REDUCED-ORDER AEROELASTIC MODELING OF A TORSIONALLY COMPLIANT UAV ROTOR BLADE

by

Montana William Marks

A thesis submitted in partial fulfillment
of the requirements for the degree

of

Master of Science

in

Mechanical Engineering

MONTANA STATE UNIVERSITY
Bozeman, Montana

April 2021

ACKNOWLEDGEMENTS

I would like express my sincere appreciation for my master's committee, Dr. Mark Jankauski, Dr. Erick Johnson, and Dr. Michael Edens for their consistent guidance and direction throughout this project.

I would like to thank my colleagues in the Bio Inspired Dynamics Lab for their support.

Finally, I would like to thank my family for their overwhelming and unrelenting support and encouragement. You inspired me to grow and become the person I am today. Without you, this would not have been possible.

TABLE OF CONTENTS

1. INTRODUCTION 1

Novelty 4
Background 5
Fluid-Structure Interaction Modeling 5
Reduced-Order Modeling 6

2. AEROELASTIC MODELING OF A TORSIONALLY COMPLIANT ROTOR BLADE 9

Simplifications and Idealizations 10
Structural Dynamics and Derivation of the Equations of Motion 11
Reference Frame 11
Kinematics 11
Kinetic and Potential Energy 14
Lagrange's Equations 17
Quasi-Static Equilibrium 19
Aerodynamic Modeling 21
The Blade Element Model 21
Material Mechanics and Torsional Stiffness 25

3. NUMERICAL SIMULATION 29

Lift and Drag Coefficients For a Flat Plate 29
System Convergence and Computation Times 41
Blade Twist Optimization 43
Solving for the Quasi-Static Equilibrium Position 46
Comparison of Rigid and Flexible Blades 48

4. CONCLUSION 55

REFERENCES CITED 57

5. 64

LIST OF TABLES

Table Page

2.1 Comparison of Actual and Approximated Torsional Parameter and Percent Error. c_{Actual} is taken from [10]. 27

3.1 Mesh Statistics for 2D Flat Plate ANSYS Fluent CFD. 32

3.2 Edge Sizing Mesh Refinement Settings for 2D Flat Plate ANSYS Fluent CFD. 32

3.3 Inflation Layer Mesh Refinement Settings for 2D Flat Plate ANSYS Fluent CFD. 32

3.4 Face Sizing Mesh Refinement Settings for 2D Flat Plate ANSYS Fluent CFD. 33

3.5 Rotor Blade and Model Properties. 49

LIST OF FIGURES

Figure Page

1.1 Example of Fixed UAV Rotor [3] 3

1.2 Resultant Aerodynamic Force and Components for 2D Airfoil. [6] 8

2.1 Section of Rotor Blade Being Modeled 10

2.2 Coordinate Frame Transformations 12

2.3 Diagram of Span-wise Discretization of Rotor into Blade Elements and variable definitions 13

2.4 Diagram of Aerodynamic Forces and Vectors on i^{th} Blade 21

2.5 Rectangular Cross Section of Rotor Blade 26

3.1 Geometry from ANSYS Fluent CFD Simulation of 2D Flat Plate 31

3.2 Geometry Detail from ANSYS Fluent CFD Simulation of 2D Flat Plate. 31

3.3 Mesh from ANSYS Fluent CFD Simulation of 2D Flat Plate, Face Sizing Refinements Shown; Near Airfoil (Red) and Wake (Blue). 33

3.4 Mesh Detail from ANSYS Fluent CFD Simulation of 2D Flat Plate. 34

3.5 Mesh Boundary from ANSYS Fluent CFD Simulation of 2D Flat Plate. 35

3.6 Inlet and Outlet Boundary Condition Settings Fluent CFD Simulation of 2D Flat Plate. 35

3.7 Residuals from ANSYS Fluent CFD Simulation of 2D Flat Plate for $\alpha = 5^0$. 36

3.8 Normal and Axial Coefficients from ANSYS Fluent CFD Simulation of 2D Flat Plate for $\alpha = 5^0$. 36

3.9 Normal and Axial Coefficient Data Obtained from ANSYS Fluent CFD Simulation of 2D Flat Plate. 37

LIST OF FIGURES – CONTINUED

Figure Page

3.10 Comparison of Lift Coefficient Data Obtained from CFD Simulation additional data taken from [22], and [32] 38

3.11 Comparison of Drag Coefficient Data Obtained from CFD Simulation additional data taken from [22], and [32] 39

3.12 Lift to Drag Ratio Data Obtained from ANSYS Fluent CFD Simulation of 2D Flat Plate. 39

3.13 Quarter Chord Moment Coefficient Data Obtained from ANSYS Fluent CFD Simulation of 2D Flat Plate. 40

3.14 Comparison of Center of Pressure location for Theoretical[6] and Numerical Simulation..... 40

3.15 Convergence of Total Drag Force for Flexible Blade for Various Rotor Speeds 42

3.16 Convergence of Total Drag Force for Flexible Blade for Various Rotor Speeds 42

3.17 Computation Time vs. Number of Blade Elements for Rigid and Flexible Rotor Blades..... 43

3.18 Contour of Total Rotor Lift Vs. 2^{nd} Order Polynomial Coefficients..... 44

3.19 Contour of Total Rotor Drag Vs. 2^{nd} Order Polynomial Coefficients..... 45

3.20 Contour of Total Lift to Drag Ratio Vs. 2^{nd} Order Polynomial Coefficients..... 45

3.21 Optimized Blade Shape Based on Grid Search 46

3.22 Comparison of Initial and Final Blade Shapes for Rigid, Flexible Untuned, and Flexible Tuned Blades..... 50

3.23 Comparison of Aerodynamic Lift Force for Rigid, Flexible Untuned, and Flexible Tuned Blades..... 51

LIST OF FIGURES – CONTINUED

Figure Page

3.24 Comparison of Aerodynamic Drag Force for Rigid, Flexible Untuned, and Flexible Tuned Blades. 52

3.25 Target Shape and Flexible Tuned Blade Shapes at Various Angles of Attack. 52

3.26 Comparison of Mechanical Power Requirements for Rigid, Flexible, and Flexible Optimized Blades. 53

3.27 Comparison of Lift to Drag Ratio for Rigid, Flexible Untuned, and Flexible Tuned Blades. 53

3.28 Comparison of Lift to Power Ratio for Rigid, Flexible Untuned, and Flexible Tuned Blades. 54

ABSTRACT

Small-scale quadrotor helicopters, or quadcopters, have increased in popularity significantly in the past decade. These unmanned aerial vehicles (UAVs) have a wide range of applications - from aerial photography and cinematography to agriculture. Increasing flight time and payload capacity are of the utmost importance when designing these systems, and reducing vehicle weight is the simplest method for improving these performance metrics. However, lighter components and structures are often more flexible and may deform during operation. This is especially the case for flexible UAV blade rotor behavior during flight. Modeling rotor blade deformations is non-trivial due to the coupling between the structure and the surrounding flow, which is called Fluid-Structure Interaction (FSI). Several methods exist for FSI modeling where the most common involves integrating Finite Element and Computational Fluid Dynamics solvers. However, these higher-fidelity models are computationally expensive and are not ideal for parametric studies that consider variable rotor geometry, material properties or other physical characteristics.

This research develops low-order modeling techniques that can be leveraged by UAV rotor designers. Here, a reduced-order FSI model of a small-scale UAV rotor blade is developed using Lagrangian mechanics paired with a blade element model. The rotor blade is discretized into rectangular elements along the span. Each blade element is constrained to uni-axial rotation about the span-wise axis and is treated as a torsional stiffness element. The quasi-static equilibrium state of the structure due to aerodynamic forces at user-defined operational conditions is then determined. The model presented is capable of producing a converged solution in as little as 0.016 seconds, as opposed to higher-order FSI models, which can take up to several orders of magnitude longer to solve. It is determined that the deflection of a flexible blade can reduce the total aerodynamic lift from 18-25% when compared to a rigid blade with the same initial geometry. It is shown that the model allows a user to tailor the initial pre-twist of the flexible rotor blade such that losses in lift are reduced to 0.68-5.7%.

1

INTRODUCTION

In recent years, lightweight, unmanned aerial vehicles (UAVs) have become popular as an alternative to manned systems [38]. These UAVs have a wide range of applications, from aerial photography and videography to agriculture. For example, Duggal et al. presented a framework for using UAVs to monitor the growth and estimate yield in pomegranate crops. Anderson et al. created an autonomous UAV robot that is capable of performing odor localization in a confined space. As these systems become increasingly advanced, engineers will continue to look for ways to improve vehicle performance. One mechanism to increase vehicle performance is to reduce the overall weight of the aircraft. Reducing the mass of the vehicle has a significant effect on both flight time and payload capacity. The most efficient way to reduce mass is to reduce the amount of material used by vehicle components. However, removing material often results in a more compliant structure. This increased flexibility can complicate the design process as understanding how a compliant structure will perform may be non-trivial.

These difficulties are especially prevalent when modeling the interaction between a fluid and a structure, where the fluid flow causes the structure to deform and the structural deformation influences the surrounding flow field. This particular type of modeling is called fluid-structure interaction modeling (FSI). Traditional FSI modeling usually involves coupling a finite element model (FEM) to a computational fluid dynamics (CFD) solver. These conventional high order models are notoriously complex and computationally expensive to solve and this long computation time is not conducive to parameter studies [25]. For these reasons, compliant structures

are often overlooked by designers, despite the fact that flexibility may enhance performance. For example, Mountcastle and Combes showed that wing damage is mitigated in yellowjackets through the use of a costal break which allows for large wing deformations to occur during a collision. This concept of using deformation to mitigate collision damage has been applied to UAVs. Mintchev et al. created a UAV frame that is capable of withstanding loads within the flight envelope but softens and folds during collisions.

In order to reduce computational expenses, engineers have developed reduced-order models (ROMs). Reduced-order FSI models incorporate new and innovative ways to mathematically resolve structural and fluid forces. ROMs have a wide range of benefits, from aiding in parametric studies to control system design.

The goal of this thesis is to create a reduced order, two-way coupled fluid structure interaction model of a flexible UAV rotor blade (Fig.1.1). As the rotor blade moves through the air, aerodynamic forces cause the blade to deform, and predicting this deformed shape is difficult. This flexibility plays an important role in the aerodynamics of the rotor blade [11]. This model allows for various user inputs such as: pre-twist, rotor radius, chord length, material properties, spin speed, and thickness changes along span. With this information, the steady state torsionally deformed, or twisted blade shape, as well as the total lift, drag, and power required by the rotor blade are determined. With this, a designer can determine if a rotor design is sufficient in providing the needed performance metrics. Because of the reduced order nature of the model, all of this can be achieved in as little as a fraction of a second (as compared to the hours or even days that high fidelity model can take).

The hope is that this model will aid in the design of a high-performance, lightweight UAV rotor. Additionally, this model may also provide a platform for flexible blade parametric studies. This research also paves the way for more advanced

reduced-order FSI models that can incorporate additional axes of rotation and more advanced blade cross-sections and geometry. Advanced versions of this model may be applied to a wide range of engineering problems - from understanding insect flight to non-hovering UAV flight.

Figure 1.1: Example of Fixed UAV Rotor [3]

To reduce mathematical complexity and overall computational costs of this model, several idealizations and assumptions have been used. Understanding which physical phenomena were not accounted for or idealized in the creation of each reduced order model is imperative to proper application. Below is a list of idealizations and assumptions that were utilized during the creation of this reduced order model.

- Span-wise and chord-wise bending have been neglected. Consequently, the blade is considered to be fully rigid in bending in the span-wise and chord-wise directions.
- Span-wise fluid flow across rotor surfaces has been neglected.
- The aerodynamic model does not resolve a fluid field, and consequently neglects effects of transient phenomena such as vortex shedding, tip losses, and turbulent structures.
- The blade cross section is assumed to be a thin rectangle.
- The vehicle is assumed to be in a stationary hovering flight condition.

Novelty

Reduced order aeroelastic modeling of large scale rotor-craft blades has been explored in a wide range of previous works [12, 15, 19, 26, 39, 49]. However, rotor-craft blade dynamics differ significantly from UAV rotors in several areas. Large scale rotor-craft operate in a significantly different dynamic range than small scale UAVs. This difference has a large effect on the aerodynamics. Additionally, traditional rotor craft have large diameter rotor blades with a slender construction resulting in very large aspect ratios. This is in contrast to a small-scale quadcopter which has much smaller aspect ratios. Traditional large scale rotor-craft also employ a verity of complex linkage systems that allow a rotor blade to flap, lead-lag, and pitch. These additional degrees of freedom can contribute to more complex equations of motion for the rotor blade. Conversely, small scale quadcopters employ a fixed rotor configuration that eliminates complex linkages. For these reasons, developing a reduced order aeroelastic model specific to small scale UAV rotor blades is necessary. However, very little work has been done in the area of reduced-order FSI modeling of small-scale fixed-rotor aircraft. For example, Pounds and Mahony produced a reduced order aeroelastic model for small scale rotors that utilized BET paired with elastic deformation integrals to simulate aeroelasticity. However, this work differs significantly from the research presented here via the structural solver. Some work as been done model the aerodynamics of a small-scale fixed rotor UAVs [13, 34]. However, these aerodynamic models neglect the interaction of the fluid and the structure. For this reason, it is believed by the author that the work presented herein is novel.

Background

Fluid-Structure Interaction Modeling

FSI modeling has a wide range of applications; from modeling the ring sail parachutes used on the Orion spacecraft [45][46], to analyzing blood flow through the cardiovascular system [8]. This type of modeling enhances our understanding of the complex interaction between fluid and structures. FSI models are either one-way coupled, meaning one physical domain may inform the other but not vice versa; or two-way coupled, meaning both physical domains interact with each other. Several methods exist for FSI modeling, the most common of which involves coupling FEM and CFD solvers. One topic of particular interest is aeroelastic coupling and its effect on aerodynamic efficiency in rotor aircraft. This phenomenon is particularly important when understanding rotor aircraft flight, where a large extent of aeroelastic coupling is present due to the slender construction of the blades [40]. The highly unsteady loading caused by the combined effect of aerodynamic interactions, blade dynamics, and complex aerodynamic-structural coupling makes aeroelastic analysis in rotor aircraft incredibly difficult [41]. Difficulties in designing aircraft and modeling aerodynamic characteristics increase when length scales are in the Micro Air Vehicle (MAV) region and FSI models can aid in this area [35]. In small-scale rotor aircraft, tip displacement can be significant [42]. Additionally, blade twist can play an important role in efficiency of rotor aircraft; low twist is beneficial for hover while high twist is helpful for forward flight [28]. For this reason, designing a rotor aircraft to operate efficiently in both hover and forward flight presents a challenge, and passive blade twist control has been studied using FSI [28]. Rotor blade FSI is also used in other industries. For example, Miao et al. developed an FSI model to study adaptive wind turbine blade technology under extreme loading conditions [29].

Many FSI models, such as the one used by Sitaraman et al., utilize CFD and computational solid dynamics or FEM. CFD is a branch of fluid dynamics that leverages numerical methods to model fluids. With the advent of the modern computer, CFD has become the standard means of modeling complex fluid flow. All of CFD, in one form or another, is based on the fundamental governing equations of fluid dynamics: the continuity, momentum, and energy equations [5]. CFD utilizes a computational mesh that is generated by discretizing the fluid volume into individual cells. The Navier-Stokes equations are then used in conjunction with boundary conditions to resolve the physical state of the fluid at the mesh nodes. FEM is a computational technique used to obtain approximate solutions of boundary value problems in engineering [21], and is often applied to structural analysis. These conventional high-order models are notoriously complex and computationally expensive to solve, and this long computation time is not conducive to parameter studies [25]. In order to reduce computation time, some FSI models only incorporate a one-way coupling. That is, the pressure distribution over a solid surface is calculated using CFD and then applied as a time-dependent load condition to an FEA solver to obtain a deformation. In these models, the reverse influence of the deformation on the fluid is neglected [36]. Although this method reduces computation time, it does not account for the impact of the deformation on the fluid. In a two-way coupled model, the reverse influence of the deformation on the fluid is not neglected. In doing so, a two-way coupled model often provides a more realistic and accurate result, and thus, a two way coupling is ideal.

Reduced-Order Modeling

One way to reduce computational expenses is to move to a lower-order modeling scheme. ROMs can be applied to a wide range of computational physics problems.

However, to stay within the scope of this research, we will focus solely on the background of FSI ROMs. These types of models incorporate a variety of tactics to reduce the mathematical complexity of the model via the fluid solver, structural solver, or both. For example, Shahverdi et al. used a boundary element method (BEM) to predict the aerodynamic forces and Galerkin's method to solve for the structural equations of motion for a hovering helicopter. Kwon et al. studied the aeroelastic behavior of hingeless rotor blades in hover using BEM based on the panel method for three-dimensional aerodynamic computations [39]. Bhasin et al. used the unsteady vortex lattice method coupled with equations of motion to analyze the non-linear dynamics of a joined wing. Another common method for reducing complexity in aerodynamic models involving wings is based on Blade Element Theory (BET) [37].

BET is a method for predicting the aerodynamic forces and moments on a rotor due to its motion through a fluid. The origins of blade element theory can be traced to the work of William Froude in 1878, but the first major treatment was developed by Stefan Drzewiecki between 1892 and 1920; see Glauert (1935) [24]. Blade element theory is essentially lifting-line theory applied to a rotating wing. BET works by taking a rotating propeller and discretizing it into small blade elements along the span. Each blade section is assumed to act as a two-dimensional airfoil to produce aerodynamic forces, where the influence of the wake can be contained entirely in an induced angle-of-attack at the blade element [24]. In treating each blade element as a 2D airfoil, the basic equations for aerodynamic lift, drag, and moment for a 2D airfoil can be utilized (Fig.1.2). The forces are then summed for all the blade elements to resolve the total lift and drag for the 3D wing. This method allows for the determination of aerodynamic forces with relatively little computational effort when compared to high fidelity methods such as CFD. Other methods can be used as

well, Schilling et al. utilized an unsteady vortex lattice method to predict transient hydrodynamic forces on a submerged propeller [27].

Additionally, methods can be employed to reduce the complexity of the structural model, such as the one used by k. Schwab et al.. These reduced order methods, when validated by traditional CFD and FEA coupled solvers and experimentation, can provide a valuable tool when performing parametric studies by drastically reducing computation time.

However, these low-order models come with limitations. When using BET, the flow field surrounding the wing or rotor is not resolved. This is important, as the three-dimensional rotor wake surrounding rotor aircraft is unsteady and complex [47].

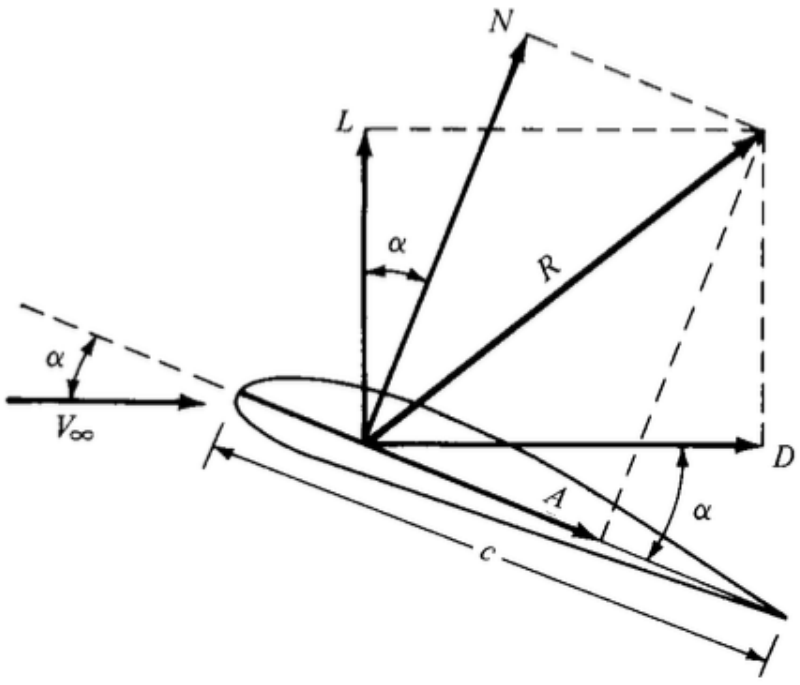

Figure 1.2: Resultant Aerodynamic Force and Components for 2D Airfoil. [6]

9

AEROELASTIC MODELING OF A TORSIONALLY COMPLIANT ROTOR BLADE

In this chapter, the two way coupled equations of motion will be derived in detail. These equations of motion form the basis for the ROM presented in this thesis. The ROM being presented is comprised of three main components, the structural model, the aerodynamic model, and the torsional stiffness model. The structural model utilizes Lagrangian mechanics to derive the non-linear equations of motion for each individual blade element. The aerodynamic model uses the blade element theory to derive the mathematical relations for the aerodynamic forces and moments applied to each blade element. The torsional model utilizes mathematical definitions to derive a relation between cross sectional geometry and torsional stiffness. The aerodynamic model is then coupled to the structural model using the principal of virtual work. When combining these three models, a set of n number of non-linear equations of motion are obtained that represent the system as a whole. These equations are then tailored to to solve for a quasi-static equilibrium state.

