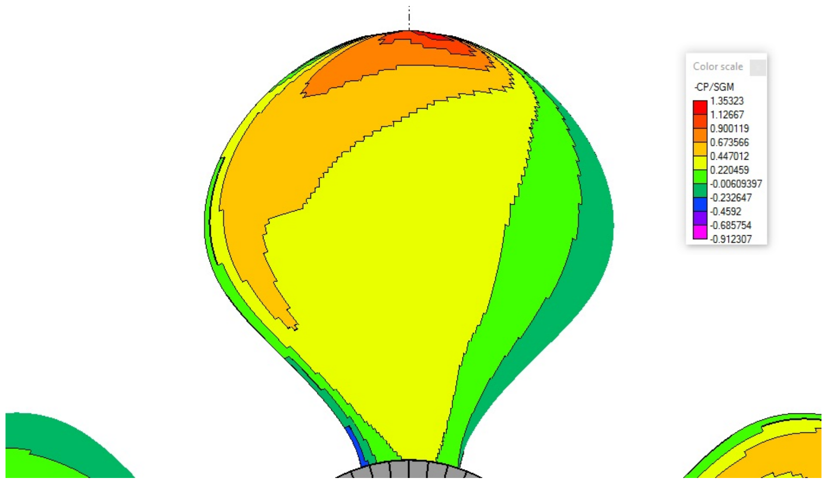

Figure 4.25: Cavitation analysis for AKPD design for propeller 2

4.2.4 Geometry Comparison

Comparison plots were made to make the differences in calculated geometry more clear. The P/D plots are seen in 4.26 and 4.27. Camber values are presented in figure 4.28, 4.29.

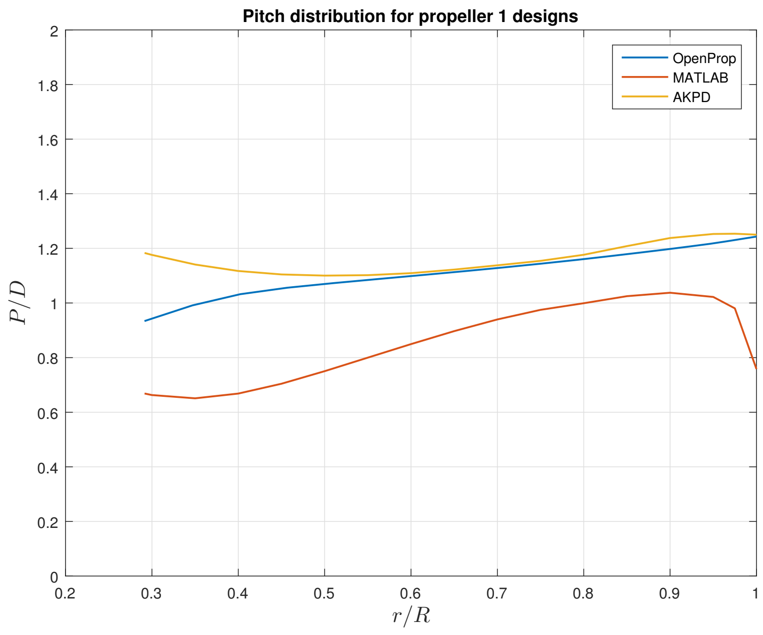

Figure 4.26: Comparison of pitch distributions for propeller 1

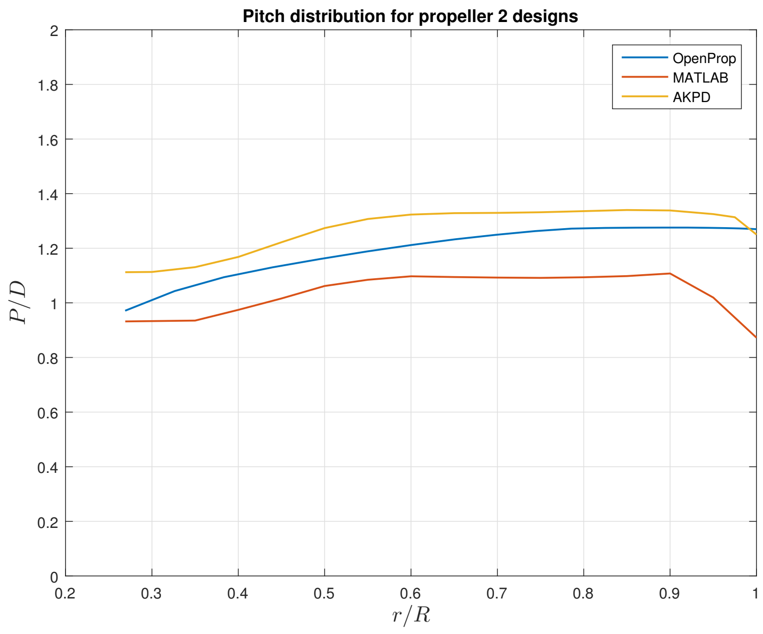

Figure 4.27: Comparison of pitch distributions for propeller 2

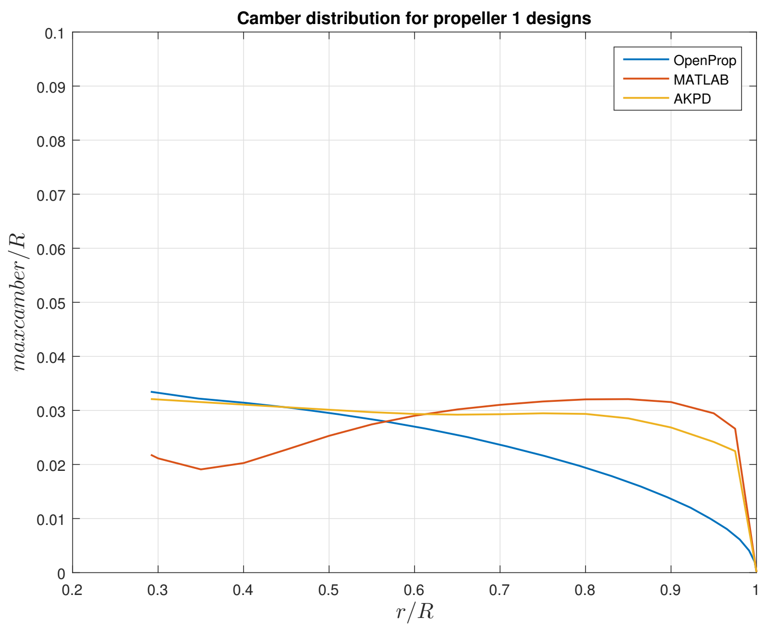

Figure 4.28: Comparison of camber distributions for propeller 1

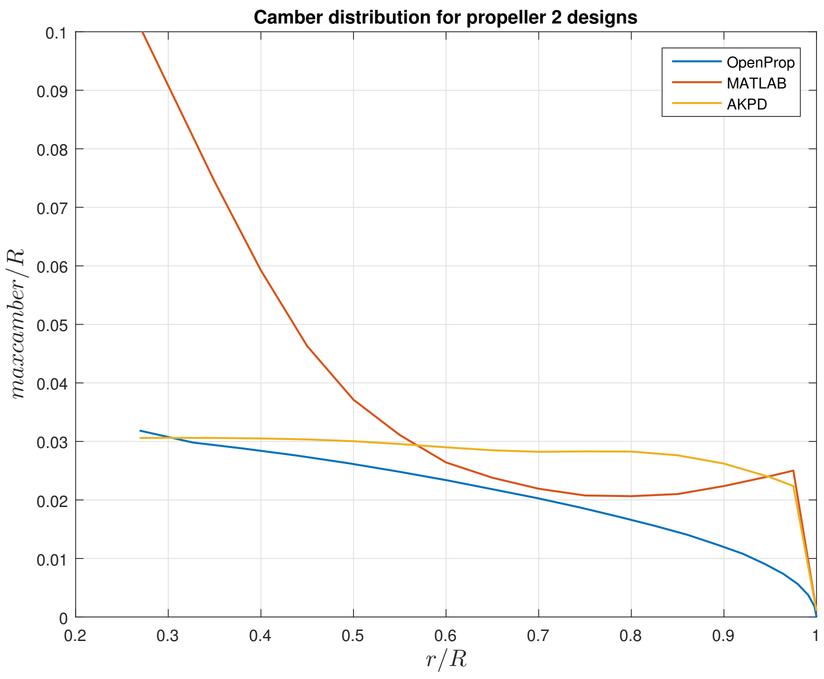

Figure 4.29: Comparison of camber distributions for propeller 2

4.2.5 Performance comparison

The six remaining plots show thrust, torque and efficiency plottet for all three design of propeller 1 if figures 4.30, 4.33 and 4.34. For propeller 2 the thrust torque and efficiency coefficients are shown in figures 4.31, 4.33 and 4.35.

