INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.
A multibody model simulating tilt-wing conversion

O'Heron, Patrick James, M.S.
The University of Arizona, 1991
A MULTIBODY MODEL SIMULATING TILT-WING CONVERSION

by

Patrick James O’Heron

A Thesis Submitted to the Faculty of the
DEPARTMENT OF AEROSPACE AND MECHANICAL ENGINEERING
In Partial Fulfillment of the Requirements
For the Degree of
MASTER OF SCIENCE
WITH A MAJOR IN MECHANICAL ENGINEERING
In the Graduate College
THE UNIVERSITY OF ARIZONA

1991
STATEMENT BY AUTHOR

This thesis has been submitted in partial fulfillment of requirements for an advanced degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

SIGNED: Parviz E. Nikravesh

APPROVAL BY THESIS DIRECTOR

This thesis has been approved on the date shown below:

Parviz E. Nikravesh
Professor of Aerospace and Mechanical Engineering

4-29-91 Date
# Table of Contents

List of Illustrations ........................................................................ 6  

Abstract .......................................................................................... 7  

1 Introduction ................................................................................... 8  
   1.1 Existing Rotor Analysis Packages ........................................... 10  
   1.2 Multibody Dynamics Packages .............................................. 11  
   1.3 Thesis Overview ..................................................................... 13  

2 Literature Review .......................................................................... 15  
   2.1 Review of the Multibody Dynamics Literature ....................... 15  
      2.1.1 Multibody Dynamics Formulations ................................. 15  
         2.1.1.1 Cartesian Coordinate Formulations ..................... 16  
         2.1.1.1.1 Implicit Formulations .................................. 16  
         2.1.1.1.2 Explicit Formulations .................................. 17  
         2.1.1.2 Relative Coordinate Formulation ...................... 19  
      2.1.2 Section Summary ......................................................... 21  
   2.2 Review of the Aerodynamics Modeling Literature ............... 21  
      2.2.1 Aerodynamic Environment .......................................... 21  
      2.2.1.1 Unsteadiness ....................................................... 22  
      2.2.1.2 Compressibility .................................................. 22  
      2.2.1.3 Stall ................................................................. 22  
      2.2.1.4 Yawed and Reversed Flow .................................. 22  
      2.2.2 Aerodynamic Modeling .............................................. 23  
         2.2.2.1 Linear Incompressible Unsteady Aerodynamics .... 24  
         2.2.2.2 Linear Compressible Unsteady Aerodynamics .... 26  
         2.2.2.3 Nonlinear Compressible Unsteady Aerodynamics .. 31  
      2.2.3 Section Summary ......................................................... 33  
   2.3 Review of the Induced Inflow Modeling Literature ............... 34  
      2.3.1 Momentum Theory Of Induced Inflow ......................... 34  
      2.3.2 Introduction of Dynamic Inflow Effects ...................... 37  
      2.3.3 Section Summary ....................................................... 38  
   2.4 Review of the Trim Control Modeling Literature ................. 39  
   2.5 Chapter Summary .............................................................. 41  

3 MBOSS Enhancements ................................................................... 42  
   3.1 Joint Coordinate Driving Constraints ................................... 42  
      3.1.1 Derivation ............................................................... 43  
      3.1.2 Numerical Aspects ................................................... 45  
      3.1.3 Implementation ....................................................... 48  
      3.1.4 Validation ............................................................. 50  
   3.2 Acceleration-Dependent Forces ............................................ 53  
      3.2.1 Derivation ............................................................... 53  
         3.2.1.1 Direct Method .................................................. 54
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.2.2.2 Effect of Collective Pitch Rate Variations</td>
<td>119</td>
</tr>
<tr>
<td>6.2.2.3 Effect of Moderate Shaft-Tilt</td>
<td>121</td>
</tr>
<tr>
<td>6.2.3 Tilt-Wing Application</td>
<td>124</td>
</tr>
<tr>
<td>6.3 Chapter Summary</td>
<td>126</td>
</tr>
<tr>
<td>7 Thrust Control System</td>
<td>128</td>
</tr>
<tr>
<td>7.1 Derivation</td>
<td>128</td>
</tr>
<tr>
<td>7.2 Implementation</td>
<td>130</td>
</tr>
<tr>
<td>7.3 Validation</td>
<td>131</td>
</tr>
<tr>
<td>7.3.1 Robustness Test</td>
<td>131</td>
</tr>
<tr>
<td>7.3.2 Altitude Control Test</td>
<td>133</td>
</tr>
<tr>
<td>7.4 Tilt-Wing Application</td>
<td>135</td>
</tr>
<tr>
<td>7.5 Chapter Summary</td>
<td>140</td>
</tr>
<tr>
<td>8 Tilt-Wing Conversion Results and Conclusions</td>
<td>141</td>
</tr>
<tr>
<td>8.1 Tilt-Wing Conversion Results</td>
<td>141</td>
</tr>
<tr>
<td>8.2 Conclusions</td>
<td>154</td>
</tr>
<tr>
<td>List of References</td>
<td>157</td>
</tr>
</tbody>
</table>
List of Illustrations

Schematic Representation of TRC and Tilt-Wing Conversion .......................... 9
Notation for the Incompressible Model ......................................................... 25
Force Resolution on a Thin Airfoil in Unsteady Flow ...................................... 30
Trailing Edge Separation .................................................................................. 33
Momentum Theory of Induced Inflow ............................................................... 35
Two Bodies Connected by a Revolute Joint ...................................................... 44
Simple Blade Model and Test Results ............................................................. 52
Model and Results for theAcceleration-Dependent Force Test ......................... 59
Typical Blade Model and Reduced Floating Model .......................................... 62
Predicted Flap, Lag and Pitch Moments ........................................................... 66
Tilt-Wing Model Schematic ............................................................................. 74
Blade Assembly 1 Schematic .......................................................................... 75
Model Test 1 Flap Response ............................................................................. 80
Model Test 2 Flap and Lag Response .............................................................. 81
Model Test 4 Commanded Pitch Angle ............................................................ 82
Model Test 4 Forward Response to Shaft-Tilt .................................................. 82
Thrust Response to Collective Pitch ............................................................... 93
Thrust and Flap Response to Moderate Pitch Rates ......................................... 95
Unsteady Thrust due to Increases in Pitch and Pitch Rate ............................... 102
Normal Force Due to Rapid Periodic Pitch Oscillation .................................... 104
Normal Force Due to Rapid Ramp Pitch Variation .......................................... 106
Nonlinear Normal Force and Moment Coefficients ....................................... 110
Inflow and Thrust Response to Pitch .............................................................. 118
Inflow and Thrust Resonse to Pitch Rate ........................................................ 120
Inflow and Thrust Response to Moderate Shaft Tilt ....................................... 122
Flap and Forward Speed Response to Moderate Shaft Tilt ............................ 123
Inflow and Thrust Response to Tilt-Wing Conversion .................................... 125
Flap Resonse to Tilt-Wing Conversion .......................................................... 126
Thrust and Pitch Results for controller test 1 ................................................. 132
Thrust, Pitch, and Altitude Results for Controller Test 2 ............................... 134
Pitch Response During Tilt-Wing Conversion ............................................... 136
Flap Response During Tilt-Wing Conversion ................................................... 137
Lag Response During Tilt-Wing Conversion .................................................... 138
Forward Speed Resonse During Tilt-Wing Conversion ................................... 139
Altitude .......................................................................................................... 143
Forward Flight Speed ..................................................................................... 144
Thrust During Tilt-Wing Conversion ................................................................ 146
Flap Response ............................................................................................... 147
Required Collective Pitch .............................................................................. 149
Required Tilt-Torque ...................................................................................... 150
Tilt Acceleration ............................................................................................. 152
Normal Force Coeffient During Conversion .................................................. 154
Abstract

A multibody model is presented which simulates the conversion process associated with tilt-wing aircraft. A multibody dynamics approach is used to derive the equations of motion for a tilting articulated rotor with flap-pitch-lag root geometry. An enhanced model is used for the near-wake aerodynamics and uniform dynamic-inflow is used for the far-wake aerodynamics. A thrust control system is computes the required trim settings. It is found that the controller can "fly" the model to a hover condition at a desired altitude, and can be used achieve desired thrust levels during conversion. It is noted that conventional blade twist is inadequate during conversion. It is observed that unsteady aerodynamics are important during conversion. Also nonlinear effects on the tilt-wing cause large variations in tilt-torque during conversion.
1 Introduction

The current work is dedicated to the analysis of vertical take off and landing aircraft which utilize rotors to generate thrust. Such aircraft will, henceforth, be simply called advanced rotorcraft. In particular, the current work is dedicated to the analysis of some of the transient dynamic phenomena which arise in such rotorcraft. In the current chapter the nature of the transient phenomena are discussed and the remainder of the thesis is outlined.

Several innovative advanced rotorcraft designs have been proposed as possible ways to achieve high speed forward flight. Two examples are the trail rotor convertiplane (TRC) and the tilt wing configurations. In each of these concepts the aircraft possesses two distinct operating conditions. The first is similar to that of a conventional helicopter: a main rotor system is used to generate the thrust needed for vertical take off and landing. The second operating condition is similar to that of conventional fixed wing aircraft: a jet engine or propeller system is used to generate the thrust necessary to achieve high speed forward flight difficult to attain by conventional rotorcraft. These two operating conditions are sometimes called helicopter and fixed-wing modes, respectively. The current work is dedicated to developing analysis tools which can model the transition between these two modes.

In the conversion process used by the TRC, the main rotor is tilted 90° aft, and the hub speed decelerated as the rotor system is folded and subsequently stopped. During the conversion and in fixed-wing mode, the required propulsive thrust is generated by a jet engine system. A schematic representation of a typical TRC conversion is depicted in figure 1 a).
The tilt-wing conversion is just the reverse of that used in the TRC concept: there are two main rotors, each attached to wings which will operate in tandem in the helicopter mode. In the conversion process the wings are tilted 90° forward, and the rotors ultimately act as propellers in fixed-wing mode, so that high speed forward flight can be achieved. In the TRC configuration the rotor system actually decreases the total thrust as it is tilted aft, so that the conversion must be accomplished quickly. In the tilt wing configuration, however,
the rotors provide propulsive thrust as they are tilted forward. A schematic representation of a tilt-wing conversion is depicted in figure 1 b). The tilt-wing configuration is currently under active research and development at McDonnell Douglas Helicopter Company, and hence, is the system to which the current work is primarily addressed.

Both the TRC and tilt-wing configurations offer sound and imaginative solutions to the urgent problem of developing vertical takeoff and landing aircraft capable of high speed forward flight. Advanced configurations present significant engineering challenges. One of the interesting problems facing the rotor dynamics community concerns the development of software tools capable of making credible analyses of the highly nonlinear dynamics governing the conversion process.

1.1 Existing Rotor Analysis Packages

Elliott and McConville [1] discuss the capabilities of the current generation of commercially available helicopter rotor analysis software. There are many dedicated rotor analysis programs, for example CAMRAD [2], designed for industrial application which have been used successfully to design rotors intended to power conventional helicopters. At the heart of these dedicated packages are analytically derived equations of motion which govern the rotor system dynamics. These equations are commonly based on assumptions including: constant flight conditions, small blade angles and angular rates, constant rotor speed, and standard root geometries. These assumptions are well justified when applied to conventional helicopter configurations and flight conditions, hence current generation rotor software packages have proven to be invaluable analysis tools.
The inherent assumptions found in dedicated rotor packages, however, preclude their applicability to the conversion process associated with the TRC, tilt-wing and other advanced configurations. In the future generation of rotorcraft, nonstandard hub geometries may be employed. Also hub accelerations, large blade rotations and rotational rates, and other nonlinear effects are expected to be significant. The current rotor analysis tools are, simply, not designed to account for these effects [1], and a new generation of rotor analysis software must be developed which will effectively model the conversion process.

As a solution to this very basic issue, the assumptions upon which current generation rotor analysis tools are based can be relaxed and the software redesigned. Unfortunately, the nonlinear differential equations which govern the motion of complex dynamical systems are tedious, if not practically impossible, to derive analytically; this fact completely justifies the assumptions made in conventional rotor analysis software. Moreover, these equations will have to be reformulated, and the software subsequently updated, in order to analyze new, as yet unforeseen, configurations. This approach is considered complex and inherently inflexible. Other technology should, therefore, be explored.

1.2 Multibody Dynamics Packages

Obviously, the need to formulate and solve the nonlinear equations of motion describing complex mechanical systems is not unique to the helicopter industry. In recognition of the fundamental nature of the problem, analysis tools have evolved which can accurately formulate and solve the equations of motion governing complex dynamical systems [3]. Such codes have found application in diverse fields, including the robotics, biomechanical, sports, rail and road transportation, and aerospace industries [3]. In fact, the
wide diversity of application attests to the modelling flexibility, accuracy, and expendability of these tools. These so-called multibody tools have been used extensively at the University of Arizona in satellite, [4], automotive, [5][6], and biomechanical, [7][8], and rail transportation, [9] applications. At McDonnell Douglas Helicopter Company a package called ADAMS Based Rotor Analysis (ABRA) [10] is currently under development.

