University of Cincinnati

Date: 5/7/2012

I, Omkar G Champhekar, hereby submit this original work as part of the requirements for the degree of Master of Science in Mechanical Engineering.

It is entitled:
Inverse Design of Two-Dimensional Centrifugal Pump Impeller Blades using Inviscid Analysis and OpenFOAM

Student’s name: Omkar G Champhekar

This work and its defense approved by:

Committee chair: Urmila Ghia, PhD
Committee member: Shaaban Abdallah, PhD
Committee member: Kirita Ghia, PhD
Committee member: Milind Jog, PhD
Inverse Design of Two-Dimensional Centrifugal Pump Impeller

Blades using Inviscid Analysis and OpenFOAM

A thesis submitted to the Graduate School of the
University of Cincinnati
in partial fulfillment of the requirements for the degree of
Master of Science
in the School of Dynamic Systems
of the College of Engineering and Applied Sciences
by
Omkar Champhekar
Bachelor of Technology, Mechanical Engineering
University of Pune, India, August 2007

July 2012
Advisor: Dr. Urmila Ghia/Dr. Kirti Ghia
Abstract

Inverse design of centrifugal pump impeller blades is a widely used technique for designing pump blades. In a typical Computational Fluid Dynamics (CFD) analysis, the geometry (flow domain) is prescribed, and the governing equations are solved over the flow domain, subject to appropriate boundary conditions. In the inverse design technique, the geometry is unknown, while the desired flow field characteristics on the geometry (blades of pump impeller) are prescribed. Starting with an initial guess for the blade shape, the final shape of the blades is obtained iteratively. The present work uses OpenFOAM, an open source CFD software, which solves the governing equations using a finite-volume method (FVM) for the inverse design of two-dimensional centrifugal pump impeller blades.

A CFD analysis using FVM requires mesh to be of good quality. Since the pump blade has high curvature, careful consideration has to be given while generating the mesh. The present work explains the geometry generation and meshing of the geometry, to obtain a good quality mesh in the Gambit software.

The inverse design technique in the present study is based on assumption of a potential flow field. In the potential flow analysis of pumps, the circulation generated by a pump is an unknown. This unknown appears in a boundary condition downstream of the blade. An iterative method has been implemented in OpenFOAM for calculating the circulation. The circulation value calculated using OpenFOAM compares well with analytical value of circulation generated for the test case of an impeller with logarithmic spiral blades. A grid-independence study shows that, as the grid is refined, the value of the circulation generated by the pump impeller approaches the analytical value.
Inverse design is an iterative process, which is carried out till a converged blade shape is obtained, while satisfying the prescribed flow characteristics. For every new shape generated during iteration, the geometry creation and grid generation are automated using a Journal file in Gambit. The mesh is then imported in OpenFOAM. Inverse design of a pump blade requires specification of swirl distribution along the blade. To accommodate this swirl distribution in the boundary condition on the blade, the OpenFOAM code is modified. The iterative process has been automated, and linked with Gambit, using a Linux shell script.

The implementation of the present inverse design process in OpenFOAM is verified for a two-dimensional case. The swirl distribution generated on the blade is calculated by analyzing potential flow in a blade-to-blade channel for an assumed blade shape. This swirl distribution is then imposed on a flat blade, and the inverse design iterations are carried out, till a converged blade shape is obtained. This inverse design blade shape matches closely with the assumed shape of the blade. The method is further used to redesign a blade which has non-zero incident flow at the leading edge. The inverse design method redesigns the blade such that the flow is tangential to the blade at leading edge. Following chapters explain the implementation of inverse design method using Gambit and OpenFOAM.
Acknowledgements

First of all, I would like to thank Dr. Urmila Ghia and Dr. Kirti Ghia for their tremendous support throughout my journey in the Master of Science Program. The courses taught by them have raised the level of my understanding in Computational Fluid Dynamics to a totally new level. Their constant encouragement kept my interest level high throughout. Along with the constant guidance, they also provided me a unique academic opportunity of teaching undergraduate students for three quarters, and a professional opportunity of working with Procter and Gamble (P&G). A special thanks to Ken Comer for being a fantastic boss at P&G.

I would like to thank Dr. Shaaban Abdallah and Dr. Milind Jog for being on my Master’s Thesis defense committee. The feedback given by them helped me improve the quality of my work. The computer labs, and various CFD software availability provided by University of Cincinnati helped me do my work at any time I wanted. I feel proud of being a graduate student of one of the best colleges in United States of America.

I would like to acknowledge all my Computational Fluid Dynamics Research Lab (CFDRL) members for giving me feedback on my work. I am also grateful to Rob Ogden for helping me with the computer cluster at CFDRL. Special thanks to Nikhil Kumar Palakurthi and Prahit Dubey for helping me with my thesis document and presentation, and their priceless guidance on the academic as well as the personal aspects of life.

Across the seven seas, praying for my success everyday – my family. Their constant encouragement, support, and guidance have helped me sail through the rough seas I faced since the day I left my home.
# Contents

<table>
<thead>
<tr>
<th>Title</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>..........................Page Number</td>
</tr>
<tr>
<td>1. Introduction</td>
<td>.......................................................... 1</td>
</tr>
<tr>
<td>1.1 Motivation for Present Work</td>
<td>.......................................................... 1</td>
</tr>
<tr>
<td>1.2 Basic Pump Theory</td>
<td>.......................................................... 3</td>
</tr>
<tr>
<td>2. Literature Survey</td>
<td>.......................................................... 8</td>
</tr>
<tr>
<td>2.1 Potential Flow Analysis of Turbomachines</td>
<td>.......................................................... 8</td>
</tr>
<tr>
<td>2.2 Inverse Design Methods</td>
<td>.......................................................... 10</td>
</tr>
<tr>
<td>2.3 Objectives of Current Work</td>
<td>.......................................................... 12</td>
</tr>
<tr>
<td>3. Mathematical Formulation</td>
<td>.......................................................... 13</td>
</tr>
<tr>
<td>3.1 Flow Domain</td>
<td>.......................................................... 13</td>
</tr>
<tr>
<td>3.2 Governing Equation for potential flow and its Solution in OpenFOAM</td>
<td>.......................................................... 14</td>
</tr>
<tr>
<td>3.3 Boundary Conditions</td>
<td>.......................................................... 15</td>
</tr>
<tr>
<td>3.4 Mesh Considerations</td>
<td>.......................................................... 18</td>
</tr>
<tr>
<td>3.5 Solution Schemes in OpenFOAM</td>
<td>.......................................................... 22</td>
</tr>
<tr>
<td>4.1 Determination of Circulation by Analytical Expression</td>
<td>.......................................................... 25</td>
</tr>
<tr>
<td>4.2 Numerical Determination of Unknown Circulation</td>
<td>.......................................................... 27</td>
</tr>
<tr>
<td>4.3 Grid-Independence Study</td>
<td>.......................................................... 28</td>
</tr>
<tr>
<td>5. Inverse Design Methodology Implementation in OpenFOAM and its Verification</td>
<td>.......................................................... 31</td>
</tr>
<tr>
<td>5.1 Introduction</td>
<td>.......................................................... 31</td>
</tr>
<tr>
<td>5.2 Flow Domain and Boundary Conditions</td>
<td>.......................................................... 31</td>
</tr>
<tr>
<td>5.3 Input Quantities for Inverse Design</td>
<td>.......................................................... 33</td>
</tr>
<tr>
<td>5.4 Constraints on the Mean Swirl Distribution</td>
<td>.......................................................... 33</td>
</tr>
</tbody>
</table>
## Table of Contents

5.5 Implementation of inverse design methodology in OpenFOAM and Gambit .......... 34
5.6 Calculation of Wrap Angle .............................................................................. 34
  5.6.1 Geometry generation in Gambit .................................................................... 37
  5.6.2 Hard linking of periodic faces in Gambit .................................................... 40
  5.6.3 Grid generation in inverse design procedure ............................................... 41
  5.6.4 Journal file in Gambit .................................................................................. 42
  5.6.5 Solver and boundary conditions in OpenFOAM .......................................... 43
  5.6.6 Calculation of velocities .............................................................................. 44
  5.6.7 Iterations using Linux Shell Script .............................................................. 44
5.7 Verification of Blade Angle Calculation Code .................................................. 45
5.8 Verification of Inverse Design Procedure for 2D Spiral Blade ......................... 49
  5.8.1 Initial geometry and boundary conditions .................................................... 49
  5.8.2 Results ........................................................................................................ 51
  5.8.3 Grid-Independence Study ........................................................................... 53

6. Pump Performance Improvement Using Inverse Design ................................. 56
  6.1 Pump Parameters ............................................................................................ 56
  6.2 Potential-Flow Analysis .................................................................................. 57
  6.3 Redesigning of Blade ...................................................................................... 59

7. Conclusions and recommendations for future work ....................................... 64
  7.1 Conclusions .................................................................................................... 64
  7.2 Future Work ................................................................................................... 65

8. References ......................................................................................................... 66

Appendix .................................................................................................................. A.1
# List of Figures

<table>
<thead>
<tr>
<th>Title</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1.1: Schematic of meridional section</td>
<td>5</td>
</tr>
<tr>
<td>Figure 1.2: Wrap angle and blade angle</td>
<td>6</td>
</tr>
<tr>
<td>Figure 1.3: Velocity triangles</td>
<td>6</td>
</tr>
<tr>
<td>Figure 3.1: Blade-to-blade passage for two-dimensional domain</td>
<td>14</td>
</tr>
<tr>
<td>Figure 3.2: Blade-to-blade passage for three-dimensional domain</td>
<td>14</td>
</tr>
<tr>
<td>Figure 3.3: Domain decomposition of blade-to-blade passage</td>
<td>19</td>
</tr>
<tr>
<td>Figure 3.4: Highly Hex-Map mesh</td>
<td>21</td>
</tr>
<tr>
<td>Figure 3.5: Combined Hex-Map and prism mesh</td>
<td>21</td>
</tr>
<tr>
<td>Figure 4.1: 3D view of impeller</td>
<td>25</td>
</tr>
<tr>
<td>Figure 4.2: Meridional section of blade</td>
<td>26</td>
</tr>
<tr>
<td>Figure 4.3: Flow chart for determination of unknown circulation</td>
<td>28</td>
</tr>
<tr>
<td>Figure 4.4: Domain and edges used for meshing</td>
<td>29</td>
</tr>
<tr>
<td>Figure 4.5: Convergence of circulation</td>
<td>30</td>
</tr>
<tr>
<td>Figure 5.1: Flow-chart for inverse design</td>
<td>32</td>
</tr>
<tr>
<td>Figure 5.2: Cylindrical and meridional coordinates</td>
<td>35</td>
</tr>
<tr>
<td>Figure 5.3: Computational stencil for discretization of the wrap angle calculation equation</td>
<td>37</td>
</tr>
<tr>
<td>Figure 5.4: Geometry Generation in Gambit – Step 3</td>
<td>39</td>
</tr>
<tr>
<td>Figure 5.5: Geometry generation in Gambit – Step 4</td>
<td>39</td>
</tr>
<tr>
<td>Figure 5.6: Geometry generation in Gambit – Step 5</td>
<td>40</td>
</tr>
<tr>
<td>Figure 5.7: Hard linking of periodic faces</td>
<td>41</td>
</tr>
<tr>
<td>Figure 5.8: Grid generation in Gambit</td>
<td>42</td>
</tr>
</tbody>
</table>
Figure 5.9: Blade angle calculation program verification – comparison of analytical and numerical wrap angles .................................................................47

Figure 5.10: Blade angle calculation code verification – comparison of analytical and numerical blade angles ..................................................................................48

Figure 5.11: Blade angle calculation code verification – comparison of analytical and numerical blade shape ..................................................................................48

Figure 5.12: Initial geometry for inverse design ..................................................................................................................50

Figure 5.13: \( \Delta \theta \) versus non-dimensional radius ...........................................................................................................................................51

Figure 5.14: Convergence of wrap angle at the leading edge of the blade .................................................................52

Figure 5.15: Blade shape comparison for various iterations ........................................................................................................................................52

Figure 5.16: 3D view blade shape for various iterations ........................................................................................................................................53