Simplifications and Idealizations

Before beginning the structural framework, the physical system must be simplified as much as possible. Given that UAV rotors are symmetrical about the primary rotating axis, we can reduce complexity and computation time by modeling a single blade of the rotor. This is visualized in Fig. 2.1 where the red outline represents the portion of the blade being modeled. This method is beneficial as it not only reduces computational expenses but it also allows for the analysis and application towards rotors with a wide range of blade configurations. Additionally, due to symmetry, any aerodynamic forces or power requirements determined by the model can easily be multiplied by the total number of blades on each rotor to obtain total rotor performance metrics. The cross-section of the rotor blade is also assumed to be rectangular and symmetrical about the center axis. This drastically simplifies both the aerodynamic model and the torsional stiffness relations. The UAV is also assumed to be in a stationary hovering condition. This assumption has an effect on both the rotating reference framework, as well as the aerodynamic modeling.

Figure 2.1: Section of Rotor Blade Being Modeled

Structural Dynamics and Derivation of the Equations of Motion

Reference Frame

The rotor blade is discretized into span wise elements called blade elements. Each blade element is assigned a body-fixed rotating coordinate frame whose origin is located at the center of rotation. A pseudo-body-fixed primary rotating reference frame rotating about the stationary space fixed z^0 axis is also defined. This pseudo-body-fixed primary rotating reference frame is called such because it represents the reference frame that is attached to the rotor blade as a whole. This scheme results in $n + 2$ total coordinate frames where n is the total number of blades and the two additional frames are the space fixed frame and the primary rotating frame. For simplicity, the space fixed frame will be represented with a null superscript (x^0, y^0, z^0). The primary pseudo-body-fixed rotating frame is represented with a one in the superscript (x^1, y^1, z^1) and is transformed by rotating about the stationary z^0 axis by angle γ. Finally, each body fixed reference frame is denoted as $(x^{i+1}, y^{i+1}, z^{i+1})$ where i is the blade index. The body fixed frame is transformed by rotating about the y^1 axis by angle β_i. By adopting this framework, mathematical complexity is reduced during derivation of the total kinetic energy, potential energy, and the angle of attack when compared to a framework that defines rotation with respect to the previous blade's coordinate frame.

Kinematics

The total angular displacement of each blade element β_i with respect to the primary rotating reference frame is equal to the sum of the pre-twist η_i and angular displacement θ_i of the blade. Therefore the total angular displacement for the i^{th}

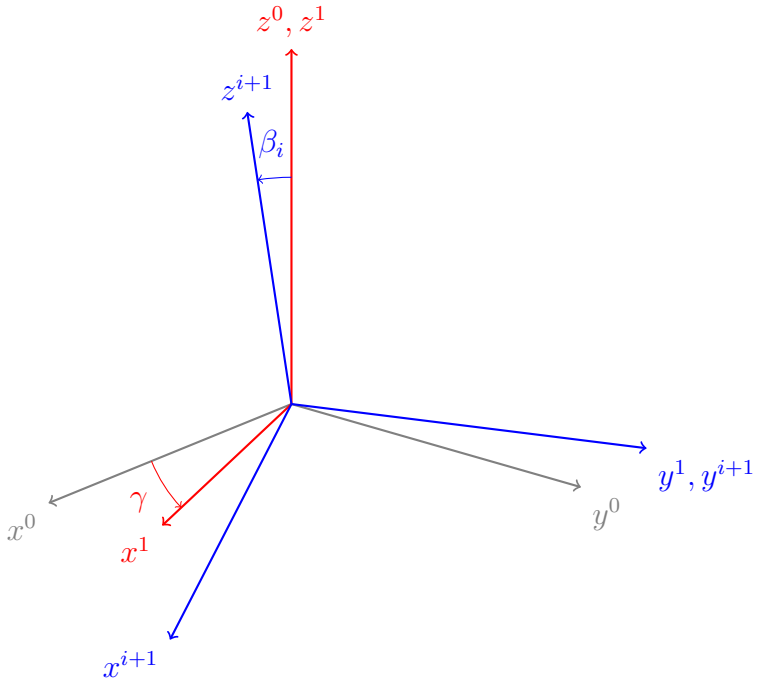

Figure 2.2: Coordinate Frame Transformations

blade is:

$$\beta_i = \eta_i + \theta_i \tag{2.1}$$

The position vector $\boldsymbol{r_i}$ from the center of rotation of each blade to an arbitrary point on the blade can be described as:

$$\boldsymbol{r_i} = x_i \boldsymbol{e}_{x^{(i+1)}} + y_i \boldsymbol{e}_{y^{(i+1)}} \tag{2.2}$$

It can be shown that the total angular velocity $\boldsymbol{\Omega}_i$ with respect to the initial stationary reference frame for the i^{th} blade is:

$$\boldsymbol{\Omega}_i = \dot{\delta}_i \boldsymbol{e}_{x^{(i+1)}} + \dot{\theta}_i \boldsymbol{e}_{y^{(i+1)}} + \dot{\lambda}_i \boldsymbol{e}_{z^{(i+1)}} \tag{2.3}$$

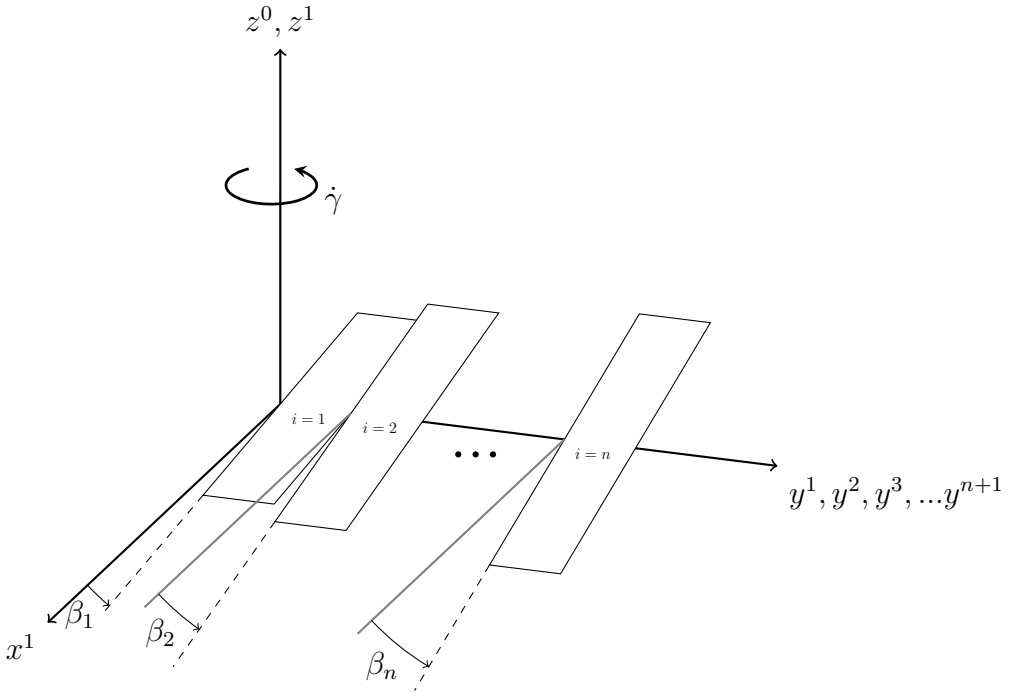

Figure 2.3: Diagram of Span-wise Discretization of Rotor into Blade Elements and variable definitions

where $\dot{\lambda}_i$ and $\dot{\delta}_i$ are relations created in order to reduce complexity and are defined as:

$$\dot{\lambda}_i = \dot{\gamma}\cos(\beta_i) \tag{2.4}$$

$$\dot{\delta}_i = \dot{\gamma}\sin(\beta_i) \tag{2.5}$$

The velocity vector $\dot{\boldsymbol{r}}_{\boldsymbol{i}}$ of any arbitrary point on each blade can be found by crossing the total angular velocity of the blade with the corresponding position vector of the blade. The velocity vector for the i^{th} blade can be shown to be:

$$\dot{\boldsymbol{r}}_{\boldsymbol{i}} = \boldsymbol{\Omega}_i \times \boldsymbol{r}_{\boldsymbol{i}} = -\dot{\lambda}_i y_i \boldsymbol{e}_{x(i+1)} + \dot{\lambda}_i x_i \boldsymbol{e}_{y(i+1)} + (-\theta_i x_i + \dot{\delta}_i y_i)\boldsymbol{e}_{z(i+1)} \tag{2.6}$$

and the velocity magnitude is

$$||\dot{\boldsymbol{r}}_{\boldsymbol{i}}||^2 = \dot{\lambda_i}^2 y_i^2 + \dot{\lambda_i}^2 x_i^2 + \dot{\delta_i}^2 y_i^2 - 2\dot{\theta}_i \dot{\delta}_i x_i y_i + \dot{\theta}_i x_i^2 \tag{2.7}$$

Kinetic and Potential Energy

In this section, the kinetic and potential energy for each blade element will be defined. The total kinetic and potential energies for the rotor blade will then be derived using the individual blade energy definitions. These relations are imperative when using Lagrangian mechanics to derive the equations of motion.

The kinetic energy for each blade element, T_i, is derived beginning with the basic definition of kinetic energy for an arbitrary differential mass element

$$T_i = \frac{1}{2} \int_m ||\dot{\boldsymbol{r}}_{\boldsymbol{i}}||^2 \, dm \tag{2.8}$$

Utilizing the relation for velocity magnitude derived in the previous section (Eq. 2.7) and plugging it into into Eq. 2.8 we obtain

$$T_i = \frac{1}{2} \int_m \dot{\lambda_i}^2 y_i^2 + \dot{\lambda_i}^2 x_i^2 + \dot{\delta_i}^2 y_i^2 - 2\dot{\theta}_i \dot{\delta}_i x_i y_i + \dot{\theta}_i x_i \, dm \tag{2.9}$$

The basic definitions for the moments and products of inertia for the i^{th} blade element with respect to the blade's coordinate system are:

$$I_{xx_i} = \int_m y_i^2 \, dm \tag{2.10}$$

$$I_{yy_i} = \int_m x_i^2 \, dm \tag{2.11}$$

$$I_{zz_i} = \int_m x_i^2 + y_i^2 \, dm \tag{2.12}$$

$$I_{xy_i} = \int_m x_i y_i \, dm \tag{2.13}$$

By plugging the basic definitions of the moments and products of inertia Eq. 2.10, 2.11, 2.12, 2.13 into Eq. 2.9 the kinetic energy then becomes:

$$T_i = \frac{1}{2}[\dot{\lambda_i^2} I_{xx_i} + \dot{\lambda_i^2} I_{yy_i} + \dot{\delta_i}^2 I_{xx_i} - 2\dot{\theta_i}\dot{\delta_i} I_{xy_i} + \dot{\theta_i} I_{yy_i}] \tag{2.14}$$

By combining the like terms and simplifying, Eq. 2.14 can be reduced to:

$$T_i = \frac{1}{2}[(\dot{\lambda_i^2} + \dot{\delta_i}^2) I_{xx_i} + (\dot{\lambda_i^2} + \dot{\theta_i}) I_{yy_i} - 2\dot{\theta_i}\dot{\delta_i} I_{xy_i}] \tag{2.15}$$

Plugging in Eq. 2.4 and Eq. 2.5 into Eq. 2.15, the following is obtained:

$$T_i = \frac{1}{2}[\dot{\gamma}(\cos^2(\beta_i) + \sin^2(\beta_i)) I_{xx_i} + (\dot{\gamma}\sin(\beta_i) + \dot{\theta_i}) I_{yy_i} - 2\dot{\theta_i}\dot{\gamma}\sin(\beta_i) I_{xy_i}] \tag{2.16}$$

Utilizing the Pythagorean identity:

$$\cos^2(\beta_i) + \sin^2(\beta_i) = 1 \tag{2.17}$$

Eq. 2.16 can be reduced to:

$$T_i = \frac{1}{2}[\dot{\gamma} I_{xx_i} + \dot{\gamma}\sin(\beta_i) I_{yy_i} + \dot{\theta}_i I_{yy_i} - 2\dot{\theta}_i \dot{\gamma}\sin(\beta_i) I_{xy_i}] \tag{2.18}$$

The total kinetic energy T is the sum of the kinetic energy of each individual blade.

$$T = \sum_{i=1}^{n} T_i \tag{2.19}$$

Where n is the total number of blade elements. Plugging Eq. 2.4 and Eq. 2.14 into Eq. 2.19 and expanding out, we obtain the relation for the total kinetic energy of the rotor blade. This relation will be utilized when deriving the equations of motion for each blade element.

$$T = \frac{1}{2}\sum_{i=1}^{n}[\dot{\gamma} I_{xx_i} + \dot{\gamma}\sin(\beta_i) I_{yy_i} + \dot{\theta}_i I_{yy_i} - 2\dot{\theta}_i \dot{\gamma}\sin(\beta_i) I_{xy_i}] \tag{2.20}$$

The potential energy of a single blade element, V_i, is dependant on its location along the length of the blade. For the first blade element closest to the root of the rotor blade ($i = 1$) the potential energy can be shown to be:

$$V_1 = \frac{1}{2} k_1 \theta_1^2 \tag{2.21}$$

For all internal blade elements located between the first and final blade elements

$(1 < i < n)$ the potential energy for the i^{th} blade can be shown to be:

$$V_i = \frac{1}{2} k_i (\theta_i - \theta_{i-1})^2 \tag{2.22}$$

For the last blade at the tip of the rotor $(i = n)$ the potential energy can be shown to be:

$$V_n = \frac{1}{2} k_n (\theta_n - \theta_{n-1})^2 \tag{2.23}$$

The total potential energy V is the sum of the potential energy for all blades.

$$V = \sum_{i=1}^{n} V_i \tag{2.24}$$

Expanding out, the total potential energy becomes

$$V = \frac{1}{2} [k_1 \theta_1^2 + \sum_{i=2}^{n-1} k_i (\theta_i - \theta_{i-1})^2 + k_n (\theta_n - \theta_{n-1})^2] \tag{2.25}$$

Here, k_i is the torsional stiffness coefficient and is defined in more detail in a later section. We have now defined the relations for the total kinetic and potential energy terms. With these terms, we can now move on to deriving the equations of motion using Lagrange's equations.

Lagrange's Equations

In this section, the relations derived in previous sections will be used in conjunction with Lagrangian mechanics in order to derive the equations of motion for each blade element. These equations of motion form the basis of the structural solver.

Lagrange's equations of motion is defined as [16]

$$\frac{\partial}{\partial t}\left(\frac{\partial T}{\partial \dot{q}_i}\right) - \frac{\partial T}{\partial q_i} + \frac{\partial V}{\partial q_i} = Q_i \tag{2.26}$$

Where q_i is the generalized coordinate and Q_i is the generalized force. A generalized coordinate is defined as a variable that uniquely defines any possible position or state of the system based on its initial position or state [17]. For this case the generalized coordinate is the angular displacement, θ_i. It should be noted that the angular velocity, $\dot{\gamma}$, could also be treated as a generalized coordinate. However, this is neglected because it would yield an equation of motion that describes the angular acceleration about the z^0 axis due to a given force input, and any acceleration about this axis would be prescribed. The generalized force can be shown to be equal to the aerodynamic moment, M_i, through the principal of virtual work. By taking the derivative with respect to the generalized coordinates the following relations are obtained.

$$\frac{\partial V}{\partial \theta_1} = k_1\theta_1 + k_2(\theta_1 - \theta_2) \tag{2.27}$$

$$\frac{\partial V}{\partial \theta_i} = k_i(\theta_i - \theta_{i-1}) + k_{i+1}(\theta_i - \theta_{i+1}) \tag{2.28}$$

$$\frac{\partial V}{\partial \theta_n} = k_n(\theta_n - \theta_{n-1}) \tag{2.29}$$

$$\frac{\partial T}{\partial \dot{\theta}_i} = \dot{\theta}_i I_{yy_i} - \dot{\gamma} sin(\beta_i) I_{yy_i} \tag{2.30}$$

$$\frac{\partial}{\partial t}\left(\frac{\partial T}{\partial \dot{\theta}_i}\right) = \ddot{\theta}_i I_{yy_i} - \ddot{\gamma} sin(\beta_i) I_{xy_i} - \dot{\gamma}\dot{\theta} cos(\beta_i) I_{xy_i} \tag{2.31}$$

By plugging 2.27-2.31 into 2.26 the Non-Linear Equations of Motion are obtained and shown to be:

For first blade ($i = 1$)

$$\ddot{\theta}_1 I_{yy_1} + \dot{\gamma}^2 sin(\theta_1 + \eta_1) cos(\theta_1 + \eta_1) I_{yy_1} + k_1\theta_1 + k_2(\theta_1 - \theta_2) = M_1 \tag{2.32}$$

For internal blades ($1 < i < n$)

$$\ddot{\theta}_i I_{yy_i} + \dot{\gamma}^2 sin(\theta_i + \eta_i) cos(\theta_i + \eta_i) I_{yy_i} + k_i(\theta_i - \theta_{i-1}) + k_{i+1}(\theta_i - \theta_{i+1}) = M_i \tag{2.33}$$

For outer blade ($i = n$)

$$\ddot{\theta}_n I_{yy_n} + \dot{\gamma}^2 sin(\theta_n + \eta_n) cos(\theta_n + \eta_n) I_{yy_n} + k_n(\theta_n - \theta_{n-1}) = M_n \tag{2.34}$$

Quasi-Static Equilibrium

The model is idealized for hovering stationary flight, this implies that the system being modeled is not undergoing any significant inertial or aerodynamic changes. Additionally, it is assumed that no mechanical vibration phenomena are present. Consequently, aerodynamic and mechanical forces, as well as the angular positions of the individual blade elements can be said to be constant. In other words, the system is said to be in a quasi-static equilibrium state. To solve for the quasi-static equilibrium point, the equations of motion must be tailored to represent the equilibrium state. Using the assumptions stated above, several relationships can be developed. First and foremost, because the angular position of each blade element is considered constant,

it can then be said that the derivative of the angular position with respect to time for each blade $\ddot{\theta}_i$ is zero.

$$\ddot{\theta}_i = 0 \tag{2.35}$$

The angular displacement for any given blade can be assumed to be equal to some displacement value at equilibrium θ_{i_0}.

$$\theta_i = \theta_{i_0} \tag{2.36}$$

The total angular displacement at equilibrium β_{i_0} can be obtained by substituting 2.36 into 2.1

$$\beta_{i_0} = \theta_{i_0} + \eta_i \tag{2.37}$$

substituting 2.35, 2.36, and 2.37 into the non-linear equations of motion 2.32-2.34, the following relations are obtained:
For first blade ($i = 1$)

$$\dot{\gamma}^2 sin(\theta_{1_0} + \eta_1) cos(\theta_{1_0} + \eta_1) I_{yy_1} + k_1\theta_{1_0} + k_2(\theta_{1_0} - \theta_{2_0}) = M_1 \tag{2.38}$$

For internal blades ($1 < i < n$)

$$\dot{\gamma}^2 sin(\theta_{i_0} + \eta_i) cos(\theta_{i_0} + \eta_i) I_{yy_i} + k_i(\theta_{i_0} - \theta_{i-1_0}) + k_{i+1}(\theta_{i_0} - \theta_{i+1_0}) = M_i \tag{2.39}$$

For outer blade ($i = n$)

$$\dot{\gamma}^2 sin(\theta_{n_0} + \eta_n) cos(\theta_{n_0} + \eta_n) I_{yy_n} + k_n(\theta_{n_0} - \theta_{n-1_0}) = M_n \tag{2.40}$$

These equations represent the structural equations of motion at a quasi-static equilibrium state. By coupling these equations with the aerodynamic and torsional stiffness models presented in the coming sections, and solving for θ_i, the steady state deflection of the rotor blade is obtained.

Aerodynamic Modeling

The Blade Element Model

To resolve the aerodynamic forces that are present on the surface of each blade element an aerodynamic model is developed using the BET. In order to reduce mathematical complexity, several idealizations are implemented. First, span-wise flow is neglected and each blade is treated as a thin airfoil.