Conceptually, multibody systems packages are not dissimilar to finite element software. A multibody system model is constructed from a library of elements including: joints, externally applied forces, springs, and dampers. The equations of motion are then assembled based on the known topology of the mechanical system, the inertial properties of the basic elements, position and type of the joints, the magnitude of the externally applied forces, spring constants, and damping coefficients. An assembly process is used to assemble the equations of motion using methods not dissimilar to those found in finite element software. Finally, the equations of motion are solved numerically and the computed data are submitted for post-processing.

Most multibody codes can handle planar and spatial mechanical systems of chain or tree structure with open or closed kinematic loops [3]. No a priori assumptions are made concerning the motion between the bodies. This latter property makes multibody dynamics packages applicable to advanced rotorcraft conversion. An articulated rotor system may be modelled as a system of bodies connected by joints, and the aerodynamic forces as user-defined applied forces. In the analysis one need not make the assumptions that are implicitly made in dedicated rotor software that were listed above.
There are several examples of general purpose multibody dynamics software packages available on the market, of which ADAMS, Automatic Dynamic Analysis of Mechanical Systems, and DADS, Dynamic Analysis and Design System, are perhaps the best known and most frequently used [3]. Other codes exist only in academe, for example MBOSS, Multibody System Simulation, [11] which is in use at the University of Arizona. Unfortunately, no known major, commercially-available code contains the helicopter rotor analysis capability required to analyze even conventional helicopter problems. All of these packages do have the flexibility necessary to include them [3][11].

1.3 Thesis Overview

The need for a new approach to rotorcraft engineering is quite clear. Highly nonlinear equations of motion must be formulated and solved accurately and efficiently, so that some of the new engineering challenges associated with advanced rotor configuration design may be breached. A very general and well tested approach will provide the most flexible and reliable analysis tool. Multibody dynamics methods have been successfully used in other industries, but do not possess the capabilities, for example aerodynamic and inflow models, needed to solve helicopter rotor dynamics problems. This raises one crucial issue: is it advantageous to redesign well-tested and commonly-used rotor analysis software, or should a rotor analysis package be developed to extend the capability of a well-tested multibody dynamics program? The present work is directed to the latter issue. In this thesis helicopter specific capabilities are incorporated in a general purpose multibody dynamics code. It is shown that a multibody dynamics package can, in fact, be readily extended to perform both
conventional and advanced rotorcraft analyses. The details regarding software design and validation are presented in the thesis. The general structure of the thesis is outlined in the following paragraph.

In chapter 2 a literature survey is conducted in which a review of multibody dynamics technology is presented. In chapter 3 several small modifications are made to a multibody code, and test results are presented. In chapter 4 two rotor models are designed to study the tilt-wing conversion process. In chapter 5 three aerodynamic models are outlined, their implementation discussed, and test results are given. In chapter 6 a simplified induced inflow model is outlined, its implementation discussed, and the full model is applied to tilt-wing conversion. In chapter 7 a thrust controller is presented, implemented and applied to the tilt-wing conversion process. In chapter 8 a full tilt-wing conversion simulation is presented, final conclusions are made regarding the application of multibody software, and some conclusions regarding tilt-wing conversion are drawn.
2 Literature Review

The literature review that is presented in the current chapter is intended to summarize some of the available multibody dynamics, aerodynamics, induced inflow, and trim control models. Specific models are selected as a direct consequence of the literature review, and each model is summarized in appropriate sections. Typically, the notation used in the references is slightly modified in order to avoid confusion. As this chapter contains a summary of two very diverse fields, the detailed derivations are best explained in the relevant references.

2.1 Review of the Multibody Dynamics Literature

In the current section three multibody dynamics formalisms are summarized, and one formulation is selected for use in rotor analysis.

2.1.1 Multibody Dynamics Formulations

There are several different techniques that are commonly used to construct the equations of motion governing general mechanical systems. These formulations are generally classified in terms of the type of coordinate system that is used to express the system configuration. Equations may either be cast in terms of a maximal set of Cartesian coordinates [12][13], in terms of a reduced set of relative coordinates [14], or in terms of a minimal set of coordinates [15]. For the open-loop systems encountered in the current work the last two formulations are equivalent, so only the first two cases need be considered.
2.1.1.1 Cartesian Coordinate Formulations

Cartesian coordinate formulations can be further classified into explicit and implicit methods. In both of these methods mixed systems of differential and algebraic equations (DAE) are derived, but the form in which the DAE sets are cast, either explicit or implicit, can lead to dramatic differences in the numerical methods used to solve them [16][17].

2.1.1.1.1 Implicit Formulations

In implicit formulations, which is found in ADAMS, the equations of motion are formulated as follows [12]:

2.1) \[ M(q, u, \dot{u}, F, t) = 0 \]

2.2) \[ N(u, q) = 0 \]

2.3) \[ \Phi(q, F, t) = 0 \]

Here the quantity \( q \) is a vector of generalized coordinates. The vector \( u \) is a vector of generalized speeds. The vector \( F \) is a vector of generalized forces. These equations represent 6 first-order dynamic equations for each body relating acceleration to force, 6 first-order kinematic equations for each body relating position to velocity, 1 algebraic equation for each scalar equation of holonomic constraint, 1 differential equation for each nonholonomic constraint, 1 algebraic equation for each force, and a set of user definable differential equations. These equations are cast in a form which maximizes system sparsity, hence improving the efficiency of the formulation [12]. The symbolic form of the equations shows that the formulation includes the derivative terms implicitly. Here the one dimensional matrices \( M, N, \) and \( \Phi \) are, respectively, a set of dynamical equations relating acceleration
and force, a set of kinematic equations relating position and velocity also including any nonholonomic constraints, and a set of algebraic equations describing holonomic constraints and forces.

Standard differential equation integration methods, for example Adams-Bashforth type integration methods, cannot be applied to DAE systems in implicit form [17]. Thus, more sophisticated numerical solution methods must be employed [12]. Because of this added algorithmic complexity, and because other methods exist, the implicit formulation need not be discussed further.

2.1.1.1.2 Explicit Formulations

In the implicit formulation discussed above, the algebraic equations of constraint are applied to the system in their exact form. Pezold [17] indicates that the system can be greatly simplified by differentiating the algebraic equations of constraint. Hence, a general system of holonomic constraint equations expressed in the canonical form:

2.4) \( \Phi(q,t) = 0 \)

can be differentiated twice to give:

2.5) \( \Phi(q,t) = \Phi_q \dot{q} + \Phi_{q} \ddot{q} + \dot{\Phi} \)

Here the matrix \( \Phi_q \) is called the Jacobian matrix of the constraints. If instead of applying the undifferentiated position-level constraint, one applies the twice differentiated acceleration-level constraint, then the system can be cast in an explicit form. In the numerical analysis literature [17] this general approach is commonly called reducing the index of the DAE system. The position-level constraint will, in principal at least, be satisfied as a result
of integrating the acceleration-level constraint. The system of equations can be cast in explicit form using the well-known Lagrange multiplier method [18] to give the system:

\[
\begin{bmatrix}
M & \Phi_q^T \\
\Phi_q & 0
\end{bmatrix} \begin{bmatrix} q \\ \Lambda \end{bmatrix} = \begin{bmatrix} F \\ \Gamma \end{bmatrix}
\]

Where the Lagrange multipliers \( \Lambda \) are directly related to the generalized reaction forces due to the constraints. The vector \( F \) is a vector of generalized applied forces which includes the centrifugal acceleration terms. The vector \( \Gamma \) contains the acceleration terms which arise from the holonomic constraint equations. This right-hand side acceleration term contains the Coriolis accelerations. This method has been used extensively in the past [19][20], and is the basic approach used in DADS [13]. The Jacobian matrix entries are dependent upon the specific joint type, and the entries for specific joint types are derived in [19].

It is important to note that the constraints are not directly satisfied in this formulation, but are satisfied as a result of numerical integration [17]. Errors, for example due to truncation error, will propagate during the simulation and may degrade the solution [19]. Position-level constraint violation is highly problem dependent [17]. The position-level constraint violations can be stabilized, at least in the sense of Liapunov, by constraint stabilization methods [21]. There are also methods to reduce this maximal coordinate formulation to a minimal set which are given attention in [19][22] and will not be discussed here.

The coordinate system used is based on a set of Cartesian coordinates, so that six second-order differential equations are required for each body, and \( 6N - k \) algebraic equations must be used to model the system. Where \( k \) is the total number of system degrees
of freedom and $N$ is the number of bodies in the system [19]. Standard numerical methods can be used to solve the DAE system as it is expressed in explicit form [17]. This latter fact is the principle advantage of the explicit over the implicit formulation.

2.1.1.2 Relative Coordinate Formulation

The Cartesian coordinate formulations generate the equations of motion very efficiently [19], but the large system of equations require large amounts of computation time to solve numerically. It is beneficial, therefore, to use the Cartesian formulation to derive the equations, and then to transform the equations into another basis of lower dimension for numerical integration: reduce the number of integration variables. A velocity transformation method, sometimes called the joint-coordinate method, may be used to reduce the number of equations which are originally cast in the explicit Cartesian form.

The absolute velocities of the bodies in the mechanical system can be written in terms of the relative velocities between the bodies in the system

\begin{equation}
\hat{q} = B\hat{\Theta}
\end{equation}

The vector $\Theta$ is a vector of relative velocities between adjacent bodies in the mechanical system which are determined by the joint type. The matrix $B$ is called the velocity transformation matrix [14], is sometimes called the orthogonal complement array [23], and in the robotics industry a portion of the matrix is confusingly called the Jacobian [24]. The Cartesian acceleration vector can be recovered from the velocity transform expression by simple differentiation

\begin{equation}
\ddot{q} = B\ddot{\Theta} + \dot{B}\hat{\Theta}
\end{equation}
Note that the velocity transformation matrix is a function of the system configuration. Now the matrix B can be shown [23] to be the orthogonal complement to the Jacobian matrix associated with the constraint equations appearing in the above section. So that:

2.9) \[ B^T \Phi_q^T = 0 \]

The explicit Cartesian formulation of the equations of motion can now be reduced to a smaller set by the transformation:

2.10) \[ B^T M (\dot{\Theta} + B \dot{\Theta}) + B^T \Phi_q^T \lambda = B^T F \]

or:

2.11) \[ B^T M B \dot{\Theta} = B^T (F - M \ddot{\Theta}) \]

As with the constraint Jacobian matrix, which arises in the explicit Cartesian formulation, the B matrix entries are determined by the type of joints used in the system. The B matrix entries for specific kinematic joints are given in [14] and are expanded upon in [25]. For systems with closed loops an extra set of algebraic equations may be included [15] in the system equations. In the joint coordinate method the equations are formulated easily and are transformed into a much smaller set of equations which require integration. Note also that the position-level constraints are enforced by the assembly of the B matrix, so that position-level constraint violation is not a problem [17].

The number of equations are reduced substantially via the velocity transformation. Consider a simple pendulum which only has one kinematic degree of freedom. In the joint coordinate formulation one must invert a $1 \times 1$ matrix and numerically integrate a single second order differential equation, but in the explicit Cartesian formulation one must invert an $11 \times 11$ matrix and integrate at least 6 second order differential equations. In [14] a
problem of significant complexity is solved using both the joint coordinate method and DADS. It was observed that the joint coordinate method was approximately 6 times faster, as measured in CPU time. The joint coordinate method was, therefore, selected for use in the rotor models.

2.1.2 Section Summary

In summary, the joint coordinate method allows for relative ease in both model description and solution. Standard and well-documented numerical methods may be used to solve the equations of motion generated by the joint coordinate method. The method is well documented and has been compared favorably to commercially available software.

2.2 Review of the Aerodynamics Modeling Literature

In the current section the aerodynamic effects which must be modeled in order to predict the rotor loads are summarized. Three models are presented which capture these effects in varying degrees.

2.2.1 Aerodynamic Environment

The aerodynamic environment to which helicopter rotors are subjected is notoriously complex. Shapiro [26] gives an excellent qualitative account of the aerodynamic environment, parts of which are summarized here. The discussion is composed of four components, each covering a specific aerodynamic effect: unsteadiness, compressibility, stall, yawed and reversed flow. Each component is intended to briefly summarize the relevant physical effects.
2.2.1.1 Unsteadiness

The angle of attack at the blade sections in forward flight is not constant, in fact, it can undergo quite wide variations from the advancing to the retreating side of the rotor disk. Sample contour plots of typical angle of attack variations in forward flight are given in [27]. The variations in angle of attack can cause hysteretic loading, and must be accounted for in the aerodynamic model [28].

2.2.1.2 Compressibility

Compressibility is generally quantified by the Mach number of the flow. The flow is termed incompressible if the Mach number is such that $M^2 < 1$. Compressibility effects the flow by causing an increase in the lift-curve slope, a dramatic reduction in the maximum attainable lift, and an increase in the drag at the blade section [29].