Figure 5.17: Blade shape (y versus x) for various meshes ........................................................................................................................................54

Figure 5.18: Wrap angle versus m for various meshes ........................................................................................................................................55

Figure 6.1: Convergence of circulation generated by impeller ..................................................................................................................57

Figure 6.2: Swirl distribution generated on blade ........................................................................................................................................58

Figure 6.3: Vector plot at leading edge of the original blade ........................................................................................................59

Figure 6.4: Comparison of swirl on original blade and the swirl used for re-designing the blade 60

Figure 6.5: Convergence of wrap angle at leading edge ........................................................................................................................................61

Figure 6.6: Comparison of original and new blade coordinates ........................................................................................................62

Figure 6.7: Comparison of wrap angle for the two blades ........................................................................................................62

Figure 6.8: Velocity vectors at leading edge of blade ..................................................................................................................63
# List of Tables

<table>
<thead>
<tr>
<th>Title</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 3.1: Element types in Gambit</td>
<td>19</td>
</tr>
<tr>
<td>Table 3.2: Meshing schemes in Gambit</td>
<td>19</td>
</tr>
<tr>
<td>Table 3.3: Comparison of Quad-Map and Tet Mesh Quality</td>
<td>22</td>
</tr>
<tr>
<td>Table 3.4: Surface normal gradient schemes in OpenFOAM</td>
<td>23</td>
</tr>
<tr>
<td>Table 3.5: Gradient schemes in OpenFOAM</td>
<td>24</td>
</tr>
<tr>
<td>Table 4.1: Pump parameters of a two-dimensional blade by Visser, Brouwers and Jonker (1999)</td>
<td>26</td>
</tr>
<tr>
<td>Table 4.2: Edge mesh count for grid-independence study</td>
<td>29</td>
</tr>
<tr>
<td>Table 4.3: Comparison of Numerical and Analytical Values of Circulation</td>
<td>30</td>
</tr>
<tr>
<td>Table 5.1: Boundary conditions for inverse design of blade</td>
<td>50</td>
</tr>
<tr>
<td>Table 5.2: Mesh-independence study for inversely designed blade</td>
<td>54</td>
</tr>
<tr>
<td>Table 6.1: Parameters of pump selected for re-design</td>
<td>56</td>
</tr>
<tr>
<td>Table 6.2: Boundary conditions for potential flow analysis</td>
<td>57</td>
</tr>
</tbody>
</table>
## Nomenclature

### Latin Letters

- \( A_{out} \) Area at outlet (m\(^2\)/s) of impeller
- \( B \) Axial width of impeller
- \( C_F \) Flow coefficient
- \( C_H \) Head Coefficient
- \( d \) Impeller inner diameter (m)
- \( D \) Impeller outer diameter (m)
- \( D_T \) Constant in OpenFOAM
- \( g \) Gravitational acceleration (m/s\(^2\))
- \( H \) Head delivered by a pump (m)
- \( m, s \) Meridional coordinates
- \( \hat{n} \) Unit vector along normal to blade surface
- \( N_s \) Specific speed pump
- \( P \) Pressure (Pa)
- \( Q \) Volume flow rate (m\(^3\)/s)
- \( Q_d \) Design flow rate (m\(^3\)/s)
- \( Q_{EAS} \) Equi-angle skewness
- \( r \) Radius (m)
- \( t \) Torque (Nm)
- \( T \) Scalar variable in OpenFOAM
- \( u \) Blade speed (m/s)
\( v \) Absolute velocity of flow (m/s)
\( \bar{v}_\theta \) Average circumferential velocity from pressure side to suction side
\( v_r \) Radial component of absolute velocity (m/s)
\( v_t \) Tangential component of absolute velocity (m/s)
\( w \) Relative velocity (m/s)
\( w_r, w_\theta, w_z \) Relative velocity components along cylindrical coordinates (m/s)
\( z \) Distance along axial direction (m)
\( Z \) Number of blades

**Greek Letters**

\( \theta_{TE} \) Wrap angle at trailing edge
\( \theta \) Wrap angle (rad)
\( \rho \) Density (kg/m\(^3\))
\( \omega \) Rotational speed (rad/s)
\( \Gamma_g \) Gamma function
\( \Gamma \) Circulation (m\(^2\)/s)
\( \tau_\beta \) Correction factor for spiral blades
\( \tau_0 \) Correction factor for straight blades
\( \phi \) Velocity potential (m\(^2\)/s)
\( \beta \) Blade angle (radians)
1. Introduction

This thesis illustrates implementation of an inverse design method for design of centrifugal pump impeller blades, based on potential flow analysis. Hence, it is necessary to understand the potential flow theory as applied to centrifugal pumps. This chapter presents motivation for the present work, the theory of potential flow in centrifugal pumps, and the inverse design technique adopted.

1.1 Motivation for Present Work

The shape of a pump blade has a significant effect on the overall performance of the pump. Blade curvature affects the flow field in the blade passage, and has an influence on the head delivered, and on the hydraulic efficiency of the pump. If the flow entering the blade is not tangential to the blade at the leading edge of the blade, flow separation can occur. This flow separation has multiple disadvantages. The separated region is unstable, and can cause oscillations and vibrations in the pump, and damage the pump over time. The separated region increases the blockage to the flow, raises local flow velocities, and thus increases the losses. Also, the separated region has low pressure, and can cause cavitation inception. Because of such negative effects of inadequate design of a pump blade, it becomes necessary to accurately calculate and understand the flow field. Advanced techniques like Computational Fluid Dynamics (CFD) give elaborated details of the flow field, which can help in identifying and rectifying the problems. Thus, blades designed using CFD analysis will perform better than blades designed using thumb rules, since more physics is incorporated in the design method of the former design method.
In the conventional approach where CFD is used for designing a product, CFD analysis and design change iterations are done manually, till a favorable design is obtained. The design modifications are done based on the experience of the designer. A trial-and-error approach is adopted to achieve the final design. This process is time consuming because, for each design modification, the generation of CAD model and the grid has to be done manually. Instead, if the design process is driven by the flow physics itself, the need for experience of the designer is reduced to a large extent. It also eliminates the trial-and-error method of designing the blade. This is the basic philosophy of inverse design. Geometry modifications are driven by constraints determined from the flow-physics rather than by the experience and intuition of the designer. Thus, for designing a pump blade, which is one of the most critical parts in an impeller, the inverse design technique is advantageous.

The focus of the present work is on use of potential flow theory for designing centrifugal pump blades. For solving the potential flow field, the OpenFOAM 2.0.1 software has been used, and for grid generation, Gambit 2.4.6 has been used. To the author's best knowledge, such kind of work using OpenFOAM has not been carried out previously.

The OpenFOAM software was selected for discretizing the flow equations governing the flow field because it is an open-source CFD software, and uses a finite-volume method. Use of OpenFOAM is increasing because, it is available without a license fee, and the source code of the software is available to the user. Over the years of development of OpenFOAM, a number of CFD models have been added to it, thus enhancing its capabilities. Being an open-source code, the user has the ability to modify the code. This presents a huge advantage, especially when the equations to be solved and the associated boundary condition types are not available in the standard sets provided by other commercial codes. This provision of code modification
eliminates the need to develop a totally new code for an application. Being an open source software, the support and documentation available for OpenFOAM is not very good. The successful implementation of potential-flow-based inverse design technique in OpenFOAM will open the door for future implementation of Euler equation and Navier-Stoke's based inverse design methods.

1.2 Basic Pump Theory

Pumps are broadly classified in two types: positive displacement type and roto-dynamic type. In positive displacement pumps, fluid displacement is achieved by change in volume of the cavity containing the fluid. In roto-dynamic type pumps, momentum change occurs as the fluid passes through rotating blade passages called impellers. This increases the kinetic energy of the fluid. This kinetic energy is then converted to pressure by making the fluid pass through a stationary passage called the diffuser or volute. A brief discussion of the theory is presented in this section. For further details, reader is referred to the studies Güllich (2008), Tuzson (2000), and Kruyt (2009). The present work done is applicable for centrifugal pump blade design.

Centrifugal Pumps

Centrifugal pumps fall into the category of rotodynamic pumps. Figure 1.1 represents a typical schematic of a centrifugal pump.

![Figure 1.1: Schematic of a centrifugal pump after Tuzson (2000)](image-url)
As the impeller blades rotate, fluid is drawn in the axial direction through the eye of the impeller labeled as 1 in Fig 1.1. The fluid passes over the impeller blades, and gains momentum and pressure. The diffuser, or the expanding-area scroll, as shown in Figure 1.1, converts the kinetic energy into pressure by passing the fluid through a passage of increasing cross-sectional area, called the scroll.

**Basic Pump Terminologies**

This section describes the basic terminology used for describing the geometry and performance parameters of a centrifugal pump.

**Net Head**

The net head \( H \) is equal to the head supplied by the pump minus the losses. For steady-state conditions, Bernoulli’s equation can be applied between inlet and exit of the pump. Neglecting viscous effects, the net head can be written as:

\[
H = \left( \frac{p}{\rho g} + \frac{v^2}{2g} + z \right)_2 - \left( \frac{p}{\rho g} + \frac{v^2}{2g} + z \right)_1
\]

(1.1)

where subscripts 1 and 2 represent inlet and outlet of pump respectively.

**Flow coefficient** \( C_F \)

The flow coefficient \( C_F \) is the non-dimensional flow rate through the pump, and is given by:

\[
C_F = \frac{Q}{\omega D^3}
\]

(1.2)

where \( Q \) is the volume flow rate, \( \omega \) is the rotational speed, and \( D \) is outer diameter of the impeller.

**Head Coefficient** \( C_H \)

The head coefficient \( C_H \) is the dimensionless head delivered by the pump, and is given by:
\[ C_H = \frac{gH}{\omega^2 D^2} \]  
(1.3)

Where \( g \) is acceleration due to gravity, and \( H \) is head delivered by the pump.

**Specific speed**

Specific speed \( N_s \) is one of the most important non-dimensional parameters used for comparing pumps of different diameters. It involves speed, head and discharge only, and is given by:

\[ N_s = \frac{\omega Q^{1/2}}{(gH)^{3/4}} \]  
(1.4)

**Meridional Section**

Since the pump impeller is axi-symmetric, it becomes convenient to use the cylindrical coordinate system \((r, \theta, z)\) instead of the Cartesian coordinate \((x, y, z)\) system. The meridional section of a blade is the projection of the blade on a plane of constant \( \theta \). The flow field is often described in terms of velocities along the meridional direction. Figure 1.1 shows the plan view of a blade and its corresponding meridional section.

![Figure 1.1: Schematic of meridional section](image)
**Blade Angle**

Blade angle describes the shape of the blade. The $\theta$-coordinate (wrap angle) of the cylindrical coordinate system represents the angle at every point on the blade with respect to a reference line of constant $\theta$. The blade angle ($\beta$), at a given point on the blade, represents the angle with respect to the meridional direction (for a 3D geometry) or the radial direction (for a 2D geometry). Figure 1.2 illustrates the difference between the wrap angle and the blade angle.

![Blade Angle Diagram](image)

**Figure 1.2: Wrap angle and blade angle**

**Velocity Triangles**

![Velocity Triangles Diagram](image)

**Figure 1.3: Velocity triangles**
Error! Reference source not found. shows the velocity triangles at the pump inlet and outlet. Here, \( u \) is the blade linear speed, \( w \) is the velocity of the fluid relative to the blade, and \( v \) is the absolute velocity of the flow. Subscripts 1 and 2 represent inlet and outlet conditions, respectively.

Applying the principle of angular momentum, the torque \( \tau \) applied is given by:

\[
t = \rho Q (r_2 v_{t2} - r_1 v_{t1})
\]

(1.5)

where \( \rho \) is the density of the fluid, \( Q \) is the volume flow rate, \( r_1 \) and \( r_2 \) are the inlet and outlet radii of the impeller, respectively, and \( v_{t1} \) and \( v_{t2} \) are the tangential components of the absolute velocity at the impeller inlet and outlet, respectively.