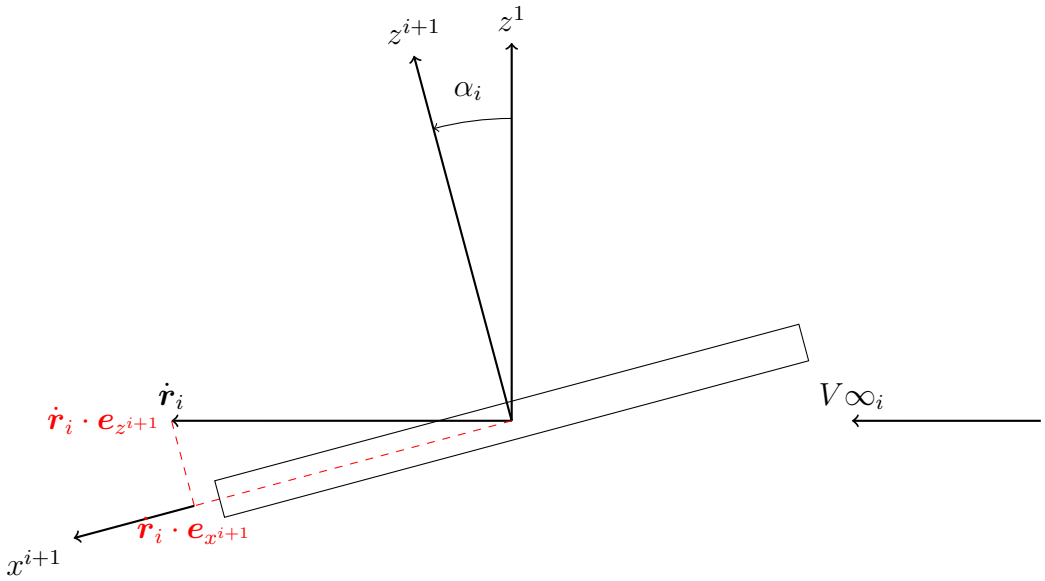

Figure 2.4: Diagram of Aerodynamic Forces and Vectors on i^{th} Blade

The velocity vector $\dot{\boldsymbol{r}}_{\boldsymbol{i}}$ for any arbitrary point on the blade was defined in the previous

section as:

$$\dot{\boldsymbol{r_i}} = -\dot{\gamma}\cos(\beta_i)y_i\boldsymbol{e}_{x^{(i+1)}} + \dot{\gamma}\cos(\beta_i)x_i\boldsymbol{e}_{y^{(i+1)}} + (-\theta_i x_i + \dot{\gamma}\sin(\beta_i)y_i)\boldsymbol{e}_{z^{(i+1)}} \tag{2.41}$$

Using Fig.2.4 and a basic trigonometric identity a relation for the angle of attack α_i for the i^{th} blade can be defined as:

$$\tan(\alpha_i) = \frac{\dot{\boldsymbol{r_i}} \cdot \boldsymbol{e}_{z^{(i+1)}}}{\dot{\boldsymbol{r_i}} \cdot \boldsymbol{e}_{x^{(i+1)}}} \tag{2.42}$$

Expanding out and solving for the angle of attack α_i becomes:

$$\alpha_i = \tan^{-1}\left(\frac{-\theta_i x_i + \dot{\gamma}\sin(\beta_i)y_i}{-\dot{\gamma}\cos(\beta_i)y_i}\right) \tag{2.43}$$

The above equation suggests that the angle of attack varies along the chord of each blade. For simplicity, the angle of attack will be referenced about the pitching axis. Therefore, $x_i = 0$ which leads to the following:

$$\alpha_i = \tan^{-1}\left(\frac{\dot{\gamma}\sin(\beta_i)y_i}{-\dot{\gamma}\cos(\beta_i)y_i}\right) = \tan^{-1}(\tan(\beta_i)) = \beta_i \tag{2.44}$$

The general forms for the aerodynamic lift L_i, drag D_i, and moment M_i were obtained from [6] and are as follows:

$$L_i = C_{li}q_{\infty i}S_i \tag{2.45}$$

$$D_i = C_{di}q_{\infty i}S_i \tag{2.46}$$

$$M_i = C_{mi} q_{\infty i} S_i X_{cp} \tag{2.47}$$

Where $q_{\infty i}$ is the dynamic pressure and S_i is the surface area of the blade element and are defined as:

$$q_{\infty i} = \frac{1}{2} \rho {V_{\infty i}}^2 \tag{2.48}$$

$$S_i = w_{bi} l_{bi} \tag{2.49}$$

Here ρ is the fluid density, X_{cp} is the center of pressure, w_{bi} and l_{bi} are the blade width and cord length respectively, C_{li}, C_{di}, and C_{mi} are the coefficients of lift, drag and moment respectively, and $V_{\infty i}$ is the free stream fluid velocity magnitude. The coefficients of lift and drag are obtained empirically and used in the calculation of the moment coefficient (Eq. 2.52).

The center of pressure, X_{cp}, in this case is idealized to be constant and is located at the quarter-chord point. However, in some cases it can be a function of the angle of attack α_i. This idealization suggests that there may also be an additional aerodynamic moment about the quarter-chord point $M_{c/4}$. In order to further reduce the complexity of the aerodynamic model, the quarter-chord moment is assumed to be near zero and is therefore neglected. This assumption is assumed to be valid because due to the idealization of the cross section as a thin symmetrical flat plate airfoil. Theoretically, the moment about the quarter chord point for a thin symmetrical airfoil is zero [6]. The validity of this assumption will be further explored in the following chapter.

In order to reduce complexity, the system is assumed to be hovering. For

stationary hovering flight, the free stream fluid velocity magnitude is equal to the velocity magnitude of the blade and there is no component of the magnitude that is a result of the center of rotation translating. The free stream fluid velocity magnitude can be obtained by taking the magnitude of the velocity vector (Eq. 2.50). However, because span-wise flow is neglected, the component of velocity along the span-wise $\boldsymbol{e}_{y^{(i+1)}}$ direction is ignored. Additionally, referencing the angle of attack about the pitching axis makes $x_i = 0$, leading to:

$$\dot{\boldsymbol{r}}_{\boldsymbol{i}} = -\dot{\gamma}\cos(\beta_i) y_i \boldsymbol{e}_{x^{(i+1)}} + \dot{\gamma}\sin(\beta_i) y_i \boldsymbol{e}_{z^{(i+1)}} \tag{2.50}$$

The free stream fluid velocity is equal to the square of the magnitude of the velocity vector:

$$V_{\infty i}{}^2 = ||\dot{\boldsymbol{r}}_{\boldsymbol{i}}||^2 = \dot{\gamma_i^2} y_i^2 \tag{2.51}$$

The generalized form of the moment coefficient is defined as:

$$C_{mi} = C_{li}\cos(\alpha_i) + C_{di}\sin(\alpha_i) \tag{2.52}$$

where the coefficients of lift C_{li} and drag C_{di} are functions of the angle of attack and are obtained empirically. By plugging in Eq.2.44, 2.48, 2.49, 2.51, 2.52 into Eq.2.45, 2.46, 2.47 the equations for lift, drag, and the generalized moment for the i^{th} blade respectively are shown to be:

$$L_i = \frac{1}{2} C_{li} \rho \dot{\gamma_i^2} y_i^2 w_{bi} l_{bi} \tag{2.53}$$

$$D_i = \frac{1}{2} C_{di} \rho \dot{\gamma_i^2} y_i^2 w_{bi} l_{bi} \tag{2.54}$$

$$M_i = \frac{1}{2}(C_{li}\cos(\beta_i) + C_{di}\sin(\beta_i))\rho\dot{\gamma_i^2}y_i^2 w_{bi} l_{bi} X_{cp} \tag{2.55}$$

Material Mechanics and Torsional Stiffness

A valuable parametric control in the design of a flexible rotor blade is the cross-sectional dimensions. Incorporating the ability for the model to modulate cross-sectional thickness and width over the length of the blade will allow for more complex blade geometries to be examined and modeled. However, changing these parameters also directly effects the local torsional stiffness. Torsional stiffness is a measure of a structure or member's resistance to torsional deflection. The torsional stiffness is largely dependent on the torsional constant, J. The torsional constant is a geometric property and is defined by the cross-sectional area. Therefore, modulating the cross-sectional dimensions over the length of the blade also modulates the torsional constant, and consequently the torsional stiffness over the length of the blade. Thus, understanding the relation between the torsional constant and the cross-sectional area is imperative. The torsional constant for non-uniform cross sections, such as an airfoil, can be complex and nontrivial to determine, and often requires the use of numerical methods to solve [23]. However, the idealization of each blade element as having a rectangular cross-section allows for the simplification of modeling the torsional stiffness. The torsional constant for a body with a rectangular cross section is well understood and documented [10].

This section will present the mathematical definitions for torsion utilized for this model. This includes the relation of the torsional constant to the cross sectional parameters, as well as how it subsequently relates to the torsion coefficient. The following torsional model is largely based off of Hooke's law. Consequently, this

implies that the material used is linearly elastic and that deformation is entirely within the elastic regime.

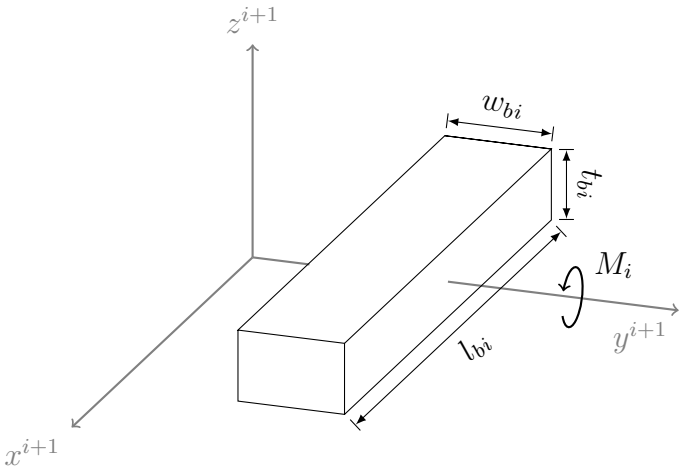

Figure 2.5: Rectangular Cross Section of Rotor Blade

Using Hooke's law and treating each element as a torsional spring, we can define the angular displacement for the i^{th} blade element as:

$$M_i = k_i \theta_i \tag{2.56}$$

Where M is the aerodynamic moment, k is the torsion stiffness coefficient, and θ_i is the angular displacement due to the moment. The relation between the angular displacement and torque applied is also defined as:

$$\theta_i = \frac{M_i w_{bi}}{G J_i} \tag{2.57}$$

G is the shear modulus, and J_i and w_b are the torsional constant and width for the i^{th} blade respectively. The width of each blade is defined as:

$$w_{bi} = \frac{R}{N_b} \tag{2.58}$$

where R is the radius of the rotor and N_b is the number of blades. Combining 2.58, 2.57, and 2.56 and solving for k_i we obtain the relation:

$$k_i = \frac{GJ_iN_b}{R} \tag{2.59}$$

Note that G, N_b, and R are known material and geometric constants, and J_i will change along the length of the blade. The torsional constant is defined by [10] re-writing it in index form leads to the following relation

$$J_i = c_i(t_{b_i})^3(l_{b_i}) \tag{2.60}$$

where c_i is the torsional parameter and is defined by

$$c_i = \frac{1}{3}\left[1 - \frac{192}{\pi^5}\frac{t_{b_i}}{l_{b_i}}\sum_{n=1,3,5,\ldots}^{\infty}\frac{1}{n^5}\tanh\frac{nl_{b_i}\pi}{2t_{b_i}}\right] \tag{2.61}$$

In order to reduce mathematical complexity, the torsional parameter will be approximated by reducing the infinite sum to the first two terms. Which leads to:

$$c_i \approx \frac{1}{3}\left[1 - \frac{192}{\pi^5}\frac{t_{b_i}}{l_{b_i}}\left(\tanh\frac{l_{b_i}\pi}{2t_{b_i}} + \frac{1}{3^5}\tanh\frac{3l_{b_i}\pi}{2t_{b_i}}\right)\right] \tag{2.62}$$

Table 2.1: Comparison of Actual and Approximated Torsional Parameter and Percent Error. c_{Actual} is taken from [10].

l_b/t_b	1	1.5	2	2.5	3	4	6	10	∞
c_{Actual}	0.141	0.196	0.229	0.249	0.263	0.281	0.299	0.312	0.333
c_{Approx}	0.1407	0.1958	0.2287	0.2494	0.2633	0.2808	0.2983	0.3123	0.3333
% Error	-0.21%	-0.10%	-0.13%	0.16%	0.11%	-0.07%	-0.23%	0.10%	0.09%

As shown in Table 2.1 the percent error between the approximated value and

average value of the torsional parameter is relatively low, with a maximum percent error magnitude of 0.23%. Therefore, the approximation of this parameter is assumed to be valid.

Combining 2.59, 2.60, and 2.62, the approximate torsional stiffness coefficient for a blade with a rectangular cross section is shown to be:

$$k_i \approx \frac{G(t_{b_i})^3(l_{b_i})N_b}{3R}\left[1 - \frac{192}{\pi^5}\frac{t_{b_i}}{l_{b_i}}\left(\tanh\frac{l_{b_i}\pi}{2t_{b_i}} + \frac{1}{3^5}\tanh\frac{3l_{b_i}\pi}{2t_{b_i}}\right)\right] \tag{2.63}$$

NUMERICAL SIMULATION

This chapter focuses on the application and possible execution of the ROM presented in this work. A particular size and dynamic range of UAV rotor will be used as a surrogate for exploring model performance as a whole. This will also allow for a preliminary analysis of flexible blade performance when compared to a rigid counterpart. The feasibility of using the model for the optimization of a flexible rotor blade will also be explored.

The chapter will begin with the methods used for the determination of aerodynamic coefficients. We will then move on to an investigation into the computational expenses and convergence criterion of the ROM presented. Additionally, a method used to determine the ideal torsional shape for a rotor blade will be presented. This ideal shape will then be used in conjunction with the ROM to analyze the performance characteristics of various blade configurations.

Lift and Drag Coefficients For a Flat Plate

The coefficients of lift and drag are of the upmost importance when predicting aerodynamic forces. These coefficients are functions of the angle of attack and are also largely dependent on the dynamic range. The coefficients can differ drastically for an airfoil of the same geometry and angle of attack in a low Reynolds number or laminar flow when compared to the same airfoil in a high Reynolds number or turbulent flow. Additionally, these coefficients vary drastically as the angle of attack changes. The relation between the coefficients and angle of attack for a flat plate are fairly well understood and have been explored in a wide range of previous works [6, 20, 22, 32, 43, 44, 48]. However, obtaining the lift and drag coefficient values for the specific dynamic range and geometry needed proved to be difficult. For example,

Jiang et al. characterized the lift and drag coefficients for a flat plate for Reynolds number is within the range of 10000 to 1000000. However, this work was only for low or high angles of attack and did not characterize the onset of stall. Mueller and Roth-Gibson presented the lift and drag coefficients for a flat symmetrical airfoil at Reynolds numbers equal to 80,000 and 140,000, this data is shown in Fig. 3.10 and Fig.3.11, and was digitized using [4]. However, the flat plate that was used in this experiment has a non rectangular cross section. Knowing the point of stall onset and characterizing the stall region for a airfoil with a rectangular cross section is imperative to the BET model framework. Therefore, the stall angle was estimated at Reynolds numbers appropriate for UAVs within this body of work. Traditionally, the coefficients of lift and drag are obtained empirically. However, for the sake of time the coefficients of lift and drag were obtained via CFD.

ANSYS Fluent, version 2020 R2 Academic, was utilized to perform the CFD simulations and obtain the coefficient information. The geometry was created in Design Modeler and can be seen in Fig. 3.1 and Fig. 3.2. A fillet was applied to the edges of the rectangular cross section in order to reduce the possibility of simulation divergence and increase numerical stability.

An unstructured quadrilateral computational mesh (Fig. 3.3) was created with the ANSYS Meshing Program. The statistics of this mesh can be seen in Table 3.1. To achieve good mesh quality, several mesh refinements were performed. Separate edge sizing refinements on the horizontal, vertical, and radius edges of the airfoil were prescribed. The number of divisions for each edge sizing refinement was chosen such that the change in element size in the areas surrounding the edges was relatively smooth (Fig. 3.5). Additionally,a thickness specified inflation layer refinement was added to all airfoil edges and adjusted until the wall Y+ values were < 1. This ensures that when using a transition viscous model, boundary layer effects are properly

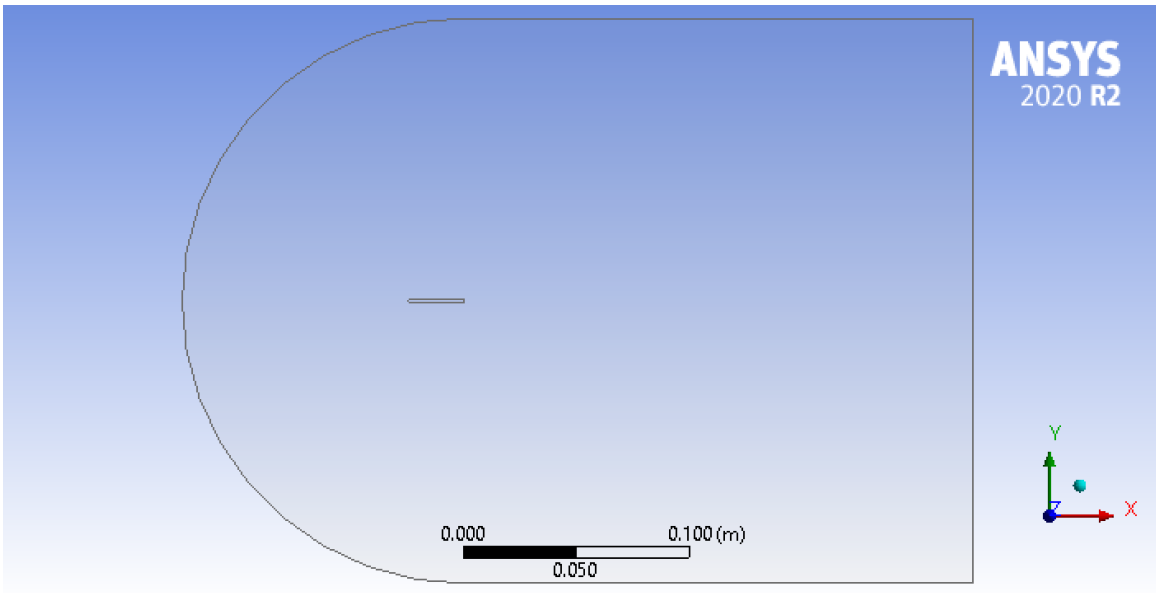

Figure 3.1: Geometry from ANSYS Fluent CFD Simulation of 2D Flat Plate.

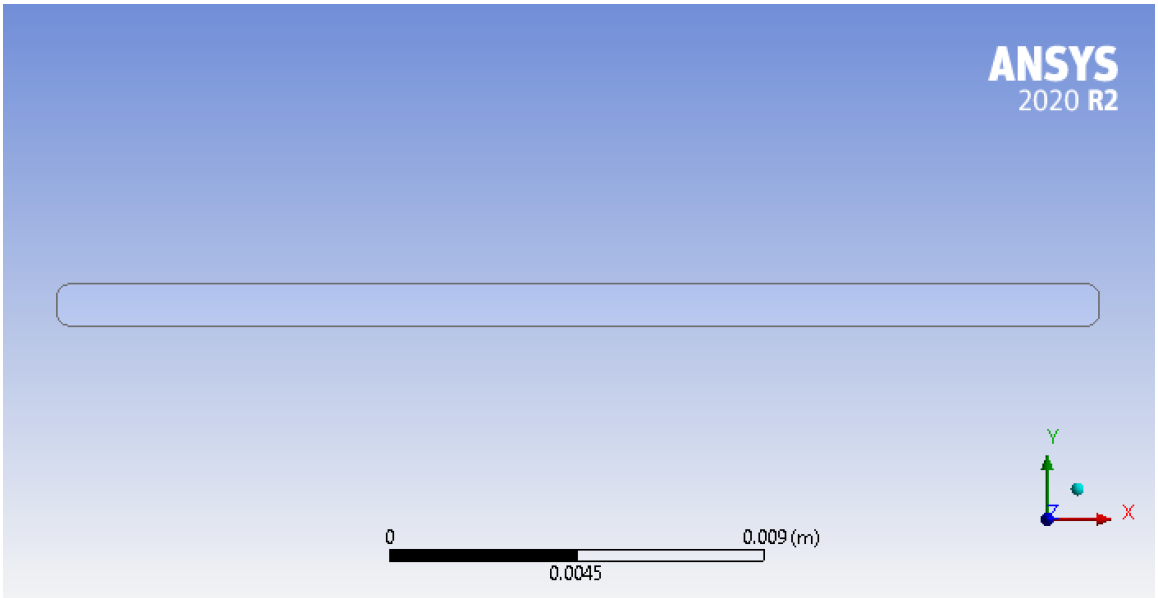

Figure 3.2: Geometry Detail from ANSYS Fluent CFD Simulation of 2D Flat Plate.

resolved. For the areas surrounding the airfoil and airfoil wake, two separate face sizing refinements were implemented. A near-airfoil face sizing refinement in the area directly surrounding the airfoil, shown in red, and a face sizing refinement in the wake

area of the airfoil, shown in blue in Fig. 3.3. These refinements ensure that mesh element sizes are small enough to properly resolve any fluid structures in the areas surrounding the airfoil. The mesh refinement settings can be found in Tables 3.2-3.4.