2.2.1.3 Stall

In the linear range, an airfoil has low drag, a lift coefficient which varies linearly with angle of attack, and a low moment coefficient [29]. As the angle of attack is increased to a critical level, on the order of $12^\circ$ [27], the airfoil is said to stall and its aerodynamic properties change dramatically. The drag coefficient increases substantially, the lift coefficient decreases with increased angle of attack, and large variations in pitching moment occur. These changes are caused by viscous effects associated with flow separation. Below stall, the flow is said to be attached and viscous effects are negligible.

2.2.1.4 Yawed and Reversed Flow

As a rotor blade sweeps the azimuth in forward flight, several azimuthal locations at which a significant component of the velocity at the blade section is directed along the
spanwise direction can be identified. This is termed a yawed flow. In neglecting this spanwise component one assumes that the velocity of the airfoil section acts in a plane perpendicular to the span. Shapiro [27] concludes, based on wind tunnel tests, that this assumption renders a good degree of approximation, and yawing may be neglected to a reasonable order of accuracy.

When in forward flight, the inboard regions of the retreating side of the disk may encounter reversed flow [27]. This is because there is a component of velocity directed opposite to the direction of blade rotation which is due to the forward velocity of the aircraft. This phenomenon is termed reversed flow, and limits the forward flight speed of conventional rotorcraft. Johnson [27] states that reversed flow is negligible for conventional helicopters operating at moderate advance ratio, so flow reversal is neglected in the current work.

2.2.2 Aerodynamic Modeling

A literature review was conducted in order to find reliable and well tested models which effectively capture the important effects outlined above. These models are also selected so that an incremental approach can be used in the design process. It is desirable to add unsteadiness, then compressibility, and then viscous effects. This is distinct from adding one unified model which captures all of the effects at once the results of which may, at first, be difficult to interpret.
2.2.2.1 Linear Incompressible Unsteady Aerodynamics

Models capturing linear, incompressible, unsteady aerodynamic effects have found wide application in industry, [28]. Essentially the same model is developed in both [27][30]. The derivation is quite involved and only the results are presented here. The notation employed here is taken directly from [30].

The lift and pitching moment per unit span acting on the blade section due to plunging and pitching oscillations are cast in the form:

\[ L = L_{NC} + L_C \]

where the subscript \( C \) indicates the circulatory component of loading and the subscript \( NC \) denotes noncirculatory or impulsive loading. The two components of the loading are given by:

\[ L_{NC} = \pi \rho b^2 (\dot{h} + U \dot{\alpha} - b a_p \ddot{\alpha}) \]

\[ L_C = 2\pi \rho U b C(k_f) [\dot{h} + U \alpha b (1/2 - a_p) \dot{\alpha}] \]

The pitching moment is also expressed in terms of circulatory and noncirculatory components:

\[ M = M_C + M_{NC} \]

where the components of moment are given by:

\[ M_{NC} = \pi \rho b^2 [b a_p \dot{h} - U b (1/2 - a_p) \dot{\alpha} - b^2 (1/8 + a_p^2) \ddot{\alpha}] \]

\[ M_C = 2\pi \rho U b^2 (a_p + 1/2) C(k_f) [\dot{h} + U \alpha b (1/2 - a_p) \dot{\alpha}] \]

Here the constant \( a_p \) is the location of the pitching axis with respect to the midchord, and the constant \( b \) is the semichord length, the quantity \( h \) is the vertical displacement of the
blade’s pitching axis, and $\alpha$ is the angle of attack. The parameter $C(k_f)$ is the Theodorson lift deficiency function. This function depends on the reduced frequency of the oscillation $k_f$ and is defined as a ratio of two Hankel functions [30]. This model is simplified and expressed in more convenient notation in chapter 5. The notation used in the above forms is given in the following figure.

![Notation for the Incompressible Model](image)

Figure 2.1 Notation for the Incompressible Model

The total unsteady lift and moment contributions consist of two distinct components: one circulatory in nature and the other noncirculatory. These two contributions arise from two entirely different sources. The circulatory terms arise from integration of the circulation. Hence, they satisfy the Kutta condition. The noncirculatory terms arise from impulsive loading resulting from the blade’s impact with the body of air, and satisfy key boundary conditions. The noncirculatory loads contain derivatives up to second order, and the constant coefficients of the second-order terms are commonly treated as apparent masses. The notion
of apparent mass arises from the fact that the speed of sound in an incompressible flow is infinite [30]. The concept of apparent mass will become more clear when discussing the interface between this model and MBOSS.

2.2.2.2 Linear Compressible Unsteady Aerodynamics

One method for accounting for compressible effects in the flow is to apply the Prandtl-Glauert compressibility correction [27]:

\[
a^\kappa_c = \frac{a^\kappa_c}{\sqrt{1 - M^2}}
\]

to the incompressible solution. Here the quantity \(a^\kappa\) is the lift-curve slope (\(a^\kappa = 2\pi\) for symmetric airfoils in an attached incompressible flow). The subscripts \(ic\) and \(c\) are intended to denote incompressible and compressible lift-curve slope. The term \(\sqrt{1 - M^2}\) is the well-known Prandtl-Glauert compressibility factor, heretoafter denoted as \(\beta_{pe}\).

In compressible unsteady flow, the noncirculatory loading no longer depends on the instantaneous accelerations and velocities of the wing, but instead depends on their time history [30]: the flow has a "memory effect" [30]. If the compressible solution is determined by simply applying the so-called Prandtl-Glauert rule to the incompressible solution, then the impulsive terms will depend on the instantaneous accelerations and velocities, hence the "memory effects" will not be captured. Thus, computation of the compressible loads via application of the Prandtl-Glauert rule is, strictly speaking, only applicable to the steady flow case [28].

In the literature, the indicial method has found much application for unsteady compressible rotor loads prediction. The early work in this field is reviewed in [30]. The indicial
response approach essentially involves estimating the lift and pitching moment responses due to, for example, step change in angle of attack, and then lift and moment due to arbitrary forcing found by superposition via Duhamel's integral. The indicial response functions can be found analytically for incompressible flow, but must be estimated for compressible flow [28].

Many of these indicial response models are derived in the frequency domain, for example [31] and are, therefore, not suitable for implementation in a time domain multibody dynamics package. In [32][33][34] the basis for a time domain model was established. Indicial response functions are validated for application in the continuous time domain in [35], and a model is expressed in a continuous time, state-space format in [28][36]. The model has been developed and tested in these references quite systematically and has found use in practical unsteady problems [37]. This model is particularly well-suited for multibody application as will be readily observed.

A very brief review of the model is in order, but the details are best discussed in the references. The response of the indicial normal force and pitching moment at the 1/4 chord to step change in angle of attack and pitch rate are approximated by:

\[
C_{Na}(s) = \left[ \frac{4}{M} \phi'_\alpha(s) + \frac{2\pi}{\beta_{pg}} \phi^C_\alpha(s) \right] \alpha
\]

\[
C_{Ma}(s) = \left[ -\frac{1}{M} \phi'_M(s) - \frac{2\pi}{\beta_{pg}} \phi^C_\alpha(s)(x_{ac}(M) - 0.25) \right] \alpha
\]

\[
C_{Na}(s) = \left[ -\frac{1}{M} \phi'_q(s) - \frac{\pi}{\beta_{pg}} \phi^C_q(s) \right] q
\]
The approximating indicial response functions \( \phi \) which are used to determine the lift and pitching moments are found in Lieshmann’s work, and need not be restated here. The superscripts \( I \) and \( C \) refer to impulsive, and circulatory components respectively. The independent variable \( s \) represents the dimensionless distance traveled by the airfoil in semichords \( s = Ut/b \). The indicial response functions are introduced here only for completeness. The reader is referred to the references for a detailed discussion of the notation and theory.

In the derivation, the indicial response functions are transformed to the Laplace domain and transfer functions are obtained which relate the angle of attack and pitch rate (the inputs) to the blade loads (the outputs). Methods commonly used in linear systems theory [41] may be employed to convert these transfer functions to equivalent state-space realizations. Hence, differential equations can be obtained which relate the output normal force and moment coefficients to the input angle of attack and pitch rate. A sample transfer function is derived and converted to state space format in [31]. The final state-space formulation is expressed in the familiar form:

\[
2.22) \quad C_{m_q}(s) = \left[ \frac{7}{12M} \phi^I_{m_q}(s) - \frac{\pi}{8\beta} \phi^C_{m_q}(s) \right] q
\]

The time variant coefficient matrices are of the form:

\[
2.23) \quad x^t = A^t x^t + B^t \begin{bmatrix} \alpha \\ q \end{bmatrix}
\]

\[
2.24) \quad \begin{bmatrix} C_N \\ C_M \end{bmatrix} = C^t x^t + D^t \begin{bmatrix} \alpha \\ q \end{bmatrix}
\]
2.25) \[ A^L = \text{diag} (a_{11}, a_{22}, a_{33}, a_{44}, a_{55}, a_{66}, a_{77}, a_{88}) \]

2.26) \[ B^L = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0.5 & 0.5 & 0 & 1 & 0 & 0 & 1 & 1 \end{pmatrix}^T \]

2.27) \[ C^L = \begin{pmatrix} c_{11} & c_{21} & c_{13} & c_{14} & 0 & 0 & 0 & 0 \\ c_{21} & c_{22} & 0 & 0 & c_{25} & c_{26} & c_{27} & c_{28} \end{pmatrix} \]

2.28) \[ D^L = \begin{pmatrix} \frac{4}{M} & \frac{1}{M} \\ -\frac{1}{M} & -\frac{7}{12M} \end{pmatrix} \]

Here the superscript L, which stands for Lieshmann, is intended to differentiate these matrices from other variables used below. The vector \( x^L \) is a vector of internal states. The entries and the specific form of the matrices are derived in [36] and elaborated upon chapter 5. The unsteady chordwise force, an important component of drag, can be recovered from the model:

2.29) \[ C_C(t) = 2\cdot\frac{\pi}{\beta_{ps}} \alpha^2_s(t) \]

where the effective angle of attack \( \alpha_s \) is given by:

2.30) \[ \alpha_s(t) = \beta_{ps}^2 \left( \frac{2U}{c} \right) (A_1^B b_1^B x_1^L + A_2^B b_2^B x_2^L) \]

The constants \( A_1^B, b_1^B, A_2^B \) and \( b_2^B \) are given in chapter 5, and may be found in [36]. The superscript \( B \) is intended to indicate that these constants are due to Beddoes. The quantity \( U \) is the freestream airspeed, and the quantity \( c \) is the airfoil chord length. The model captures
the unsteady pressure drag which may be recovered from the normal force, chord force, and angle of attack:

\[ C_D(t) = C_N(t) \sin \alpha(t) - C_C(t) \cos \alpha(t) \]

Here the functional dependency on time is explicitly shown. The forces and angles of attack are illustrated in the following figure.

![Figure 2.2 Force Resolution on a Thin Airfoil in Unsteady Flow](image)

The vector \( x^L \) contains eight internal states. The states are not unique, as the state space realizations of the transfer functions are not unique. The states, unfortunately, do not directly represent any physical quantities. They may, however, be indirectly associated with distinct portions of the loading. States 1 and 2 are associated with change in circulatory normal force and moment coefficients due to a step change in angle of attack. State 3 is associated with change in impulsive normal force due to change in angle of attack. State 4
is associated with change in impulsive normal force due to change in pitch rate. States 5 and 6 are associated with change in impulsive moment loading due to change in angle of attack. States 7 and 8 are associated with change in impulsive moment loading due to change in pitch rate.

Several comments are in order regarding the physical interpretation of the model. The fact that the model consists of a system of first-order differential equations implicitly captures the Bisplinghoff idea of "memory effect". Secondly, the apparent mass analogy, valid only for incompressible flow, is lost as the acceleration-dependent terms do not arise. The model is easily adapted to multibody dynamic formulations as the coefficient matrices $A^L, B^L, C^L,$ and $D^L$ may be computed with minimal programming and computational effort, as will be seen in chapter 5.

2.2.2.3 Nonlinear Compressible Unsteady Aerodynamics

The Lieshmann model can be directly extended to include dynamic stall effects [39]. This model, however, adds four more states to the overall aerodynamics model, and also adds considerable physical complexity to the system. The Lieshmann model does, however, allow for static trailing edge stall to be modelled algebraically. This model is based on Kirchhoff [40] theory, and is the basis for a portion of Lieshmann's dynamic stall model.

The physical process of trailing edge separation is well known and is perhaps best presented in [40]. As the angle of attack is increased past the linear range, a separation point, denoted here as $f_s$, appears at the trailing edge. As the angle of attack is further increased, the separation point moves forward as the adverse pressure gradient at the rear of the airfoil is increased. During further increases in angle of attack smaller and smaller
amounts of vorticity are shed into the wake until finally the lift reaches its maximum value.