From Eq. 1.5, the head delivered by the pump can be derived. The resulting expression for the head delivered is given by:

\[
H = \frac{1}{g} (u_2 v_{t2} - u_1 v_{t1})
\]

(1.6)

The terminologies explained in this section define the basic input and output parameters involved in design of pump. In the inverse design method, the pump head and the meriodional section are the primary inputs, while the blade angle is the output. These input and output parameters can be expressed in various other forms, and used to fix design criteria. The following section summarizes the work done in potential flow analysis and inverse design of centrifugal pumps.
2. Literature Survey

This section reviews the work done in potential flow analysis of turbomachines, and the inverse design techniques used for designing blades of turbomachines.

2.1 Potential Flow Analysis of Turbomachines

Full three-dimensional Navier-Stokes solution of flow in passages of rotating machines has gained momentum due to improvements in turbulence modeling, and enhancement of computational power. However, considerable simplification of the core flow field in the blade passage of a rotating machine can be achieved, without losing overall validity, by assuming incompressible, inviscid, irrotational flow because of high values of the prevailing flow Reynolds number. Also, the potential flow analysis does not require the zero-slip boundary condition on the blades, and this fact can be used to inversely design the blades, as explained in Section 4.

Vast literature exists for potential flow analysis of flow through blade-to-blade passages of rotating machines. For two-dimensional cases, analytical expressions have been formulated. A method of conformal mapping and superposition of flows was used for obtaining solutions to the potential flow equation. Acosta (1952) implemented this technique to obtain solutions for straight and logarithmic spiral blades. Expressions for flow variables were obtained in integral form, and the evaluation of the integrals was done using numerical techniques. Solutions in closed form were also obtained by Visser (1994) for straight and logarithmic spiral blades. Closed-form solutions were obtained for parameters important for evaluating the performance of
radial centrifugal pumps. Visser et al., (1999) compared the results from this study with the experimental results for flow inside a blade-to-blade passage of a low specific speed pump with logarithmic spiral blades. The analytical results were found to be in good agreement with the experimental results for the core of the blade passage flow.

In potential flow analysis of a pump blade-to-blade passage, part of the challenge is determining the value of the circulation generated by the impeller. This unknown circulation appears in the boundary condition while carrying out a numerical analysis. The unknown circulation is calculated by satisfying the Kutta condition. Details for the Kutta condition application have been documented by Crighton (1985). Visser (1994) formulated analytical expressions to compute the circulation for a two-dimensional geometry. For a three-dimensional geometry, several approaches have been developed. Maiti et al., (1989) have described a technique to satisfy the Kutta condition at the trailing edge by using superposition of elementary flows to obtain the complete solution for the Laplace’s equation for the velocity potential. This method was implemented using a Finite Element Analysis (FEA). A FEA analysis was carried out, and the results were compared with experimental values. Good agreement was observed between the experimental and FEA results for the range of flow rates and pump speeds investigated. Kruyt et al., (1999) developed a superelement-based method, which was used to simulate the three dimensional unsteady flow involving rotor-stator interactions also. In this method, the Kutta condition was implicitly satisfied, eliminating the need for any special treatment. In the following section, the work done in field of inverse design of turbo-machines is summarized.
2.2 Inverse Design Methods

Inverse design techniques have been implemented vastly for designing airfoils. The earliest work can be traced back to that of Lighthill (1945) who formulated a technique based on conformal mapping to design airfoils with desired velocity distribution. Since then, inverse designing of airfoils gained momentum, and panel-method based codes like XFOIL (1986) and XFLR5 (2003) have been extensively used for designing airfoils and wings. The panel methods assume inviscid, irrotational flow. Several methods have been further developed which account for the viscous effects. Dulikravich (1990, 1992, 1995) has outlined and compared these methods. Dulikravich and Baker (1999) formulated a new mathematical model in which aerodynamic surfaces were treated as elastic membranes, and the redesign was carried out based on a prescribed surface pressure distribution. The method was formulated such that it could be used with any available flow field solver, and was applied for both two-dimensional and three-dimensional shapes.

Hawthorne et al., (1984) first designed a three-dimensional turbomachine using a prescribed mean swirl distribution for annular cascades. The flow field was assumed as inviscid and incompressible, and Biot-Savart’s law was used to calculate the flow field. The method was extended by Borges (1990) to radial turbomachines for arbitrary meridional geometry. Borges applied this method to redesign a low-speed radial inflow turbine, and experimentally demonstrated the improvement in efficiency from the original design. The same method was applied by Borges (1993) to design a mixed flow pump. Zangeneh (1991) further extended this method for compressible flows. A derivative of mean-swirl was used as a design parameter. Zangeneh (1996, 2004), Zangeneh et al., (1998) applied the method for suppressing secondary flows for a mixed-flow impellers, a compressor vaned diffuser, and a compressor stage. Goto et
al., (2002) further extended the practicality of this method by integrating together 3D CAD modeling, automatic grid generation, CFD analysis and inverse design, and applied the method for designing an impeller, a diffuser, and a volute casing. Along the same lines, Westra et al., (2005) developed an inverse design technique using a Finite-Element Method (FEM). The potential flow solution in the blade passage is obtained using FEM, and the blade shape is updated by satisfying the impenetrability condition. The method was applied to re-design radial and mixed flow impellers. The current work is based on the work done by Westra (2008). The method was chosen because it is straightforward in nature, and a detailed analysis is described in the thesis.

Demeulenaere and Braembussche (1998) used an Euler solver to calculate the flow field. An inversely designed shape was obtained for a targeted pressure distribution over the blade. The method was applied to redesign a transonic compressor rotor blade, and a low aspect ratio turbine blade. The same approach was used by Veress and Braembussche (2004) for designing a return channel for a multistage centrifugal compressor and by Vito et al., (2003) to design turbine blades for transonic and supersonic flows.

The method developed by Zangeneh solves an inviscid flow field and calculates the blade shape by satisfying the flow-tangency condition. Viscous effects are accounted for, by introducing blockage ratio while solving the flow field. Daneshkhah (2007) developed a viscous inverse design method for turbomachines in which Reynolds-Averaged Navier Stokes equations are solved to calculate the flow field. A virtual velocity is then computed based on the desired and actual pressure distribution over the blade. This virtual velocity drives the change of the blade shape. In this approach, the unsteadiness of the flow has been taken into account. Prescribed parameters for the inverse design included blade loading and thickness distribution. Daneshkhah
and Ghaly (2009) successfully applied the method for redesigning a subsonic turbine and a transonic compressor blade. Ramamurthy et al., (2010) used the same method for inverse design of an airfoil subjected to external flow conditions.

2.3 Objectives of Current Work

The work carried out till now in the field of inverse design has been done by developing codes. The capabilities of available commercial codes have not been utilized for inverse design problems. The current work focuses on inverse design of a two-dimensional pump blade, and implementing the inverse design procedure in OpenFOAM. Following are the specific objectives of this work:

1. Calculate circulation generated by a centrifugal pump impeller using OpenFOAM.
2. Modify the OpenFOAM code to determine potential flow and accommodate boundary conditions required for implementing inverse design methodology.
3. Automate geometry creation and mesh generation in Gambit for both two-dimensional and three-dimensional blade geometries.
4. Automate the iterative process of mesh generation, setting boundary conditions, solving for potential flow, and calculating new blade shape using a Linux shell script.
5. Verify inverse design methodology implementation, and apply it to improve pump performance.

The next chapter explains the flow domain, governing equation, boundary conditions, and the grid generation strategy adopted for solving the potential flow in a blade-to-blade passage.
Chapter 3

3. Mathematical Formulation

The present analysis is based on the assumption of an isolated impeller, in which the effect of the pump volute has not been considered. This assumption is justified when the pump has a well-designed vaneless diffuser, and operating at Best Efficiency Point as explained by Westra (2008). Also, the thickness of the blades has been neglected, and the blades are assumed to be thin. This section explains the axi-symmetric flow domain over which the potential flow equation is solved, the boundary conditions, grid generation using Gambit, and the discretization schemes used in OpenFOAM.

3.1 Flow Domain

The domains for a two-dimensional analysis and a three-dimensional analysis are as shown in Figure 3.1 and Figure 3.2 respectively. In the two-dimensional case, there is no variation of $r$ and $\theta$ coordinates. The domain is an isolated blade-to-blade channel. Blades are assumed to be of zero thickness. Also, the impeller is assumed to be detached from the stationary volute, so the effect of volute or diffuser on the flow is not considered.
3.2 Governing Equation for potential flow and its Solution in OpenFOAM

Since a potential flow is solved over the domain, the Laplace equation for the velocity potential $\phi$ has to be satisfied. Hence, the equation to be solved is:
In the standard version of OpenFOAM, a solver, named laplacianFoam is available, which solves the following equation:

$$\nabla^2 \phi = 0 \quad (3.1)$$

where $D_T$ is a constant, and $T$ is a scalar variable. Equation (3.2) simulates unsteady flow. Since the equation to be solved in the present analysis is a steady state equation, the solver has been modified to remove the time dependence. The modified solver solves the following equation:

$$\nabla^2 T = 0 \quad (3.3)$$

where $T$ is a scalar variable, which will correspond to the velocity potential $\phi$. The original code in OpenFOAM is:

```cpp
fvm::ddt(T)+fvm::laplacian(DT,T) \quad (3.4)
```

The code has been modified to:

```cpp
fvm::laplacian(T) \quad (3.5)
```

The modified solver is used to solve potential flow over the blade-to-blade passage, subject to the boundary conditions explained in the following section.

### 3.3 Boundary Conditions

At the inlet of the blade-to-blade channel, for a swirl-free flow, it is necessary to have a constant value of the velocity potential over the inlet surface. Hence, the boundary condition at inlet is taken as:

$$\nabla^2 \phi = 0 \quad (3.1)$$
\( \varphi = 0 \quad (3.6) \)

Since the blade-to-blade section of the impeller is considered for the analysis, the blades, and the boundaries upstream and downstream of the leading and trailing edges of the blade are rotationally periodic in nature. In terms of velocity potential, this boundary condition can be written as:

\[ \varphi_{ps} - \varphi_{ss} = C \quad (3.7) \]

where \( ps \) and \( ss \) represent the pressure side and the suction side, respectively. Upstream of the blade, the flow is without any swirl. So \( C = 0 \) for the upstream section labeled inletPeriodic0 (pressure side) and inletPeriodic1 (suction side) in Fig. 3.1 and Fig. 3.2. Hence downstream of the trailing edge,

\[ \varphi_{ps} - \varphi_{ss} = 0 \quad (3.8) \]

Downstream of the trailing edge of the blade, the swirl will be equal to the swirl corresponding to the circulation generated by the impeller. Hence,

\[ \varphi_{ps} - \varphi_{ss} = \Gamma \quad (3.9) \]

On the impeller blades, a free-slip boundary condition exists since the blades are solid surfaces, and the flow normal to the blade surface is zero in the inertial frame of reference. The corresponding relation between the absolute velocity, relative velocity and the blade speed is given by:

\[ \vec{V} = \vec{W} + \vec{\omega} \times \vec{r} \quad (3.10) \]

In the direction normal to the blade, Eq. (3.10) has the component:

\[ V_n = W_n + (\vec{\omega} \times \vec{r}) . \hat{n} \quad (3.11) \]
Since the blades are solid surfaces, no flow will pass through them. Hence, on the blade surface, the relative velocity in the normal direction will be zero, i.e., $V_n = 0$. Therefore, $V_n = (\omega \times \hat{r}).\hat{n}$ on the blade surface. In terms of velocity potential, this condition can be written as:

$$\frac{\partial \phi}{\partial n} = (\omega \times \hat{r}).\hat{n}$$  \hspace{1cm} (3.12)

This boundary condition is not available in the standard set of boundary conditions provided in OpenFOAM. To implement this boundary condition, a utility known as swak4Foam has been used. swak4FOAM (2011) is a library for OpenFOAM to facilitate easy implementation of complex boundary conditions. The boundary condition can then be written in ‘T’ file located in the ‘0’ folder as follows:

```plaintext
blade_half0
{
    type            groovyBC;
    fractionExpression  "  0 " ;
    gradientExpression "(vector(0,0,\omega)^pos()&normal())" ;
    //value           uniform 0;
}
```

where $\omega$ is the rotational speed.