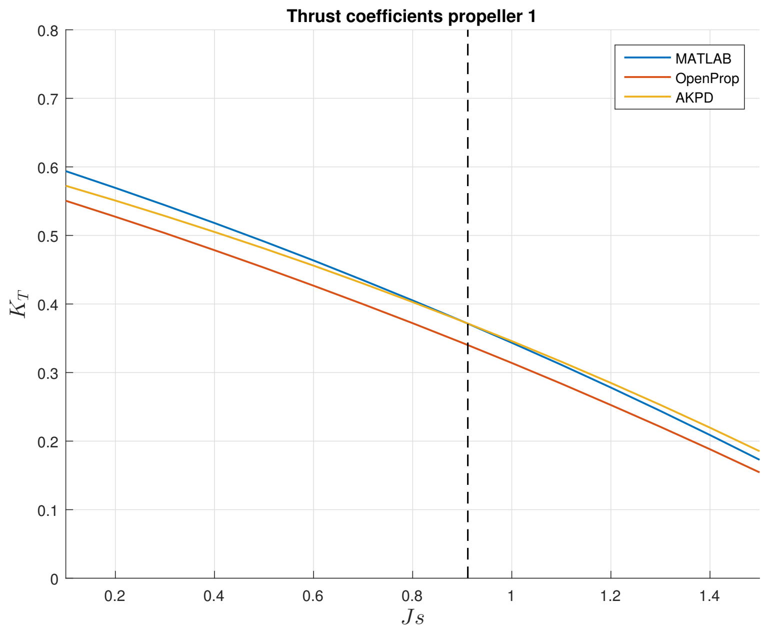

Figure 4.30: Thrust coefficients for propeller 1

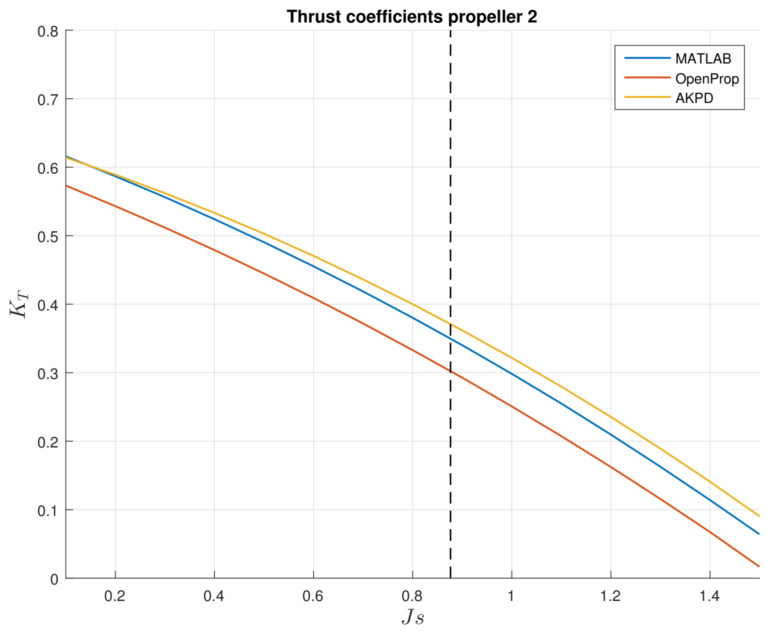

Figure 4.31: Thrust coefficients for propeller 2

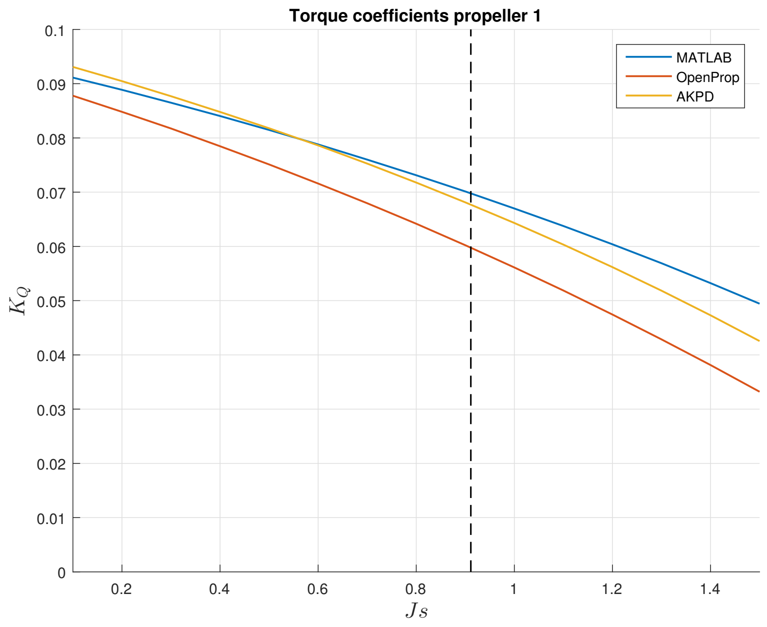

Figure 4.32: Torque coefficients for propeller 1

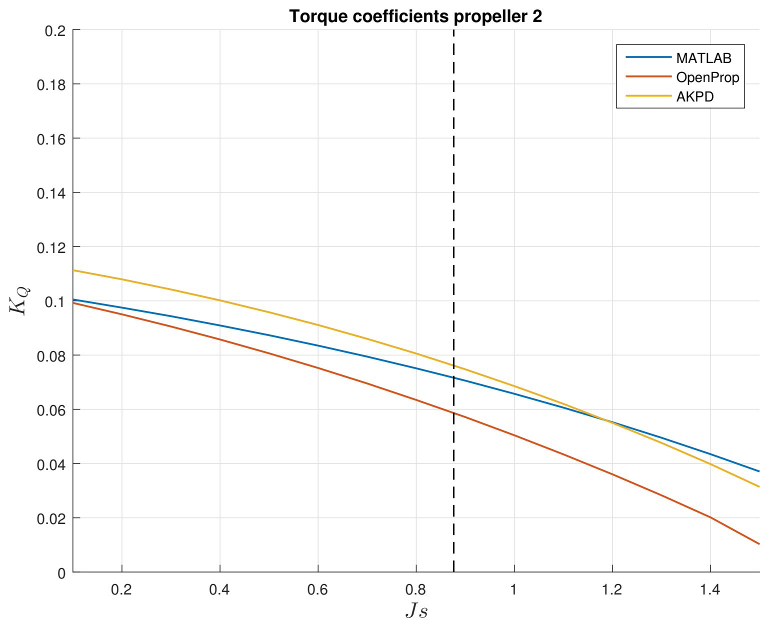

Figure 4.33: Torque coefficients for propeller 2

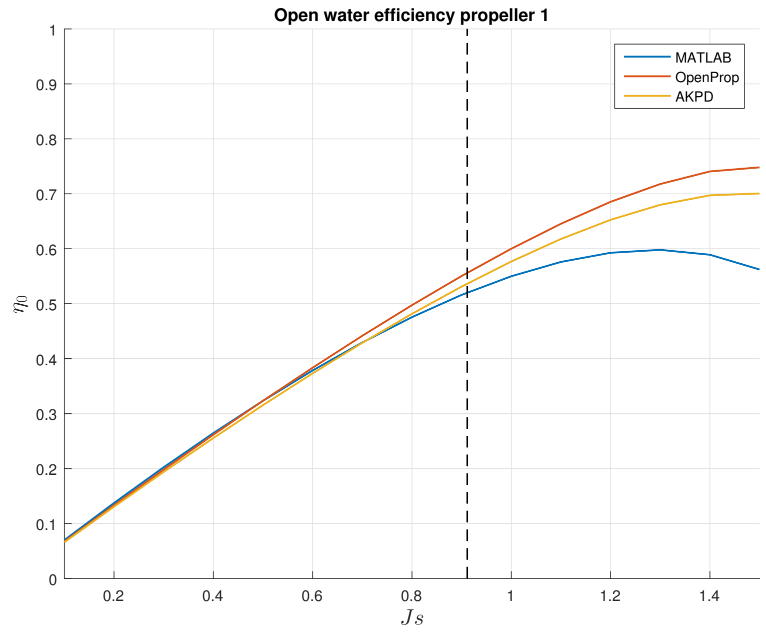

Figure 4.34: Open water efficiencies η_0 for propeller 1

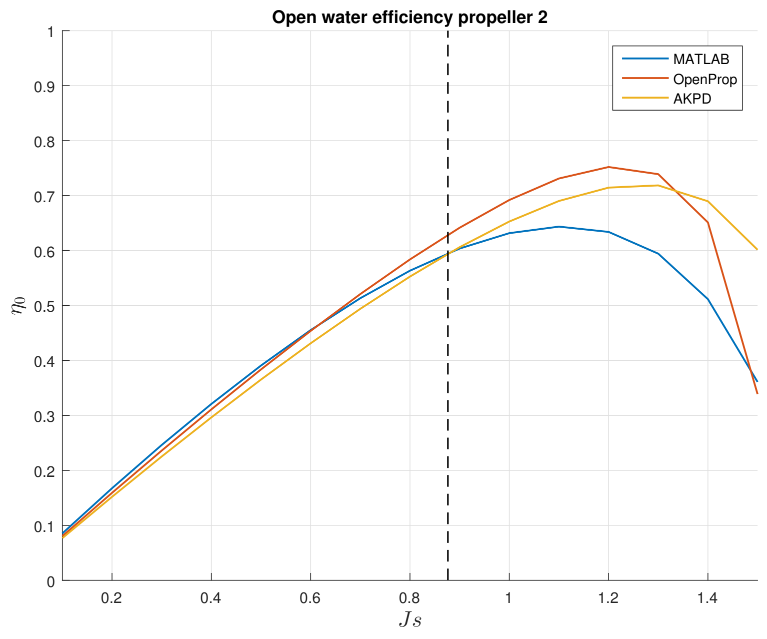

Figure 4.35: Open water efficiencies η_0 for propeller 2

Chapter 5

Discussion

5.1 Validation

The KVLCC2 propeller has been for validation purposes as the propeller has undergone extensive testing and experimental data on performance is readily available (SIMMAN, 2008). The propeller has been analyzed for open water conditions using both AKPA and OpenProp. The results can be seen in figure 4.3, 4.4 and 4.5. As the performance curves in figure 4.3 shows, both AKPA and OpenProp both predict a higher performance for advance numbers below 0.67. OpenProp shows a closer fit for both thrust and torque coefficients for higher advance numbers. The results are less accurate for lower advance numbers as would be expected as the method applied in OpenProp is not accurate for highly loaded propellers. It's important to note that the K_Q-values should be used only as a guide due to the small size of the propeller model. AKPA was chosen as the preferred analysis tool based on the higher accuracy expected from the analysis method.

The design tools on OpenProp and AKPD have been tested with the KVLCC2 geometry. The resulting pitch and camber distributions can be seen in table 4.1. OpenProp prescribes a pitch distribution with very small variations. A small increase can be seen at the tip and a decrease at the root of the blade. Camber values from OpenProp are very low compared to the original propeller. The higher values are found at the blade root and are decreasing to an insignificant amount at 70% of the radius. The pitch distribution on the

AKPD design is steadily decreasing in value from 0.691 at the root to 0.571 at the tip. While the camber values are significantly lower than the original, they are a lot closer than the OpenProp design. In the original geometry the pitch distribution has the lowest values at the root and reaches a maximum at 70% of the radius. Neither AKPD nor OpenProp prescribes a pitch or camber distribution that resembles original geometry.

5.2 Comparison of Designed Geometry

The geometries of the two propellers can be seen in table 4.2, 4.3 and 4.4, and are visualized in figure 4.6-4.9, 4.14, 4.15, 4.20 and 4.21. The initial geometry of chord and thickness distributions was calculated and optimized by the MATLAB program. Thickness distributions were based on scaled values from the Wageningen B-series. The chord distributions were based on a parametric form function, which in hindsight might have been given too much trust. Based on 4 parameters, the form function was capable of producing a range of blade shapes where several passed the thrust requirement and the cavitation check. Some considerations were made to avoid the more extreme designs. It was especially important in the blade sections near the root and tip. Too high gradients in in the chord distribution in these areas could cause problems with both convergence in AKPD and OpenProp, and cavitation. The two propellers ended up with relatively similar blade shapes. Both blades have a relatively short chord length at the root section compared to the maximum chord around 70% of the radius. The difference is more severe for propeller 2. A relatively wide maximum chord resulted in rapidly decreasing chord length towards the tip.