Table 3.1: Mesh Statistics for 2D Flat Plate ANSYS Fluent CFD.

Description	Element Count	Node Count	Growth Rate	Element Size
Mesh Stats.	122988	124181	1.12	5e-3 m

Table 3.2: Edge Sizing Mesh Refinement Settings for 2D Flat Plate ANSYS Fluent CFD.

Description	Number of Divisions	Behavior	Bias
Horizonal Edge	1050	Hard	No Bias
Virtical & Radius Edge	40	Hard	No Bias

Table 3.3: Inflation Layer Mesh Refinement Settings for 2D Flat Plate ANSYS Fluent CFD.

Description	Inflation Option Setting	Number of Layers	Maximum Thickness
Inflation Layers	Total Thickness	30	5e-4 m

Due to the dynamic range of the simulation and the necessity to predict near wall turbulence so that drag is accurately predicted, a Transition $k - k\omega$ viscous model was implemented. This model was also chosen because it proved to provide the most stable steady state solution when compared to an invicid or $k\omega$ SST model.

Table 3.4: Face Sizing Mesh Refinement Settings for 2D Flat Plate ANSYS Fluent CFD.

Description	Element Size	Influence Radius
Near Foil Face	5e-4 m	3e-2 m
Wake Face	5e-4 m	4e-2 m

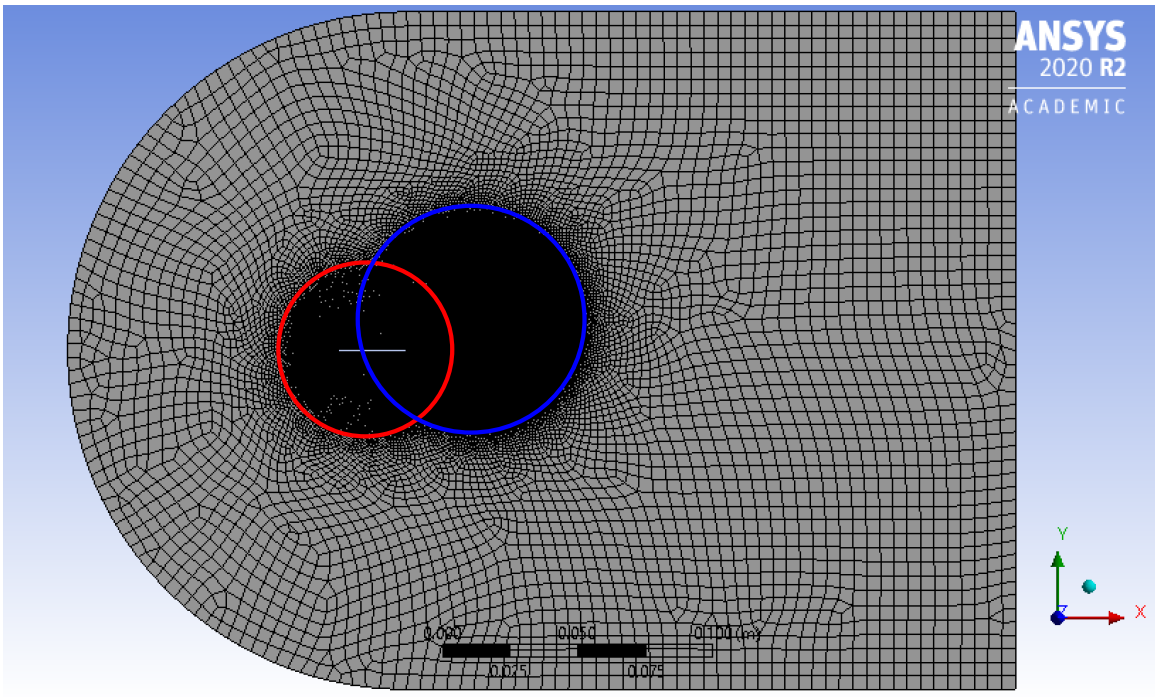

Figure 3.3: Mesh from ANSYS Fluent CFD Simulation of 2D Flat Plate, Face Sizing Refinements Shown; Near Airfoil (Red) and Wake (Blue).

Velocity inlet and pressure outlet boundary conditions are defined around the outside of the 2D fluid domain. Figure 3.6 shows which domain boundaries were assigned to inlet (shown in blue) and outlet (shown in red) conditions. Additionally, no-slip wall boundary conditions were set for the airfoil boundary. To induce the angle of attack, X and Y velocity components were specified in the velocity inlet settings using the relations shown in Eq. 3.1 and Eq. 3.2. Where V_∞ is the velocity magnitude of a

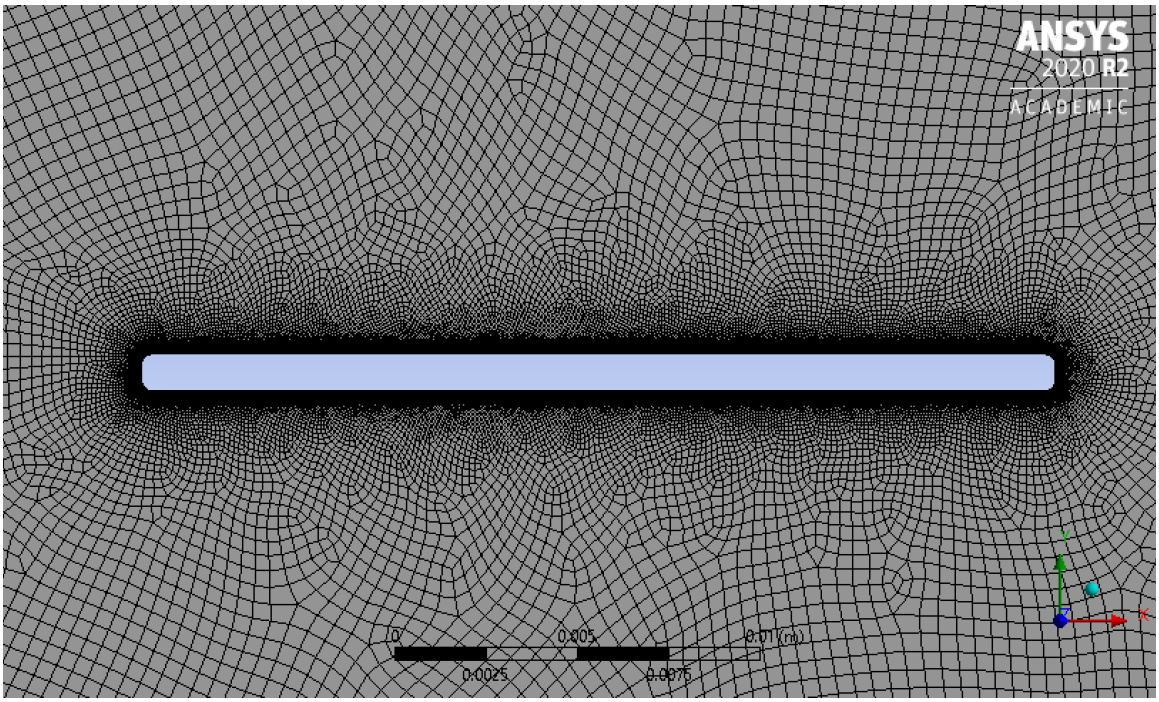

Figure 3.4: Mesh Detail from ANSYS Fluent CFD Simulation of 2D Flat Plate.

point at half span of a 10 cm radius blade rotating at 1000 rad/s. With a chord length of 2.5 cm, this places the rotor blades Reynolds number around 80,000. The simulation was repeated for multiple angles of attack and would run for anywhere from 4000 to 8000 iterations to ensure solution convergence. Figures 3.9 and 3.7 show the normal and axial coefficients, as well as residual information for a particular converged solution.

$$X_\infty = V_\infty \cos(\alpha) \tag{3.1}$$

$$V_\infty = V_\infty \sin(\alpha) \tag{3.2}$$

Normal and axial coefficient data, as well as quarter chord moment coefficient data, was recorded from the ANSYS simulation and can be found in Fig. 3.9 and Fig.

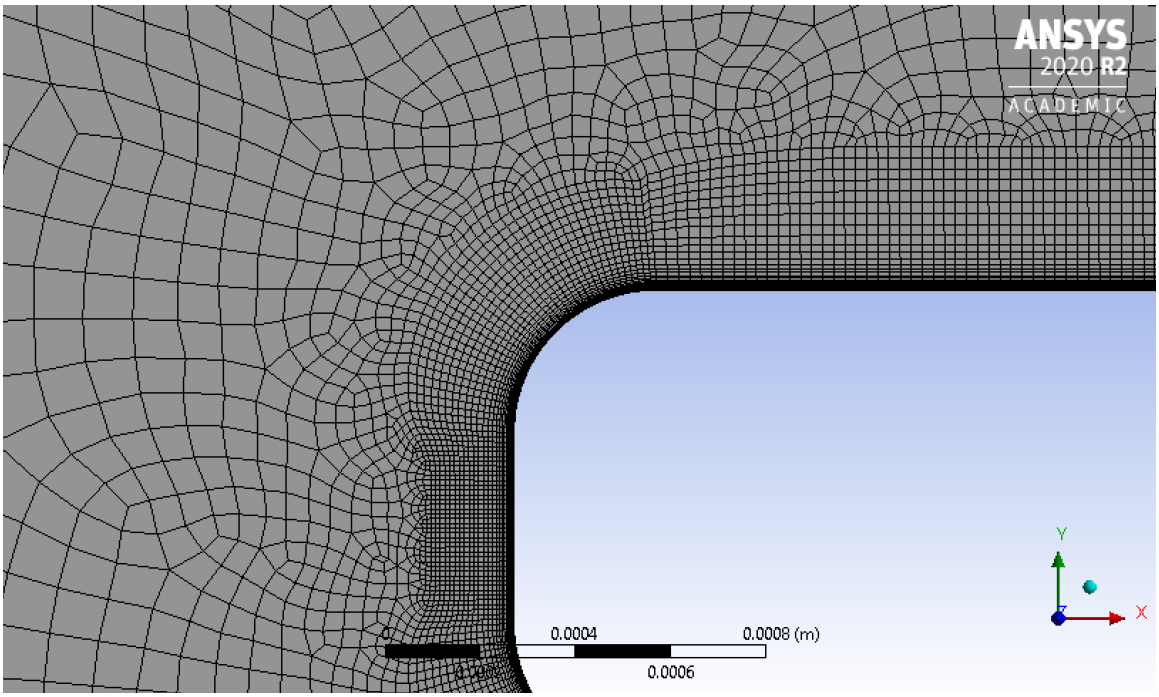

Figure 3.5: Mesh Boundary from ANSYS Fluent CFD Simulation of 2D Flat Plate.

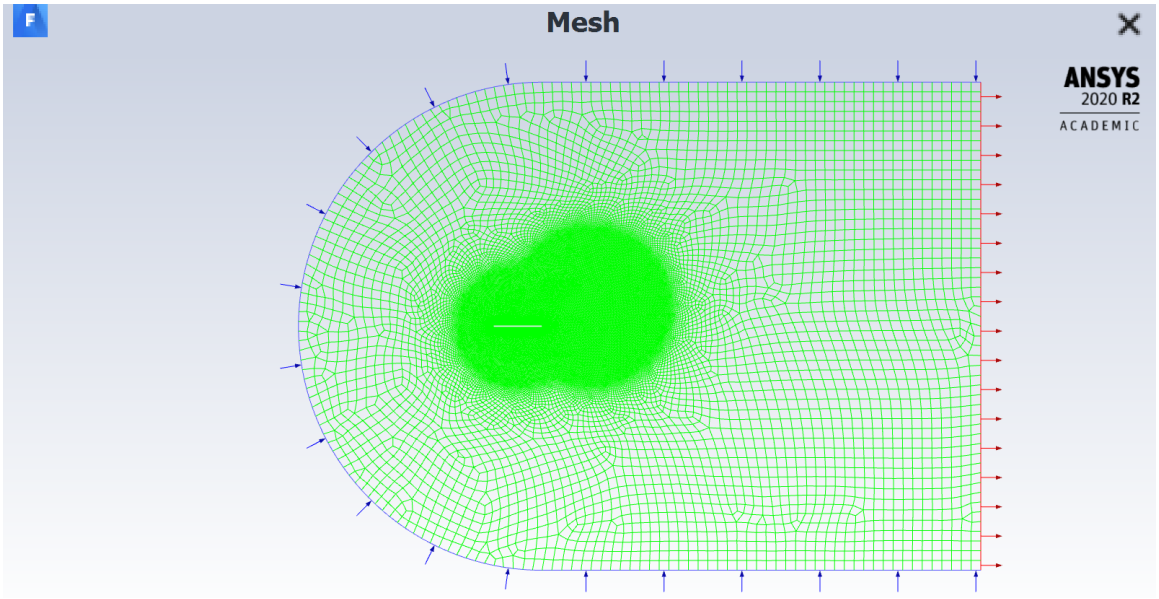

Figure 3.6: Inlet and Outlet Boundary Condition Settings Fluent CFD Simulation of 2D Flat Plate.

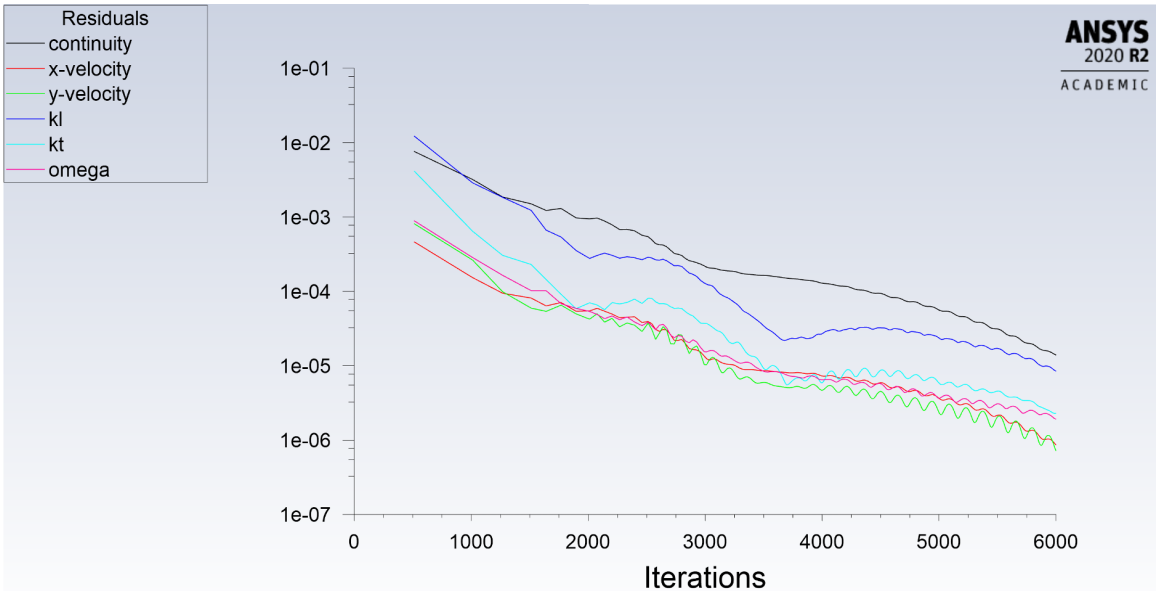

Figure 3.7: Residuals from ANSYS Fluent CFD Simulation of 2D Flat Plate for $\alpha = 5^0$.

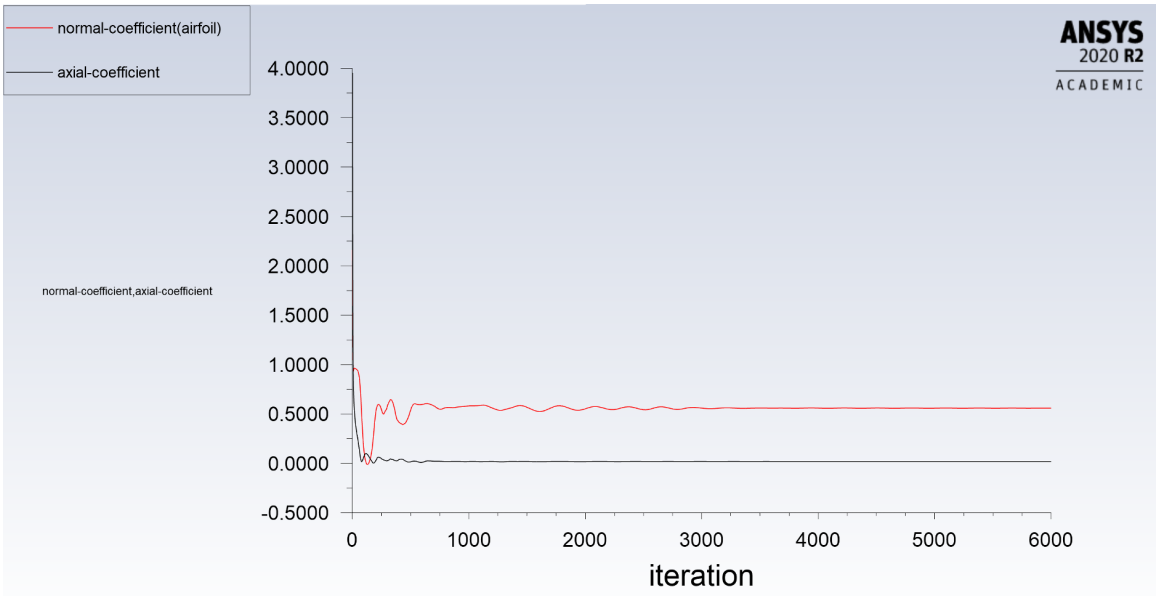

Figure 3.8: Normal and Axial Coefficients from ANSYS Fluent CFD Simulation of 2D Flat Plate for $\alpha = 5^0$.

3.13. This data was then converted to lift and drag coefficient data using a relation developed from equations obtained from [6]. The conversion from normal and axial

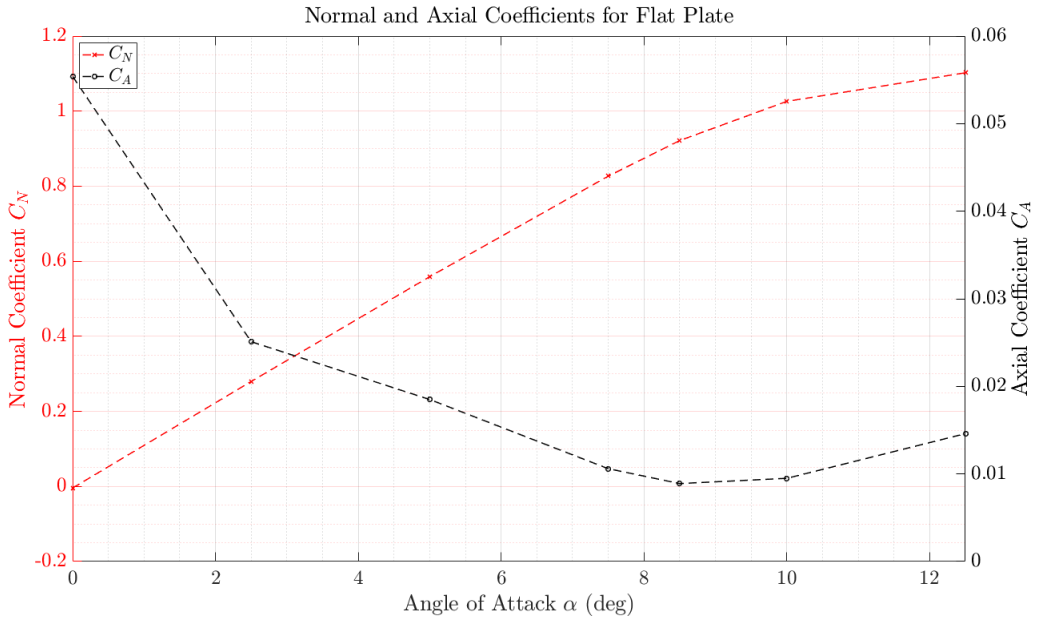

Figure 3.9: Normal and Axial Coefficient Data Obtained from ANSYS Fluent CFD Simulation of 2D Flat Plate.

coefficients to lift and drag coefficients is

$$C_l = C_N \cos(\alpha) - C_A \sin(\alpha) \tag{3.3}$$

$$C_d = C_N \sin(\alpha) + C_A \cos(\alpha) \tag{3.4}$$

The resulting lift and drag coefficients can be seen in Fig. 3.10 and Fig. 3.11. When comparing the results from the CFD simulation to the estimated lift and drag coefficients obtained from Jiang et al. shown in Fig. 3.10 and Fig. 3.11, it can be seen that the lift data obtained from CFD agrees very well where the CFD analysis indicates the onset of stall at roughly $\alpha = 8.5°$. The drag data obtained from the CFD simulation agrees less with Jiang et al.. This may be because skin friction drag is being neglected when plotting the reference data. While the CFD simulation is accounting

for skin friction drag. However, the data obtained from the CFD simulation agrees less with the data obtained from [32]. The reason for this discrepancy may be due to the fact that Mueller and Roth-Gibson used a non-rectangular cross section.