At the outlet, the flow is assumed to be uniform in the direction normal to the outlet face. Hence at the outlet:

$$V_n = \frac{\partial \phi}{\partial n} = \frac{Q}{A_{out}B}$$  \hspace{1cm} (3.13)

The solver explained in Section 3.2 is solved over the blade-to-blade domain subject to boundary conditions mentioned above. The next section explains the steps to generate mesh for solving the governing equation.
3.4 Mesh Considerations

For the two-dimensional case, there is no variation in axial direction. The axial coordinates of all the points on the hub are the same. Thus, the hub surfaces (hub 0, hub1, and hub 2), shown in Figure 3.2, can be considered as a single surface for the purpose of meshing. The same applies for the shroud also, for the two-dimensional case. Hence, the hub can be meshed, and the mesh can be extruded from hub to shroud.

For the three-dimensional geometry, the flow enters axially, and exits in the radial or a mixed axial-radial direction. As a result, the axial coordinate varies along both hub and shroud. In this case, the extrusion of the mesh from hub to shroud cannot be done using Gambit, and so, the domain has to be decomposed to facilitate meshing.

So, to generalize the grid generation for both two-dimensional and three-dimensional geometries, the blade-to-blade passage has been decomposed into three domains. The domain decomposition is as shown in Figure 3.3.
Figure 3.3: Domain decomposition of blade-to-blade passage

Gambit provides various options for element types, and also provides the meshing schemes that can be used to generate surface and volume meshes. The description of the elements and schemes, as provided in the documentation of Gambit (2009) is given in Table 3.1 and Table 3.2.

<table>
<thead>
<tr>
<th>Table 3.1: Element types in Gambit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Element Type</strong></td>
</tr>
<tr>
<td>Quad</td>
</tr>
<tr>
<td>Tri</td>
</tr>
<tr>
<td>Quad/Tri</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 3.2: Meshing schemes in Gambit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Scheme Type</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Map</td>
<td>Creates a regular, structured grid of mesh elements</td>
</tr>
<tr>
<td>Submap</td>
<td>Divides an unmappable face into mappable regions and creates structured grids of mesh elements in each region</td>
</tr>
<tr>
<td>Pave</td>
<td>Creates an unstructured grid of mesh elements</td>
</tr>
<tr>
<td>Tri Primitive</td>
<td>Divides a three-sided face into three quadrilateral regions and creates a mapped mesh in each region</td>
</tr>
<tr>
<td>Wedge Primitive</td>
<td>Creates triangular elements at the tip of a wedge-shaped face and creates a radial mesh outward from the tip</td>
</tr>
</tbody>
</table>

While meshing the three domains shown in Figure 3.3, a face mesh is first generated on the hub, and this mesh is then extruded along the blade from the hub to the shroud. So the three dimensional-meshes consist of hexahedral or prismatic elements. The meshes have been compared for quality in terms of equi-angle skewness. While generating the meshes, the number of cells has been kept the same on the boundaries.

**Mesh 1:**

All the domains are meshed with Hex elements and Map scheme. It can be observed in Figure 3.4, in the domain 2, that the skewness of the cells is very high, rendering this mesh inappropriate in domain 2.
Mesh 2: Domain 2 is meshed with Tri elements using a Pave scheme. Domain 2 thus contains prism elements, while domains 1 and 2 contain structured hexahedral elements.

A quantitative comparison of the two meshes in domain 2 is given in Table 3.3. The quantity compared is the equiangle skewness in domain 2, since the mesh in domain 1 and domain two for the two cases is the same. Equiangle skewness is defined as given in Eq. 3.14.
\[ Q_{EAS} = \max \left\{ \frac{\theta_{\text{max}} - \theta_{eq}}{180 - \theta_{eq}}, \frac{\theta_{eq} - \theta_{\text{min}}}{\theta_{eq}} \right\} \]  

(3.14)

<table>
<thead>
<tr>
<th>Equisize skew range</th>
<th>% of total count</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Quad-map</td>
</tr>
<tr>
<td>0-0.1</td>
<td>0</td>
</tr>
<tr>
<td>0.1-0.2</td>
<td>0</td>
</tr>
<tr>
<td>0.2-0.3</td>
<td>0</td>
</tr>
<tr>
<td>0.3-0.4</td>
<td>0</td>
</tr>
<tr>
<td>0.4-0.5</td>
<td>0</td>
</tr>
<tr>
<td>0.5-0.6</td>
<td>0</td>
</tr>
<tr>
<td>0.6-0.7</td>
<td>3.08</td>
</tr>
<tr>
<td>0.7-0.8</td>
<td>53.38</td>
</tr>
<tr>
<td>0.8-0.9</td>
<td>42</td>
</tr>
<tr>
<td>0.9-1.0</td>
<td>1.54</td>
</tr>
<tr>
<td><strong>Total Count</strong></td>
<td><strong>2400</strong></td>
</tr>
</tbody>
</table>

In table 3.3, the Tet mesh has more cells near zero value of skewness indicating a better mesh. Thus, Tet-Pave meshes in domain 2 yield a better quality mesh than the Quad-Map mesh. So in the analysis carried out in the following sections, a mesh with tet elements is used in domain, while hex elements are used for the meshes in domains 1 and 3.

### 3.5 Solution Schemes in OpenFOAM

In OpenFOAM, the schemes available for solving the flow equations are specified in the folder `/0/system/fvSchemes/`. Since \( \nabla^2 \Phi \) is a Laplacian equation, schemes are specified for the laplacian operator. In OpenFOAM, three inputs are required for specifying the scheme for the laplacian solver.

1. Discretization scheme for the Laplacian operator
2. Scheme for interpolation (from cell center to face center) of \( D_T \), coefficient as explained in Eq. (3.2).
3. Scheme for surface-normal gradient

Since the solver has been modified, and the constant $D_T$ no longer exists in the equation, specification of any scheme for $D_T$ does not influence the solution.

In OpenFoam, a second-order Gaussian scheme is available for laplacian term. The Gaussian scheme is based on summing values on cell faces. The details are given in User’s Guide of OpenFOAM (2011).

For surface normal gradients, OpenFOAM provides the following schemes listed in Table 3.4. The details of the schemes are given in the Programmer’s Guide of OpenFOAM (2011).

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Corrected</td>
<td>Explicit non-orthogonal correction</td>
</tr>
<tr>
<td>Uncorrected</td>
<td>No non-orthogonal correction</td>
</tr>
<tr>
<td>Limited</td>
<td>A blend of corrected and uncorrected schemes</td>
</tr>
<tr>
<td>Bounded</td>
<td>Bounded correction for positive scalars</td>
</tr>
<tr>
<td>Fourth</td>
<td>Fourth order</td>
</tr>
</tbody>
</table>

In the present work, the corrected scheme has been used. The corrected scheme accounts for the non-orthogonality of the mesh. A non-orthogonality correction is applied to applied to cells with non-orthogonal faces based on the over-relaxed approach given in Jasak (1996). The non-orthogonality correction is done explicitly. Hence, it is necessary to specify the number of “non-orthogonality” correctors in the file /system/fvSolution. The number of these correctors has to be decided by observing the convergence of the solution. Higher the non-orthogonality, larger
the number of correctors required. In the current study, 30 non-orthogonality correctors have been used to achieve a convergence to residual level of 1e-11.

After a converged solution for $\varnothing$ is obtained, the velocities can be determined by calculating $\nabla \varnothing$. For calculation of the gradients, the schemes available in OpenFOAM are listed in Table 3.5.

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gauss &lt;interpolationScheme&gt;</td>
<td>Second order, Gaussian integration</td>
</tr>
<tr>
<td>leastSquares</td>
<td>Second order, least squares</td>
</tr>
<tr>
<td>fourth</td>
<td>Fourth order, least squares</td>
</tr>
<tr>
<td>cellLimited &lt;gradScheme&gt;</td>
<td>Cell limited version of one of the above schemes</td>
</tr>
<tr>
<td>faceLimited &lt;gradScheme&gt;</td>
<td>Face limited version of one of the above schemes</td>
</tr>
</tbody>
</table>

In the current work, a fourth-order scheme has been used since it has highest order of accuracy. The description of fourth order scheme is given in file –

“/src/finiteVolume/finiteVolume/gradSchemes/fourthGrad/fourthGrad.C”

In the fourth-order scheme for calculating gradients, first, the standard least-squares gradient is assembled. Then, the fourth-order correction is added to the second-order accurate gradient to complete the accuracy. Use of fourth-order schemes ensures high accuracy in calculation of velocities. The domain, mesh, boundary conditions and the discretization schemes used for potential flow analysis of the blade-to-blade passage of the impeller has been explained in this section. The next chapter describes the potential flow analysis carried out for a two-dimensional pump impeller.

While carrying out potential flow analysis of a pump blade-to-blade channel, the circulation generated by the impeller is an unknown. This unknown comes in the boundary condition downstream of the blade as described in Section 3.3. For a two-dimensional geometry, the circulation value can be analytically determined as explained by Visser (1994). Sections 4.1 to 4.3 describe potential flow analysis using OpenFOAM.

4.1 Determination of Circulation by Analytical Expression

A low specific speed pump used by Visser, Brouwers and Jonker (1999) has been considered in the present analysis. The blade geometry and the meridional section are shown in Figure 4.1 and Figure 4.2.

Figure 4.1: 3D view of impeller

[Image of impeller with labels: Hub, Impeller Blade]
The pump parameters are shown in Table 4.1.

**Table 4.1: Pump parameters of a two-dimensional blade by Visser, Brouwers and Jonker (1999)**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specific speed ((N_s))</td>
<td>0.4</td>
</tr>
<tr>
<td>Blade angle ((\beta))</td>
<td>(-70^\circ)</td>
</tr>
<tr>
<td>Impeller ID ((d))</td>
<td>0.32 m</td>
</tr>
<tr>
<td>Impeller OD ((D))</td>
<td>0.64 m</td>
</tr>
<tr>
<td>Axial width ((B))</td>
<td>25 mm</td>
</tr>
<tr>
<td>Design flow rate ((Q_d))</td>
<td>0.008 m(^3/s)</td>
</tr>
</tbody>
</table>

In Table 4.1, the specific speed \((N_s)\) is given by:

\[
N_s = \frac{\omega Q^{1/2}}{(gH)^{3/4}}
\]  

(4.1)

The blades are assumed to be of zero thickness. For equiangular blades, i.e., blades of constant blade angle, of zero thickness, the volume flow rate \((Q)\), blade angle \((\beta)\), leading-edge radius \((r)\), angular speed \((\omega)\), and the impeller width \((B)\) are governed by following equation (Visser (1994)):

\[
-\tan(\beta) = \tau_\beta(n, \beta) 2\pi r_{ie}^2 \omega \frac{B}{Q}
\]  

(4.2)

where \(\tau_\beta(n, \beta)\) is the correction factor for equiangular blades given by:

\[
\tau_\beta(n, \beta) = 1 + (\tau_0(n) - 1)\cos(\beta)
\]  

(4.3)

and \(\tau_0(n)\), the correction factor for straight blades, is given by:
\[ \tau_0(n) = 2^{4/n} \frac{\Gamma(1 - \frac{4}{n})}{\Gamma(1 - 2/n)^2} \]  

(4.4)

The correction factors are used to determine the flow rate for an incidence-free flow at the impeller inlet. For all other flow rates, the flow will not be tangential to the blade at the leading edge. For seven blades and constant blade angle of \(-70^\circ\), the volumetric flow rate is 0.008012 m\(^3\)/s, which is approximately the same as the design flow rate of 0.008 m\(^3\)/s. The corresponding velocity at the outlet is 0.127515 m/s.

For a two-dimensional case, the pump circulation and the head are related by:

\[ gH = \frac{Z\omega\Gamma}{2\pi} \]  

(4.5)

Using the specific speed value of 0.4 for calculating the head (H), the circulation can be obtained, as all other parameters in the Eq. (4.5) are known. The analytical circulation value for the given parameter values is thus 0.19675 m\(^2\)/s. The next section explains the numerical method for determining the circulation.