A comparison of the pitch distributions calculated by the design softwares is presented in figure 4.26 and 4.27. The MATLAB program clearly calculates the lowest pitch values for both propellers. In both cases AKPD calculates the highest pitch values. For propeller 1 OpenProp and AKPD seems to produce similar values for a large part of the blade length, but deviate in the root sections. The propeller 2 designs have larger differences in pitch designs from each design tool. The larger variations close to the root might be a result of differences in how hub effects are taken into account. Other reasons for differences in pitch from the design tools are are how OpenProp corrects for 3-dimensional effect and for finite number of blades.

The camber distributions for each design are shown in figure 4.28 and 4.29. It becomes evident that the MATLAB program calculates very high camber values in the root area of the blade. The reason for this is how the hub effect were accounted for. This is explained in more detail in section 3.1.1. By allowing for non-zero circulation at the blade root the resulting optimized circulation distribution reaches very high values close to the hub. The root camber is especially large for propeller 2 which has the thinnest blade width at the hub. AKPD and OpenProp produces similar values at the root sections but deviates towards the tips. OpenProp produces a smooth declining curve while AKPD maintains a relatively stable camber towards the tip. In the AKPD design the camber is slowly decreasing from $r/R = 0.8$ to a non-zero value at the tip.

5.3 Comparison of Predicted Performance

The AKPA performance analyses are shown individually in figure 4.10, 4.11, 4.16, 4.17, 4.22 and 4.23. The predicted performance are plotted for comparison in figure 4.30, 4.31, 4.32, 4.33, 4.34 and 4.35. For both propellers the AKPD and MATLAB designs produce higher thrust and torque than the OpenProp design. The curves for thrust and torque coefficients for the AKPD and OpenProp design stay approximately equidistant for all advance numbers. The MATLAB design for propeller 1 produce the same thrust as the AKPD design at the design point. The thrust is somewhat higher for low advance numbers and lower at the higher advance numbers. For propeller 2 the thrust coefficient for the MATLAB design falls between the OpenProp and AKPD designs, but closer to the AKPD design at the design point. The torque coefficient for the MATLAB design of propeller 1 is the highest at the design point and higher advance numbers. For propeller 2 the AKPD design shows the highest torque coefficient except for advance numbers higher that $J_S = 1.2$.

Both propellers have a blade area ratio A_E/A_0 of approximately 0.55. This means the performance can be compared to the Wageningen B-series diagram in figure 2.1. With a mean pitch ranging from 0.93-1.17 for propeller 1 and 1.04-1.27 for propeller 2 it becomes clear that the thrust and torque coefficients are higher than what would be expected. This is most likely due to the high camber values of the blades.