As stated in the previous chapter, the center of pressure X_{cp} is assumed to be at the quarter chord point for all angles of attack. This assumption is based largely on thin symmetrical airfoil theory [6]. This assumption was tested using relations gathered from [6] and data gathered from the CFD simulation. The relation for the center of pressure location normalized to the chord length can be seen in Eq. 3.5. Using this relation and the data from Fig. 3.9 and Fig. 3.13, the quarter chord location can be computed and then compared to the theoretical assumption. The results of this can be seen in Fig. 3.14. This comparison shows that the assumption of the center of pressure being constant at the quarter chord point, and that the moment about that point is approximately zero is valid.

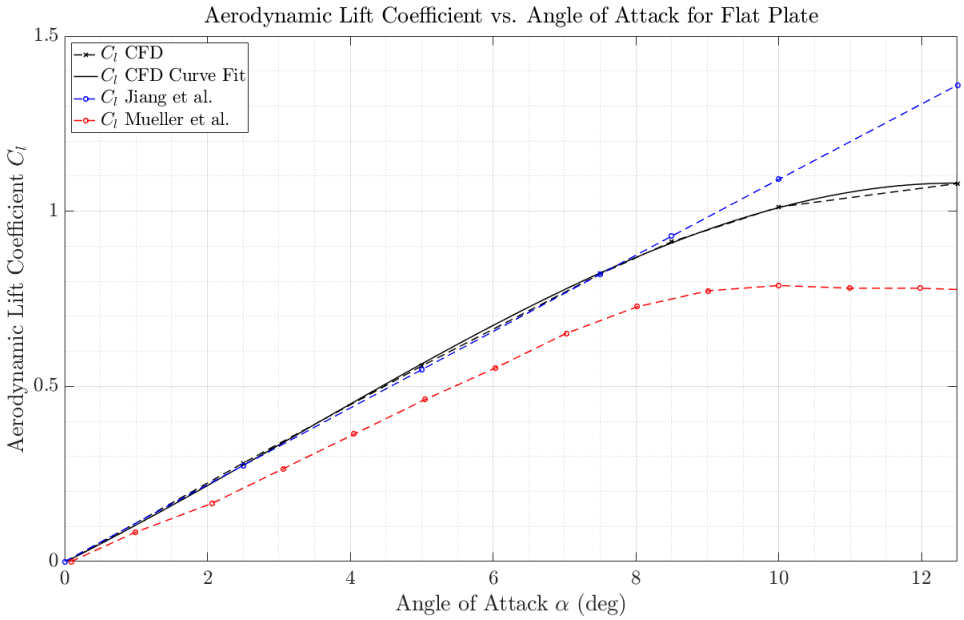

Figure 3.10: Comparison of Lift Coefficient Data Obtained from CFD Simulation additional data taken from [22], and [32]

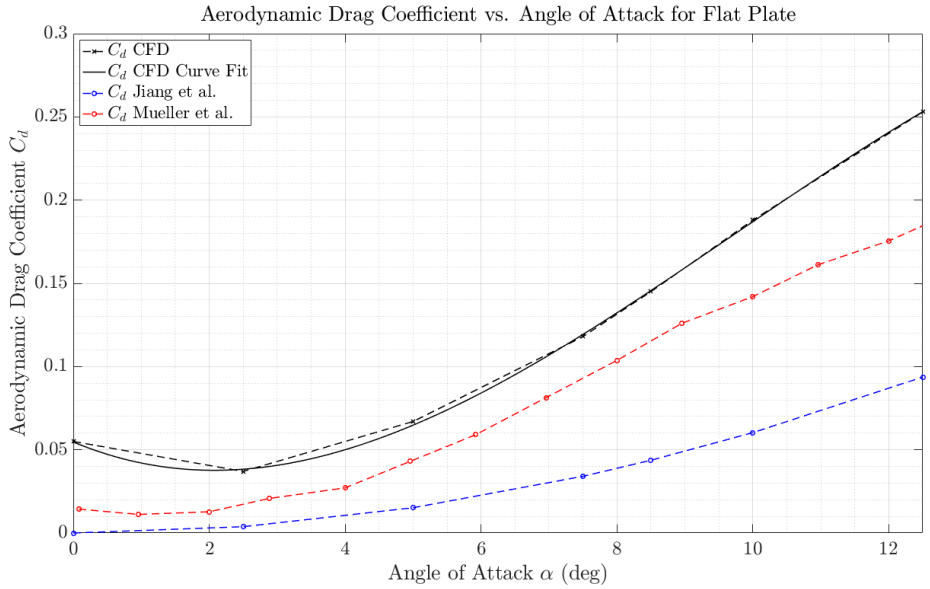

Figure 3.11: Comparison of Drag Coefficient Data Obtained from CFD Simulation additional data taken from [22], and [32]

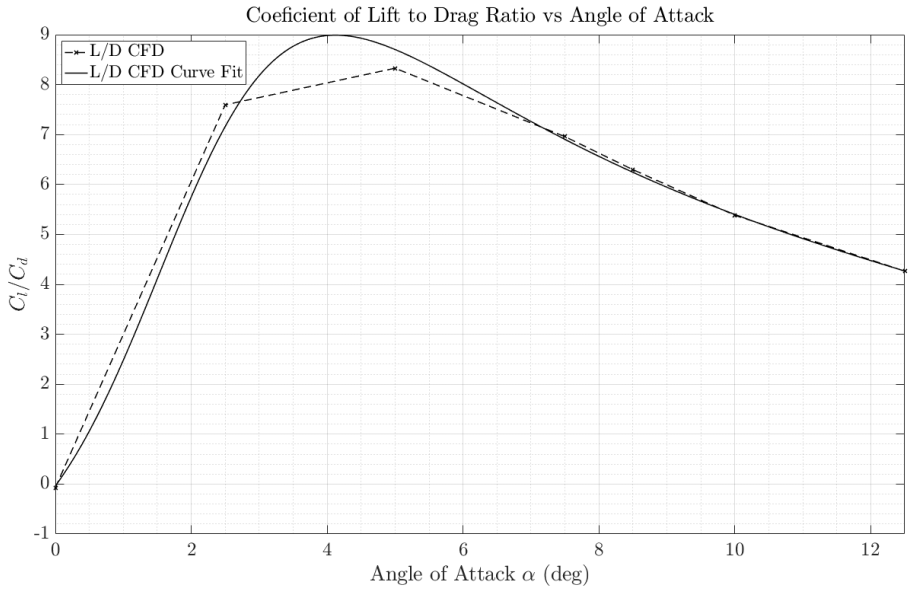

Figure 3.12: Lift to Drag Ratio Data Obtained from ANSYS Fluent CFD Simulation of 2D Flat Plate.

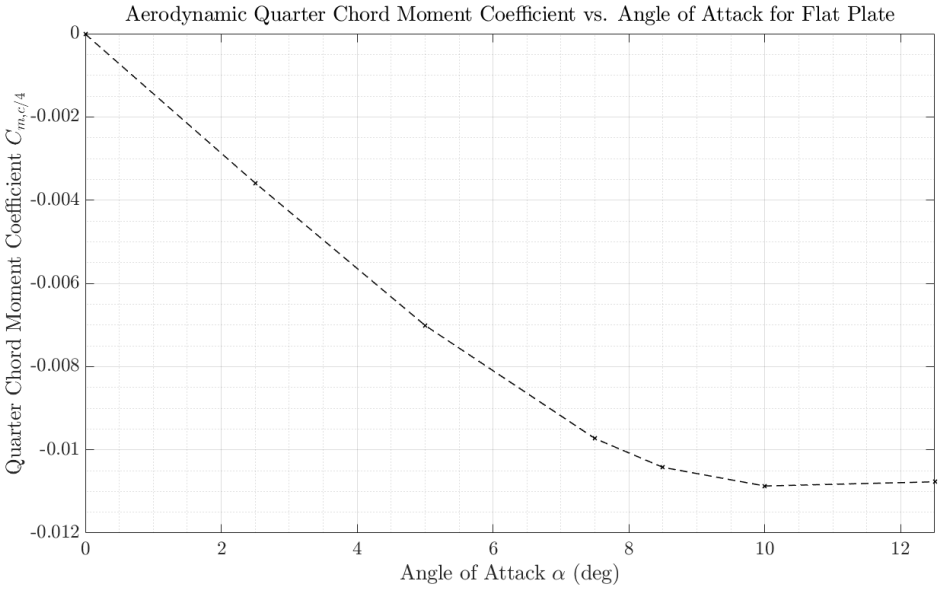

Figure 3.13: Quarter Chord Moment Coefficient Data Obtained from ANSYS Fluent CFD Simulation of 2D Flat Plate.

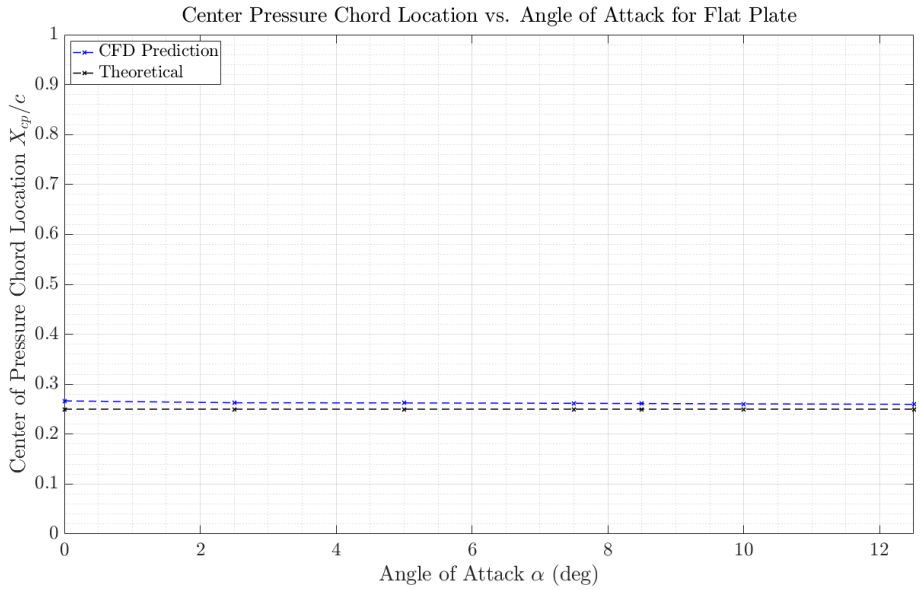

Figure 3.14: Comparison of Center of Pressure location for Theoretical[6] and Numerical Simulation

$$\frac{X_{cp}}{c} = \frac{1}{4} - \frac{C_{m,c/4}}{C_N} \tag{3.5}$$

System Convergence and Computation Times

This section will be focused on the exploration of Numerical convergence and total computation. The calculations were performed on a 2012 Apple MacBook Pro with a 2.3 GHz Quad-Core Intel Core i7 and 16 GB 1600 MHz DDR3 RAM running MATLAB R2020a. Convergence values were determined by modulating the number of blade elements and spin speed and recording the total lift and drag values for each case. By plotting this information, the amount of blade elements necessary to achieve convergence can be determined by analyzing the point when lift and drag values no longer change with respect to an increase in blade count. Figures 3.15 and 3.16 show the lift and drag force values respectively compared with total number of blade elements. Additionally, the total computation time compared with the number of blade elements was determined using a similar manner stated above in conjunction with the MATLAB stopwatch timer feature [2]. The data collected shows that higher spin speeds require a higher number of blade elements to reach numerical convergence on lift and drag forces. When analyzing these plots it can be seen that, even for the highest spin speed, convergence is achieved at roughly 80 blade elements. When comparing this to Fig. 3.17, which shows the computation times for both multiple and single values of $\dot{\gamma}$, it can be shown that computation times can take as little as 0.016s to reach a solution.

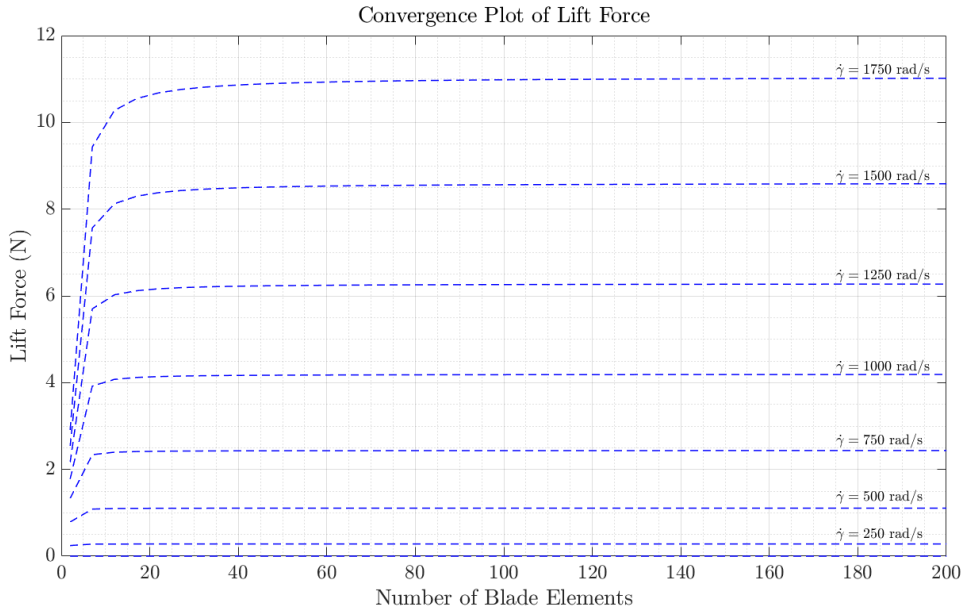

Figure 3.15: Convergence of Total Drag Force for Flexible Blade for Various Rotor Speeds

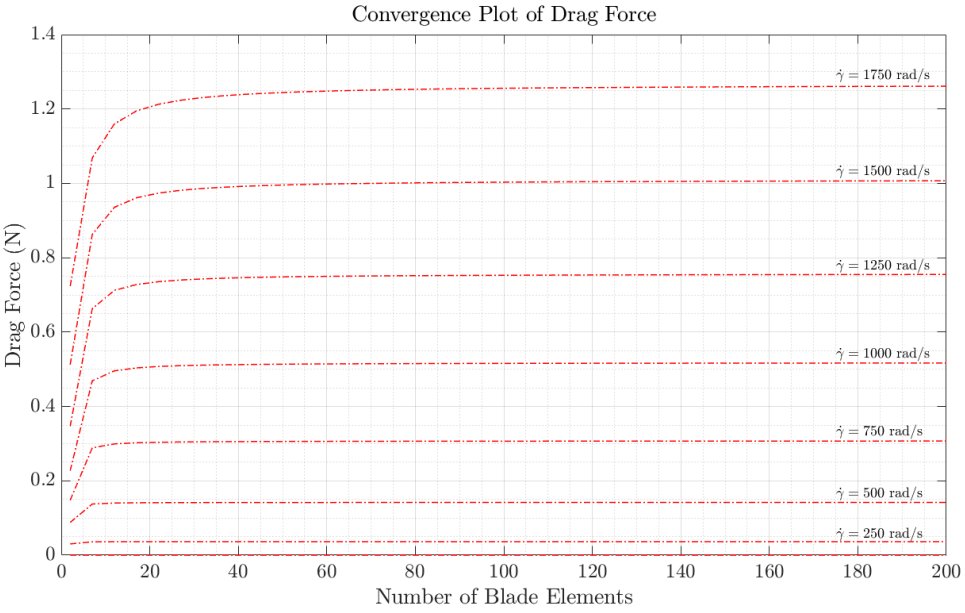

Figure 3.16: Convergence of Total Drag Force for Flexible Blade for Various Rotor Speeds

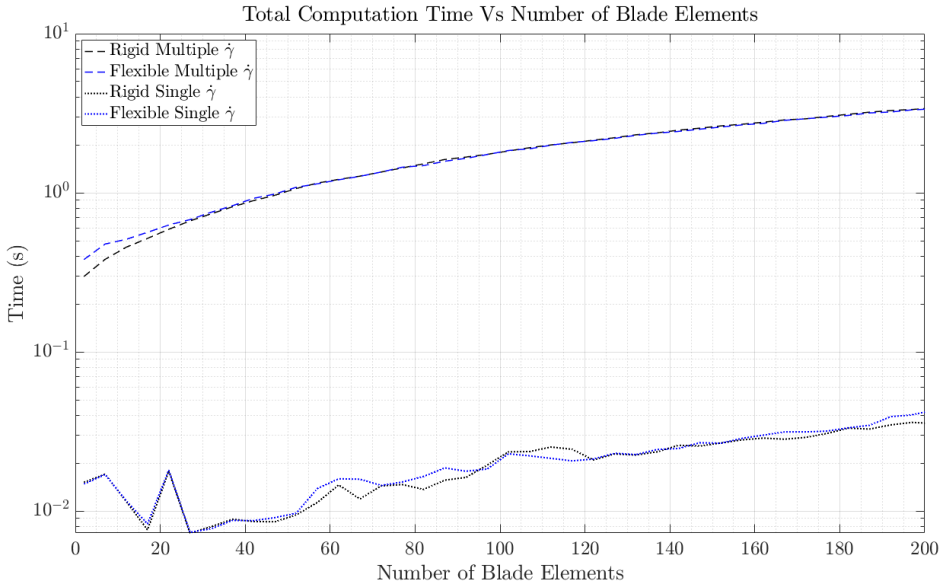

Figure 3.17: Computation Time vs. Number of Blade Elements for Rigid and Flexible Rotor Blades.

Blade Twist Optimization

An optimized torsional shape was determined to aid in the analysis of different blade configurations. This optimal shape is the torsional shape of a rotor blade such that the total lift to drag ratio is maximized. This shape will be used to define the shape of a fully rigid blade, the innitial shape of a flexible blade, and act as a target shape for an optimized flexible blade. In doing this, we can easily see the effects that flexibility might have on aerodynamic performance while also aiding in the exploration of flexible blade optimization.