### 4.2 Numerical Determination of Unknown Circulation

Analytical solution of the circulation exists for two-dimensional geometry. For three-dimensional geometries, a circulation can be determined by numerical methods. Various methods have been proposed for calculation of the unknown circulation, as explained by Visser (1994) and Maiti, Seshadri and Malhotra (1989). In the present work, an iterative method has been adopted for determining the unknown circulation. Starting with an initial guess, the potential flow equation is solved over the flow domain as shown in Fig (3.1). Then, \(\Delta\phi\) at the trailing edge is calculated by calculating the difference in \(\phi\) between the pressure and suction sides. This \(\Delta\phi\) is then applied as a boundary condition on the periodic section (outletPeriodic0
and outletPeriodic1) downstream of the blade. The procedure can be summarized by the flowchart given in Fig 4.3.

1. Assume initial value of circulation. Apply it as a boundary condition on the boundary condition downstream of the blade.

2. Solve $\nabla^2 \phi = 0$ over the domain shown in Figure 3.1 subject to boundary conditions explained in section 3.3

3. Calculate circulation at the trailing edge

4. If the calculated circulation is not equal to the applied circulation, apply the calculated circulation as a boundary condition, and repeat the process from step 2.

**Figure 4.3: Flow chart for determination of unknown circulation**

The iterative process is continued till a converged value of circulation is obtained. Convergence, in this case, is said to be achieved when the absolute value of the difference in the circulation values between two successive iterations drops below $1e^{-05}$.

The converged value of the circulation has been calculated for three different grids to ensure grid independence of the values. The meshes used, and the results of the grid-independence study are explained in the next section.

### 4.3 Grid-Independence Study

Figure 4.4 shows the edges of the domain on which the grid distribution has been specified.

Table 4.2 shows the interval count on each edge, for all the meshes. The interval count on each edge has been chosen such that the approximate interval size on each edge is the same for all the edges.
Figure 4.4: Domain and edges used for meshing

Table 4.2: Edge mesh count for grid-independence study

<table>
<thead>
<tr>
<th>Edge Name</th>
<th>Number of Cells</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mesh 1</td>
</tr>
<tr>
<td>Inlet</td>
<td>14</td>
</tr>
<tr>
<td>InletPeriodic0</td>
<td>8</td>
</tr>
<tr>
<td>InletPeriodic1</td>
<td>8</td>
</tr>
<tr>
<td>Edge1</td>
<td>14</td>
</tr>
<tr>
<td>Blade0</td>
<td>47</td>
</tr>
<tr>
<td>Blade1</td>
<td>47</td>
</tr>
<tr>
<td>Edge2</td>
<td>28</td>
</tr>
<tr>
<td>OutletPeriodic0</td>
<td>8</td>
</tr>
<tr>
<td>OutletPeriodic1</td>
<td>8</td>
</tr>
<tr>
<td>Outlet</td>
<td>28</td>
</tr>
<tr>
<td>Approximate interval size on blades</td>
<td>0.01</td>
</tr>
</tbody>
</table>
Figure 4.5: Convergence of circulation

Table 4.3: Comparison of Numerical and Analytical Values of Circulation

<table>
<thead>
<tr>
<th>Approximate Interval Size on Blade (m)</th>
<th>Circulation Generated by Impeller (m²/s)</th>
<th>% Difference with respect to analytical value</th>
<th>% Difference with respect to finest mesh</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Numerical</td>
<td>Analytical</td>
<td></td>
</tr>
<tr>
<td>0.01</td>
<td>0.190027</td>
<td>-3.42</td>
<td>-1.70</td>
</tr>
<tr>
<td>0.005</td>
<td>0.190921</td>
<td>-2.96</td>
<td>-1.24</td>
</tr>
<tr>
<td>0.0025</td>
<td>0.192829</td>
<td>-1.99</td>
<td>-0.25</td>
</tr>
<tr>
<td>0.00125</td>
<td>0.193316</td>
<td>-1.75</td>
<td>0.00</td>
</tr>
</tbody>
</table>

The results for the grid-independence study are summarized in Table 4.3. From these results, it can be observed that as the interval count on the edges is increased, the numerical circulation value approaches the analytical value. Also, the percentage difference of the circulation for successive meshes reduces. For the finest grid, the numerical value is only -1.75% less than the analytical value. The next chapter describes the implementation of inverse design method in OpenFOAM and the verification.
5. Inverse Design Methodology Implementation in OpenFOAM and its Verification

5.1 Introduction

In this chapter, an inverse design method based on potential flow theory is presented. Implementation of this method is done in OpenFOAM using Gambit as the grid generation software. The method is then applied, and verified for a two-dimensional case.

5.2 Flow Domain and Boundary Conditions

The flow domain is as shown in Figure 3.1 and Figure 3.2. The boundary conditions are as mentioned in Chapter 3 except on the blade. In the potential flow analysis part, a zero normal velocity boundary condition in non-inertial frame of reference is applied. In case of inverse design, a mean swirl is imposed on the blade. The mean swirl distribution is related to the velocity potential as shown in the work of Westra (2008), and is given by Eq. (5.1).

\[
\Delta \phi (r,z) = \phi_{ss} - \phi_{ps} = \int_{ps}^{ss} \frac{\partial \theta}{\partial s} ds = \int_{\theta_{ps}}^{\theta_{ss}} v_{\theta} (m, \theta, s) r \, d\theta = \frac{2\pi}{Z} r \overline{v_{\theta}}(r,z) \tag{5.1}
\]

where \( \overline{v_{\theta}} \) is the average of circumferential velocity between the suction side and the pressure side, and \( \frac{\partial \theta}{\partial s} = v_{\theta} (m, \theta, s) \). So the specification of the mean swirl distribution on the blades translates to specifying the difference of velocity potential between the pressure and suction sides.
The specification of mean swirl on the blade does not guarantee zero normal velocity in the non-inertial frame on the blade surface. To ensure this, the blade shape is calculated such that the blade becomes tangential to the flow field. The flowchart in Fig 5.1 explains the method of the inverse design methodology:

1. Guess initial shape of blade

2. Solve \( \nabla^2 \phi = 0 \) over the domain shown in Figure 3.1 subject to mean swirl distribution on the blade. Conditions on the other boundaries are as explained in Section 3.3

3. Calculate velocities on the blades by computing \( \nabla \phi \) from the converged solution for \( \phi \)

4. Calculate new blade shape by making the blade tangential to the velocities calculated in step 3

5. Check for convergence of blade shape. If the blade shape is not converged, repeat from step 2 using the new shape of blade as the initial shape

**Figure 5.1: Flow-chart for inverse design**
5.3 Input Quantities for Inverse Design

The blade shape is determined for a fixed set of given input parameters. A change in any of these inputs alters the shape of the blade. Following are the necessary and sufficient input quantities required:

- Desired head delivered by the pump (H)
- Flow rate through impeller (Q)
- Number of blades (Z)
- Rotational speed of the impeller (ω)
- Meridional section of the blade (r, z coordinates of blade)
- Wrap angle at trailing edge (θ_{TE})
- Mean swirl distribution \( r\bar{\nu}_\theta(r, z) \)

5.4 Constraints on the Mean Swirl Distribution

The mean swirl distribution imposed on the blade has to satisfy the following conditions as described by Westra (2008):

1. The flow should enter the impeller passage with zero-swirl; hence,

   \[ (r\bar{\nu}_\theta)_{LE} = 0 \]  \hspace{1cm} (5.2)

2. The impeller is designed to deliver a desired head. This imposes a restriction on the value of mean swirl at the trailing edge.

   \[ (r\bar{\nu}_\theta)_{TE} = \frac{gH}{\omega} \]  \hspace{1cm} (5.3)
3. A zero-loading or incidence-free flow is desired at the leading edge. This condition is satisfied by the following constraint:

$$\left( \frac{\partial (r \vec{v}_\theta)}{\partial m} \right)_{LE} = 0$$  \hspace{1cm} (5.4)

where coordinate m is as explained in Section 5.5.

4. The Kutta condition at trailing edge is satisfied by following constraint:

$$\left( \frac{\partial (r \vec{v}_\theta)}{\partial m} \right)_{TE} = 0$$  \hspace{1cm} (5.5)

The thesis of Westra (2008) provides a detailed discussion of these constraints.

### 5.5 Implementation of inverse design methodology in OpenFOAM and Gambit

Potential flow analysis is carried out using OpenFOAM software, while the geometry creation and the grid generation are done using Gambit software. Starting with an initial blade shape, the final blade shape is obtained iteratively. For each iteration, it is necessary to generate the geometry, mesh the geometry, apply boundary conditions, solve for the potential flow, post-process the results, and calculate the new blade shape. Sections 5.6.1 to 5.6.6 explain how each step is implemented in OpenFOAM and Gambit.

### 5.6 Calculation of Wrap Angle

The velocities on the blade, determined from the computed potential flow over the domain, may not be tangential to the blade. A new blade shape is calculated such that the velocities become tangential to the blade. Mathematically, the condition to be satisfied by the new blade shape is given by:
\( \vec{w} \cdot \vec{n} = 0 \) \hspace{1cm} (5.6)

where \( \vec{w} \) is the relative velocity at a given point on the blade surface, and \( \vec{n} \) is the unit vector normal to the blade surface at that point.

For a three-dimensional geometry, the edges of the meridional section are not necessarily aligned along the \( r \) and \( z \) axes of the cylindrical coordinate system. In these cases, it becomes convenient to express the grid on the blade surface in terms of meridional coordinates, and solve the equations in these coordinates. The cylindrical coordinates and the meridional coordinates are shown in Error! Reference source not found.

Figure 5.2: Cylindrical and meridional coordinates

The curvilinear coordinates are not used to solve the potential flow equation since OpenFOAM is used, and OpenFOAM solves the equations in Cartesian coordinates only. For solving the Eq. (5.6), the meridional coordinates are used. In terms of the meridional coordinates, the Eq. (5.6) transforms to the following form:

\[
\frac{\partial \theta}{\partial m} = a(m, s) \frac{\partial \theta}{\partial s} + b(m, s) \hspace{1cm} (5.7)
\]

where,

\[
a(m, s) = \frac{w_r \frac{\partial z}{\partial m} - w_z \frac{\partial r}{\partial m}}{w_r \frac{\partial z}{\partial s} - w_z \frac{\partial r}{\partial s}} \hspace{1cm} (5.8)
\]
On the hub and shroud, the Eq. (5.7) reduces to the form:

\[ \frac{\partial \theta}{\partial m} = b(m, s) \]  \hspace{1cm} (5.10)

An initial condition is specified by a distribution of wrap angle on trailing edge; \( \theta(1, s) = \theta_{TE} \). Then the wrap angles on the hub and shroud are calculated using equation (5.10) by marching along m-direction. Using the values of wrap angle on the hub and shroud, the values of wrap angle on all the interior points are calculated using Eq. (5.7). Details of the derivation are described in the thesis of Westra (2008). Equation (5.10) is solved on the hub and shroud, while Eq. 5.7 is solved in the interior points on the blade meridional section.

Equation 5.7 is discretized using using the Crank-Nicholson scheme at the point \((i+1/2, j)\), second order accurate in both \(\Delta m\) and \(\Delta s\) as done by Westra (2008). The computational stencil is as shown in Fig 5.3. Along the s-direction, the equation is solved implicitly, while the solution is marched along m-direction from the trailing edge to leading edge. Accordingly, the discretized form of Eq. (5.7) is:

\[ \frac{\theta_{i+1,j} - \theta_{i,j}}{\Delta m} = \frac{a_{i+1,j}(\theta_{i+1,j+1} - \theta_{i+1,j-1}) + a_{i,j}(\theta_{i,j+1} - \theta_{i,j-1})}{4\Delta s} + \frac{b_{i+1,j} + b_{i,j}}{2} \]  \hspace{1cm} (5.11)

Equation 5.10 is discretized as:

\[ \frac{\theta_{i+1,j} - \theta_{i,j}}{\Delta m} = \frac{b_{i+1,j} + b_{i,j}}{2} \]  \hspace{1cm} (5.12)
A C++ program has been written for solving Eqs. (5.11) and (5.12). Input for this program is the velocity field calculated by OpenFOAM, while the output of the program is the wrap angle distribution on the blade surface.