The OpenProp design is predicted by AKPA to have the highest efficiency for both propeller 1 and 2 at the design point. The MATLAB design shows the lowest efficiency at the design point compared to the other designs for propeller 1. For propeller 2 the predicted performance of the MATLAB design is equivalent to the AKPD design.

5.4 Comparison of Cavitation Analyses

The results of the cavitation analyses done in AKPD for each design are presented in figure 4.12, 4.13, 4.18, 4.19, 4.24 and 4.25. The cavitation analyses of the MATLAB designs predicts cavitation at the tip of the blade. The visualisation also reveals some issues with the tip geometry. This is an artifact from AKPA's inability to handle the zero chord length tip section calculated by the MATLAB program chord distribution function. This tip is the cause of some of the predicted cavitation, but $\frac{-C_P}{\sigma}$ values can be observed to be above the cavitation limit some distance away from the tip section as well. As most propellers are design with some cavitation, the cavitation areas seem to be limited and the within a reasonable ratio. The OpenProp design show barely any cavitation in the tip region for propeller 1 and no cavitation for propeller 2. Lower camber in the tip sections of the designs seems to effectively prevent cavitation. As for the AKPD design the analyses show some cavitation occurring in the tip region. For propeller 1 the cavitation area is relatively limited but the $\frac{-C_P}{\sigma}$ ratio is higher. The opposite is the case for propeller 2.

5.5 Comment on the Lack of CFD Simulation Results

The problem formulation includes the use of CFD for analyzing the propeller designs. An effort was made to complete the simulations, but due to time limitations satisfying results could not be produced. Several software issues occurred during this process. With no prior experience with the CFD solver STAR-CCM+, the author were provided with templates for the simulation set-up. To efficiently use these templates, the propeller geometry had to be imported as parasolid files. These parasolids needed surfaces defined in a specific way. As AKPA is not made to export these files, the blade geometries had to be created using an old meshing tool called Gambit. Gambit used to be a part of the ANSYS Fluent software tool, but is no longer supported. The author managed to get hold of a copy of Gambit version 2.4.6 with support from Klas Johansson at EDR Medeso, but was not able to acquire a working license. EDR Medeso is an engineering company that provides ANSYS support. With help from Senior Research Scientist at MARINTEK, Vladimir Krasilnikov, one set of files were produced by using his copy of Gambit. Another problem surfaced during the geometry mesh generation in STAR-CCM+. The software was at several occasions unable to produce a mesh, due to a low memory capacity on the author's computer. The mesh generation was tested on higher capacity computer owned by another student. A few test simulations were run, but either failed to converge to a solution or crashed due to lack of memory.

Chapter 6

Conclusion and Recommendations for Further Work

6.1 Conclusion

In this master thesis a propeller design program based on the lifting line method was developed. The program was capable of designing a propeller based on two sets of input data and two circumferentially averaged radial wake fields. It calculates the thickness, chord length, pitch and camber distributions to find the most efficient, non-cavitating propeller within the limits of its method. The program, together with AKPD and OpenProp, was used to generate in total six designs based on two different set of characteristics. All six designs were analysed in AKPA to compare the geometry, predicted performance and cavitation. The thesis was not able cover a CFD analysis of the propeller designs within the time limitations. However, based on the work done in this thesis it is able to make the following conclusions:

– Based on the propellers designs analyzed in this thesis, OpenProp was able to produce the better designs. Both of the OpenProp propeller designs achieved the highest efficiency as well as showing the least cavitation.

– OpenProp has an advantage in time required to produce a design. It is able to design and run a performance analysis in a matter of seconds. AKPD requires several minutes to produce a full design if the number of unsteady calculation iterations are set to 5 or above (which is recommended by the author for convergence).

– AKPD is the only design tool of the two which is able to account for effects from skew and rake. Skew is often preferred in modern propeller design in order to reduce cavitation, noise and vibrations.

– AKPD is set up for a seamless transition to AKPA. If AKPA is the preferred analysis program, making the designs in AKPD may end up saving time in the design process.

– While both AKPD and OpenProp are restricted to circumferentially averaged inflow, AKPD can account for inflow in the radial as well as the axial and tangential direction.

6.2 Recommendations for Further Work

The work on this thesis has in many ways been similar to the process of designing a propeller. Each iteration has improved the authors knowledge on the field as well as uncover potential improvements. The thesis did not cover every objective that was initially intended and thus some recommendations for further are more self-evident than others. The authors recommendations are the following:

– Include an analysis of the propeller designs using a CFD software. A RANS simulation may help in giving a more realistic performance estimate as well as highlight cavitation issues and viscous effects. The STAR-CCM+ templates that were provided to the author are still, to the authors knowledge, a valid and time efficient tool for preparing the simulations. However, The Author will suggest that an alternative to Gambit is used to prepare the geometries, since the software is no longer supported by ANSYS.

– Experimental testing of the propeller designs may also provide more data to validation and compare with the analysis results.

– The MATLAB design program developed by the author may be improved in several ways. The program was developed specifically for the work on this thesis and not with the intention of extended use. Should a reader of this thesis be in the process of developing their own lifting line software, these are some suggestions for improvement:

1. Find a better parametric shape function for the chord length distribution. In the authors experience too many of the generated blade shapes have a bad transition toward tip and/or root.
2. Find a way of increasing the computational efficiency of the program. The program is at its present stage a relatively slow runner. One way may be to avoid the use of objects in the MATLAB code, or to use another program language.
3. Implement the Lerbs method (Lerbs, 1952) for higher accuracy in the calculations and to more effectively account for non-zero circulation at the blade root section. This should be done along with point 2 as the implementation in the current code will most certainly increase the already bad running times.

– Look into how skew and rake effects affect the propeller designs in AKPD. A systematic analysis of a propeller with increasing skew or rake values could give some indication of when AKPD would be the preferred design tool.

Bibliography

Abbott, I. H. and Doenhoff, A. E. V. (1959). *Theory of wing sections, including a summary of airfoil data.* Dover Publications.

Achkinadze, A. S. and Krasilnikov, V. I. (2012). *AKPA Manual.*

Achkinadze, A. S., Krasilnikov, V. I., Stepanov, I. E., and Berg, A. (2003). Interactive program system for the design/analysis of marine propulsors. *ISNAOE'03 - Shanghai, China - Sept 23-26, 2003.*

Amini, H. (2011). *Azimuth Proplusors in Off-design Conditions.* PhD thesis, NTNU.

Bernitsas, M., Ray, D., and Kinley, P. (1981). Kt, kq and efficiency curves for the wageningen b-series, technical report, department of naval architecture propellers. Technical report, The University of Michigan.

D'Epagnier, K. P., Chung, H.-L., Stanway, M. J., and Kimball, R. W. (2007). An open source parametric propeller design tool.

Epps, B. P. (2010). Openprop v2.4 theory document.

Epps, B. P. and Kimball, R. W. (2013). Unified rotor lifting line theory. *Journal of Ship Research*, 57(4).

Falkner, V. M. (1947). The solution of lifting-plane problems by vortex-lattice theory. *Aeronautical Research Concil Reports And Memoranda*, (2591).

Froude, R. E. (1889). On the part played in propulsion by differences of fluid pressure. *Transactions of the Royal Institution of Naval Architects*, 30:390–405.

Glauert, H. (1947). *The Elements of Aerofoil and Airscrew Theory.* Cambride University Press.

Goldstein, S. (1929). On the vortex theory of screw propellers. *Proceedings of the Royal Society of London*, 123.

Greeley, D. S. and Kerwin, J. E. (1982). Numerical methods for propeller design and analysis in steady flow. *Transactions of the Society of Naval Architects and Marine Engineers.*

Hess, J. and Valarezo, W. O. (1985). *Calculation of steady flow about propellers by means of a surface panel method.* American Institute of Aeronautics and Astronautics.

Hess, J. L. and Smith, A. M. O. (1964). Calculation of non-lifting potential flow about arbitrary three dimensional bodies. *Journal of Ship Research*, 8.