In order to determine an optimum blade twist shape a grid search method was utilized. To do this, first the optimized blade shape is assumed to take the form of the second order polynomial shown in Eq. 3.6. A blade root pretwist, or c value, is defined and a script iterates over a and b values. The lift and drag forces are

calculated for each blade element and then summed to produce a total lift and drag force for that particular set of a and b values. These total lift and drag force values are then used to calculate the total lift to drag ratio for a particular set of a and b values. All of these values are stored in matrix form and are plotted in Fig. 3.18, Fig. 3.19, and Fig. 3.20. Utilizing a grid search method and locating the point of highest value on Fig. 3.20 the a and b coefficients for the blade shape with the highest lift to drag ratio is found. This resulting torsional blade shape can be seen in Fig. 3.21 and Eq. 3.7 is the corresponding equation.

$$\beta_{optim} = ay^2 - by + c \tag{3.6}$$

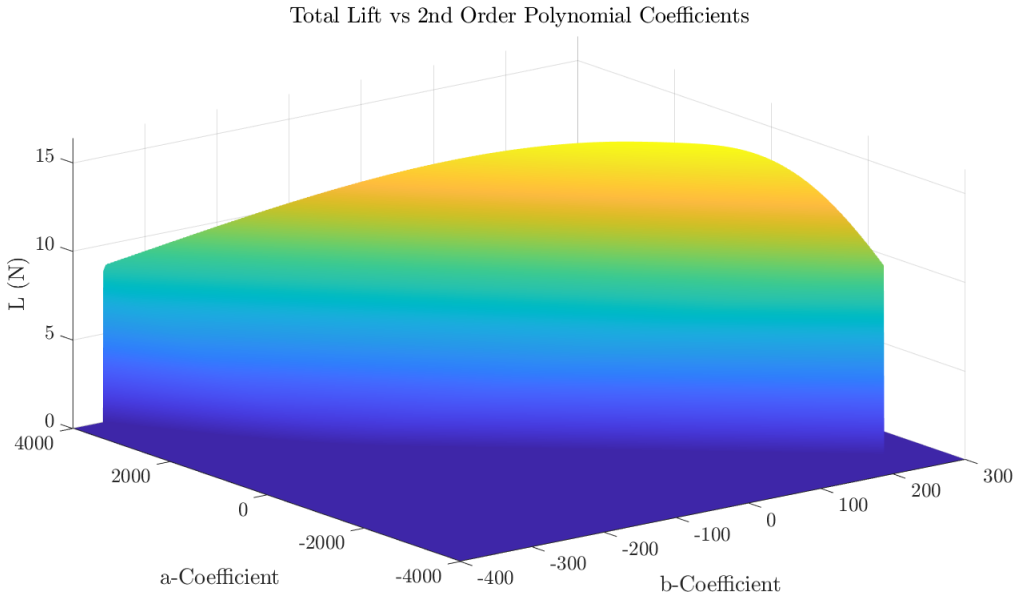

Figure 3.18: Contour of Total Rotor Lift Vs. 2^{nd} Order Polynomial Coefficients

$$\beta_{optim} = 742.3712y^2 - 106.5533y + 8 \tag{3.7}$$

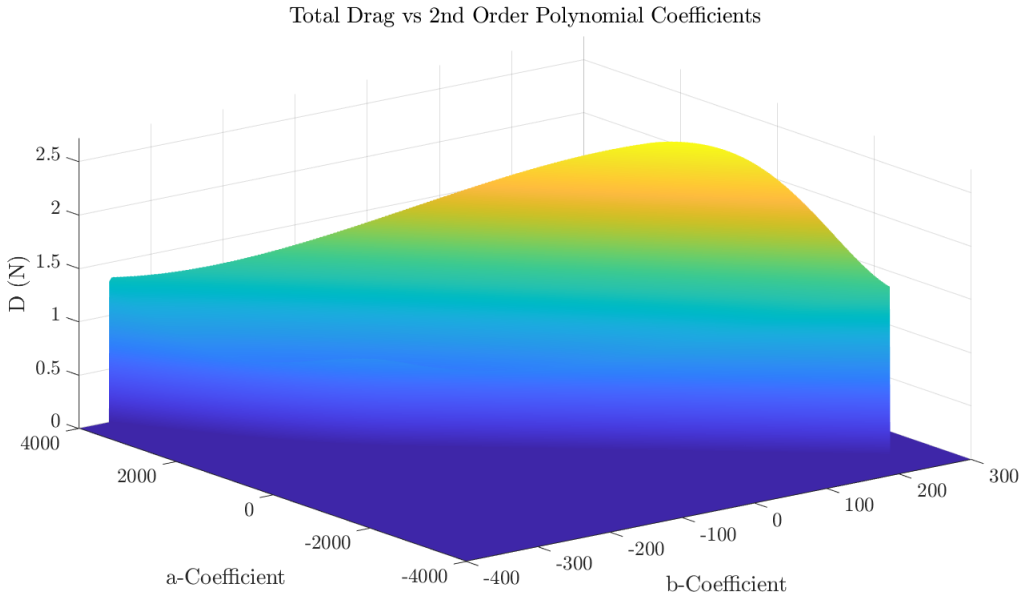

Figure 3.19: Contour of Total Rotor Drag Vs. 2^{nd} Order Polynomial Coefficients

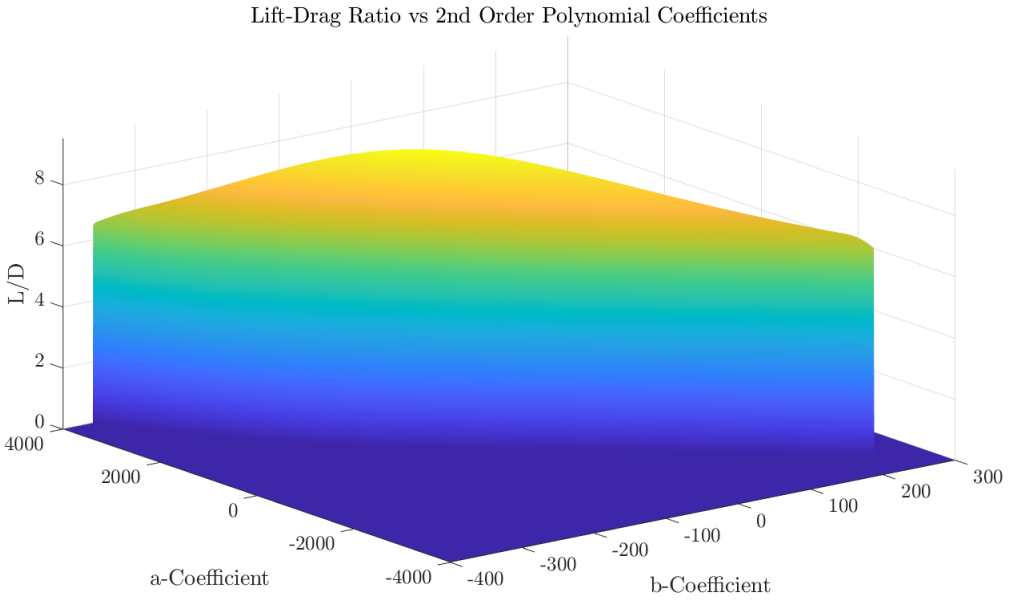

Figure 3.20: Contour of Total Lift to Drag Ratio Vs. 2^{nd} Order Polynomial Coefficients

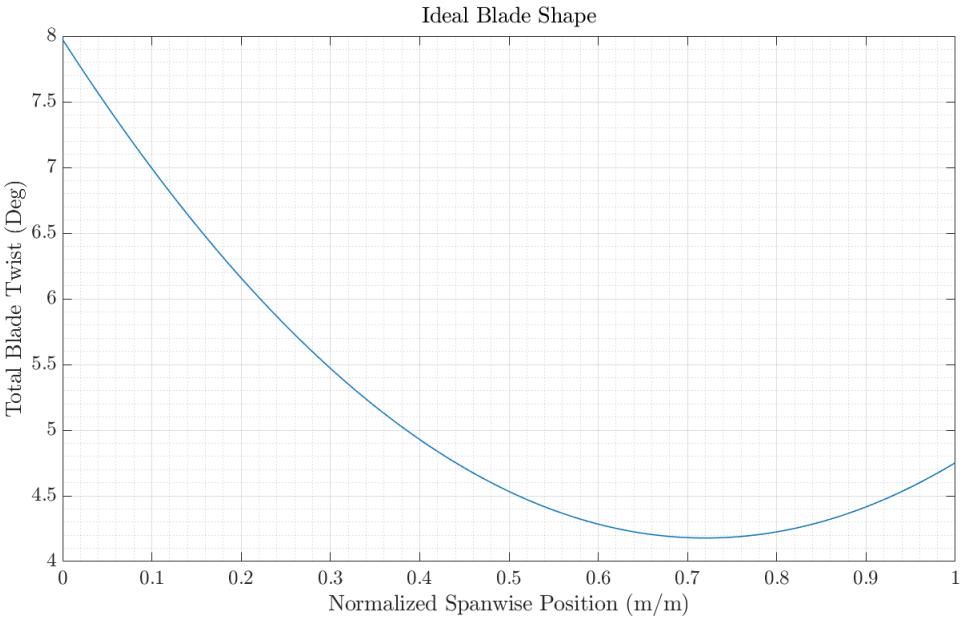

Figure 3.21: Optimized Blade Shape Based on Grid Search

Solving for the Quasi-Static Equilibrium Position

Using the rotor blade model derived in Ch. 2, we can solve for the quasi-static equilibrium configuration of the blade at non-zero angular velocities. MATLAB R2020a was utilized to solve the set of non-linear equations of motion. For this solution, $n = 200$ blade elements were utilized. From the previous section on numerical convergence, Fig. 3.15 and Fig. 3.16 show that this number of blade elements is well beyond the number of elements required to achieve convergence. This number of blade elements was chosen because it meets convergence criterion and allows for highly resolved and smooth blade-shape curves. Additionally, given the very low solution times shown in the previous section, time to solve was of little concern when choosing the blade elements count.

First, we define system constants such as spin speed, rotor blade geometry,

number of blade elements, material properties, and fluid properties. This information is used to calculate blade element characteristics such as surface area, center of mass y-location, dynamic pressure, moments of inertia, and torsional stiffness. To solve for these values, a function script is created in MATLAB and the user input information is imported into it. The function iterates over the number of blade elements and calculates properties for each element. Because these values are independent of the generalized coordinates and remain the same throughout the solution process, they can be pre-calculated and stored for later use, further improving the computational efficiency of the model. To solve for the equilibrium point, MATLAB's fsolve function is utilized to solve Eqs. 2.38-2.40 and Eq. 2.55. This intrinsic MATLAB function allows a user to solve a set of n non-linear equations with n unknown variables by leveraging a trust-region dogleg algorithm [1]. In order to achieve this, the constant blade characteristics and user-defined constants are imported into the equilibrium solver function along with an array of random θ_i values. The array of random displacement values acts as a set of initial values for MATLAB's fsolve function to begin iterating with. With this information, MATLAB is able to solve for the values of θ_i such that the set of n number of non-linear equations are balanced. This solution is output in the form of an array containing the angular displacement values of each blade element. With this information, the displacement array is then summed with the pre-twist array to obtain the total angular displacement of each blade element. Once the total angular displacement array is calculated the aerodynamic forces are then re-calculated outside of the fsolve function. The re-calculation of aerodynamics forces is performed by iterating over each blade element and solving for the aerodynamic forces using the updated total angular displacement. These forces can then be shown as a function of position along the span of the blade, or can be summed across the span to obtain the total aerodynamic force. This information

is then output via a number of plots. In order to obtain the information for multiple spin speeds, the fsolve function can be looped over for multiple values of $\dot{\gamma}$. The MATLAB scripts and functions used can be found in Appendix 5.

Comparison of Rigid and Flexible Blades

The optimal blade shape was then utilized to examine the effects of flexibility on blade performance. For the purposes of analyzing different blade shapes, this optimal blade shape will act as a target shape. Three different cases were examined: (1) a rigid blade with the same initial pre-twist as the optimal shape from Fig. 3.21, (2) a flexible blade with the same initial pre-twist as the rigid, and (3) a flexible blade with an initial pre-twist tuned such that the final blade shape is close to the optimized shape. The flexible blades are modeled by modulating the cross-sectional thickness linearly from 2mm to 0.4mm from root to tip. The properties of the different blade configurations can be seen in Table 3.5. Note that the modulus of rigidity, as well as the thickness of the rigid blade case are not included as these factors influence the flexibility of the blade and this configuration is assumed to be entirely rigid. The initial and final blade shapes can be seen in Fig. 3.22. Each of the cases were modeled with 200 blade elements. The total lift generation for the rigid, flexible, and flexible optimized blades can be seen in Fig. 3.23.

When comparing total lift generation of the rigid, flexible untuned, and flexible tuned blades, it is found that the lift performance of the flexible untuned blade degrades with increasing spin speed with a maximum decrease of 24.06%. However, the flexible tuned blade performs significantly better in total lift generation when compared to the flexible untuned, with only a 5.69% maximum decrease in total lift generation compared to the rigid blade at the highest spin speed. Additionally, from analysis of Fig. 3.23, it can be seen that at one particular angular velocity, roughly

Table 3.5: Rotor Blade and Model Properties.

Blade Type	Rotor Radius	Chord Length	Rigidity Modulus	Root Thickness	Tip Thickness	Fluid Density	Number of Blade Elements
Rigid Blade	0.1 m	25 mm	N/A	N/A	N/A	1.2754 Kg/m^3	200
Rigid Blade	0.1 m	25 mm	4.1x10$^9 Pa$	2x10$^{-3} m$	4x10$^{-4} m$	1.2754 Kg/m^3	200

$\dot{\gamma} = 1500$ rad/s, the total lift generation of the flexible tuned and rigid blades are very similar with only a 0.68% difference between the two total lift values. This would suggest that the flexible tuned blade shape is very similar to the shape of the rigid blade at that particular spin speed. To confirm this theory, a comparison of the target shape (rigid shape) and flexible tuned blade shape at $\dot{\gamma} = 1500$ rad/s is shown in 3.25. A calculation of the correlation coefficient between the two shapes shows a strong correlation exists between the two shapes with $R^2 = 0.9998$.

When comparing the rate of change of total lift and drag for each of the cases, it can be theorized that the flexible blades may be less responsive to a dynamic input, such as a sudden change in spin speed or a collision. However, this cannot be confirmed with the current model due the the quasi-static nature of the solver.

When analyzing Fig. 3.26, the tuned flexible blade shows a higher power requirement at lower angular velocities than both the rigid, and the untuned flexible blade. The power requirement curves for the rigid and flexible tuned blades appear to intersect at about $\dot{\gamma} = 1500$ rad/s with a 0.60% difference in total power required at this point. This finding falls in line with earlier statements regrading shape similarities

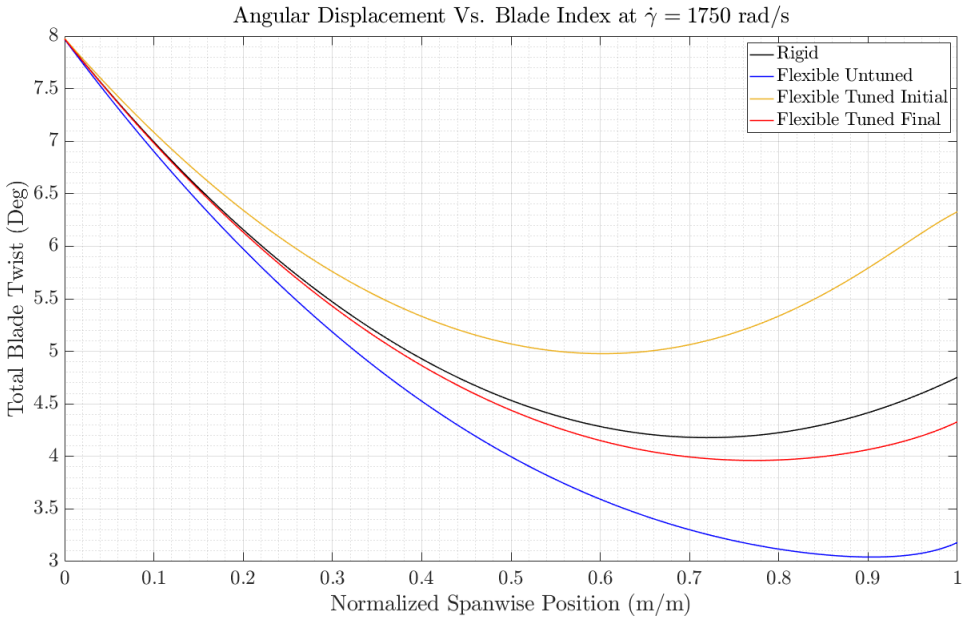

Figure 3.22: Comparison of Initial and Final Blade Shapes for Rigid, Flexible Untuned, and Flexible Tuned Blades.

at this angular velocity. Conversely, the untuned flexible blade and rigid blade power curves diverge with an increase in angular velocity with a maximum percent difference in power of 20.18%. This finding is unsurprising, as it shows that the flexible blade is twisting out of the fluid flow, generating less aerodynamic drag.

The lift to drag ratio, as well as lift to power ratio (at various angular velocities for each rotor blade configuration) were also explored. The results of this exploration are displayed in Fig. 3.27 and Fig. 3.28. Figure 3.27 shows that the lift to drag ratio of the tuned flexible blade is highly reduced by 3.51% at low angular velocities when compared to the rigid blade. However, when at $\dot{\gamma} = 1500$ rad/s, the difference in lift to drag ratio between the rigid and flexible tuned is reduced to 0.004%. This is a stark contrast to the untuned flexible blade witch at this point shows a reduction of 4.02% in lift to drag performance when compared to the rigid blade. The lift to power ratio of the tuned flexible blade is also adversely affected at low angular velocities.

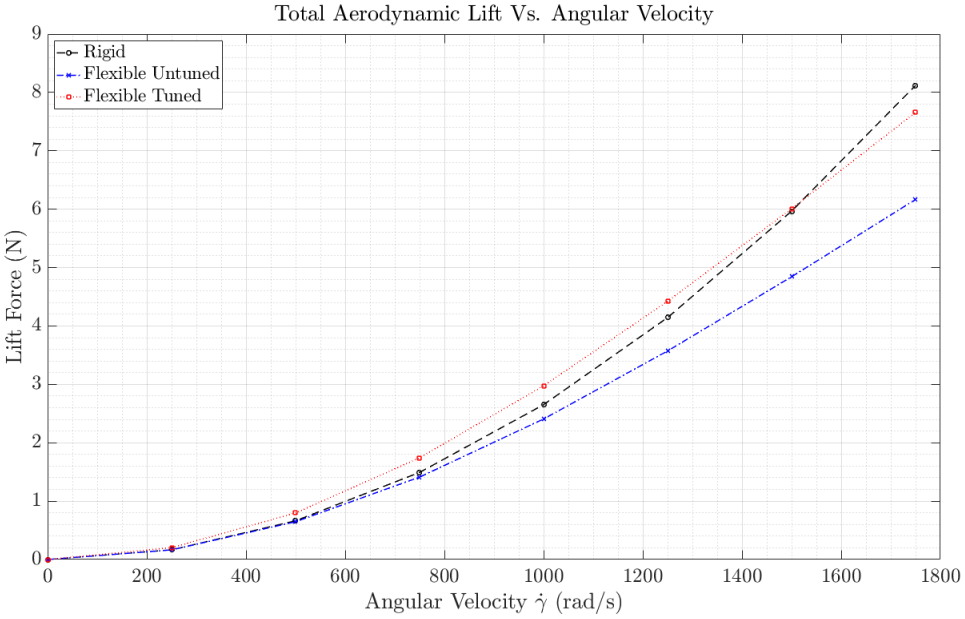

Figure 3.23: Comparison of Aerodynamic Lift Force for Rigid, Flexible Untuned, and Flexible Tuned Blades.

With a maximum reduction in lift to power ratio of 5.79% when compared to the rigid blade. All of this points to the conclusion that the tuned flexible blade is idealized for a specific operational envelope and that moving outside of that envelope results in a degradation of performance.

Overall this case study shows that this model can aid in the design of a lightweight, and consequently, flexible blade that is tuned for a specific operating envelope. However, it also demonstrates that a tuned flexible blade will have adverse performance characteristics when operating outside its operating envelope when compared to a rigid counterpart. These findings are a good demonstration of the ROM's value as a tool for parametric rotor blade design.

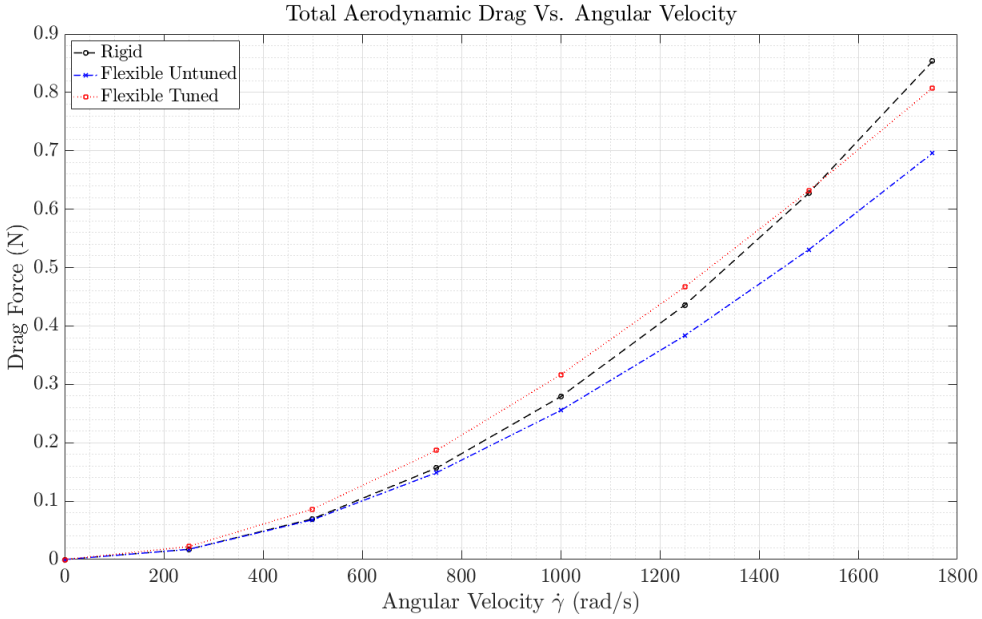

Figure 3.24: Comparison of Aerodynamic Drag Force for Rigid, Flexible Untuned, and Flexible Tuned Blades.

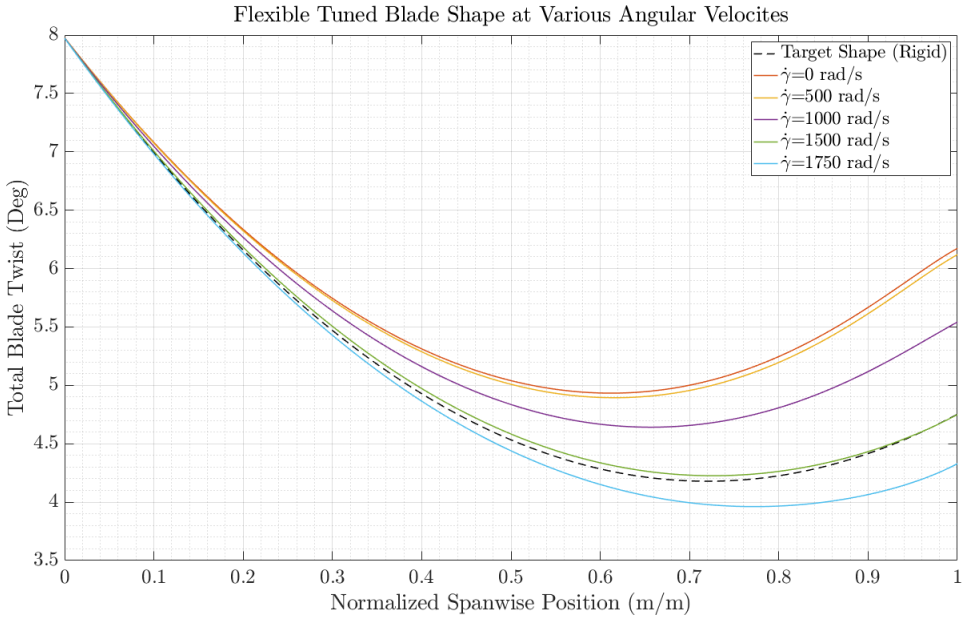

Figure 3.25: Target Shape and Flexible Tuned Blade Shapes at Various Angles of Attack.