**5.6.1 Geometry generation in Gambit**

Because of the curved and twisted (for 3D) shape of the blade, generation of the blade surface is not a straightforward task. To capture the complex shape, the following steps, illustrated in Figs have been adopted to generate the geometry:

Step 1: The blade surface is split into strips along the s-direction. The strips are then stacked to get the complete shape. Along the m-direction, a NURBS curve is fitted for the blade points.

Figure 5.4: illustrates this process:
Step 2: The process outlined in step 1 is repeated for the other periodic surface of the blade.

Figure 5.5 : Geometry generation in Gambit – Step 2

Step 3: The faces along $\theta$-coordinate axis are created at the inlet of the channel, trailing edge, leading edge, and the outlet of the channel.
Step 4: It was observed that Gambit failed to rotate the periodic surfaces to create a volume. So to create a volume, it was necessary to split the curved domain between the blades into a number of smaller parts. To facilitate this, five additional surfaces are created in between the two periodic blade surfaces as shown in Figure 5.5. Dividing the domain between the blades into 6 smaller parts has been found to accurately create the volume between the two blades.

Figure 5.4: Geometry Generation in Gambit – Step 3

Figure 5.5: Geometry generation in Gambit – Step 4
Step 5: Faces are constructed on the hub and shroud using the wireframe option in Gambit. Using all the generated faces, volumes are formed, as shown in Fig. 5.7.

Figure 5.6: Geometry generation in Gambit – Step 5

The periodic faces of the geometry are then linked with a command called ‘hard linking’ in Gambit.

5.6.2 Hard linking of periodic faces in Gambit

The blade faces, and the faces upstream and downstream of the blade are periodic. This implies that there should be a one-to-one correspondence between the meshes generated on these faces.

In Gambit, this achieved by Operation $\rightarrow$ Mesh $\rightarrow$ Face $\rightarrow$ Link $\rightarrow$ Face $\rightarrow$ Meshes.

The process is illustrated in Error! Reference source not found..
The next section describes the type of grid used for meshing the geometry.

### 5.6.3 Grid generation in inverse design procedure

The grid generation follows the same steps as explained in Section 3.4. Volumes 1 and 8, shown in Fig. 5.7, are meshed with hex elements, while volumes 2 through 7 are meshed with prism elements. Gambit uses the bottom-to-top approach for meshing. Accordingly, the edges are meshed first, then faces, and then volumes. The hub and shroud between the blade surfaces are meshed with triangular elements, and this mesh is then extruded along the blade surfaces to yield prism elements in volumes 2 through 7. This ability of Gambit to extrude elements along a surface, rather than along a particular direction or a curve, facilitates generation of a structured grid along the blade surfaces, but at the same time places triangular elements on the hub and shroud, which are surfaces of revolution. This facilitates assigning the swirl boundary condition.
on the center of the cell faces on the blade surface. The grid generated is shown in Fig. 5.9.

Figure 5.8: Grid generation in Gambit

The procedure summarized in sections 5.6.1 through 5.6.3 can be adopted to generate geometry and mesh in Gambit. This procedure is carried out for every iteration of the inverse design process. The next section describes automation of this process in Gambit using Journal file.

5.6.4 Journal file in Gambit

The process of geometry creation, grid generation and hard-linking of periodic faces, summarized in sections 5.6.1 through 5.6.4, is carried out for each iteration of inverse design. So, it becomes convenient to automate all these steps. The automation has been done using Journal files in Gambit. Journal files are text files that contain Gambit program commands given in documentation of Gambit (2009). For each new blade shape, the journal file imports the vertices of the blade in Gambit; builds edges, faces, volumes, then builds edge mesh, face mesh,
volume mesh; hard links the periodic faces; labels the boundaries, and finally exports the mesh. This mesh is then used by OpenFOAM to solve for the potential flow.

5.6.5 Solver and boundary conditions in OpenFOAM

As explained in Section 3.2, the original solver of OpenFOAM is modified to solve $\nabla^2 \phi = 0$. Another modification that has been done in OpenFOAM is for accommodating the swirl distribution and hence, the jump in the potential function $\phi$. In the standard version of OpenFOAM, a boundary condition called as “fan” is available. This fan boundary condition calculates the jump in $\phi$ for each face cell, as follows:

\[
\text{jump} = f_0 + f_1 * U_n + f_2 * U_n^2 + f_3 * U_n^3 + f_4 * U_n^4 + \ldots
\]

Where $f_i$ indicates the jump coefficients given as input by the user, and $U_n$ is the normal velocity (calculated by OpenFOAM), and pow is the power function.

The code for the above expression can be found in the folder


So in mathematical notation, the code for the jump can be written as:

\[
\text{jump} = f_0 + f_1 * U_n + f_2 * U_n^2 + f_3 * U_n^3 + f_4 * U_n^4 + \ldots
\]
The coefficients \( f_0, f_1, f_2 \ldots \) are specified by the user depending on the relation between jump and velocities, while \( U_n \) is calculated by OpenFOAM. For the current case, the code has been modified to take the jump value directly from the user, instead of calculating the jump using the local normal velocity component values, and the coefficients. Accordingly, the new code is:

```c
for (label i=1; i<f_.size(); i++)
{
    jump_[i] = f_[i];
}
```

In the input file (0/T), the jump is to be specified for each face on the boundary. For example, on a face with a 60*30 grid, 1800 values have to be supplied in the input file.

In this way, the swirl distribution, i.e., jump in the value of \( \phi \), can be specified by the user on the blades. The next section explains how to calculate velocities in OpenFOAM.

### 5.6.6 Calculation of velocities

After obtaining a converged solution for \( \phi \), the velocity field is obtained by calculating \( \nabla \phi \). The gradient vector \( \nabla \phi \) can be calculated in OpenFOAM by adding the command `fvc::grad(T)` to the code of the solver. OpenFOAM then automatically writes the velocity components after the solution is converged in Cartesian coordinates.

### 5.6.7 Iterations using Linux Shell Script

Since the blade design is carried out iteratively, it is necessary at every iteration to build the geometry, generate the mesh, apply the boundary conditions, solve the potential flow equation, and post-process the results. Automation of these tasks has been done by using a Shell Script in Linux. The shell scripts executes the journal file in Gambit, imports the mesh generated by
Gambit in OpenFOAM, runs OpenFOAM to solve for potential flow, and executes the C++ program for calculating the blade angles.

Sections 5.6.1 through 5.6.7 explain the implementation of the inverse design method in OpenFOAM. To check if the implementation is correct, it is necessary to verify any new code written, and any modification done in OpenFOAM. Section 5.7 summarizes the verification of C++ code written for calculating the blade angle, while Section 5.8 summarizes the verification of the implementation of inverse design process in OpenFOAM.

5.7 Verification of Blade Angle Calculation Code

The blade angle calculation code demonstrates the new blade angles, based on the velocities obtained after solving the potential flow equation to determine the velocity potential $\Phi$. Westra (2008) has given a procedure for a good verification case for this blade angle calculation code.

For the blade angle calculation code, the input required is the velocity field, while the output is the wrap angle distribution on the blade. To verify the code, the analytical solution for given input conditions is needed. One such flow for which the analytical solution for velocity is available is the potential flow solution for superimposed line source and a vortex flow. The analytical expression for the velocity potential is given by

$$\Phi(r, \theta) = \frac{q}{2\pi} \ln(r) + \frac{\Gamma}{2\pi} \theta \quad (5.13)$$

Where $q$ is the flow rate per unit width, and $\Gamma$ is the circulation. Relative velocities along the $r$ and $\theta$ directions can be calculated using the expressions:

$$w_r = \frac{\partial \Phi}{\partial r} = \frac{q}{2\pi r} \quad (5.14)$$

and
For a two-dimensional case, Eq. (5.10) reduces to the form:

\[ w_\theta = \frac{1}{r} \frac{\partial \phi}{\partial \theta} - \omega \frac{\Gamma}{2\pi r} = \omega r \]  

(5.15)

Substituting Eqs. (5.14) and (5.15) for the velocity in Eq. (5.16) and integrating, we get

\[ \frac{d\theta}{dr} = \frac{w_\theta}{rw_r} \]  

(5.16)

For given values of \( \theta_{TE} \), \( \Gamma \), \( q \), \( r_{TE} \), and \( \omega \), Eq. (5.17) provides an analytical expression for the blade shape as a function of \( r \).

To verify the blade shape calculation code, analytical values of velocities are used to generate the numerical shape of blade. The numerically generated \( \theta \) values for the blade are then compared with the analytical values.

The values of the parameters chosen for calculating the analytical values of velocity and blade shape are as follows:

\[ \Gamma = 10 \frac{m^2}{s}; \quad Q = \frac{m^2}{s}; \quad \omega = 10 \frac{rad}{s}; \quad r_{TE} = 0.5m; \quad r_{LE} = 0.25 \text{ m}; \quad \theta_{TE} = 0 \text{ rad} \]

Using these values, analytical values of the velocity components are calculated using Eqs. (5.14) and (5.15), and the present code was then used to calculate blade angle and the blade wrap angle for 50 intervals along the blade. Figure 5.9 shows the comparison of numerical values of the wrap angle with the analytical values.
Figure 5.9: Blade angle calculation program verification – comparison of analytical and numerical wrap angles

Figure 5.11 compares the blade angle calculated using the present C++ code with the analytical values. Figure 5.11 compares the numerical shape of the blade with the analytical values. The maximum percentage difference between the numerical and analytical values is less than 0.5. Note that the blade is hypothetical, and so the shape does not resemble a realistic shape. Even though the verification is not done for a realistic shape, the hypothetical shape has large curvature, and so this hypothetical case will be a more stringent case for verification, than the case with a realistic blade shape.
Figure 5.10: Blade angle calculation code verification – comparison of analytical and numerical blade angles

Figure 5.11: Blade angle calculation code verification – comparison of analytical and numerical blade shape
The maximum percentage error reported for all the plots above is below 0.5% of the analytical value. This accurate reproduction of angles verifies that the code written is correct. This section described the verification of the wrap angle calculation code. The next section describes the verification of the whole implemented inverse design procedure.

5.8 Verification of Inverse Design Procedure for 2D Spiral Blade

The inverse design procedure has been applied for a two-dimensional spiral blade. The pump parameters and the geometry are the same as used in the Section 4.

To verify the implementation of the inverse design method, the results from direct analysis have been used as inputs for an inverse design. Specifically, the swirl distribution and the circulation generated have been used from the potential flow analysis done in Section 4. Starting with a flat blade (zero blade angle), the swirl distribution from the direct analysis has been applied on the flat blade. The final shape of the inversely designed blade is then determined and this has been compared with the original shape. The details are described in next three sections.

5.8.1 Initial geometry and boundary conditions

The initial geometry is a flat blade with a constant blade angle of zero degrees throughout the blade. The initial geometry is as shown in Figure 5.12.
Figure 5.12: Initial geometry for inverse design

The boundary conditions are as shown in Table 5.1. Note that the boundary condition on the blade is different for the inverse design case compared to the one in chapter 4.