Hoshino, T. (1989). Hydrodynamic analysis of propellers in steady flow using a surface panel method. *Journal of The Society of Naval Architects of Japan*, 165.

Kerwin, J. E. and Hadler, J. B. (2010). *Propulsion.* The Principles of Naval Architecture. The Society of Naval Architects and Marine Engineers, 601 Pavonia Avenue, Jersey City, New Jersey 07306, 1 edition.

Kerwin, J. E., Kinnas, S. A., Lee, J.-T., and Shih, W.-Z. (1987). A surface panel method for the hydrodynamic analysis of ducted propellers. *Transactions of the Society of Naval Architects and Marine Engineers*, 95.

Kerwin, J. E. and Lee, C.-S. (1978). Prediction of steady and unsteady marine propeller performance by numerical lifting-surface theory. *Transactions of the Society of Naval Architects and Marine Engineers*, 86.

Krasilnikov, V. and Sileo, L. (2012). Cfd modelling of marine propulsors. *MARINTEK Review nr.2 2012*, 2.

Krasilnikov, V. I. (2015). Application of boundary element methods to the analysis of marine propulsion systems. Guest lecture presentation slides.

Lammeren, W. P. A., van Manen, J. D., and Oosterveld, M. W. C. (1969). The wageningen b-screw series. *Journal of The Society of Naval Architects and Marine Engineers*, 77.

Lerbs, H. W. (1952). *Moderately Loaded Propellers With A Finite Number Of Blades And An Arbitrary Distribution Of Circulation.* The Society of Naval Architects and Marine Engineers, 29 West 39th Street, New York 18, N.Y.

Morino, L. and Kuo, C.-C. (1974). Subsonic potential aerodynamics for complex configurations : A general theory. *AIAA Journal*, 12(2):191–197.

Phillips, A. B., Turnock, S. R., and Furlong, M. (2009). Evaluation of manoeuvring coefficients of a self-propelled ship using a blade element momentum propeller model coupled to a reynolds averaged navier stokes flow solver. *Ocean Engineering*, 36(15):1217–1225.

Politis, G. K. (2004). Simulation of unsteady motion of a propeller in a fluid including free wake modeling. *Engineering Analysis with Boundary Elements*, 28(6):633 – 653.

Praefke, E. (2011). The marine propeller design spiral. *Second International Symposium on Marine Propulsors.*

Prantl, L. (1921). Application of modern hydrodynamics to aeronautics. technical report naca tr-116. Technical report, National Advisory Committee for Aeronautics.

Prantl, L. and Tietjens, O. (1934). *Applied Hydro- and Aerodynamics: Based on Lectures of L. Prandtl.* Engineering societies monographs. McGraw-Hill.

Rankine, W. J. M. (1865). On the mechanical principles of the actions of propellers. *Transactions of the Royal Institution of Naval Architects*, 1(13):13–30.

SIMMAN (2008). Kvlcc2 propeller openwater data (nmri model).

Skåland, E. K. (2015). Accuracy of propeller design methods. Master's thesis, NTNU.

Steen, S. (2014). *LECTURE NOTES TMR4220 Naval Hydrodynamics, Foil and Propeller Theory.* Department of Marine Technology, Norwegian University of Technology and Science, Nardoveien 12, 7005 Trondheim.

Wrench, J. (1957). *The Calculation of Propeller Induction Factors: AML Problem 69-54.* Report: David W. Taylor Model Basin. Navy Department, David Taylor Model Basin.

Appendix A

MATLAB code

A.0.1 MAIN.m

```
%*******************************************************************%
%—————————————————————————propDesign———————————————————————————————%
%*******************************************************************%
% This MATLAB code was developed as part of the master thesis in
% marine hydrodynamics at NTNU titled"The Influence of the Choice
% of Propeller Design Tool on Propeller Performance"
%   by  Edvard Knutsen Skaaland                spring 2016
%
% This is propeller design code based on lifting line theory with
% Goldstein factors to calculate induced velocities. Foil section
% camber and thickness values are based on NACA a = 0.8 and NACA 66
% tables. The circulation distribution is set to reach zero at the
    hub
% center to have a more realistic non-zero circulation value at the
% root.
%
% NOTE TO USER:
% The design approach used in this program is based on the method
    described
% in the lecture notes by Sverre Steen for the course Naval
    Hydrodynamics
% at NTNU. This code was only ever intended to generate some basic
% geometries which could be further developed by other design
    programs.
% If someone is reading this with the intention of using this code, I
     refer
% to the recommendation section of my thesis for ways to improve it.


% INPUT DATA
D = 3.5; % propeller diameter [m]
Z = 4; % number of blades
dHub = 1.020; % hub diameter [m]
rHub = dHub/2; % hub radius [m]
RPM = 150; % revolutions per minute
Vs = 15.5; % ship speed [knots]
h = 4.5; % shaft immersion [m]
NoS = 100; % number of strips

% Radial wake field had to be assigned in either radialWake_model.m
    if
% model measurements are given, or in radialWake.m for full scale
    wake.

RShip = 285040; % Ship resistance in newton
tShip = 0.289;  % thrust deduction
Thrust = RShip/(1-tShip); % Required thrust
cavSafe = 0.8; % Cavitation safety factor
% OPTIMIZATION PARAMETERS
a = linspace(0.0,0.5,6);
m = [0.2:0.1:1.0];
c_0 = 0.4*D/2; % Chord length at root
c_1 = 0.9; % Max chord length

opt = optimizeChordAndCirc(c_1,a,m,prop,Thrust);
print = 0; % Print values if print = 1
opt = opt.optimize(c_0,print);
prop = Propeller(D,Z,rHub,RPM,Vs*0.5144444444,h,NoS,Lship,Lmodel);
```

```

% After optimization procedure is done, change parameters to the ones
% reccomended in
a = 0.5;
m = 0.5;
prop.c = chordLength(prop.x, c_0, c_1, a, m);

k = 5.8197;
a = 0.5;
m = 0.6;

prop.Gamma = gammaCalc(prop.r./prop.R,k,a,m);
prop = prop.calculateVelocities();
prop = prop.calculateForces();
prop = prop.cavitationCheck(cavSafe);
prop = prop.calculateGeometricVariables();
```