The Leishmann model can be modified to include this effect without the introduction of additional dynamic states. This stall model is, however, a static model as unsteady stall effects are not included. Unsteady effects are retained in the attached portion of the solution, however. The basic idea is as follows. The normal force computed by the linear unsteady model is modified for static trailing edge separation by the relation:

\[ C_{N}^{nl} = C_{N}^{l}\left(1 + \sqrt{f_{s}}\right)^{2} \]

The superscript \( nl \) and \( l \) are intended to denote the linear and nonlinear normal force coefficient. The function \( f_{s} \) is an assumed dimensionless separation point location given by the semi-empirical function:

\[ f_{s} = \begin{cases} 
1 - 0.3\exp\left(\frac{\alpha - \alpha_{1}}{S_{1}}\right) & \text{if } \alpha \leq \alpha_{1} \\
0.04 + 0.66\exp\left(\frac{\alpha_{1} - \alpha}{S_{2}}\right) & \text{if } \alpha > \alpha_{1}
\end{cases} \]

The Mach dependent parameters \( S_{1} \) and \( S_{2} \) define an assumed separation characteristic, and the Mach dependent parameter \( \alpha_{1} \) defines a critical angle of attack at which static stall is initiated. Data for these parameters are given as a function of Mach number in [32]. The pitching moment may be determined by the semi-empirical form:

\[ \frac{C_{M}^{nl}}{C_{M}^{l}} = K_{0} + K_{1}(1 - f_{s}) + K_{2}\sin(\pi f_{s}^{m}) \]

where parameters \( K_{0}, K_{1}, K_{2} \) are again Mach and airfoil dependent. The quantity \( m \) can be taken to equal 2. Finally the chord force coefficient can be approximated by the form:
2.35) \[ C_C = \eta C_{p} \alpha^2 \sqrt{f_s} \]

where the parameter \( \eta \) compensates for the fact that an airfoil cannot achieve 100% of its potential leading edge suction. Typically \( \eta = 0.95 \). This approach is tested in [40] against experimental airfoil data and the results are excellent. The separation model is illustrated in the following figure.

![Figure 2.3 Trailing Edge Separation](image)

2.2.3 Section Summary

In summary, three models have been presented which together calculate the aerodynamic loading to which the rotor is subjected. The physical effects were determined to be important from a literature review, and three models were found so that these effects can be included in the multibody model incrementally. In chapter 5 the specific implementation of the aerodynamics models is given, and test results are presented which show that the
models are interfaced with the multibody dynamics model correctly.

2.3 Review of the Induced Inflow Modeling Literature

In rotor calculations the aerodynamic environment can be split into two distinct components [27]. A near wake component in which the blade section loads are calculated by the aerodynamics models presented above. A far wake solution is also needed to account for the wake of the rotor itself. In the current section, the induced velocity effects, or far wake effects, which must be modeled in order to predict the rotor loads are summarized. Two models, one static and the other dynamic, are presented based in a review of the literature of the field. First, however, the physical origin and implications of induced velocity are explained.

2.3.1 Momentum Theory Of Induced Inflow

A rudimentary physical treatment of induced inflow is given in both [26][27]. The basic concepts are briefly outlined in the current section for completeness. In the figure a rotor disk is placed within a control volume. The flow is assumed to cross the control volume walls only at positions 1 and 4. There must be a pressure differential across positions 2 and 3 as the rotor is generating thrust. The pressure at the upper surface must be lower than the pressure at the lower surface, and the pressures at position 1 and position 4 are equal. The existence of this pressure differential across the rotor disk, therefore, is attributed to an abrupt change in velocity of the flow somewhere in the disk. This change in velocity is termed the induced velocity or inflow.
Figure 2.4 Momentum Theory of Induced Inflow

Inflow has the effect of reducing the angle of attack at a given blade section, and can reduce the total power generated by the rotor by as much as 60% [27]. Since this effect is indeed significant, it must be accounted for. It is sometimes helpful to view the power generated by the rotor system as consisting of two components, one of which generates thrust and the other inflow.
The control volume explanation is helpful for a rudimentary explanation of induced inflow, but modern explanations are more complete. A recent work by Johnson [41] provides a much more detailed, qualitative discussion.

The momentum theory model can be used to derive the inflow velocity as a function of total rotor thrust. In this section all of the quantities are expressed in a dimensionless form, consistent with the conventions used in the literature. For the case of steady flight the induced velocity can be found [27] in terms of the forward flight speed and thrust, and cast in the following nonlinear form:

\[
\lambda_i = \frac{C_T}{2\sqrt{\mu^2 + \lambda^2}}
\]

where \( \mu \) is the dimensionless forward flight speed called the advance ratio, and \( \lambda \) is the dimensionless velocity of the flow through the rotor disk given by:

\[
\lambda = \mu \tan \alpha_d + \lambda_i
\]

The quantity \( \alpha_d \) is the rotor disk angle of attack. For the case of hover \( \mu = 0 \) the inflow expression reduces to:

\[
\lambda_i = \sqrt{\frac{C_T}{2}}
\]

The quantity \( C_T \) is the thrust coefficient. For the case of high speed forward flight, \( \mu \gg \lambda \), the inflow expression reduces to:

\[
\lambda_i = \frac{C_T}{2\mu}
\]
As can be readily observed, the inflow velocity and thrust are coupled nonlinearly. The thrust produces inflow, and inflow reduces the thrust. In this momentum theory approach, the inflow and the thrust must be computed simultaneously, and as the expression for inflow is not directly invertible, the inflow and thrust calculations must be done by iteration. A Newton-Raphson formula for the iteration process is given in [27]. The model is an important first-order approximation, and as such was selected for use with the multibody model. The model does not, however, account for dynamic effects.

2.3.2 Introduction of Dynamic Inflow Effects

The simple momentum theory inflow model is restrictive in two distinct ways: the inflow field is assumed to be uniform, and it is also assumed to be steady. In several comprehensive review papers, for example [42], the effects of nonuniformity and dynamics in the inflow field are discussed. In [43] a model is suggested which accounts for both unsteadiness and also nonuniformity. The model originated in [44] and was validated for the cases of hover and forward flight in [45]. The model is customarily called the Pitt-Peters model. It is, in its original form, cast in a linearized perturbation context, but was later extended in [46] into a nonlinear model suitable for use in time-marching simulations.

The basic idea behind the approach is as follows. The induced velocity is assumed to have the following functional variation over the rotor disk:

\[ \lambda_{\psi} = \lambda_{0} + \frac{r}{R} [\lambda_{S} \sin \psi + \lambda_{c} \cos \psi] \]

The three inflow components are commonly called the uniform and first sine and first cosine components. Inflow components are shown in [44] to approximately obey the following
differential equations:

\[
\begin{bmatrix}
\lambda_0 \\
\lambda_2 \\
\lambda_L \\
\lambda_L
\end{bmatrix} + \begin{bmatrix}
\lambda_0 \\
\lambda_2 \\
\lambda_L \\
\lambda_L
\end{bmatrix} = \begin{bmatrix}
C_T \\
C_L \\
C_M
\end{bmatrix}
\]

Here the derivatives are with respect to rotor time \(\Omega t\). The matrices \(M_3\) and \(L_3\) are called apparent mass and coupling matrices respectively, and their form will be elaborated upon in chapter 6. The term \(C_L\) and \(C_M\) are the rotor roll and pitch moment coefficients.

The advantage to this model is that the inflow is may be found from an integration of the differential equations. The angle of attack reduction due to induced inflow can be accounted for directly with the solution to the differential equations, and iteration on the rotor loads is not needed. There is, however, a need to iterate to find a momentum theory induced velocity term which appears in the expressions, but this iteration process may be accomplished once the total thrust is known, and not during the thrust prediction.

2.3.3 Section Summary

In summary, inflow effects must be included in the rotor models so that a reasonable prediction of rotor thrust can be made. Two models are to be used: one model assumes that the inflow field is static in time, and the other allows unsteady effects to be modelled. Both of these models have been used in the past, they are well tested, and they have sound foundation in theory. The dynamic inflow model is more easily implemented in multibody dynamics packages than the static model is, but either, in principal at least, can be included.
2.4 Review of the Trim Control Modeling Literature

In steady flight, the aircraft is in a state of kinetostatic equilibrium: all forces and moments must be balanced. This condition is known as trim. The pitch, roll, and yaw moments generated by the rotor system must therefore be eliminated. The yaw moment is eliminated by a tail rotor or NOTAR system in a conventional helicopter, and will be eliminated by the other main rotor in the tilt-wing configuration. The pitch and roll moments are commonly eliminated by the application of cyclic pitch variations. Small variations in blade pitch are well known to have dramatic effects on both propulsive thrust and the pitch and roll moments. The pitch and roll moments can, therefore, be removed by appropriate variations in blade pitch as the blade sweeps the azimuth. The amplitudes of the pitch oscillations are commonly called longitudinal and lateral cyclic pitch, and the average value of pitch over one complete rotation is called the collective pitch. The blade pitch, as generated by a swash-plate control system, can be expressed in terms of the collective and cyclic components [27]:

\[ \theta = \theta_0 + \theta_\psi \sin \psi + \theta_\phi \cos \psi \]

\( \theta_0 \) is called the collective pitch and represents the average value of pitch over an entire rotor revolution. The \( \theta_\psi \) is called longitudinal and \( \theta_\phi \) the lateral cyclic pitch components. The blade section angles of attack depend upon the blade pitch, so that in order to achieve a state of trimmed flight, the collective and cyclic components of pitch must be known. The required blade pitch is, unfortunately, not known a priori, so the required pitch components must be determined during the calculation.
Several methods have been proposed to determine the required pitch components. In [46] a brief review of the methods is given. Some researchers use iterative methods iteration to determine them over a rotor revolution. Other investigators have used periodic shooting methods to find them. A different approach involves the use of a control system which is designed to "fly" the rotor to trim during the calculation. The latter method is by far the best for implementation in multibody models as the internal equation solution methods are not directly adaptable to either iterative or periodic shooting methods.

The controller was initially proposed in [46] and was validated for hover. In a later work [47] it was modified for performance in low and moderate speed forward flight and good results were obtained. In [47] the controller is cast in the form:

\[
2.43) \quad T \begin{bmatrix} \dot{\theta}_0^p \\ \dot{\theta}_r^p \\ \dot{\theta}_c^p \\ \dot{\theta}_r^c \\ \dot{\theta}_c^c \end{bmatrix} + \begin{bmatrix} \dot{\theta}_0^p \\ \dot{\theta}_r^p \\ \dot{\theta}_c^p \\ \dot{\theta}_r^c \\ \dot{\theta}_c^c \end{bmatrix} = KA_p \begin{bmatrix} \Delta C_T \\ \Delta C_L \\ \Delta C_M \end{bmatrix}
\]

Again the derivatives are with respect to rotor time. The matrices T, K are appropriate time constants and gains, and the matrix A_p is a coupling matrix. These matrices will be expanded upon in chapter 7. The errors in thrust, pitching and roll moments are scaled by appropriate gains and used to determine the second derivatives of collective and cyclic pitch. The time constants are used to filter out oscillatory components in the pitch response. This simple feedback control is directly adaptable to multibody modelling methods, and the first and second derivatives of pitch can be appended to the system vector and integrated in time.
2.5 Chapter Summary

Following a literature review, a multibody dynamic formulation has been selected, three aerodynamic models, two inflow models, and one method for calculating trim settings during flight have also been chosen. The models and methods all have been well-tested, have a sound foundation in theory, and have proven to capture the desired effects well in application.
3 MBOSS Enhancements

The multibody dynamics code at the University of Arizona, MBOSS, was designed to simulate the motion of arbitrary mechanical systems using the joint coordinate method, which was outlined in chapter 2. MBOSS has been well tested and has been successfully used in the past. There were, however, three capabilities, important to the rotor analyst, that were not present in the original version of the software: the ability to apply driving constraints to specific joint coordinates, the ability to calculate reaction forces, and moments, due to kinematic joints, and the ability to apply acceleration-dependent forces to the model. In this chapter, algorithms are presented which implement these capabilities. The formulations are discussed in general terms as they are to become intrinsic MBOSS capabilities. Their specific application will be discussed in relevant chapters below.

3.1 Joint Coordinate Driving Constraints

In the rotor simulations which are outlined in the next chapter, it is desirable to specify the blade pitch, shaft tilt angle, and hub speed as explicit functions of time. Each of these quantities is associated with one of the joint coordinates used by MBOSS to describe the motion of the rotor system. A method must be constructed to force specific joint coordinates to assume some time dependent functional form.

Clearly, the configuration of a mechanical system is solely determined by the resultant of the generalized forces acting on it. If, for example, a simulation requires that the hub speed of the rotor remain constant, then the resultant generalized force associated with the hub speed must be identically equal to zero throughout the simulation. The resultant of the externally applied generalized forces associated with a given joint coordinate will, typically,
not be zero, so that an additional constraint force must be applied. The value of the required
constraint force is not known a priori and hence it must be calculated during the simulation.
A method, known as the method of driving constraints [19] may be used to determine the
required constraint force. This method may be used to systematically determine the con­
straint forces required to achieve commanded joint coordinate values. The approach is
discussed in this section. The derivation is quite general, so that the method may be applied
to any of the joint coordinates describing the system. The results of an example test problem
are presented, some comments concerning computational aspects of the method are made,
finally an algorithm implementing the method is outlined and validated.