Table 5.1: Boundary conditions for inverse design of blade

<table>
<thead>
<tr>
<th>No.</th>
<th>Boundary Name</th>
<th>Boundary Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>inlet</td>
<td>( \varphi = 0 )</td>
</tr>
<tr>
<td>2</td>
<td>inletPeriodic0</td>
<td>( \varphi_{\text{inletPeriodic0}} = \varphi_{\text{inletPeriodic1}} )</td>
</tr>
<tr>
<td></td>
<td>inletPeriodic1</td>
<td>( \frac{\partial \varphi}{\partial n}<em>{\text{inletPeriodic0}} = - \frac{\partial \varphi}{\partial n}</em>{\text{inletPeriodic1}} )</td>
</tr>
<tr>
<td>4</td>
<td>blade0</td>
<td>( \varphi_{\text{blade0}} - \varphi_{\text{blade1}} = \Gamma(r,z) )</td>
</tr>
<tr>
<td></td>
<td>blade1</td>
<td>( \frac{\partial \varphi}{\partial n}<em>{\text{blade0}} = - \frac{\partial \varphi}{\partial n}</em>{\text{blade1}} )</td>
</tr>
<tr>
<td>7</td>
<td>outletPeriodic0</td>
<td>( \varphi_{\text{outletPeriodic0}} - \varphi_{\text{outletPeriodic1}} = \Gamma \text{ at trailing edge} )</td>
</tr>
<tr>
<td></td>
<td>outletPeriodic1</td>
<td>( \frac{\partial \varphi}{\partial n}<em>{\text{inletPeriodic0}} = - \frac{\partial \varphi}{\partial n}</em>{\text{inletPeriodic1}} )</td>
</tr>
<tr>
<td>8</td>
<td>outlet</td>
<td>( \frac{\partial \varphi}{\partial n} = 0.1275 ) (based on flow rate and flow area at outlet)</td>
</tr>
</tbody>
</table>
The circulation $\Gamma(r,z)$ on the blade is obtained from the direct analysis. For mesh 2 of the potential flow analysis explained in Section 4, the variation of the swirl distribution on the blade is as shown in Figure 5.13. The difference of velocity potential on the pressure and the suction side of the blade gives the variation of the swirl distribution.

![Graph of $\Delta\Phi$ versus non-dimensional radius](image)

**Figure 5.13: $\Delta\Phi$ versus non-dimensional radius**

### 5.8.2 Results

As the inverse design iterations are carried out, the blade shape changes from the initial flat shape to a converged curved blade. The convergence of the blade shape is determined by monitoring the value of the wrap angle at the leading-edge. Figure 5.14 shows the convergence of the leading edge wrap angle. It can be observed that the blade shape has nearly converged after 10 iterations approximately. Figure 5.15 and Figure 5.16 show the blade shape at selected iterations.
Figure 5.14: Convergence of wrap angle at the leading edge of the blade

Figure 5.15: Blade shape comparison for various iterations
Figure 5.15 shows that the blade shape remains unchanged after the 9th iteration, indicating that convergence has been nearly achieved by this iteration.

Figure 5.17 shows a three-dimensional view of the blade for selected iterations.

![Figure 5.16: 3D view blade shape for various iterations](image)

The final blade shape compares well with the original blade shape. The difference between the wrap angles of the original blade and the inverse designed blade is less than 1%. The analytical and the numerical values compare well for the mesh size used in this section. In the next section, a grid-independence study is done for various grid sizes.

### 5.8.3 Grid-Independence Study

Section 5.8.2 illustrated the application of the inverse design method implemented in OpenFOAM for a two-dimensional problem. In this section, the variation of the blade shape with grid refinement has been examined, and the results have been compared with the analytical blade shape. For the grid-independence study, the mesh used is the same as explained in Section 4.
Table 5.2 shows the variation of the wrap angle at the leading edge with grid refinement.

**Table 5.2: Mesh-independence study for inversely designed blade**

<table>
<thead>
<tr>
<th>Mesh</th>
<th>Wrap angle at leading edge (Degrees)</th>
<th>% difference with respect to analytical</th>
<th>% difference with respect to finest grid</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>109.31</td>
<td>0.18</td>
<td>0.16</td>
</tr>
<tr>
<td>2</td>
<td>109.26</td>
<td>0.13</td>
<td>0.11</td>
</tr>
<tr>
<td>3</td>
<td>109.19</td>
<td>0.07</td>
<td>0.05</td>
</tr>
<tr>
<td>4</td>
<td>109.14</td>
<td>0.02</td>
<td>0.00</td>
</tr>
</tbody>
</table>

For the coarsest mesh, the leading-edge wrap angle is within 0.2% of the analytical value. With grid refinement, the numerically determined value result approaches the analytical value within 0.2%. Figure 5.17 compares wrap angles for the four meshes. Figure 5.18 shows the comparison between the analytical and numerical blade shapes. It can be observed that the inversely designed numerical blade shape obtained using inverse design methodology compares well with the original blade, the difference between the analytical and numerical values less than 0.2%.

![Figure 5.17: Blade shape (y versus x) for various meshes](image)
Figure 5.18: Wrap angle versus m for various meshes

This section verifies the implementation of inverse design method in OpenFOAM. The next section uses this implementation to redesign and improve performance of a blade.
6. Pump Performance Improvement Using Inverse Design

The previous section described the verification of the inverse design method implemented in OpenFOAM. The swirl distribution imposed on the blade is obtained from the potential flow analysis. But, while re-designing a pump using the inverse design method, a swirl distribution has to be selected that satisfies the constraints explained in Section 5.4.

The need to re-design a pump often arises when a pump suffers from cavitation or when the losses are high. In such situations, the blade shape can be redesigned to reduce flow separation, and the associated losses. The following sections demonstrate how the inverse design method can be used to improve the flow in the pump blade-to-blade channel.

6.1 Pump Parameters

In this section, the inverse design technique implemented in OpenFOAM has been used to modify the blade geometry to improve the performance of a pump. The pump parameters are as shown in Table 6.1.

<table>
<thead>
<tr>
<th>Impeller OD: 0.32 m</th>
<th>Blade angle (): -70°</th>
<th>Impeller ID: 0.16 m</th>
</tr>
</thead>
<tbody>
<tr>
<td>Angular Velocity: 125.664 rad/s</td>
<td>Axial width (z): 25 mm</td>
<td>Flow rate (Qₚ): 0.2876 m³/s</td>
</tr>
</tbody>
</table>

First, a potential flow analysis is been done to predict the pump performance. Then, the inverse design method has been applied to redesign the blade, and improve the pump performance. The next two sections describe these two steps.
6.2 Potential-Flow Analysis

The domain used for the analysis is the same as shown in Figure 3.1. The boundary conditions are as shown in Table 6.2: Boundary conditions for potential flow analysis.

<table>
<thead>
<tr>
<th>Boundary Conditions</th>
<th>Equations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inlet</td>
<td>( \partial \phi \big/ \partial n = 0 )</td>
</tr>
<tr>
<td>Outlet</td>
<td>( \partial \phi \big/ \partial n = \frac{Q}{2 \pi r_{TE} W} = 4.578 \text{ m/s} )</td>
</tr>
<tr>
<td>Blade0, Blade1</td>
<td>( \partial \phi \big/ \partial n = (125.664 \hat{k} \times \hat{r}) \cdot \hat{n} )</td>
</tr>
<tr>
<td>inletPeriodic0, inletPeriodic1</td>
<td>cyclic, ( \phi_{inletPeriodic0} = \phi_{inletPeriodic1} )</td>
</tr>
<tr>
<td>outletPeriodic0, outletPeriodic1</td>
<td>cyclic, ( \phi_{outletPeriodic1} - \phi_{outletPeriodic0} = \Gamma )</td>
</tr>
</tbody>
</table>

Here, \( \Gamma \) is the unknown circulation generated by the impeller. This unknown circulation is determined iteratively as described in Section 4. The mesh used is the same as Mesh 3 described in Section 4. Figure 6.1 shows the convergence of the circulation with iterations. The value of the circulation converges after 80 iterations, when the residual levels fall below 1e-05.

![Figure 6.1: Convergence of circulation generated by impeller](image)

\( \Gamma \) versus Number of Iterations
The final converged value of circulation is 4.9039 m²/s. The swirl distribution on the blade is shown in Figure 6.2.

![Swirl Distribution (Γ) on Blade versus m](image)

**Figure 6.2: Swirl distribution generated on blade**

From Figure 6.2, it can be observed that the swirl distribution at the leading edge is not zero, and the gradient of the swirl distribution here is also not zero. This indicates that the flow at the inlet to the blade passage has non-zero incidence. The non-zero incidence causes flow separation and increases loss. The vector plot in Figure 6.3 also illustrates the non-zero incident flow at the leading edge of the blade. Flow separation is not desired in the blade-to-blade passage as it can lead cavitation and increased head loss. Hence, there is a need to redesign the blade, such that it delivers the desired blade, and at the same time is tangential to the flow at the leading edge. The next section describes the modification of the blade geometry to make the flow tangential to the blade at leading edge using the implemented inverse design procedure.
6.3 Redesigning of Blade

The blade is been redesigned by changing the swirl distribution over the blade. The new swirl distribution, which is the input for the inverse design method, is chosen to satisfy the four constraints described in Section 5.4. Equation 5.1 shows a polynomial expression, which satisfies the four constraints.

\[ \Delta \theta = \Gamma = (3m^2 - 2m^3) \Gamma_{TE} \]  

(5.1)

where, \( \Gamma_{TE} \) is calculated from the potential flow analysis in Section 5.1, its value being 4.9039 m²/s. The significance of \( \Gamma_{TE} \) is that it is directly related to the head delivered by the pump as follows:
Keeping $\Gamma_{TE}$ fixed while redesigning the blade ensures that the impeller with the new blade design delivers the same head as delivered by the original impeller.

Figure 6.4 shows the comparison of the swirl distribution on the original blade, and the swirl distribution used for redesigning the blade, given by Eq. 5.1.

**Figure 6.4: Comparison of swirl on original blade and the swirl used for re-designing the blade**

The swirl distribution is specified as a boundary condition on blade. Other boundary conditions are the same as in Section 5.2.

Figure 6.5 shows the convergence of the blade shape. The wrap angle at the leading edge has been monitored for assessing the convergence. When the difference in successive values of the
leading edge wrap angle falls below $10^{-5}$, the solution is said to be converged. A new converged shape of the blade is obtained in 12 iterations.

![Wrap Angle at Leading Edge versus Iteration number](image)

**Figure 6.5: Convergence of wrap angle at leading edge**
Figure 6.6: Comparison of original and new blade coordinates

The new blade is slightly shorter compared to the original blade. Figure 6.7 compares the original and the re-designed blade.

Figure 6.7: Comparison of wrap angle for the two blades
Figure 6.8: Velocity vectors at leading edge of blade

The resulting improvement in the flow direction at the leading edge can be observed in the relative velocity plot shown in Figure 6.8. The flow enters the new blade passage tangentially.

This section has demonstrated that, using the inverse design method implemented in OpenFOAM, the flow separation at leading edge can be avoided, and the losses can be reduced. Thus, the current work can be used to reshape the blade to improve pump performance. The next section summarizes the conclusions drawn from the current work, and also recommends work that can be done to enhance the capability of the current implementation.
7. Conclusions and recommendations for future work

7.1 Conclusions

OpenFOAM has been modified and used to determine potential flow in a two-dimensional blade-to-blade pump impeller passage. The circulation generated by the pump impeller is an unknown during the potential flow analysis. This unknown circulation forms a part of the boundary condition downstream of the blade. An iterative procedure for calculating this circulation accurately predicts the value of circulation generated by the impeller, when compared with the corresponding analytical value for a verification case.

Availability of the OpenFOAM source code made it possible to modify the solver and the boundary conditions. The OpenFOAM code has been successfully modified to enable incorporation of the swirl distribution boundary condition on the blade, and on the periodic boundaries downstream of the blade. The inverse design method has been successfully verified for a two-dimensional logarithmic spiral blade. A potential flow analysis of the blade-to-blade channel of a pump impeller was done. From this analysis, the swirl distribution generated on the blade was calculated. This swirl distribution was taken as a boundary condition for inverse designing the same blade. Starting with a flat blade, inverse design iterations were carried out, and the resulting converged blade shape was found to match closely with the analytical blade shape.

The inverse design process was also applied to redesign a pump blade. On performing potential flow analysis, the original pump blade showed a non-zero incidence at the leading edge of the blade. The blade was redesigned using a modified swirl distribution. A potential-flow analysis
of the new blade showed that the inlet flow was tangential to the leading edge of the new blade shape. Thus, a zero-incidence flow was obtained for the new shape.

Successful implementation in OpenFOAM has now opened the door for further development in the inverse design process.

7.2 Future Work

The current methodology is based on a potential-flow analysis. The work can be extended to Euler’s inviscid equation. Both these approaches do not account for viscous effects. The viscous effects can be included by introducing a blockage ratio, as outlined by Zangeneh (1991).