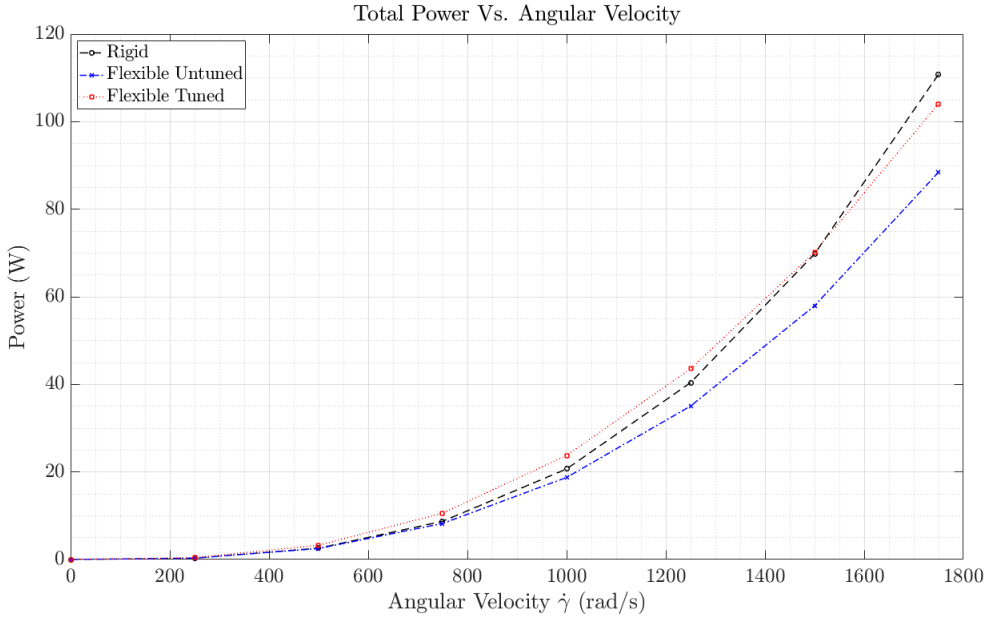

Figure 3.26: Comparison of Mechanical Power Requirements for Rigid, Flexible, and Flexible Optimized Blades.

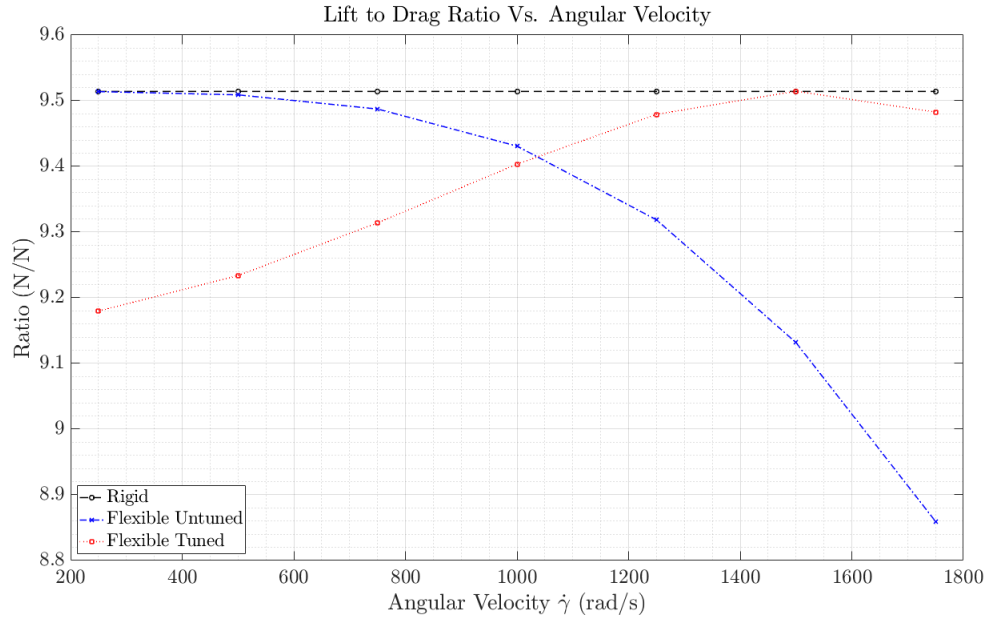

Figure 3.27: Comparison of Lift to Drag Ratio for Rigid, Flexible Untuned, and Flexible Tuned Blades.

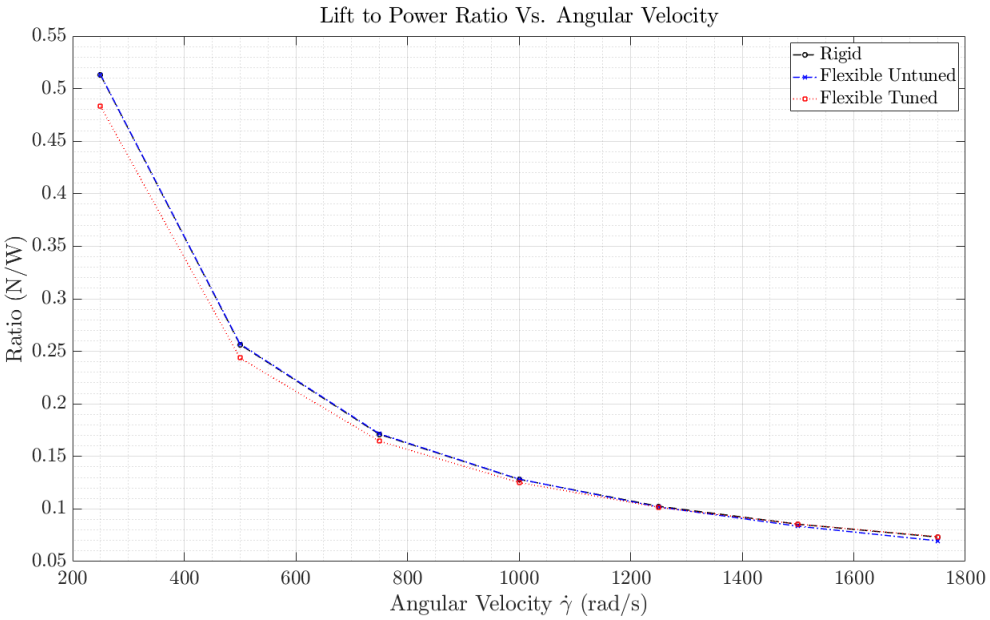

Figure 3.28: Comparison of Lift to Power Ratio for Rigid, Flexible Untuned, and Flexible Tuned Blades.

CONCLUSION

Through this research, a reduced order aeroelastic model of a torsionally flexible rotor blade was developed. The structural framework was developed via the Lagrangian formulation and subsequently coupled to a BET aerodynamic scheme. The set of nonlinear equations of motion was populated with pre-determined physical constraints and idealizations. These equations of motion were then tailored for an equilibrium position solution and solved by leveraging trust-region dogleg based algorithm within MathWorks MATLAB R2020a. With this model, blade flexibility has been shown to diminish aerodynamic performance, and that performance can be recovered by tailoring the pre-twist of a flexible rotor blade.

The reduced order FSI model presented in this work has shown some initial promise towards describing the deformation and aerodynamic forces of a UAV rotor blade and therefore benefits blade design. This allows a user to tailor blade geometry and predict the deformed state during defined operating conditions. The model presented is capable of producing a converged solution in as little as 0.016 seconds, showing a potential for parametric study in rotor blade design. It is shown that the deflection of a flexible blade can reduce the total aerodynamic lift from 18-25% when compared to a rigid blade with the same initial geometry. This model allows a user to tailor the initial pre-twist of the flexible rotor blade such that losses in lift are reduced to 0.68-5.7%. Furthermore, this research lays the groundwork for more advanced models capable of application towards a wider range of physical systems. This would inevitably add more complexity to the model. However, current solution times are small enough that this may be of little concern.

There are several ways in which this model could be advanced with future work. The most obvious are the elimination of idealizations and assumptions. Some

idealizations are believed to have the potential for larger effect on the outcome, while others constrain the model to specific applications. For example, it is believed that the neglecting of span-wise bending may play a large role in the outcome of the solution. For this reason, the incorporation of the accounting of span-wise bending within the structural and aerodynamic framework may prove to be a valuable addition to model capabilities. Likewise, building in the capability to prescribe additional axes of rotation would allow for application towards such things as insect flight, providing the model is moved to a non-quasi static solution scheme. A more advanced torsional stiffness model may also be implemented. Because the current torsional stiffness model is limited to only a rectangular cross section, adapting this model to accommodate for a wider range of non-uniform cross sections would allow for the study of more advanced rotor blade designs. Pairing all of these advancements together may produce a highly adaptable and powerful tool for engineers.

However, it should be noted that this framework has yet to be validated by physical experimentation and this would be one of the first steps in the future exploration of this model. Validation is an imperative step in numerical simulation and reduced order modeling and this research is no exception. This important step may reveal that model solutions are accurate and may negate the need for the elimination of idealizations. I believe that validation may be further explored by comparing the results to a high fidelity CFD and FEA coupled solver. Although this approach is no substitute for physical experimentation, it may act as a form of verification.

REFERENCES CITED

[1] Matlab fsolve information. https://www.mathworks.com/help/optim/ug/fsolve.html, . Accessed: 2021-04-08.

[2] Matlab stopwatch feature information. https://www.mathworks.com/help/matlab/ref/tic.html,, . Accessed: 2021-04-08.

[3] Photo of uav rotor. https://www.nicepng.com/ourpic/u2e6a9o0u2t4i1o0_delta-drone-rotor-blades-rotor-blade-for-drone/. Accessed: 2021-03-25.

[4] Webplotdigitizer. https://automeris.io/WebPlotDigitizer/. Accessed: 2021-04-07.

[5] John D. Anderson. *Computational Fluid Dynamics*. McGraw-Hill, 1995.

[6] John D. Anderson. *Fundamentals of Aerodynamics*. McGraw-Hill, 2011.

[7] Melanie Joyce Anderson, J. Sullivan, T. Horiuchi, S. Fuller, and T. Daniel. A bio-hybrid odor-guided autonomous palm-sized air vehicle. *Bioinspiration biomimetics*, 2020.

[8] Yuri Bazilevs, Victor Calo, Thomas Hughes, and Yongjie Zhang. Isogeometric fluid-structure interaction: Theory, algorithms, and computations. *Computational Mechanics*, 43:3–37, 12 2008. doi: 10.1007/s00466-008-0315-x.

[9] Samarth Bhasin, Ping Chen, Zhicun Wang, and Luciano Demasi. *Dynamic Nonlinear Aeroelastic Analysis of the Joined Wing Configuration*. doi: 10.2514/6.2012-1791. URL https://arc.aiaa.org/doi/abs/10.2514/6.2012-1791.

[10] Schmidt Richard J. Boresi, Arthur P. *Advanced Mechanics of Materials*. John Wiley Sons, Inc., 2003.

[11] Pierre-Jean Bristeau, Philippe Martin, Erwan Salaün, and Nicolas Petit. The role of propeller aerodynamics in the model of a quadrotor uav. In *2009 European Control Conference (ECC)*, pages 683–688, 2009. doi: 10.23919/ECC.2009.7074482.

[12] Maeng Hyo Cho and In Lee. Aeroelastic stability of hingeless rotor blade in hover using large deflection theory. *AIAA Journal*, 32(7):1472–1477, 1994. doi: 10.2514/3.12217. URL https://doi.org/10.2514/3.12217.

[13] Patricia Ventura Diaz and Steven Yoon. *High-Fidelity Computational Aerodynamics of Multi-Rotor Unmanned Aerial Vehicles*. doi: 10.2514/6.2018-1266. URL https://arc.aiaa.org/doi/abs/10.2514/6.2018-1266.

[14] V. Duggal, M. Sukhwani, K. Bipin, G. S. Reddy, and K. M. Krishna. Plantation monitoring and yield estimation using autonomous quadcopter for precision agriculture. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5121–5127, 2016. doi: 10.1109/ICRA.2016.7487716.

[15] Dominique Fleischmann, Simone Weber, and Mohammad M. Lone. *Fast Computational Aeroelastic Analysis of Helicopter Rotor Blades*. doi: 10.2514/6.2018-1044. URL https://arc.aiaa.org/doi/abs/10.2514/6.2018-1044.

[16] Jerry H. Ginsberg. *Advanced Engineering Dynamics*. Cambridge University Press, 1998.

[17] Jerry H. Ginsberg. *Engineering Dynamics*. Cambridge University Press, 2008.

[18] H. Glauert. *Airplane Propellers*, pages 169–360. Springer Berlin Heidelberg, Berlin, Heidelberg, 1935. ISBN 978-3-642-91487-4. doi: 10.1007/978-3-642-91487-4_3. URL https://doi.org/10.1007/978-3-642-91487-4_3.

[19] Dewey H. Hodges and Robert A. Ormiston. Stability of elastic bending and torsion of uniform cantilever rotor blades in hover with variable structural coupling. *National Aeronautics and Space Administration*, 1976.

[20] Lin Huang, You-Lin Xu, and Haili Liao. Nonlinear aerodynamic forces on thin flat plate: Numerical study. *Journal of Fluids and Structures*, 44: 182–194, 2014. ISSN 0889-9746. doi: https://doi.org/10.1016/j.jfluidstructs.2013.10.009. URL https://www.sciencedirect.com/science/article/pii/S0889974613002284.

[21] David V. Hutton. *Fundamentals of Finite Element Analysis*. McGraw-Hill, 2004.

[22] Hai Jiang, Yan Li, and Zhong Cheng. Relations of lift and drag coefficients of flow around flat plate. *Applied Mechanics and Materials*, 518:161–164, 02 2014. doi: 10.4028/www.scientific.net/AMM.518.161.

[23] David Johnson. *Advanced Structural Mechanics*. Thomas Telford, 2000.

[24] Wayne Johnson. *Rotorcraft Aeromechanics*. Cambridge University Press, 2013.

[25] Ryan k. Schwab, Heidi E. Reid, and Mark A. Jankauski. Reduced-order modeling and experimental studies of two-way coupled fluid-structure interaction in flapping wings. 2019.

[26] Oh Joon Kwon, Dewey H Hodges, and Lakshmi N. Sankar. Stability of hingeless rotors in hover using three-dimensional unsteady aerodynamics. *Journal of the American Helicopter Society*, 36(2):21–31, 1991. doi: https://doi.org/10.4050/JAHS.36.21. URL https://www.ingentaconnect.com/content/ahs/jahs/1991/00000036/00000002/art00002.

[27] Schilling M. Kumar J. Wurm F.-H. Laß, A. Rotor dynamic analysis of a tidal turbine considering fluid–structure interaction under shear flow and waves. *International Journal of Naval Architecture and Ocean Engineering*, 2019.

[28] Peng Lv, Sebastien Prothin, Fazila Mohd-Zawawi, Emmanuel Benard, Joseph Morlier, and Jean-Marc Moschetta. Performance improvement of small-scale rotors by passive blade twist control. *Journal of Fluids and Structures*, 55: 25–41, 2015. ISSN 0889-9746. doi: https://doi.org/10.1016/j.jfluidstructs.2015.01.008. URL https://www.sciencedirect.com/science/article/pii/S0889974615000122.

[29] Weipao Miao, Chun Li, Yuanbo Wang, Bin Xiang, Qingsong Liu, and Yunhe Deng. Study of adaptive blades in extreme environment using fluid–structure interaction method. *Journal of Fluids and Structures*, 91:102734, 2019. ISSN 0889-9746. doi: https://doi.org/10.1016/j.jfluidstructs.2019.102734. URL https://www.sciencedirect.com/science/article/pii/S0889974619302038.

[30] S. Mintchev, S. de Rivaz, and D. Floreano. Insect-inspired mechanical resilience for multicopters. *IEEE Robotics and Automation Letters*, 2(3):1248–1255, 2017. doi: 10.1109/LRA.2017.2658946.

[31] Andrew M. Mountcastle and Stacey A. Combes. Biomechanical strategies for mitigating collision damage in insect wings: structural design versus embedded elastic materials. *Journal of Experimental Biology*, 217(7):1108–1115, 2014. ISSN 0022-0949. doi: 10.1242/jeb.092916. URL https://jeb.biologists.org/content/217/7/1108.

[32] Thomas J. Mueller and Roth-Gibson. Aerodynamic measurements at low reynolds numbers for fixed wing micro-air vehicles. pages 161–164, 09 1999.

[33] Pauline Pounds and Robert Mahony. Small-scale aeroelastic rotor simulation, design and fabrication. 03 2012.

[34] Andres M. Pérez, Omar Lopez, and Svetlana V. Poroseva. *Free-Vortex Wake and CFD Simulation of a Small Rotor for a Quadcopter at Hover*. doi: 10.2514/6.2019-0597. URL https://arc.aiaa.org/doi/abs/10.2514/6.2019-0597.

[35] Vijayanandh R, Naveen Kumar K, Senthil Kumar M, Raj Kumar G, Naveen Kumar R, and Ahilla Bharathy L. Material optimization of high speed micro aerial vehicle using fsi simulation. *Procedia Computer Science*, 133:2–9, 2018. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2018.07.002. URL https://www.sciencedirect.com/science/article/pii/S1877050918309463. International Conference on Robotics and Smart Manufacturing (RoSMa2018).

[36] H Schmucker, F Flemming, and S Coulson. Two-way coupled fluid structure interaction simulation of a propeller turbine. *IOP Conference Series: Earth*

and Environmental Science, 12:012011, aug 2010. doi: 10.1088/1755-1315/12/1/012011. URL https://doi.org/10.1088/1755-1315/12/1/012011.

[37] Ryan Schwab, Erick Johnson, and Mark Jankauski. A novel fluid–structure interaction framework for flapping, flexible wings. *ASME. J. Vib. Acoust*, 2019.

[38] Pau Segui-Gasco, Yazan Al-Rihani, Hyo-Sang Shin, and Al Savvaris. A novel actuation concept for a multi rotor uav. *Journal of Intelligent Robotic Systems*, 74, 04 2014. doi: 10.1007/s10846-013-9987-3.

[39] H. Shahverdi, A.S. Nobari, M. Behbahani-Nejad, and H. Haddadpour. Aeroelastic analysis of helicopter rotor blade in hover using an efficient reduced-order aerodynamic model. *Journal of Fluids and Structures*, 25(8):1243–1257, 2009. ISSN 0889-9746. doi: https://doi.org/10.1016/j.jfluidstructs.2009.06.007. URL https://www.sciencedirect.com/science/article/pii/S0889974609000759.

[40] J. Sitaraman and B. Roget. Prediction of helicopter maneuver loads using a coupled cfd / csd analysis. 2008.

[41] J Sitaraman, A Datta, James Baeder, and Inderjit Chopra. Coupled cfd/csd prediction of rotor aerodynamic and structural dynamic loads for three critical flight conditions. 01 2005.

[42] Pedro J. Sousa, Francisco Barros, Paulo J. Tavares, and Pedro M.G.P. Moreira. Displacement analysis of rotating rc helicopter blade using coupled cfd-fea simulation and digital image correlation. *Procedia Structural Integrity*, 17:812–821, 2019. ISSN 2452-3216. doi: https://doi.org/10.1016/j.prostr.2019.08.108. URL https://www.sciencedirect.com/science/article/pii/S2452321619303142. 3rd International Conference on Structural Integrity, ICSI 2019, 2-5 September 2019, Funchal, Madeira, Portugal.

[43] Quanhua Sun and Iain Boyd. Flat-plate aerodynamics at very low reynolds number. *Journal of Fluid Mechanics*, 502:199 – 206, 03 2004. doi: 10.1017/S0022112003007717.

[44] QUANHUA SUN and IAIN D. BOYD. Flat-plate aerodynamics at very low reynolds number. *Journal of Fluid Mechanics*, 502:199–206, 2004. doi: 10.1017/S0022112003007717.

[45] Kenji Takizawa, Creighton Moorman, Samuel Wright, Timothy Spielman, and Tayfun E. Tezduyar. Fluid-structure interaction modeling and performance analysis of the orion spacecraft parachutes. *International Journal for Numerical Methods in Fluids*, 65(1-3):271–285, January 2011. ISSN 0271-2091. doi: 10.1002/fld.2348. Copyright: Copyright 2011 Elsevier B.V., All rights reserved.