A.0.2 Propeller.m

```
%
    ***************************************************************************

% ROUTINE: Propeller.m
%
    ___________________________________________________________________________

%
% Object: This routine holds most of the functions used to calculate
      the
% propeller. The routine creates a Propeller object holding all the
% calculated velocities, forces, geometry variables defining the
% propeller.
%
% Inputs:
% Name        Description
%
    _______________________________________________________________________________

% D           Propeller diameter
% Z           Number of Blades
% rBoss       Hub radius
% RPM         Revolutions per minute
% Uship       Ship speed in m/s
% h           Shaft immersion
% nrStrips    Number of strips used in calculations
%
%
    _______________________________________________________________________________

%
    ===============================================================================

% Written by: Edvard Knutsen Skaaland
% Last edited: 10.05.16
%
    ****************************************************************************


classdef Propeller
    properties
        D;        % Diameter
        R;       % Radius
        Z;       % Number of blades
        rBoss;   % Hub radius
        RPM;     % Revolutions per minute
        omega;   % Angular velocity
        Uship;   % Ship speed in m/s
        Lship;   % Length of ship (Only used for scaling of model wake
            )
        Umodel;  % Model ship speed
        Lmodel;  % Length of model ship
        h;       % Shaft immersion
        nrStrips; % Number of Strips used in calculations

        rho;     % Density of sea water
        g;       % Gravitational acceleration
        pv;      % Vapor pressure
        patm;    % Atmospheric pressure
        nu;      % Kinematic viscosity

        r;       % Vector of radial calculation points. 0 at center of
             hub
        x;       % Vector of radial calculation point. from 0 to 1
            root to tip
        dr;      % length between calculation points
        c;       % Vector of sectional chord lengths
        thickness; % Vector of sectional max thickness
        Gamma;      % Vector of circulation distribution
        sigma;      % cavitation numbers
        Cl;         % Lift coefficients
        Cd;         % drag coefficients
```

```
    dL;          % section lift
    dD;          % section drag
    dT;          % section thrust
    dQ;          % section torque
    CpMin;       % minimum pressure coefficient
    %Velocities
    Rn;          % Reynold numbers
    w;           % Radial wake
    U_A;         % Axial velocity
    U_T;         % Tangential Velocity
    Umag;        % Magnitude of sectional velocity vector
    U_A0;        % Axial velocity based on ship speed - wake
    U_T0;        % Tangential velocity based on rotational velovity
    % Induced velocities
    u_Amom;      % Induced axial velocity momentum theory
    u_Tmom;      % Induced tangential velocity momentum theory
    u_Agol;      % Induced axial velocity based on goldstein
        factors
    u_Tgol;      % Induced tangential velocity based on goldstein
        factors
    u_A;         % Induced axial velocity used in calculations
    u_T;         % Induced tangential velocity used in calculations
    % Pitch
    beta;        % angle between U_A0 and U_T0
    beta_i;      % angle axial and tangential velocities including
        induced
    beta_iMom; % angle based on momentum theory
    beta_iGol; % angle based on goldstein theory
    alpha_i_2D;% angle of attack for 2D section
    alpha_i;     % angle of attack 3D
```

```
            phi;          % pitch angle
            PD;           % Pitch / Diameter
            camber;       % Camber
            % Forces
            Thrust;
            Torque;
            eta0;

            % Cavitation;
            cavitation;  % Cavitation string 'yes' for cav 'no' for no
                cav
        end
        methods
            function obj = Propeller(D,Z,rBoss,RPM,Uship,h,nrStrips,Lship
                ,Lmodel)
                % Environmental parameters
                obj.g = 9.81;         % Gravitational acceleration [m/s^2]
                obj.rho = 1025;       % Density of seawater [kg/m^3]
                obj.patm = 101325;    % Atmospheric pressure [Pa]
                obj.pv = 1500;        % Vapor pressure [Pa]
                obj.nu = 10e-6;       % Kinematic  viscosity [m^2/s]

                % Propeller characteristics
                obj.D = D;
                obj.Z = Z;
                obj.rBoss = rBoss;
                obj.RPM = RPM;
                obj.h = h;
                obj.nrStrips = nrStrips;
                obj.R = obj.D/2;
                obj.omega = 2*pi*obj.RPM/60;

                % Ship data
                obj.Uship = Uship;
                obj.Lship = Lship;
                obj.Lmodel = Lmodel;
                obj.Umodel = obj.Uship*sqrt(obj.Lmodel/obj.Lship);

                % Blade variables
                obj.dr = (obj.R-obj.rBoss)/(obj.nrStrips-1);

                obj.r = zeros(1,obj.nrStrips);
                obj.x = zeros(1,obj.nrStrips);

                for i = 1:obj.nrStrips;
                    obj.r(i) = obj.rBoss+(i-1)*obj.dr;
                    obj.x(i) = (obj.r(i) - obj.rBoss)/(obj.R-obj.rBoss);
                end

                obj.c = zeros(1,obj.nrStrips);
                % Calculating thickness distribution
                t = Thickness(obj.D,obj.Z);
                obj.thickness =t.t(obj.r);

                obj.Gamma = zeros(1,obj.nrStrips);
                obj.sigma = zeros(1,obj.nrStrips);

                obj.Cl = zeros(1,obj.nrStrips);
                obj.Cd = zeros(1,obj.nrStrips);
                obj.dL = zeros(1,obj.nrStrips);
                obj.dD = zeros(1,obj.nrStrips);
                obj.dT = zeros(1,obj.nrStrips);
                obj.dQ = zeros(1,obj.nrStrips);
                obj.CpMin = zeros(1,obj.nrStrips);
                % Velocities
                obj.Rn = zeros(1,obj.nrStrips);
                obj.w = radialWake(obj,obj.r);
                obj.U_A = zeros(1,obj.nrStrips);
                obj.U_T = zeros(1,obj.nrStrips);
                obj.Umag = zeros(1,obj.nrStrips);

                obj.U_A0 = obj.Uship*(ones(1,obj.nrStrips)-obj.w);
                obj.U_T0 = obj.omega*obj.r;

                % Induced velocities
                obj.u_Amom = zeros(1,obj.nrStrips);
                obj.u_Tmom = zeros(1,obj.nrStrips);

                obj.u_Agol = zeros(1,obj.nrStrips);
                obj.u_Tgol = zeros(1,obj.nrStrips);

                obj.u_A = zeros(1,obj.nrStrips);
                obj.u_T = zeros(1,obj.nrStrips);

                % Pitch
                obj.beta = atan(obj.U_A0./obj.U_T0);
                obj.beta_i = zeros(1,obj.nrStrips);
                obj.beta_iMom = zeros(1,obj.nrStrips);
                obj.beta_iGol = zeros(1,obj.nrStrips);

                obj.alpha_i_2D = zeros(1,obj.nrStrips);
                obj.alpha_i = zeros(1,obj.nrStrips);
                obj.phi = zeros(1,obj.nrStrips);
                obj.PD = zeros(1,obj.nrStrips);
                obj.camber = zeros(1,obj.nrStrips);

                % Forces
                obj.Thrust = 0;
                obj.Torque = 0;
                obj.eta0 = 0;

                % Cavitation
                obj.cavitation = '';
                obj.cavitationExtra1 = '';
                obj.cavitationExtra2 = '';
            end
            function obj = momentumTheoryVelocities(obj)
               % Calculate induced velocities and hydrodynamic pitch
                   angle
               % based on momentum theory
               obj.u_Tmom = obj.Z*obj.Gamma./(2*pi*obj.r);

               a = 0.5;
               b = obj.U_A0;
               cc = -obj.u_Tmom.*(obj.U_T0 - 0.5.*obj.u_Tmom);

               obj.u_Amom = 0.5.*((-b+sqrt(b.^2-4.*a.*cc))./(2*a));
               obj.beta_iMom = atan2(obj.U_A0+obj.u_Amom./2, obj.U_T0-obj
                   .u_Tmom./2);
            end
            function obj = goldsteinFactorVelocities(obj)
                % Calculate velocities based on goldstein factors
               xx = obj.r./obj.R;
               Xi = zeros(1,obj.nrStrips);
               for i = 1:obj.nrStrips
                   Xi(i) = goldsteinFactor(xx(i),obj.beta_iMom(i));
               end
               obj.u_Tgol = obj.u_Tmom./Xi;
               obj.u_Agol = obj.u_Amom./Xi;

               obj.beta_iGol = atan2(obj.U_A0+obj.u_Agol./2, obj.U_T0-obj
                   .u_Tgol./2);
            end
            function obj = calculateVelocities(obj)
                % Calculate induced velocities
                obj = obj.momentumTheoryVelocities();
                % Update based on Goldstein factors
                obj = obj.goldsteinFactorVelocities();
                % Change actual velocities to Golstein
                obj.u_T = obj.u_Tgol;
                obj.u_A = obj.u_Agol;
                obj.U_T = obj.U_T0 - obj.u_T./2;
                obj.U_A = obj.U_A0 + obj.u_A./2;

                obj.Umag = sqrt(obj.U_T.^2+obj.U_A.^2);
                obj.beta_i = atan2(obj.U_A, obj.U_T);

            end
            function obj = calculateForces(obj)
                % Calculate lift
                obj.dL = obj.rho.*obj.Gamma.*obj.Umag;

                % Calculate lift coefficient
                obj.Cl = obj.dL./(0.5*obj.rho.*obj.c.*obj.Umag.^2);
                obj.Cl(1) = 2*obj.Cl(2) - obj.Cl(3);
                obj.Cl(end) = 2*obj.Cl(end-1) - obj.Cl(end-2);

                % Calculate drag coefficient
                obj.Rn = obj.Umag.*obj.c./obj.nu;

                obj.Cd = dragCoef(obj.Rn, obj.Cl, obj.thickness./obj.c);
                obj.Cd(1) = 2*obj.Cd(2) - obj.Cd(3);
                obj.Cd(end) = 2*obj.Cd(end-1) - obj.Cd(end-2);

                % Calculate drag
                obj.dD = 0.5*obj.rho*obj.Cd.*obj.c.*obj.Umag.^2;

                % Calculate thrust
                obj.dT = obj.rho.*obj.Gamma.*obj.U_T - obj.dD.*sin(obj.
                    beta_i);

                % Calculate torque
                obj.dQ = obj.rho.*obj.Gamma.*obj.U_A.*obj.r + obj.dD.*cos
                    (obj.beta_i);

                % Integrate to find global forces
                obj.Thrust = obj.Z.*trapz(obj.r,obj.dT);
                obj.Torque = obj.Z.*trapz(obj.r,obj.dQ);

                obj.eta0 = mean(obj.Thrust.*obj.U_A0./(obj.omega.*obj.
                    Torque));
            end
            function obj = cavitationCheck(obj, cavsafe_f)
                % Cavitation check with radial wake
                obj.sigma = (obj.patm - obj.pv + obj.rho.*obj.g*(obj.h-obj
                    .r))./(0.5.*obj.rho.*obj.Umag.^2);
                obj.CpMin = minPress(obj.Cl, obj.thickness./obj.c);
                obj.CpMin(1) = 2*obj.CpMin(2) -obj.CpMin(3);
                obj.CpMin(end) = 2*obj.CpMin(end-1) - obj.CpMin(end-2);

                % Check for radial cavitation
                if all(-obj.CpMin <= cavsafe_f.*obj.sigma)
                    obj.cavitation = 'False';
                else
                    obj.cavitation = 'True';
                end

            function obj = calculateGeometricVariables(obj)
                % Calculate Geometric variables
                obj.alpha_i_2D = idealAngle(obj.Cl);

                % 3D correction factors
                xx = obj.r./obj.R;
                k_c = 1.6946 + 0.5048.*xx - 4.0012.*xx.^2 + 4.3283.*xx
                    .^3;
                k_a = 1 + 1.46.*xx.^3;
                k_t = 2.5*obj.Z.*obj.c.*cos(obj.beta_i)./(obj.D.*xx);
                % camber corrections
                obj.camber = k_c.*maxCamber(obj.Cl);