Extension of the current methodology to three-dimensional flows is also a vital need in order to make it more practical. In the current method, a no-slip boundary condition cannot be used on the blade. This limits the applicability of the method to potential or inviscid flows only. Alternately, the approach by Daneshkhah and Ghaly (2009) enables use of Navier-Stokes equation for calculating the flow field. This method can also be implemented in OpenFOAM, to get a more accurate prediction of the real flow in the pump impeller passage.

The appendix included in the thesis represents an attempt to consider three-dimensionality, although with some issues yet to be resolved.
8. References


[14] Gambit Documentation 2.4.6,2009


[34] XFLR5, 2003, "Analysis tool for airfoils, wings and planes operating at low Reynolds Numbers."

[35] XFOIL, 1986, "Interactive program for the design and analysis of subsonic isolated airfoils."


Appendix

Inverse design of 3-D impeller blades

Introduction

This appendix describes an attempt to carry out inverse design of a 3-D impeller. In the first part, potential flow analysis results are explained. Potential flow analysis of the original geometry shows that the flow at the inlet to the pump is not tangential to the blade loading edge, because the swirl distribution at the inlet does not have a zero gradient. This provides motivation to redesign the pump so as to achieve a zero derivative of swirl distribution at inlet of the impeller. The next section describes the configuration of the pump used for inverse design.

Pump Configuration

The pump configuration is given in work of Combes and Rieutord (1992) and Westra (2008). The meridional section of the pump blade is as shown in Fig. A.1. The geometry is constructed in Gambit from the given data. Figure A.2 shows the three dimensional view of the blade-to-blade channel.

Flow parameters

The pump flow parameters are as shown in Table A.1.

<table>
<thead>
<tr>
<th>Impeller ID (d)</th>
<th>0.22 m</th>
<th>Impeller OD (D)</th>
<th>0.4 m</th>
</tr>
</thead>
<tbody>
<tr>
<td>Axial width (B)</td>
<td>30 mm</td>
<td>Design flow rate (Q_d)</td>
<td>0.1118 m³/s</td>
</tr>
<tr>
<td>Angular Velocity</td>
<td>1200 rpm</td>
<td>Head (m)</td>
<td>31 m</td>
</tr>
<tr>
<td>Number of blades (Z)</td>
<td>7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table A.1: Boundary conditions for potential flow analysis of SHF impeller
Figure A.1: Meridional Section of 3-D pump impeller blade

Figure A.2: Three dimensional view of SHF impeller with sections upstream and downstream of blade
**Potential flow analysis**

An iterative method is used for calculating inviscid head. Laplace’s equation is solved for velocity potential. Boundary conditions are as shown in Table A.2.

**Table A.2: Boundary conditions for potential flow analysis of SHF impeller**

<table>
<thead>
<tr>
<th>No.</th>
<th>Boundary Name</th>
<th>Boundary Condition (OpenFOAM Notation)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>inletPeriodic0</td>
<td>$\phi_{ps} - \phi_{ss} = 0$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\left( \frac{\partial \phi}{\partial n} \right)<em>{PS} = - \left( \frac{\partial \phi}{\partial n} \right)</em>{ss}$</td>
</tr>
<tr>
<td>2</td>
<td>inletPeriodic1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>blade0</td>
<td>$\frac{\partial \phi}{\partial n} = (\vec{\omega} \times \vec{r}).\hat{n}$</td>
</tr>
<tr>
<td>4</td>
<td>blade1</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>outletPeriodic0</td>
<td>$\phi_{ps} - \phi_{ss} = \Gamma$;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\left( \frac{\partial \phi}{\partial n} \right)<em>{PS} = - \left( \frac{\partial \phi}{\partial n} \right)</em>{ss}$</td>
</tr>
<tr>
<td>6</td>
<td>outletPeriodic1</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>inlet</td>
<td>$\phi = 0$</td>
</tr>
<tr>
<td>8</td>
<td>outlet</td>
<td>$V_n = \frac{\partial \phi}{\partial n} = \frac{Q}{A_{outB}} = 1.977 \text{ m/s}$</td>
</tr>
</tbody>
</table>

**Schemes used in OpenFOAM**

Details of the schemes are as given in Section 3.5 of the thesis.

**Results of potential flow analysis**

The unknown swirl distribution is calculated by using the iterative procedure explained in Chapter 4 of the thesis. The swirl distribution is as shown in Fig A.3.
Figure A.3 shows the swirl distribution for the original design of SHF impeller. It can be observed that at the leading edge the derivative of the swirl distribution is non-zero. This implies that the flow is not entering the blade passage tangential to the blade.

**Inverse Design**

Next, the inverse design of SHF impeller is attempted. A swirl distribution, satisfying the four constraints explained in Section 5.4 of the thesis, is imposed on the blades as boundary condition. Figure A.4 illustrates the comparison of the original and modified swirl distributions on the blades.
Figure A.4: Swirl distribution – new and original

Initial Geometry

Initial geometry is assumed to be flat blade of zero thickness. The initial geometry is as shown in Fig. A.5. It can be observed that the blade has constant wrap angle of zero degrees.

Laplace’s equation for the velocity potential is solved over the domain shown in Fig. A.5. Boundary conditions are same as given in Table A.2, except on the blades. On the blades, the swirl distribution, as shown in Fig. A.4 is applied.
Difficulties encountered in the current inverse design of 3-D impeller blade

Velocities are calculated by OpenFOAM from the potential flow solution. The velocities are then used to calculate the wrap angles using Eq. (5.7) in the thesis. The wrap angles at the hub and the shroud are compared in Fig. A.6. The wrap angles calculated show a very high twist in the blade shape.
Figure A.6: Blade shape after first iteration

Figure A.7 shows the 3D view of the blade after the first iteration. It can be observed that there is a very high twist between the hub and the shroud, which causes divergence. This shape is obtained at the end of the first iteration of the inverse design. Under-relaxation also yields the same shape of the blade, but after a larger number of iterations. The number of iterations depends on the value of under-relaxation.

Figure A.7: Blade shape after first iteration
Possible causes for the high twist of the blade shape

In this section, the results after the potential flow analysis are presented, and their interpretation explaining possible causes of divergence of the blade shape is also explained.

1 Non-zero coefficient ‘a’ at hub and shroud

Recalling Eqs. 5.7-5.9 from the thesis, the blade angle is calculated from the velocities using the following expression:

\[
\frac{\partial \theta}{\partial m} = a(m, s) \frac{\partial \theta}{\partial s} + b(m, s) \quad (5.18)
\]

where,

\[
a(m, s) = \frac{w_r \frac{\partial z}{\partial m} - w_z \frac{\partial r}{\partial m}}{w_r \frac{\partial z}{\partial s} - w_z \frac{\partial r}{\partial s}} \quad (5.19)
\]

\[
b(m, s) = \frac{w_z \frac{\partial r}{\partial m} \frac{\partial z}{\partial s} - \frac{\partial z}{\partial m} \frac{\partial r}{\partial s}}{w_r \frac{\partial z}{\partial s} - w_z \frac{\partial r}{\partial s}} \quad (5.20)
\]

At the hub and the shroud, coefficient ‘a’ should be zero as explained by Westra (2008). But the values of ‘a’ calculated using velocities from OpenFOAM are non zero. Also, some oscillations are observed in the variation of ‘a’ in the interior.

2 High variation of coefficient ‘b’ between hub and shroud

At the hub and the shroud, with ‘a’ equal to zero, Eq. (5.8) reduces to:

\[
\frac{\partial \theta}{\partial m} = b(m, s) \quad (5.7)
\]

This equation shows that the gradient of the wrap angle along the meridional direction depends only on the coefficient ‘b’. A variation of this coefficient between hub and shroud i.e. along line
of constant ‘m’ causes a large difference in the values of wrap angle between the hub and the shroud.

![Coefficient 'b' along Hub and Shroud](image)

**Figure A.8: Coefficient ‘b’ on the hub and the shroud**

The cause of this large difference, and hence, the divergence in the shape of the blade, has not yet been established in the present study.

### 3 Extrapolation of velocities to the hub and shroud

Since OpenFOAM calculates the velocities at cell center, the velocities in the interior are extrapolated to the hub and shroud lines. This extrapolation is done in the computational domain. The process is explained in Fig. A.14. Lagrange’s polynomial is used for extrapolation. The polynomial is given by:

\[
S(\xi) = \sum_{n=1}^{N} L_n(\xi) S_n
\]

where \(S\) is the Lagrange polynomial which passes through the data known at \(N\) points, \(\xi\) is the curvilinear coordinate, and \(L_n(\xi)\) is a polynomial given by:
In the current calculation, two points are used to form Lagrange polynomial

\[ L_n(\xi) = \frac{[\prod_{l=1}^{n-1}(\xi - \xi_l)][\prod_{l=n+1}^{N}(\xi - \xi_l)]}{[\prod_{l=1}^{n-1}(\xi_n - \xi_l)][\prod_{l=n+1}^{N}(\xi_n - \xi_l)]} \]

In the current calculation, two points are used to form Lagrange polynomial.

As a first step in finding out the exact cause of the high twist of the blade, the C++ code is verified for a three dimensional case using Method of Manufactured Solution. The process is described in the next section.
Verification of blade angle calculation code using Method of Manufactured Solution

The C++ code for calculating wrap angle of the blade using Eqs. (5.7-5.9) is verified using Method of Manufactured Solution. In this method, an analytical solution is assumed for the variable such that the governing equation is satisfied by the analytical solution. The numerical values generated by the code are then compared with the analytical values.

Manufactured solution

Following is the manufactured solution for the wrap angle $\theta$:

$$\theta = m^2 + s^2 + \cos(m) \cdot \sin(s)$$

.................. (A)

Using this expression in Eq. (5.7),

$$\frac{\partial (m^2 + s^2 + \cos(m) \cdot \sin(s))}{\partial m} = a(m, s) \frac{\partial (m^2 + s^2 + \cos(m) \cdot \sin(s))}{\partial s} + b(m, s)$$

Simplifying, we get: $2m - \sin(m) \cdot \sin(s) = a(m, s) \cdot (2s + \cos(m) \cdot \cos(s)) + b(m, s)$

Simplifying, we get: $b(m, s) = 2m - \sin(m) \cdot \sin(s) - a(m, s)(2s + \cos(m) \cdot \cos(s))$

.................. (B)

The value of the coefficient $a(m, s)$ is chosen such that it satisfies the values at the boundaries.
At the trailing edge, \( m = 1 \), \( s \) varies from zero to one. So at trailing edge,

\[
\theta = \cos(1) \times \sin(s)
\]

At hub and shroud, \( a = 0 \). So \( a(m, 0) = 0 \), and \( a(m, 1) = 0 \). A function that satisfies these conditions is,

\[
a(m, s) = m \times \sin(s \times \pi)
\]

\[
\text{..........................(C)}
\]

Substituting Eq. (C) in Eq. (B), we get \( b(m, s) \) in terms of \( m \) and \( s \) only, as

The C++ code is verified using the above analytical expressions. The variables supplied to the code are \( m, s \). Theta is then calculated numerically, and the values are compared with the analytical values.
Comparison of numerical and analytical values

The numerical solution has been computed over 60*60 grid points. Results are presented for wrap angles on the hub, shroud, mid-span, leading edge, trailing edge, and in middle of leading and trailing edges in Figs A.9 to A.13.

Figure A.11: Wrap angle versus m on hub

Figure A.12: Wrap angle versus m on mid-span
Figure A.1: Wrap angle versus m on shroud

Figure A.13: Wrap angle versus m on shroud

Figure A.14: Wrap angle versus s on leading edge
The maximum percentage difference between the numerical and the analytical values is less than 0.01% for all the locations. This suggests that the C++ code is correct for a 3-D case.
Conclusions

The two coefficients ‘a’ and ‘b’ depend on the velocities calculated by OpenFOAM, and the blade angle calculation program. The blade angle calculation code has been verified for a two-dimensional case. For three dimensional case, the code has been verified using Method of Manufactured Solution. Oscillations have been observed in coefficient ‘a’, which depends on the velocities generated by OpenFOAM. Therefore, there is a need to verify the velocities generated by OpenFOAM for the potential flow solution and swirl distribution boundary condition. This verification case will lead to identifying the reason for the high twist in the blade for the 3-D case.