3.1.1 Derivation

In figure 3.1 two bodies, i and j, are connected by a joint, in this case a revolute joint.
The joint coordinate between the bodies is \( q_t \), and the relative axis of rotation is the joint
axis a-a. Assume that body j is to rotate relative to body i in some desired manner. Thus,
the joint coordinate, \( q_t \), is to assume the functional form:

\[
q_t = \Phi_i(\Theta, t)
\]

Equation 3.1 is termed a rheonomic-holonomic, equation of constraint [18]. The term
rheonomic refers to the fact that the constraint is an explicit function of time and the term
holonomic describes the fact that the constraint is imposed on the configuration of the system.
Such configuration constraints are algebraic in nature. Nonholonomic constraints, by con­
trast, are cast in terms of differential equations. The constraint equation is coupled to the
system equations:

\[
B^T M \ddot{\Theta} = B^T (F - MB\dot{\Theta})
\]
The joint coordinate constraint equation may be differentiated twice with respect to time as was discussed in chapter 2 in connection with the explicit Cartesian multibody dynamics formulation. The acceleration-level constraint can be cast in the canonical form:

\[ \Phi_\theta\ddot{\theta} = \Gamma(\theta, \dot{\theta}, t) \]

Here the generalized coordinates are assumed to be the joint coordinates describing the system. Now the acceleration level constraint can be appended to the system equations to give the form seen in chapter 2. Recall that this general form arose in connection with the explicit Cartesian coordinate multibody dynamics formulation:
The term \( \Lambda \) is a vector of generalized constraint forces associated with the constraint. This term is directly analogous to the generalized constraint forces which appear in the explicit Cartesian formulation which was given in chapter 2. Procedures for constructing the Jacobian matrix for such constraints are outlined in [19]. If \( N \) is the total number of differential equations, \( k \) the number of system degrees of freedom, and \( m \) the total number of joint coordinate constraints, the system represents a mixed set of \( N \) differential and \( m + (N - k) \) algebraic equations in terms of the joint accelerations and the generalized constraint forces. The joint accelerations, and constraint forces are calculated by simply inverting the explicit system of differential and algebraic equations.

In the above discussion the constraints have been explicitly applied to the joint coordinate values, in some applications, it may be more convenient to constrain the joint velocities:

\[
3.5) \quad q_i = \Phi(\Theta, t)
\]

This constraint may be differentiated once with respect to time to obtain the acceleration-level constraint. The acceleration-level constraint may be directly appended to the differential equations of motion to form equation 3.6.

### 3.1.2 Numerical Aspects

Several comments regarding numerical and mathematical aspects of these joint coordinate driving constraints are in order. The algebraic equations of constraint must be
sufficiently differentiable to derive the acceleration level constraint. Higher differentiability may be desirable from a numerical point of view as discontinuous joint accelerations may cause the integration algorithm to fail.

One method to insure sufficient differentiability is to use polynomials for the constraint equations. Fourth order polynomials are sufficient to impose five conditions on the joint coordinate or joint velocity which is to be constrained. A sufficient set of conditions for a fourth order joint coordinate driving constraint is:

\[
\begin{bmatrix}
\theta(t_0) \\
\dot{\theta}(t_0) \\
\ddot{\theta}(t_0) \\
\theta(t_f) \\
\dot{\theta}(t_f)
\end{bmatrix}
= \begin{bmatrix}
\theta_0 \\
\dot{\theta}_0 \\
\ddot{\theta}_0 \\
\theta_f \\
\dot{\theta}_f
\end{bmatrix}
\]

The subscripts here refer to the starting time at which the constraint is turned on, and the time at which it is turned off. Higher order interpolation schemes are necessary if more conditions are required.

When using polynomial constraints it is actually necessary to compute the coefficients of the polynomial. This problem is known to potentially become ill-conditioned [48]. The use of polynomials of order higher than four is, therefore, not recommended. Should more than five conditions be required cubic spline interpolation is preferable [48]. Note that in the current implementation the acceleration at the time at which the constraint is turned off is, in general, discontinuous. This can be altered by going to fifth order polynomials should the discontinuity become a problem. Fourth order polynomials were found to perform well in practice, as will be seen in subsequent chapters, so that higher order polynomials are
apparently not needed.

Fourth order polynomial constraints are sufficient to describe the joint coordinate
constraints that are required in the rotor models. If the polynomial is well-conditioned, then
it is very simple to compute the second derivatives of the constraint equations. The poly-
nomial constraint equations can be constructed in MBOSS quite systematically. Constraint
equations are cast in the form:

\[ \theta^0(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 \]

Now the end-points of the polynomial are defined to be the values of time at which the
constraint is to be turned on, and then turned off. The constraints are assumed to be once
differentiable during the whole simulation, that is the polynomial values and first derivatives
are continuous. The polynomial is assumed to have specified function values and first
derivative values at the end points, and zero second derivative at the first boundary condition.
These boundary conditions are to be input by the user at run time. Once these boundary
conditions are known, the polynomial coefficients may be found from the expression:

\[
\begin{bmatrix}
1 & t_0 & t_0^2 & t_0^3 & t_0^4 \\
0 & 1 & 2t_0 & 3t_0^2 & 4t_0^3 \\
0 & 0 & 2 & 6t_0 & 12t_0^2 \\
1 & t_f & t_f^2 & t_f^3 & t_f^4 \\
0 & 1 & 2t_f & 3t_f^2 & 4t_f^3
\end{bmatrix}
\begin{bmatrix}
a_0 \\
a_1 \\
a_2 \\
a_3 \\
a_4
\end{bmatrix} =
\begin{bmatrix}
\theta^0_0 \\
\theta^0_1 \\
\theta^0_2 \\
\theta^0_3 \\
\theta^0_4
\end{bmatrix}
\]

The source of potential ill-conditioning can be readily observed from the structure of
the coefficient matrix. If boundary conditions are imposed on the polynomial at widely
split values of time, then the coefficient matrix may become ill-conditioned. The computed
polynomial coefficients will be influenced by numerical errors committed in inverting their
coefficient matrix. The coefficients, under these circumstances, might not be determined accurately enough for the simulation requirements [48]. It is a trivial matter to assess the condition of the interpolating polynomials. Once the coefficients are found, the interpolating polynomial's condition can be determined by simply evaluating it at the end points. Any excessive discrepancy between the desired, and the calculated polynomial values at the end points can be identified, and an error flag triggered. Once the coefficients of the polynomial are known, it is simple to compute the required joint accelerations and subsequently assemble the system equations.

Other types of interpolating functions are also allowed, for example, trigonometric functions. These functions are also included in MBOSS as options.

3.1.3 Implementation

The joint coordinate driving constraint algorithm, as implemented in MBOSS is as follows.

1. Enter the data entry phase of the simulation.
   A. Specify which of the joint coordinates are to be constrained.
   B. Check for redundancy. Make sure that the number of constraints applied at any given time does not exceed the number of system degrees of freedom.
   C. Determine the exact form of the constraint function by either solving for the fourth order polynomial coefficients, or using a sine function.
   D. Check to insure that the constraints are satisfied at the end points within some error tolerance. This check insures that the polynomials
are sufficiently conditioned.

2 Enter the dynamic simulation. At the current time step do the following.

A Calculate the vector of externally applied forces and centrifugal acceleration terms.

B Determine the joint coordinate right hand side acceleration terms.

Loop over the total number of joint coordinate driving constraints.

i Determine the right hand side constraint acceleration term for the current \(i^{th}\) constraint \(\Gamma_i\).

Where \(\Gamma_i = 2a_{2i} + 6a_{3i}t + 12a_{4i}t^2\)

ii Append the current right hand side constraint acceleration term to an array.

C Assemble the mass matrix and carry out the joint coordinate transformation. Note the matrix is symmetric so that only the lower half need be stored. This matrix is \(NJC \times NJC\) where \(NJC\) is the number of joint coordinates.

D Assemble the constraint Jacobian matrix into the transformed mass matrix. Loop over the number of joint coordinate constraints

i For the current \(i^{th}\) joint coordinate constraint loop over number of joint coordinates

a If the current constraint contains the current \(j^{th}\) joint acceleration then place the \(\bar{\theta}_j\) coefficient in the \(j^{th}\) column of the \((NJC + i)^{th}\) row of the transformed mass matrix.

b If the current constraint does not contain the current joint
acceleration then place the number 0 in the \( j^{th} \) column of the \((NJ + i)^{th}\) row of the transformed mass matrix.

E Append the joint coordinate acceleration terms to the transformed vector applied forces.

F Invert the \((NJ + NJCC) \times (NJ + NJCC)\), where \(NJCC\) is the number of constraints, to the system.

3 Pass the joint accelerations and velocities into the integration code to compute the joint velocities and coordinates at the next time step.

3.1.4 Validation

In this formulation the constraint is satisfied only at the acceleration level, and the possibility for error does exist in the position and velocity level constraint [17]. In order to test the formulation, and evaluate the potential for lower-level constraint violation the following simple test problem was designed.

Figure 3.2 shows a rigid blade attached to a rigid hub which is in turn attached to the inertial reference frame. The system possess two degrees of freedom: the rotation of the hub about the local \( z \), described by \( \theta_1 \), and a rotation of the blade about the local \( y \) axis described by \( \theta_2 \). The joint coordinates were constrained to vary according to the expressions:\( \theta_1(t) = 1.0 \sin(4\Omega t) \) \( \theta_2(t) = 25.4 \) rad/s. Where \( \Omega = 25.4 \) rad/s. For this example the Jacobian matrix is simply given by:

\[
\Phi_\theta = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}
\]
and the right hand side acceleration vector is simply:

\[ r = \left[ -16\pi^2 \sin(4\pi t)/180 \right] \]

The fourth order polynomial velocity constraint formulation is used for \( \theta_2 \), and a sine function position constraint is used for \( \theta_1 \).
Figure 3.2 Blade Model and Test Results
The system equations consist of two differential equations describing the motion of the system, and two algebraic equations of constraint. The simulation was performed and the results are shown in figures 3.3 and 3.4. From these test results it is clear that the constraint formulation works well, and that, at least for this test case, lower-level constraint violation is insignificant.

This is a simple and effective method for applying joint coordinate driving constraints to the system equations. As will be shown in chapter 4 these constraints are integral components of the rotor models. Also the method will be used to implement a thrust control system which is discussed in chapter 7.

3.2 Acceleration-Dependent Forces

The ability to apply acceleration-dependent forces to a body was not available in the original version of MBOSS. This capability is required in order to implement the incompressible aerodynamics model that is outlined in chapter 2. The application of acceleration dependent forces is controversial, and some authors contend that the practice is "...not consistent with Newtonian mechanics" [49]. Pars does, however, show that under certain circumstances linearly acceleration-dependent forces can be applied such that a unique solution to the system equations exists. In the current chapter a method to apply acceleration dependent forces is presented. It should be noted that forces can be applied which will cause singularity in the system equations. These forces are not physically realistic, however.

3.2.1 Derivation

The algorithmic difficulty encountered in acceleration dependent forcing rests in the method used in MBOSS to determine the joint accelerations. At the start of integration the
joint coordinates and velocities are known from the initial conditions. Thus the system mass matrix, and the position and velocity-dependent forces can be assembled, and the accelerations can be found algebraically as outlined above. The accelerations, and velocities are then numerically integrated to obtain the velocities and joint positions at the next integration time step. The process is then repeated over the requested time interval. The difficulty in acceleration-dependent forcing arises when calculating the applied force vector. The accelerations, and hence the acceleration dependent forces, are not known until after the applied force vector is assembled, and the system equations inverted. This basic difficulty can be solved using two distinct approaches; one direct and the other iterative. Both methods are outlined in this section.

3.2.1.1 Direct Method

Consider a general acceleration-dependent force in canonical functional form:

\[ F = F(q, \dot{q}, \ddot{q}) \]  

3.11)

This equation may superficially depend upon any nonlinear combination of accelerations, velocities and coordinate positions. Pars, however, clearly states that the acceleration dependent force can, under no circumstances, be nonlinear in the \( \ddot{q} \) terms. Note that the linear incompressible aerodynamics model that was outlined in chapter 2 is linear in these acceleration terms. The general acceleration-dependent force must, therefore, be assumed to be linear in the accelerations.

\[ F = G(q, \dot{q}) + A\ddot{q} + B \]

Here the matrix \( B \) can be absorbed into the \( G(q, \dot{q}) \) terms without loss of generality. For the current application, the acceleration-dependent force must be expressed in terms of the
Cartesian coordinates, so that it can be transformed into the joint coordinate space via the velocity transformation matrix:

\[ f = B^T(G(q, \dot{q}) + A\ddot{q}) \]

and subsequently appended to the system equations:

\[ B^T M \dot{\Theta} = B^T (F_{vp} - M \dot{\Theta} + F_a(q, \dot{q}) + A\ddot{q}) \]

Here the subscript \( vp \) is intended to denote the velocity and position-dependent forces initially present and the subscript \( a \) is intended to denote velocity and position dependent forces contributed by the acceleration dependent force. These velocity and position-dependent forces can be combined, and the relation between the joint and Cartesian accelerations employed to reduce the system to a more manageable form:

\[ B^T M \dot{\Theta} = B^T (F - M \dot{\Theta} + A\dot{\Theta} + A\ddot{\Theta}) \]

Now the system may be simplified by combining like terms:

\[ B^T (M - A) \dot{\Theta} = B^T (F - (M - A) \ddot{\Theta}) \]

The presence of the term \( M - A \) shows clearly that acceleration-dependent forces are identically equal to mass matrix modification. It is clear that the acceleration-dependent forcing may conceivably cause the mass matrix to become ill-conditioned, or even singular. Also the inherent symmetry and positive definiteness of the mass matrix may be lost. The possibility for singularity in the mass matrix, and hence the nonuniqueness of the solution is one reason why acceleration-dependent forces are controversial. The above method is termed the direct method. This method is elegant, but preliminary investigation indicated that it is difficult to implement in practice. The mass modification matrix is difficult to
formulate in general problem independent terms as it may be difficult to derive the acceleration dependent forces in terms of the Cartesian forces. The acceleration-dependent forces may depend on the second derivative of any of the possible generalized coordinates, and not necessarily the Cartesian coordinates. In principle, however, the mass modification matrix can be found.