```

```
                a_i3 = obj.alpha_i_2D.*k_a;
                a_t = k_t.*obj.thickness./obj.c;

                obj.alpha_i = a_i3 + a_t;
                % Pitch Angle
                obj.phi = obj.alpha_i.*pi/180 + obj.beta_i;
                % Pitch/Diameter
                obj.PD = 2*pi.*obj.r.*tan(obj.phi)./obj.D;
            end
        end
end
```

A.0.3 radialWake_model.m

```
function w_m = radialWake_model(prop,r)
      % This function defines the model wake field
      % Values in the two vectors below need to be changed to
          required
      % radial wake field. The wake is only given in the axial
          direction
      r0 = [0.01, 0.606, 0.847, 1.087, 1.202, 1.327]*prop.R;
      w0 = [0.496, 0.496, 0.300, 0.193, 0.184, 0.160];

      d0 = [-r0(end:-1:1),r0];
      fullwake = [w0(end:-1:1),w0];
    spl = spline(d0,fullwake);
    w_m = ppval(r,spl);
end
```

A.0.4 radialWake.m

```
function w = radialWake(obj,r,t)
    % This function is scaling model wake to full scale wake
    % Inputs are the propeller object, radus vector and thrust
        deduction
    w_m = radialWake_model(obj,r);

    %t = 0.289; % estimation of thrust deduction

    Rn = obj.Uship*obj.Lship/obj.nu;
    Rn_m = obj.Umodel*obj.Lmodel/obj.nu;

    CF = 0.075/(log10(Rn)-2)^2;
    CF_m = 0.075/(log10(Rn_m)-2)^2;

    w = zeros(1,length(w_m));

    for i = 1:length(w_m)
        w(i) = 0.04+t+(w_m(i)-0.04-t)*CF/CF_m;
        if w(i) > w_m(i);
            w(i) = w_m(i);
        end
    end
end
```

A.0.5 optimizeChordAndCirc.m

```
% This MATLAB class finds the optimum chord and circulation
    distribution
% by running through a^2 * m^2 possible forms
classdef optimizeChordAndCirc
    properties
```

```
        Treq; % Required thrust
        m;    % form function parameter
        mNr;  % number of m parameters
        a;    % form function parameter
        aNr;  % number of a parameters
        mGrid;% grid of m parameters
        aGrid;% grid of a parameters
        eta0; % Propeller efficiency
        thrust;     % calculated thrust
        cavitation; % cavitation
        k;    % form function parameter
        prop; % propeller object
        eta0_max; % highest achieved efficiency
        m_max; % m parameter for highest efficiency
        a_max; % a parameter for highest efficiency
        k_max; % k parameter for highest efficiency
        c;     % chamber form function parameter
        k_circ; % k parameter for circulation
        a_circ; % a parameter for circulation
        m_circ; % m parameter for circulation
        k_circMax; % k parameter for highest efficiency
        a_circMax; % a parameter for highest efficiency
        m_circMax; % m parameter for highest efficiency
    end
    methods
        function obj = optimizeChordAndCirc(c,a,m,prop,Treq)
            obj.Treq = Treq;
            obj.m = m;
            obj.mNr = length(obj.m);

            obj.a = a;
            obj.aNr = length(obj.a);

            obj.c = c;

            [obj.mGrid,obj.aGrid] = meshgrid(m,a);


            obj.eta0 = zeros(obj.aNr,obj.mNr);
            obj.thrust = zeros(obj.aNr,obj.mNr);
            obj.cavitation = zeros(obj.aNr,obj.mNr);
            obj.k_circ = zeros(obj.aNr,obj.mNr);
            obj.a_circ = zeros(obj.aNr,obj.mNr);
            obj.m_circ = zeros(obj.aNr,obj.mNr);
            obj.prop = prop;
            obj.eta0_max = 0;
            obj.m_max = 0;
            obj.a_max = 0;
            obj.k_max = 0;
            obj.k_circMax = 0;
            obj.a_circMax = 0;
            obj.m_circMax = 0;
        end
        function obj = optimize(obj,c_0,print)
            for i=1:obj.aNr
                for j = 1:obj.mNr
                    prop1 = obj.prop;
                    a1 = obj.a(i);
                    m1 = obj.m(j);
                    disp(['Analyzing a = ',num2str(a1),' and m = ',
                        num2str(ml)]);
                    prop1.c = chordLength(prop1.x,c_0,obj.c,a1,ml);

                    opt = optimizeCirculation(obj.a,obj.m,prop1,obj.
                        Treq);
                    opt = opt.optimize(print);
                    obj.eta0(i,j) = opt.eta0_max;
                    obj.a_circ(i,j) = opt.a_max;
                    obj.m_circ(i,j) = opt.m_max;
                    obj.k_circ(i,j) = opt.k_max;
                end
            end
            if numel(find(obj.eta0))>0

                [maxval,maxloc] = max(obj.eta0(:));
                [maxval_row,maxval_col] = ind2sub(size(obj.eta0),
                    maxloc);
                obj.eta0_max = maxval;
                obj.m_max = obj.m(maxval_col);
                obj.a_max = obj.a(maxval_row);

                obj.k_circMax = obj.k_circ(maxval_row,maxval_col);
                obj.a_circMax = obj.a_circ(maxval_row,maxval_col);
                obj.m_circMax = obj.m_circ(maxval_row,maxval_col);
            else
                obj.eta0_max = 0;
                obj.m_max = 0;
                obj.a_max = 0;
                obj.k_circMax = 0;
                obj.a_circMax = 0;
```

```
                    obj.m_circMax = 0;
                end
                disp(['Optimal chord length (a,m): ',num2str([obj.a_max,
                    obj.m_max])]);
                disp(['Optimal circulation distribution (k,a,m): ',
                    num2str([obj.k_circMax,obj.a_circMax,obj.m_circMax])]);
                disp(['eta_0 = ', num2str(obj.eta0_max)]);
            end
        end
end
```