[46] Tayfun Tezduyar, Sunil Sathe, Matthew Schwaab, Jason Pausewang, Jason Christopher, and Jason Crabtree. Fluid-structure interaction modeling of ringsail parachutes. *Computational Mechanics*, 43:133–142, 12 2008. doi: 10.1007/s00466-008-0260-8.

[47] Liangquan WANG, Guohua XU, and Yongjie SHI. High-resolution simulation for rotorcraft aerodynamics in hovering and vertical descending flight using a hybrid method. *Chinese Journal of Aeronautics*, 31(5):1053–1065, 2018. ISSN 1000-9361. doi: https://doi.org/10.1016/j.cja.2018.03.001. URL `https://www.sciencedirect.com/science/article/pii/S1000936118300761`.

[48] Justin Winslow, Hikaru Otsuka, Bharath Govindarajan, and Inderjit Chopra. Basic understanding of airfoil characteristics at low reynolds numbers (104–105). *Journal of Aircraft*, 55(3):1050–1061, 2018. doi: 10.2514/1.C034415. URL `https://doi.org/10.2514/1.C034415`.

[49] K. Yuan and Peretz Friedmann. Aeroelasticity and structural optimization of composite helicopter rotor blades with swept tips. 06 1995.

APPENDIX

EXAMPLE CODE

Main MATLAB script used to solve for the quasi-static equilibrium of the rotor blade.

```
%% Montana Marks Masters Thesis

clear; clc;
format compact

tic

% Set Latex Interpreter
set(groot,'defaulttextinterpreter','latex');
set(groot, 'defaultAxesTickLabelInterpreter','latex');
set(groot, 'defaultLegendInterpreter','latex');

%% User Inputs and Constants
L=0.1;                  % Radius of rotor
Nb=3000;                 % Number of blades
LE=@(y) -0.0125;        % Leading Edge Function
TE=@(y)  0.0125;        % Trailing Edge Function
g_dot=1000;             % Starting Angular Velocity of Blade rad/sec
    (1350)
G=4.1e9;                % Rigidity Modulus (Shear Modulus) (Pa) 4.1e9
    ;
Bm=3/1000;              % Total Mass of Rotor in Kg
rho=1.2754;             % Fluid Density (Kg/m^3) 1.2754
eta_Rt=12;               % Angle in deg of root pre twist
eta_Nd=13;              % Angle in deg of end pre twist
dg=250;

% Turn off/on Plotting
Plot_Optimization=1; % Turn on/off Optimization Countour Plots
    (1=on,0=off)
Plot_Blade_Shape=1;  % Turn on/off Blade Shape Plots (1=on,0=off)
Plot_Shape_Foces=1; % Turn on/off blade shape vs velocity Plots
    (1=on,0=off)

% Alter Thickness linearly along length
t_Rt=0.0033;
t_Nd=0.00048;
tb=linspace(t_Rt,t_Nd,Nb);

%% Lift and Drag Coefficient Curve Fit Inputs
alpha_max=12.5;      % Maximum Allowable Angle of Attack (Degrees)
```

```
% Lift Curve Fit Coefficients a_l*x^3+b_l*x^2+c_l*x+d_l
a_l=-5e-4;
b_l=4.5e-3;
c_l=0.1024;
d_l=-0.0023;
% Drag Curve Fit Coefficients a_d*x^3+b_d*x^2+c_d*x+d_d
a_d=-2e-4;
b_d=46e-4;
c_d=-17.1e-3;
d_d=0.0551;

%% Define Bounds of Optimization Coefficients
aLwr = -40;  % a Lower Bound
aUpr = 100;   % a Upper Bound
bLwr = -10;   % b Lower Bound
bUpr = 2;     % b Upper Bound
N = 500;     % Number of Divisions of Coefficients

%% Pre-Calculations
y   = linspace(0,L,2*Nb+1);          % Creat grid points
x   = linspace(0,L,Nb);              % Create x vector for plotting
    later
a   = linspace(aLwr,aUpr,N);         % Create a Coefficient Vector
b   = linspace(bLwr,bUpr,N);         % Create b Coefficient Vector
eta=linspace(eta_Rt,eta_Nd,Nb);      % Create values for pre-twist
eta=deg2rad(eta);                    % Switch pre-twist values from
    deg to rad
wb =L/Nb;                            % Width of each blade
m   = Bm/Nb;                         % Mass of each blade
%k   = (G*J)/wb;                      % Tosinal spring stiffness
eta_Rt = deg2rad(eta_Rt);            % Convert pre-twist to radians
alpha_max=deg2rad(alpha_max);        % Convert max AOA to radians
B_count=linspace(1,Nb,Nb);

%% Preallocate
I=eye(3);                % Identity Matrix
M=zeros(Nb);             % Mass Matrix

%% Compute Constant Blade Characteristics
[LEcom,TEcom,Xcp,lb,S,COMy,q_inf,Iyy,J,k,K_J]=
    Constant_Blade_Characteristics(Nb,LE,TE,rho,wb,y,g_dot,m,tb,G)
    ;

%% Run Optimization Routine to Obtain Optimum Blade Twist
    Polynomial
```

```
[a_index,b_index,L_Tot,D_Tot,L_D_r]=Blade_Twist_Optimizer(Nb,N,
    eta_Rt,q_inf,S,COMy,alpha_max,a_l,b_l,c_l,d_l,a_d,b_d,c_d,d_d,
    a,b);

%% Plot Contours and Blade Shape
if Plot_Optimization
    % Plot lift contour
    figure(1); clf(1)
    surf(b,a,L_Tot)
    shading interp
    title('Total Lift vs 2nd Order Polynomial Coefficients')
    ylabel('b-Coefficient')
    xlabel('a-Coefficient')
    zlabel('L')
    xlim([bLwr bUpr])
    ylim([aLwr aUpr])
    zlim([0 inf])
    set(gca,'Fontsize',25);

    % Plot drag contour
    figure(2); clf(2)
    surf(b,a,D_Tot)
    shading interp
    title('Total Drag vs 2nd Order Polynomial Coefficients')
    ylabel('b-Coefficient')
    xlabel('a-Coefficient')
    zlabel('D')
    xlim([bLwr bUpr])
    ylim([aLwr aUpr])
    zlim([0 inf])
    set(gca,'Fontsize',25);

    % Plot lift-drag ratio contour
    figure(3); clf(3)
    surf(b,a,L_D_r)
    shading interp
    title('Lift-Drag Ratio vs 2nd Order Polynomial Coefficients')
    ylabel('b-Coefficient')
    xlabel('a-Coefficient')
    zlabel('L/D')
    xlim([bLwr bUpr])
    ylim([aLwr aUpr])
    zlim([0 inf])
    set(gca,'Fontsize',25);
```

```
    % Plot blade shape
    figure(4); clf(4)
    plot(x,rad2deg(a(a_index)*x.^2+b(b_index)*x+eta_Rt))
    title('Optimized Blade Shape')
    xlabel('x-Location')
    ylabel('Angular Displacement (Deg)')
    set(gca,'Fontsize',25);
end

%% Solve For Deformed Blade

% Create a vector of random numbers to input into nonlinear
    solver
for i=1:Nb
    xmin=0.001;
    xmax=0.0015;
    T0g(i)=xmin+rand(1)*(xmax-xmin);
end
T0g(1)=0;

% Create function handle and input arguments for nonlinear solver
fhandle = @(T0)nonlinear_equilibrium_solver_WithAero_V2...
                (T0,Nb,eta,Iyy,g_dot,k,a_l,b_l,c_l,d_l,a_d,b_d,
                   c_d,d_d,S,Xcp,q_inf);
% Change fsolve function tolerances
options = optimoptions(@fsolve,'FunctionTolerance',1.0e-12,'
   MaxIterations',4000,'StepTolerance',1.0e-12,'
   MaxFunctionEvaluations',100000*Nb); % Changes algorithm for
   non-square system
% Run fsolve to obtain equilibrum points
[T0,fval]=fsolve(fhandle,T0g,options);

%% Plot Optimized Shape Vs. Deformed Shape
if Plot_Blade_Shape
    figure(5); clf(5)
    plot(x,rad2deg(a(a_index)*x.^2+b(b_index)*x+eta_Rt))
    hold on
    plot(x,rad2deg(T0+eta))
    hold on
    plot(x,rad2deg(eta));
    title('Optimized Shape Vs. Deformed Shape')
    xlabel('x-Location')
    ylabel('Angular Displacement (Deg)')
    legend({'Optimized Shape','Deformed Shape','Undeformed Shape'
       },'Location','northwest')
```

```
    set(gca,'Fontsize',25);
end

dfin=g_dot;
g_dot=0;
q=1;     % Index Value



%% Look at blade shape as angular velocity increases
% Drop into loop over g_dot
while g_dot<dfin+1

    %% Solve for Equilibrium Point

    % Create a vector of random numbers to input into nonlinear
        solver
    for i=1:Nb
    xmin=0.001;
    xmax=0.0015;
    T0g(i)=xmin+rand(1)*(xmax-xmin);
    end
    T0g(1)=0;

    % Create function handle and input arguments for nonlinear
        solver
    fhandle = @(T0)nonlinear_equilibrium_solver_WithAero_V2...
                    (T0,Nb,eta,Iyy,g_dot,k,a_l,b_l,c_l,d_l,a_d,
                        b_d,c_d,d_d,S,Xcp,q_inf);
    % Change fsolve function tolerances
    options = optimoptions(@fsolve,'FunctionTolerance',1.0e-12,'
        MaxIterations',4000,'StepTolerance',1.0e-12,'
        MaxFunctionEvaluations',100000*Nb); % Changes algorithm
        for non-square system
    % Run fsolve to obtain equilibrum points
    [T0,fval]=fsolve(fhandle,T0g,options);

    min(fval);
    max(fval);
    T0deg=rad2deg(T0);
    etadeg=rad2deg(eta);
    %% Calculate Steady State Aerodynamic Forces for each Blade
    for i=1:Nb
        q_inf(i)=0.5*rho*COMy(i)^2*g_dot^2;
```

```
        Cl(i)=a_l*(T0(i)+eta(i))^3+b_l*(T0(i)+eta(i))^2+c_l*(T0(i
            )+eta(i))+d_l;
        Cd(i)=a_d*(T0(i)+eta(i))^3+b_d*(T0(i)+eta(i))^2+c_d*(T0(i
            )+eta(i))+d_d;
        L(i)=Cl(i)*q_inf(i)*S(i);
        D(i)=Cd(i)*q_inf(i)*S(i);
        Torque(i)=D(i)*COMy(i);
    end
    % Sum Lift and Drag
    Total_Lift=sum(L);
    Total_Drag=sum(D);
    % Calculate Power
    Power=sum(Torque)*g_dot;
    Lift_Drag_Ratio=Total_Lift/Total_Drag;
    T0=zeros(1,Nb);


    % Save Blade Shapes For each g_dot
    Blade_Shape(q,:)=T0deg+etadeg;

    % Save Lift and Drag for each g_dot
    Blade_Lift(q,:)=L;
    Blade_Drag(q,:)=D;
    Tot_Lift(q)=sum(L,'all');
    Tot_Drag(q)=sum(D,'all');

    P(q)=Power;


    q=q+1;
    g_dot=g_dot+dg;
end

if Plot_Shape_Foces
    g_dotvec=linspace(0,dfin,(dfin/dg)+1);


    % Plot Blade Shapes for Each g_dot
    figure(6); clf(6)
    plot(B_count,Blade_Shape)
    legend({'$\dot{\gamma}$=0','$\dot{\gamma}$=250','$\dot{\gamma
        }$=500','$\dot{\gamma}$=750','$\dot{\gamma}$=1000'},'
        Location','northwest');
    set(gca,'fontsize',25)
    title('Angular␣Displacement␣Along␣Blade␣Length')
```

```
    xlabel('Blade Index')
    ylabel('Total Angular Displacement (Deg)')


    % Aerodynamic Lift
    figure(7); clf(7)
    plot(B_count, Blade_Lift)
    title('Steady State Aerodynamic Lift Vs. Blade Index')
    ylabel('Lift Force (N)')
    xlabel('Blade Index')
    legend({'$\dot{\gamma}$=0','$\dot{\gamma}$=250','$\dot{\gamma
       }$=500','$\dot{\gamma}$=750','$\dot{\gamma}$=1000'},'
       Location','northwest');
    set(gca,'fontsize',25)


    % Aerodynamic Drag
    figure(8); clf(8)
    plot(B_count, Blade_Drag)
    title('Steady State Aerodynamic Drag Vs. Blade Index')
    ylabel('Drag Force (N)')
    xlabel('Blade Index')
    legend({'$\dot{\gamma}$=0','$\dot{\gamma}$=250','$\dot{\gamma
       }$=500','$\dot{\gamma}$=750','$\dot{\gamma}$=1000'},'
       Location','northwest');
    set(gca,'fontsize',25)


    % Plot Natural Frequency
    figure(9); clf(9)
    plot(g_dotvec, Tot_Lift)
    hold on
    plot(g_dotvec, Tot_Drag)
    set(gca,'fontsize',25)
    title('Total Drag \& Lift Vs. Angular Velocity')
    legend({'Lift','Drag'},'Location','northwest')
    xlabel('Angular Velocity $\dot{\gamma}$ (rad/s)')
    ylabel('Force (N)')

end

timeElapsed = toc
```

Function used to calculate constant blade characteristics.

```
function [LEcom,TEcom,Xcp,lb,S,COMy,q_inf,Iyy,J,k,K_J]=
    Constant_Blade_Characteristics...
     (Nb,LE,TE,rho,wb,y,g_dot,m,tb,G)


%% Preallocate
Iyy=zeros(Nb,1);        % Moment of inertia
q_inf = zeros(Nb,1);    % Dynamic Pressure Matrix
COMy = zeros(Nb,1);     % Center of Mass Matrix
S = zeros(Nb,1);        % Surface Area Matrix
lb = zeros(Nb,1);       % Blade Length Matrix
LEcom = zeros(Nb,1);    % Leading Edge Matrix
TEcom = zeros(Nb,1);    % Trailing Edge Matrix
Xcp = zeros(Nb,1);      % Center of Pressure Matrix
K_J=zeros(Nb,1);        % Torsional Constant Coefficient
J=zeros(Nb,1);          % Torsional Constant
k=zeros(Nb,1);          % Element Stiffness

%% Calculate Constant Blade Characteristics
for i=1:Nb

    % Calculate LE and TE values at center point of each blade
    LEcom(i)=LE(y(2*i)); % Leading Edge x value
    TEcom(i)=TE(y(2*i)); % Trailing Edge x value

    % Center of Pressure of Each Blade
    Xcp(i)=LEcom(i)+((abs(LEcom(i))+abs(TEcom(i)))/4);

    % Calculate Length of Each Blade
    lb(i)=abs(LEcom(i))+abs(TEcom(i));

    % Surface Area of Each Blade
    S(i)=lb(i)*wb; % Calculate surface area of single blade

    % Calculate COM Locations
    COMy(i)=y(2*i); % y location

    % Calculate Dynamic Pressure for Each Blade
    q_inf(i)=0.5*rho*COMy(i)^2*g_dot^2;

    % Calculate Moments of Inertia
    Iyy(i)=(m/(3*(abs(TEcom(i))+abs(LEcom(i)))))*((TEcom(i))^3-(
        LEcom(i))^3);
```

```
    % Torsional Constant Calculations
    K_J(i)=(1/3)*(1-(192/(pi^5))*(tb(i)/lb(i))*((1/(1^5))*tanh
        ((1*pi*lb(i))/(2*tb(i)))+(1/(3^5))*tanh((3*pi*lb(i))/(2*tb
        (i)))));
    J(i)=K_J(i)*(lb(i))*(tb(i))^3;

    % Element Stiffness
    k(i)=(G*J(i))/wb;


end
```

Function used for calculating the optimal blade twist based on user inputs.

```
function [a_index,b_index,L_Tot,D_Tot,L_D_r]=
    Blade_Twist_Optimizer(Nb,N,eta_Rt,q_inf,S,COMy,alpha_max,a_l,
    b_l,c_l,d_l,a_d,b_d,c_d,d_d,a,b)

%% Preallocation of Matricies
L_Tot = zeros(N);
D_Tot = zeros(N);
L_D_r = zeros(N);
Lift_b = zeros(Nb,1);
Drag_b = zeros(Nb,1);

%% Perform Calculations for Lift and Drag for all Polynomial
    Coefficient Values
%Loop over a coefficient values
for i = 1:N
    % Loop over b coefficient values
    for j = 1:N
        % Loop over blades
        for k = 1:Nb
            % Calculate AOA for kth blade
            alpha=a(i)*COMy(k)^2+b(j)*COMy(k)+rad2deg(eta_Rt);

            % Check to make sure that the AOA is within limits
            if alpha > rad2deg(alpha_max)
                Lift_b(:)=0;
                Drag_b(:)=0;
                break
            elseif alpha < 0
                Lift_b(:)=0;
                Drag_b(:)=0;
                break
            else

            % Calculate Lift Coefficient
            L_Coef=a_l*(alpha)^3+b_l*(alpha)^2+c_l*(alpha)+d_l;

            % Calculate Drag Coefficient
            D_Coef=a_d*(alpha)^3+b_d*(alpha)^2+c_d*(alpha)+d_d;

            % Calculate Lift for blade
            Lift_b(k) = L_Coef*q_inf(k)*S(k);

            % Calculate Drag for blade
            Drag_b(k) = abs(D_Coef*q_inf(k)*S(k));
```

```
            end
        end
        % Sum lift and drag over entire blade and store value
        L_Tot(i,j) = sum(Lift_b);
        D_Tot(i,j) = sum(Drag_b);
        % Calculate Lift-Drag ratio and store value
        L_D_r(i,j) = L_Tot(i,j)/D_Tot(i,j);
        % Check Lift to Drag Ratio for NaN and Change to 0
        if isnan(L_D_r(i,j))
            L_D_r(i,j) = 0;
        end

    end
end

%% Determine Maximum Lift-Drag Ratio and Coresponding a and b
    coefficients
% Print out maximum lift-drag ratio
Maximum_Lift_to_Drag=max(L_D_r(:));
% Find indices of Max Lift-Drag Ratio
[a_index,b_index]=find(L_D_r == max(L_D_r(:)));
a_Optim=a(a_index);
b_Optim=b(b_index);
```

Function used to solve for the equilibrium state of the rotor blade.

```
function fval=nonlinear_equilibrium_solver_WithAero_V2(T0,Nb,eta,
    Iyy,g_dot,k,a_l,b_l,c_l,d_l,a_d,b_d,c_d,d_d,S,Xcp,q_inf)

% Preallocate
fval = zeros(Nb,1);

    %% Solve for Internal Blades
for i=2:Nb-1
    % Compute for Internal Blades
    fval(i) = g_dot^2*sin(T0(i)+eta(i))*cos(T0(i)+eta(i))*Iyy(i)
        +(k(i)*(T0(i)-T0(i-1))+k(i+1)*(T0(i)-T0(i+1))) -...
         q_inf(i)*Xcp(i)*((a_l*(T0(i)+eta(i))^3+b_l*(T0(i)+eta(i))
             ^2+c_l*(T0(i)+eta(i))+d_l)*cos(T0(i)+eta(i))+...
         (a_d*(T0(i)+eta(i))^3+b_d*(T0(i)+eta(i))^2+c_d*(T0(i)+eta
             (i))+d_d)*sin(T0(i)+eta(i)))*S(i);
end
    %% Solve for Blade 1
    % Compute for Blade 1
    fval(1 )= g_dot^2*sin(T0(1)+eta(1))*cos(T0(1)+eta(1))*Iyy(1)
        +(k(1)*T0(1)+k(2)*(T0(1)-T0(2))) -...
         q_inf(1)*Xcp(1)*((a_l*(T0(1)+eta(1))^3+b_l*(T0(1)+eta(1))
             ^2+c_l*(T0(1)+eta(1))+d_l)*cos(T0(1)+eta(1))+...
         (a_d*(T0(1)+eta(1))^3+b_d*(T0(1)+eta(1))^2+c_d*(T0(1)+eta
             (1))+d_d)*sin(T0(1)+eta(1)))*S(1);


   %% Solve for Blade Nb
   % Compute for Blade Nb
   fval(Nb)= g_dot^2*sin(T0(Nb)+eta(Nb))*cos(T0(Nb)+eta(Nb))*Iyy(
       Nb)+k(Nb)*(T0(Nb)-T0(Nb-1)) -...
         q_inf(Nb)*Xcp(Nb)*((a_l*(T0(Nb)+eta(Nb))^3+b_l*(T0(Nb)+
             eta(Nb))^2+c_l*(T0(Nb)+eta(Nb))+d_l)*cos(T0(Nb)+eta(Nb
             ))+...
         (a_d*(T0(Nb)+eta(Nb))^3+b_d*(T0(Nb)+eta(Nb))^2+c_d*(T0(Nb
             )+eta(Nb))+d_d)*sin(T0(Nb)+eta(Nb)))*S(Nb);
```