3.2.1.2 Iterative Method

The acceleration-dependent forces may also be determined by an iterative procedure. The acceleration-dependent forces can be deliberately retained on the right hand side of the system equations:

$$B^T MB \ddot{\Theta} = B^T (F - M \ddot{\Theta} + F_a(q))$$

An iterative process can then be constructed to solve for the joint accelerations. The values of the joint accelerations from the last time step may be used to determine an initial estimate of the value of the acceleration dependent portion of the applied forces. The generalized mass matrix may be constructed in the conventional manner without regard to the acceleration dependent forces. An improved estimate for the current joint accelerations can now be made by inverting the system:

$$\ddot{\Theta}_{i+1} = (B^T MB)^{-1} B^T (F - M \ddot{\Theta} + F_a(q))$$

Now, the acceleration-dependent forces can be updated and the process repeated until convergence. Thus, the acceleration-dependent forces can be left in the most convenient form, and mass matrix modification terms need not be formulated. This iterative process is a modified version of the well-known Gauss-Jordan algorithm [50].
3.2.2 Numerical Aspects

Several comments are in order regarding the application of the iterative method. The generalized mass matrix itself is not modified, this fact has two distinct ramifications. First, the mass matrix will not be singular, will be symmetric, and positive definite. The system equations may, however, be singular, or negative definite. Second, the mass matrix does not change during a given time step. It, therefore, need only be calculated and factorized once at the beginning of the iteration process. One advantage to the direct method is that the rank of the mass matrix can be easily checked by simple evaluation of its determinant, in the iterative method, however, the rank of the system cannot found as easily. The iterative method is much more flexible than the direct approach, and should work just as well. It was, therefore, selected for use in MBOSS. The iterative method is just a modified version of the Gauss-Jordan method. It can be shown [50], that the Gauss-Jordan method will converge if and only if the system is positive definite. As no physical system can be negative definite (i.e. negative mass), the iteration process can be expected to converge for physical problems.

3.2.3 Implementation

The iterative acceleration-dependent forcing algorithm is summarized as follows.

1. At the current time step:
   A. Assemble the generalized mass matrix and the generalized force vector using the joint acceleration at the previous time step as an initial guess for the acceleration dependent portion of the force vector.
   B. Invert the system equations to get an improved estimate for the joint accelerations.
C Save the L-U factorized mass matrix and the velocity and position-dependent portion of the force vector.

D Iterate until convergence.
   a Convert the updated joint accelerations to Cartesian accelerations.
   b Compute the acceleration-dependent portion of the applied force vector based on the improved accelerations.
   c Convert the acceleration-dependent force vector into joint coordinate space.
   d Add the acceleration dependent force vector to the velocity and position-dependent forces.
   e Using the L-U factorized mass matrix determine the new joint accelerations.
   f Check for convergence by the form: \( error = \sum_{i=1}^{NJC} (\ddot{\theta}_i - \ddot{\theta}^{i+1})^2 \)

E When the process has converged save the joint accelerations for use in the next time step

2 Pass the joint accelerations and velocities to the integration code.

3.2.4 Validation

The iterative method is potentially subject to three sources of failure: ill-conditioning of the system, singularity in the system equations, and negative definiteness in the system equations. The latter two failure modes should not occur in practice, and need not be considered further. The first failure mode can be examined by a simple example problem.
Consider the simple mass spring system shown in figure 3.5. An external force is applied to the mass that is directly proportional to acceleration, and the equation of motion can be written:

\[ M\ddot{q} + Kq = P\ddot{q} \]
For this problem the physical parameters were selected to be \( M = 10 \) slug, and \( K = 1000.0 \text{ lbf/ft} \). The initial conditions on the system were chosen to be \( q(0) = 0 \) and \( \dot{q}(0) = 1.0 \). Two cases were considered for illustration purposes. In the first, the constant \( P \) was set equal to 10\% of the mass of the body, and in the second 99\%. In the first case no numerical problems were expected, but in the second the system was expected to become ill-conditioned.

The results of the test cases are plotted in figure 3.6 and be compared to the analytical solution:

\[
q(t) = \sqrt{\frac{M - P}{K}} \sin(\sqrt{K/(M - P)} t)
\]

The quality of the \( P = 0.99M \) solution shows that ill-conditioning was not observed in this particular example. The results of figure 3.6 show the dramatic physical ramifications to acceleration-dependent forcing. The apparent mass of the system is in fact reduced by the applied force. This can be seen by the increase in frequency observed in the response. Acceleration-dependent forces effectively change the mass of the system as is shown in the direct formulation above. The formulation is again exercised in chapter 5 against a much more complex and interesting problem.

3.3 Reaction Force Recovery

In velocity transformation methods, such as the joint coordinate method, the reaction forces due to the configuration constraints imposed by the joints are not calculated automatically. The reaction forces caused by geometric constraints, constraints imposed by the
action multiple load paths, are automatically calculated, however. The physical reason for this is that the system equations are expressed in a reaction free space [51]. This is the physical reason for the relation between the velocity transformation matrix and the Jacobian matrix associated with the equations of constraint:

$B^T \Phi_q^T = 0$

Some [51] argue that the applicability of velocity transformation methods is severely limited because the reaction forces are not automatically calculated. This is, however, not the case. Maximal coordinate formulations calculate the reaction forces at each time step. In rotor analysis the reaction forces are, however, required only for output. Thus, calculating them at each time step is inefficient. One need only calculate them at the reporting time step and not at each intermediate time step. This observation is crucial when considering the efficacy of velocity transformation methods. In [25] a method consistent with the joint coordinate method is presented in which only the reaction forces and moments acting at specific joints are calculated. The procedure is outlined below.

3.3.1 Derivation

In figure 3.7 a typical blade assembly is presented. Consider the problem of calculating the reaction forces acting at the pitch bearing. The system can be cut at the pitch bearing yielding the reduced system depicted in figure 3.8.
Figure 3.4 Typical Blade Model and Reduced Floating Model

The reaction forces caused by joint by the flap hinge acting at the flap hinge are explicitly shown. The reduced system may now be modelled as a floating base body system with known joint coordinates, velocities and accelerations, but unknown reaction forces. The B matrix entries for the floating base body problem can be found in [25], for example.
Now the B matrix for the reduced system can be evaluated, and the reduced system equations of motion can be constructed.

3.21)

$$B_r^T M B_r \ddot{\Theta}_r = B_r^T (F_r - \ddot{M}_r B_r \Theta_r)$$

Here the subscript $r$ is intended to denote that these are the system equations for the reduced system. Now using the joint coordinate constraint method the coordinates for the floating base body can be constrained, and all equations not containing the floating base body accelerations may be discarded. The new reduced system of equations is simply:

3.22)

$$I_{6 \times 6} \Lambda = -[(B^T M B) \ddot{\Theta} + B^T (F - M \ddot{B} \Theta)]_6$$

Here the subscript 6 is intended to indicate that only the first six rows are to be used.

The vector of Lagrange multipliers can now be obtained by simple matrix multiplication. These Lagrange multipliers now represent the generalized forces acting in body $i$ due to the action of joint $j$ resolved into a force couple system acting at the center of mass of body $i+1$. This may now be resolved into an equivalent force couple system acting at any desired location.

3.3.2 Implementation

The reaction force recovery algorithm is summarized as follows:

1. At the current time step assemble the mass matrix and force vector.
   Compute the joint accelerations.

2. If the current time step is a reporting time step enter the reaction force recovery algorithm.
   A. Loop over the total number of joints at which the reaction forces are requested.
a. For the $i^{th}$ requested joint find the reduced $B$ matrix.

b. Assemble the reduced system mass matrix, force vector and carry out the multiplication.

c. Resolve the computed reaction forces into an equivalent force couple system acting on the requested body.

B. Report the reaction forces and moments to output.

3. Continue with the simulation.

3.3.3 Validation

A simple test problem is suggested by Elliott and McConville [1] in which the root moments caused by the higher harmonic pitching motion of a simple rigid blade are to be determined. Consider the system shown in figure 3.2. Joint constraints were applied to the system as in section 1.1.4.

3.23) \[ \Theta(t) = \frac{\pi}{180} \sin(4\Omega t) \]

A simple application of Newton's second law shows that the root moments, in the absence of gravity, are given by:

3.24) \[
\begin{align*}
\begin{bmatrix} M_\eta \\ M_\zeta \\ M_\xi \end{bmatrix} &= \begin{bmatrix} I_{\eta\eta} \ddot{\Theta} + \Omega^2 \cos \Theta(I_{\xi\xi} - I_{\zeta\zeta}) \\ \Omega \dot{\Theta}(\cos^2 \Theta(I_{\eta\eta} + I_{\xi\xi} - I_{\zeta\zeta}) - \sin^2 \Theta(I_{\xi\xi} - I_{\eta\eta} - I_{\zeta\zeta})) \\ 2\Omega \dot{\Theta} \cos \Theta(I_{\xi\xi} - I_{\zeta\zeta}) \end{bmatrix}
\end{align*}
\]

The $\eta \xi$ and $\zeta$ directions are the basis vectors describing the blade body fixed reference frame. $\eta$ is directed in the spanwise direction, and $\zeta$ is positive up. $\xi$ is given by the right-hand rule.
The numerically determined pitching moments are plotted in figures 3.9 and 3.10. The curves corresponding to the analytically derived results are overlaid. The above formulation is apparently sound.
Figure 3.5 Predicted Flap Lag and Pitch Moments
3.4 Chapter Summary

In summary, three distinct capabilities have been added to MBOSS: joint coordinate driving constraint equations can now be appended to the differential system equations, acceleration-dependent forces can be applied to a body, and a method to calculate the reaction forces due to the joints is now available. It is shown that the formulations work well when applied to simple test problems for which analytical solutions were available. In the next chapter the rotor models are designed, and tested. The formulations and results presented in this chapter are integral components of the models developed in the next chapter.
4 Multibody Rotor Modeling

As was stated in the introduction, a mechanical system is modelled in a multibody dynamics package by connecting a set of bodies with kinematic joints, springs, dampers, other externally applied forces, driving constraints, or control systems. When designing the dynamic model, it is essential to construct it such that the salient physical features of the mechanical system are consistently modelled. In the current chapter, two rotor models are constructed; one in which the rotor system is free to translate in the horizontal direction, and another in which the system is free to translate in the vertical plane. Before the models are constructed, it will be necessary to present very briefly some terminology unique to the multibody dynamics field. After the terminology is summarized, the flight path of the system will be outlined, so that the rotor models can be constructed.

4.1 Selected Multibody Dynamics Terminology

There is a small amount of terminology that must be defined before multibody system modelling can be discussed properly. The concept of a kinematic joint, mentioned in passing above, must be explicitly defined here. A kinematic joint limits the motion of one body with respect to another. Several types of joints exist, for example, Hooke or universal joints, revolute joints, prismatic joints, spherical joints, cylindrical joints, and others. All of these and other joints can be used in an MBOSS model, but only two types of joints were used in the rotor models, hence, only these two need be discussed here. The interested reader is referred to [19] for an excellent theoretical and computational discussion using the explicit Cartesian formulation, and [25] for $B$ matrix entries necessary for joint coordinate implementation.
The most commonly used type of joint in the models is the revolute joint. Revolute joints allow only one rotational degree of freedom, an arbitrarily large rotation about the joint axis between the two connected bodies, bodies i and j. This single relative degree of freedom between bodies i and j can be uniquely defined by the rotation of body j with respect to body i about the joint axis. This rotation is described by the joint coordinate $\theta$. In the unconstrained state, each of the rigid bodies possess 6 (3 translational and 3 rotational) degrees of freedom. The presence of the revolute joint prevents bodies i and j from translating with respect to each other, and also prevents relative rotation about axes perpendicular to the joint axis. The joint removes 5 degrees of freedom from the original 12 degree-of-freedom system. The final system, composed of two bodies connected by a revolute joint, has 7 degrees of freedom.