A.0.6 optimizeCirculation.m

```
% This MATLAB class finds the optimum circulation distribution
% based on the propeller geometry
classdef optimizeCirculation
    properties
        Treq; % Required thrust
        m;      % form function parameter
        mNr;    % number of m parameters
        a;      % form function parameter
        aNr;    % number of a parameters
        mGrid;% grid of m parameters
        aGrid;% grid of a parameters
        eta0; % Propeller efficiency
        thrust;       % calculated thrust
        cavitation; % cavitation
        k;      % form function parameter
        prop; % propeller object
        eta0_max; % highest achieved efficiency
        m_max; % m parameter for highest efficiency
        a_max; % a parameter for highest efficiency
        k_max; % k parameter for highest efficiency
    end
    methods
        function obj = optimizeCirculation(a,m,prop,Treq)
            obj.Treq = Treq;
            obj.m = m;
            obj.mNr = length(obj.m);

            obj.a = a;
            obj.aNr = length(obj.a);

            [obj.mGrid,obj.aGrid] = meshgrid(m,a);
            %obj.aGrid = meshgrid(m,a);

            obj.eta0 = zeros(obj.aNr,obj.mNr);
            obj.thrust = zeros(obj.aNr,obj.mNr);
            obj.cavitation = zeros(obj.aNr,obj.mNr);
            obj.k = zeros(obj.aNr,obj.mNr);

            obj.prop = prop;
            obj.eta0_max = 0;
            obj.m_max = 0;
            obj.a_max = 0;
            obj.k_max = 0;
        end
        function obj = optimize(obj,print)
            for i=1:obj.aNr
                for j = 1:obj.mNr
                    prop1 = obj.prop;
                a1 = obj.a(i);
                m1 = obj.m(j);
                if print == 1;
                    disp(['Analyzing a = ',num2str(a1),' and m = '
                        ,num2str(m1)]);
                end
                % Adjust k to get necessary thrust
                thrust_k = kThrust(prop1, a1, m1);
                %disp(obj.Treq)

                k1 = thrust_k.kAdjust(obj.Treq,7);

                if k1~= 0
                    % set circulation distribution
                    prop1.Gamma = gammaCalc(prop1.r./prop1.R,k1,a1
                        ,m1);
                    % calculate velocities
                    prop1 = prop1.calculateVelocities();
                    % calculate forces
                    prop1 = prop1.calculateForces();
                    % check for cavitation
                    prop1 = prop1.cavitationCheck(0.8);

                    obj.eta0(i,j) = prop1.eta0;

                    obj.thrust(i,j) = prop1.Thrust;
                    obj.k(i,j) = k1;
                    % Check for cavitation
                    if strcmp(prop1.cavitation,'True')
                        % If true
                        obj.cavitation(i,j) = 1;
                        obj.eta0(i,j) = 0; % Set efficiency to zero
                    end
                else

                    obj.eta0(i,j) = 0;
                    obj.k(i,j)    = 0;
                    obj.thrust(i,j) = 0;

                end

            end
        end
        if numel(find(obj.eta0))>0

            [maxval,maxloc] = max(obj.eta0(:));
            [maxval_row,maxval_col] = ind2sub(size(obj.eta0), maxloc);

            obj.eta0_max = maxval;
            obj.m_max = obj.m(maxval_col);
            obj.a_max = obj.a(maxval_row);
            obj.k_max = obj.k(maxval_row,maxval_col);
        else
            obj.eta0_max = 0;
            obj.m_max = 0;
            obj.a_max = 0;
            obj.k_max = 0;
        end
```

```
            if print == 1
                disp(['Optimal circulation distribution (k, a, m) = ',
                    num2str([obj.k_max, obj.a_max, obj.m_max])])
            end
        end
    end
end
```

A.0.7 goldsteinFactor.m

```
function GF = goldsteinFactor(x, beta_i)
    % This functions returns interpolated goldstein factors
    % to fit the beta_i and x vector
    beta_int = linspace(0,70,15)*pi/180;

    r = [0.95, 0.9, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2];
    Xi =[1.000, 1.000, 1.000, 1.000, 1.000, 1.000, 1.000, 1.000,
        1.000;
         0.804, 0.949, 0.997, 0.999, 1.000, 1.000, 1.000, 1.000,
             1.000;
         0.620, 0.810, 0.959, 0.993, 0.998, 0.999, 0.999, 0.997,
             0.997;
         0.514, 0.696, 0.890, 0.966, 0.989, 0.994, 0.992, 0.983,
             0.993;
         0.440, 0.609, 0.813, 0.921, 0.969, 0.983, 0.982, 0.964,
             0.988;
         0.385, 0.539, 0.742, 0.868, 0.938, 0.967, 0.970, 0.946,
             0.984;
         0.341, 0.483, 0.679, 0.814, 0.902, 0.948, 0.959, 0.933,
             0.984;
         0.307, 0.437, 0.624, 0.763, 0.864, 0.927, 0.950, 0.926,
            0.989;
        0.279, 0.400, 0.578, 0.717, 0.828, 0.906, 0.944, 0.927,
            1.001;
        0.257, 0.369, 0.539, 0.678, 0.795, 0.886, 0.941, 0.935,
            1.020;
        0.240, 0.345, 0.507, 0.644, 0.766, 0.869, 0.941, 0.951,
            1.050;
        0.225, 0.325, 0.481, 0.617, 0.741, 0.854, 0.944, 0.973,
            1.091;
        0.214, 0.309, 0.460, 0.594, 0.721, 0.843, 0.949, 1.000,
            1.145;
        0.205, 0.297, 0.440, 0.576, 0.705, 0.834, 0.956, 1.033,
            1.214;
        0.198, 0.288, 0.431, 0.562, 0.694, 0.829, 0.965, 1.068,
            1.300];

    intData = zeros(1,15);

    for i=1:15
        spl = spline(r(end:-1:1),Xi(i,(end:-1:1)));
        intData(1,i) = ppval(x,spl);
    end
spl = spline(beta_int,intData);
GF = ppval(beta_i,spl);
end
```

A.0.8 Thickness.m

```
classdef Thickness
    % This class sets the propeller blade max thickness based on the
    % Wageningen B-series
    properties
        D;
        R;
        Z;
        r;
        Ar;
        Br;
    end
    methods
        function obj = Thickness(D,Z)
            obj.D = D;
            obj.R = obj.D/2;
            obj.Z = Z;

            obj.r = linspace(0.0,1,11)*obj.R;

            obj.Ar = [0.0650, 0.0588, 0.0526, 0.0464, 0.0402, 0.0340,
                0.0278, 0.0216, 0.0154, 0.0092, 0.0030]*obj.D;
            obj.Br = [0.0050, 0.0045, 0.0040, 0.0035, 0.0030, 0.0025,
                0.0020, 0.0015, 0.0010, 0.0005, 0.0000]*obj.D;
        end
        function t = t(obj,r)
            Ar_int = interp1(obj.r,obj.Ar,r);
            Br_int = interp1(obj.r,obj.Br,r);
            t = Ar_int - Br_int*obj.Z;
        end
    end
end
```

A.0.9 dragCoef.m

```
function Cd = dragCoef(Rn, Cl, t_max)
    % This function calculates the drag coeffient from Reynolds
        number,
    % lift coefficient and maximum sectional thickness
    Cd = (0.075./(log10(Rn)-2).^2).*(1+2.*t_max).*(1+(Cl.^2)./8);
end
```

A.0.10 chordLength.m

```
%FUNCTION chordLength
%This function calculates the radial chord length distribution

function [chord_length] = chordLength(x,c_0,c_1,a,m)
    chord_length = c_0*(1-x).^1 + c_1.*(sin(pi*x) - a.*sin(2*pi.*x))
        .^m;
end
```