The other joint type that is used in the rotor models is the prismatic (or translational) joint. Two bodies i and j may be connected by a prismatic joint. The prismatic joint allows only relative translation along the joint axis. Translation along axes perpendicular to the joint axis is not permitted. Relative rotation between the bodies is also not permitted. The joint coordinate associated with the prismatic joint is described by the translation of body j with respect to body i along the joint axis. Again the final system has 7 degrees of freedom.

Two models are developed in this chapter: one called the "floating" model and the other called the "horizontal" model. The floating model employs the concept of a floating-base body. If two bodies i and j are connected by a joint, then one of the bodies is free to rotate and translate with respect to the Newtonian reference frame arbitrarily. The name floating-base body arises because the floating-base body has six independent degrees
of freedom. The joint coordinates that are used for the floating body are the three Cartesian
translational coordinates, and a set of 4 Euler parameters [19]. Euler parameters are used
because they cannot become singular [19] as 3 parameter sets, such as Euler angles, can.

The notion of an applied force is well understood. A word of caution is required, however. In multibody dynamics externally applied forces are usually resolved into
equivalent force-couple systems acting at the center of mass of the rigid body to which they
are applied. All of the equivalent force-couple systems are then summed to obtain the total
resultant applied force acting on the body. Note that the terms force and moment are both,
unambiguously, called generalized forces.

Finally, vector and tensor quantities may be measured in, and resolved into, many
different coordinate systems. By convention, each body is fitted with one, so called,
body-fixed coordinate system. A body-fixed coordinate system is rigidly attached to a
known point, possibly the center of mass of each body, and is commonly aligned with the
principal coordinate directions. The absolute motion of each body is measured in terms of
a Newtonian reference frame sometimes, unambiguously, called the global reference frame.
The three basis vectors used to denote the body fixed coordinate directions are expressed
in the lower case Roman letters \( x, y \) and \( z \), and the Newtonian reference frames are expressed
in upper case Roman letters \( X, Y \) and \( Z \). There exists a clearly defined transformation between
the coordinate systems:

\[
4.1) \quad r_{x,y,z} = T_{x,y,z}^{-1} \hspace{1cm} r_{x,y,z}
\]

The large amount of coordinate systems used in multibody dynamics can easily become
confusing. The convention used in the current work is as follows. Transformation matrices,
denoted by $T$, will always map the body fixed basis vectors into the Newtonian set of basis vectors. The mapping from the global coordinate system to the body fixed coordinate systems is simply $T^T$ [19]. For clarity, certain body fixed coordinate systems will be given names. If a coordinate system does not have a name, the body number to which the body is attached will be used to denote the coordinate system. The matrix $T_{hub}$ will map the basis vectors associated with the reference frame called hub into the global reference frame. In the event that a set of basis vectors associated one body fixed coordinate system are to be expressed in terms of a set of basis vectors associated with another noninertial coordinate system then the matrix $T_{a}^{b}$ will map frame a into frame b. All vector quantities will be measured in the inertial reference frame unless clearly stated. The transformation matrices can easily be constructed from the Euler parameters associated with the appropriate body [19], which can be found from the joint coordinates and the known topology [25].

4.2 Specified Flight Path

The rotor models must perform well under a wide range of operating conditions: rotor spin up, hover, forward flight, and conversion. The system must be able to convert between conditions smoothly during the simulation. The desired flight maneuver sequence consists of five distinct phases: rotor spinup, blade feathering, steady-state hover, shaft tilt, and forward flight. The simulation is to begin with the rotor system at rest in a state of static equilibrium. During the rotor spinup phase the rotor hub speed is increased from 0.0 rad/s to some final speed. The blades will then be feathered, from 0.0° to some final value of blade pitch, allowing rotor thrust and inflow to develop. Once a steady state of hover is achieved the rotor shaft will be tilted forward, thus allowing the model to simulate forward flight.
The final values of hub speed, blade pitch, and shaft tilt angle, and the associated rates are, for the sake of modelling generality, left arbitrary. In the next two sections rotor models are constructed to accomplish this flight sequence.

4.3 Horizontal Rotor Model

In the horizontal model the rotor system is to "fly" in the horizontal direction: no vertical flight is allowed. The rotor system is modelled as an articulated rotor. The lag and flap hinges and the pitch bearing are modelled as revolute joints. There are also flap and lag springs and dampers. The flap and lag springs and dampers are required to accomplish the spinup phase of the maneuver outlined above.

The hub is allowed to rotate relative to the inertial reference frame by the application of a revolute joint. The shaft may be tilted with respect to the base by the application of yet another revolute joint. Finally the base may translate along the ground with the application of a prismatic joint.

Figure 4.1 illustrates the final model. The rotor assembly (A) is composed of four blade assemblies (D and E) which are attached to a rigid hub (C). The hub speed (B) is to be varied by the action of a driving joint coordinate constraint applied to joint G. The blades are to be feathered by the application of constraints to the pitch bearings which are shown in figure 4.2. The shaft (F) and tilting wing (H) may be tilted about (J) by the application of a driving constraint. Note that the action of the tilting should cause a translation along (K) of the whole system in the (M) direction, thus simulating forward flight. The joint coordinates of the system are, as can be seen in figure 4.1, simply the horizontal position of the (L), the shaft tilt angle (I), the azimuthal orientation of the (B), and the four flap, four
lag and four blade pitch angles which are shown if figure 4.2. The algebraic signs of the coordinates are in accordance with helicopter convention: hub speed is positive counterclockwise, flap is positive up, pitch is positive nose-up, and lag is positive aft. There are a total of 15 joint coordinates and 6 joint coordinate driving constraints. As can be seen, each of the joint coordinates used to describe the system are of specific interest to the rotor analyst. More abstract coordinates, for example Euler angles, are not needed, but may be recovered from the joint coordinates if so desired.
Figure 4.1 Tilt-Wing Model Schematic
The blade assembly is detailed in figure 4-2. The flap hinge (A) is attached to an offset link (B) which is attached to a pitch bearing (J). The pitch bearing is attached to an offset link (D), which is attached to a lag hinge (E). The lag hinge is attached to rigid blade (H). The lag hinge is attached to the blade at the 1/4 chord location. The blade has a body fixed coordinate system (G) which is aligned as shown. The positive directions for flap, pitch and lag are shown by I, C and K respectively.

The base body is aerodynamically active. It cannot produce lift, but it will generate an appreciable amount of drag. The drag coefficient is chosen to be that of a flat plate oriented perpendicular to the freestream flow $C_D = 0.88$ direction.
The tilting wing in figure 4.1 is also aerodynamically active. It must generate lift as the shaft is tilted forward, so that thrust may be transferred from the rotor system to a lifting surface. The aerodynamics here are greatly simplified. Nonlinear lift, drag and moment coefficients have been taken from the ABRA users guide [10]. The lift, drag and moment coefficient data are interpolated, with cubic splines, so that values at any angle of attack may be determined.

All geometric and inertial properties are taken directly from the ABRA users guide [10].

The blades are the most complex components. The rotor loads are to be calculated by a numerical integration over the blade span. Each blade, therefore, is covered with a grid consisting of a variable amount Gaussian integration points. These points are points of interest at which vector quantities, such as velocity and lift, are to be calculated. Finally, the blade is to be provided with a linear distribution of twist. The blade twist and Gauss integration will be discussed in chapter 5.

The details of the blade assemblies were taken from a mixture of papers. The flap-pitch-lag root geometry was selected to show the modelling generality of the multibody approach, and because it is the root geometry used in the ABRA users guide.

The final model consists of an aerodynamically active base body which is free to translate with respect to the ground. There is an aerodynamically active lifting surface corresponding to that used in the tilt-wing configuration. There is a rigid hub which can undergo large rotations and translations. Finally, there is a set of four blades attached to
the hub by a system of flap and lag offset hinges. The flap and lag hinges are each fitted with springs and dampers. Note that absolutely no assumptions have been or will be made regarding the motion of the bodies other than the assumption that the bodies are rigid.

The initial model, without the joint coordinate constraints, consists of 15 degrees of freedom: four associated with lag motion, four with blade pitch, four with flap, one with the hub speed, one with shaft tilt, and one with forward flight speed. Joint coordinate constraints must be applied to the shaft tilt, hub speed and each of the blade pitch degrees of freedom. Therefore, a total of six algebraic equations of constraint are to be appended to the 15 second-order differential equations for each of the joints. The total number of equations is 21 and the number of kinematic degrees of freedom of the final system is 8.

4.3.1 Validation

Any multibody model must be carefully tested upon the completion of its design. This validation process must be designed so that errors may be found and eliminated. In the testing procedure the model is to be exercised in a vacuum. The model will undergo the full flight schedule outlined above. The test procedure will be outlined here, and the numerical results of the test summarized. The test procedure is to have several distinct phases, each with its own particular set of test objectives.

Test 1: The initial condition of the model is to be such that all bodies are at rest, and all flap, lag, pitch, hub, and shaft tilt angles are to be set to zero. The blades should fall, by the action of gravity, to the static equilibrium position.

Objective: Check to see that the joints are connected properly, that gravity is working and that the flap modes are reasonably damped. The nonrotating flap response can be
observed and compared to theoretical results. Figure 4.5 show the results of test 1. A very simple moment balance shows that the differential equation describing the flap motion is given by:

\[ I_{\beta} \ddot{\beta} + C_{\beta} \dot{\beta} + K_{\beta} \beta = -gL_{\eta}M \]

These results are in excellent agreement with the numerical data shown in figure 4.5. The objectives of test phase 1 have been met.

**Flap Response During Blade Drop**

![Flap Response During Blade Drop](image)

Figure 4.3 Model Test 1 Flap Response
Test 2: Once the transient vibration has decayed, the hub speed will be increased from 0.0 to 25.4 rad/s in 5 seconds.

Objective: Insure that the lag degrees of freedom are properly damped and that Coriolis coupling exists between the flap and lag modes.

Now the rotor spinup phase may be examined. If all flap coupling terms are neglected, the linearized equation of motion for the lag response is given by:

\[ M [l_2 \dddot{\xi} + \Omega l_1 (l_o + l_1) - \Omega (l_1 + l_o) \dot{\xi} l_1 + \Omega^2 l_0 l_1 \dot{\xi}] + [\Omega + \dddot{\xi}] I_\phi + K_\phi \xi + C_\phi \dot{\xi} = 0 \]

Comparison between the numerical results and this expression may be made in order to ascertain the effect of flap-lag coupling. In figure 4.6 the hub acceleration variation is plotted. In figure 4.7 the lag acceleration is shown for both the computed and approximate cases, and in figure 4.8 the computed flap acceleration is depicted. The linearized lag curve has a minimum value which is coincident with the maximum in the hub acceleration. The minimum value in the computed lag response, however, is coincident with the minimum in the flap acceleration response. This indicates that the coupling between the flap and lag degrees of freedom is, in fact, being modelled. Also the computed lag acceleration is much smaller than that for the approximate case, indicating that the stiffening effect of Coriolis flap-lag coupling is being modeled. After the spinup phase the lag degree of freedom undergoes a slight, underdamped vibration which is predicted by the approximate form. This indicates that the rotating lag frequency is being properly calculated in the model. The slight oscillation is caused by the kink in the hub acceleration curve.
Figure 4.4: Model Test 2: Flap and Lag Response
3: Test: Once the system has equilibrated, the blades will be feathered from 0°-8° of collective pitch.

Objective: Insure that the pitch bearings are working properly, that constraint violation is small, and that the fourth order polynomials are properly conditioned.

![Figure 4.5 Commanded Pitch Response](image)

Figure 4.5 Commanded Pitch Response

Figure 4.10 illustrates the blade pitch as a function of time during the feathering phase. The blade pitch does not exhibit significant error even after 30 s of simulation errors less than 0.5° were observed. The objective of test phase 4 has been met.

4: Test: The shaft will then be tilted a full 90° forward. This will cause the base body to slide backward by conservation of momentum.

Objective: Insure that the tilt degree of freedom is working properly, and that the system responds properly to shaft tilt.
By modelling the rotor system as a rigid rotating disk, the equation of motion for the base body during shaft tilt can be found by simple application of Newton’s second law:

\[ (m_1 + m_2)\ddot{y} = m_2 [\ddot{\alpha}l_2 \cos \alpha - \dot{\alpha}^2 l_2 \sin \alpha] \]

where \(\alpha\) is the shaft tilt angle, \(l_2, m_2\) and \(m_1\) are the length of the shaft, mass of the disk, and mass of the base body respectively. This equation was integrated numerically in time, and the solution is plotted in figure 4.12 along with computed data. The results are in excellent agreement, given the assumptions made in the derivation of the analytical model. The objective of test phase 5 has been met.

![Figure 4.6 Model Test 4 Forward Response to Shaft-Tilt](image)

It may be concluded that the horizontal model is fully operational. The data are in excellent correlation with theoretically derived differential equations, and it may be concluded that the dynamic portion of the rotor model is fully operational.
4.4 Flying Rotor Model

The horizontal rotor model is good for wind tunnel type simulations, but is somewhat limited because the model cannot take off: vertical flight cannot be simulated. It is a simple task, however, to add vertical flight capability to the model. The concept of a floating base body can be employed, in order to give the model the ability to "fly" in the vertical plane.

In order to augment the model with vertical flight capability, one need only replace the base body with a floating base body. The three rotational degrees of freedom must, for dynamic stability purposes, be constrained with driving joint coordinate constraints. The addition of the floating base body adds 6 degrees of freedom to the model. The three rotational degrees of freedom must be removed by constraints. Thus, 6 differential and 3 algebraic equations must be added to the model. Note that the third translational degree of freedom need not be constrained as the model cannot generate force that will accelerate the model in the direction perpendicular to the vertical plane.

The constraint formulation was used to model the ground. When the rotor generates enough thrust to overcome gravity these constraints are removed, and the helicopter takes off. The same test suite that was outlined above was applied to the floating base body, the numerical results were as above.

4.5 Chapter Summary

In summary, two rotor models have been designed in order to simulate helicopter and tilt-wing aeromechanics. The models have been tested and the results of the tests are satisfactory. Helicopter specific effects may now be added to the multibody models, so that realistic flight simulations can be conducted.
5 Aerodynamic Modeling

In the current chapter the specific methods used to interface the aerodynamic models given in chapter 2 with MBOSS are discussed, and results are given to show that the models are working properly.

5.1 Blade Modeling

The aerodynamic forces, that are to be discussed in the following sections, must be integrated over the blade span. The blades must also be twisted. The methods used to fit the blades with numerical integration points, and to twist them are not standard MBOSS capabilities. In the following sub-sections algorithms are presented which accomplish these tasks.

5.1.1 Gauss Integration

The numerical method most suited to the problem of integrating a distributed load over the blade span is, perhaps, the well-known Gauss-Legendre quadrature rule, henceforth simply called Gauss integration. In Gauss integration the integrand is evaluated at specific points along the domain of integration. The sampled integrand values are then weighted by an appropriate factor and the products of the sampled integrand values and the weighting factors are summed to give an approximation to the desired integral. The Gauss integration formula is commonly written as [48].

\[ \int_b^a f(x)dx = \sum_{k=1}^{N} \omega_k f(\chi_k) \]

Here the quantity \( \chi_k \) is the value of the quadrature abscissa, and the quantity \( \omega \) is the value of a weight which is determined by \( \chi \). This is termed an \( N^{th} \) order quadrature rule as \( N \)
sampling points are used over the domain of integration. Note that an \( N^* \) order quadrature rule can be shown to integrate a polynomial of degree \( 2N - 1 \) exactly. It is known, [48] that the abscissae for an \( N^* \) order Gauss quadrature rule are given by the locations of the \( N \) roots of the \( N^* \) order Legendre polynomial. Such polynomials may be found by the recurrence relation [48]:

\[
(n + 1)P_{n+1}(\chi) = (2n + 1)\chi P_n(\chi) - nP_{n-1}(\chi)
\]

where:

\[
P_0(\chi) = 1
\]

\[
P_1(\chi) = \chi
\]

The roots of which may be found using standard methods such as Newton’s method. The associated weights are given by the expression:

\[
\omega_{\chi} = \frac{2}{1 - \chi^2} \frac{1}{P'_{n}(\chi_w)^2}
\]

where the \( \chi_w \) are the roots of the appropriate order Legendre polynomial. So that once the order of the quadrature rule has been decided upon the location of the abscissae and the values of the associated weights can be found numerically. Source code which calculates the abcissa locations and the assoccoated weights for quadrature rules of arbitrary order may be found in [48]. In the simulations presented below \( 8^\text{th} \) quadrature rules are used. The Gauss points are number such that Gauss point 1 is at the most outboard spanwise location, and are numbered inward consecutively.
5.1.2 Blade Twist Distribution

Helicopter blades are typically twisted, so in MBOSS the blades must be aerodynamically twisted. The direct effect of twisting the blades is to alter the blade section angle of attack. This may be accomplished by measuring the absolute velocity of a specific location along the blade, and rotating this velocity vector into a twisted reference frame. The twisted reference frame is defined by a rotation by the angle \( \theta' \) about an axis parallel to the blade span, and is positive for rotations directed inward toward the hub.

The twist distribution used in the current work is assumed to be linear, and it is assumed that the twist at the 0.75R location is zero. The assumed twist distribution may be found in [27]. Once the location of each Gauss point has been determined, then twist angle at each Gauss point can be found. A rotation matrix which rotates a given vector from the untwisted frame into the twisted frame associated with each Gauss point may be found as:

\[
T'_{\text{tw}} = \begin{bmatrix}
\cos \theta' & 0 & \sin \theta' \\
0 & 1 & 0 \\
-\sin \theta' & 0 & \cos \theta'
\end{bmatrix}^T
\]

The superscript t is intended to denote the fact that these are twist angles. This matrix can be automatically constructed for each Gauss point once the Gauss point locations are determined. The initial twist angle \( \theta'(0) \), which appears in the twist distribution equation, is a parameter which will be studied in subsequent chapters.

5.2 Linear Quasi-Steady Aerodynamics Model

The incompressible unsteady model is not implemented in its full form. This is because the Theodorson lift deficiency function \( C(k_f) \) is expressed in terms of reduced frequency,
and is, therefore, difficult to implement in a time domain code. Instead of using the full unsteady model conventional quasi-steady assumptions are made. The Lieshmann model captures unsteady effects, so that no functionality will be lost by the quasi-steady assumptions. The quasi-steady assumptions are that $C(k_f) \approx 1$ and $U$ changes slowly. For incompressible flow over symmetrical airfoils the center of pressure acts at the 1/4 chord location [29]. If all pitching is assumed to be about the 1/4 chord location then, using Bisplinghoff’s notation, $a = -1/2$, and the model reduces to:

$$5.7) \quad L = \pi \rho b^2(\ddot{h} + U \ddot{\alpha} + b/2\ddot{\alpha}) + 2\pi \rho b C(k)(\dot{h} + U \alpha + b \dot{\alpha})$$

$$5.8) \quad M = -\pi \rho b^2(b/2\ddot{h} + U b \dot{\alpha} + b^2(3/8)\ddot{\alpha})$$

Note that in the models developed in chapter 4, the lag hinge is outboard of the pitch bearing. Therefore, the pitching motion is not about the 1/4 chord location. The pitching motion is assumed to be about the 1/4 chord location because the model is derived under the assumption that the pitching moment axis location is constant. Therefore, a fixed pitching moment axis must be selected arbitrarily. In the ABRA package the pitching axis is assumed to be at the 1/4 chord location, so the same assumption is made here. The precise effects of the variable $a_p$ on the model’s performance are not known. One could speculate, however, that the inability of the model to capture the location of the pitching axis properly may yield errors in stability analyses. Hence, the model should be used with caution.

The form of the above equations is still somewhat difficult to implement directly, but may be reduced still further by making the following observations. Following Bisplinghoff, the quantity:

$$5.9) \quad w_{34} = \dot{h} + U \alpha + b \dot{\alpha}$$
can be observed to be equal to the velocity of the air, called upwash, at the 3/4 chord location. The upwash is simply the velocity of the air relative to the blade normal to the blade. The quantity:

\[ w_{1/2} = \dot{h} + U \alpha - \frac{b}{2} \dot{\alpha} \]  

is the upwash at the 1/2 chord location. Now differentiating the upwash once with respect to time and making appropriate substitutions yields:

\[ L = \pi \rho b^2 \dot{w}_{1/2} + 2\pi \rho U b w_{3/4} \]  

\[ M = -\pi \rho b^3 \dot{w}_{1/2} - \pi \rho b^3 (U \dot{\alpha}/2 + b \ddot{\alpha}/8) \]  

The direction and sign of the forces and moments are somewhat counterintuitive from a multibody dynamics point of view. The circulatory lift force acts in a direction normal to the free stream velocity, the noncirculatory component, however, acts in a direction perpendicular to the blade. The pitching moment acts at the 1/4 chord location and is positive in the "nose up" direction. The directions of the lift forces are determined by their origin as circulatory and impulsive forces. The pitching moment expression implicitly resolves the lift into an equivalent force couple system acting at the 1/4 chord location.

Recall that these expressions are for lift and moment per unit span, and are not constant over the span, and that numerical integration is required to determine the total lift and moment.
5.2.1 Implementation

The acceleration dependent force algorithm is used to determine the noncirculatory lift contribution, but the circulatory portion of the loading does not depend on the accelerations, so it can be determined directly. The algorithm is, therefore, split into two components. The circulatory portion of the lift is found, and then the acceleration dependent noncirculatory contributions are found. The two algorithms are as follows.

Circulatory force calculation:

1. enter the externally applied force portion of MBOSS.

   A. Rotate the absolute velocity of each blade into the blade's body fixed reference frame.

   B. Loop over each Blade
      a. Compute the absolute velocity at each Gauss point. The Gauss points are assumed to be at the 1/4 chord location
      b. Rotate the absolute velocity of each Gauss point into the Gauss point's twisted reference frame.
      c. Find a unit vector acting in the direction of positive circulatory lift. This acts normal to the flow. This unit vector is $\mathbf{u}^{\text{circ}}$.
      d. Use only the z component of the absolute velocity at each Gauss point $v_{z}^{\text{twist}}$. Compute the circulatory lift at each Gauss point by the form $w_{k}(2\pi \rho U b v_{z}^{\text{twist}})u_{k}^{\text{circ}}$

Note the presence of the Gauss quadrature weight.
e Rotate the lift for each Gauss point into the untwisted frame.

f Replace them with equivalent force couple systems acting at the center of mass of the blade, and add the contributions of each Gauss point. Thus, obtaining the total circulatory lift for the current blade.

g Rotate the total circulatory lift for the current blade into the global reference frame, and add it to the vector of externally applied forces.

C Resolve the circulatory forces and moments for each blade into equivalent force couple systems acting at the center of the hub, and add the contribution of each blade together.

D Rotate the total aerodynamic forces acting on the hub into the hub’s body fixed coordinate system. The result is called the thrust vector.

2 Continue with the applied force portion of the code.

Noncirculatory force calculation:

1 Enter the acceleration dependent force calculation algorithm. Compute the acceleration-dependent forces as follows.

A Rotate the current approximation to the absolute Cartesian accelerations of each blade into the blades body fixed reference frame.

B Loop over each blade.

a Loop over each Gauss point
1 Compute the acceleration at the 1/2 chord location which
   is located 1/4 chord length behind the current Gauss
   point.

2 Rotate this acceleration into the Gauss point’s twisted frame.

3 Use the z component of this rotated acceleration to compute the
   noncirculatory portion of the lift \( w_z(\pi \rho b^3) \dot{\gamma}_{\text{twist}} \).

4 Compute the moment term \( w_z(-\pi \rho b^3) \gamma_{\text{twist}} \).

5 Compute the moment due to pitch rate and acceleration. Using the
   Spanwise component of angular velocity and acceleration compute
   \( w_z(-\pi \rho b^3)(U \ddot{\alpha}/2 + b \ddot{\omega}/8) \).

6 The lift acts in the twisted z location at the 1/4 chord
   location. Rotate the lift back to the untwisted frame.

7 Resolve the lift and moment into and equivalent force couple
   system acting at the center of mass of the blade.

b Sum the contributions of lift and moment due to each Gauss point
   to get the total noncirculatory lift and moment acting on the blade.

C Rotate the total lift and moment acting on each blade into the global
   coordinate system, and add them to the current estimate of the vector
   of externally applied forces.

D Resolve the noncirculatory forces and moments acting on each blade
   into equivalent force couple systems acting at the center of the hub.
E Add these contributions to get the current estimate of the thrust due
to noncirculatory loading.
2 Continue with the acceleration dependent force algorithm.

5.2.2 Validation

This linear incompressible quasi-steady aerodynamics model is to be validated by a two-point test procedure. In the first test, the blades are feathered fairly slowly to three different values of collective pitch. In the second test, the blades are feathered more rapidly. The tests are designed to show that the model is working correctly, and that the interface between the aerodynamics and the multibody dynamics models is sound.

5.2.2.1 Response to Collective Pitch Variations

In this test the aerodynamics model is interfaced with the horizontal rotor model. The hub speed is increased from 0.0 – 25.4 rad/s. Once the system has stabilized three different blade feathering tests are considered: first, the blades are feathered from 0°-5° in 5.0 s, in the second the blades are feathered from 0°-6° in 6.0 s, and in the third the blades are feathered from 0°-7° in 7.0 s. Note that in this test case the blades are not twisted. The computed thrust levels for all three cases are plotted in figure 5.2. For comparison purposes the thrust may be computed from the calculated collective pitch by the expression:

\[ \theta_{0.75R}^{p} = \frac{6C_{T}}{\sigma a_{e}^{L}} \]

Where \( \sigma \) is the rotor solidity and the lift-cure slope is taken to be 2\( \pi \). The thrust for test case 1 is computed by the approximate form and is included in figure 5.2. The results for the test cases are considered to be excellent.
5.2.2.2 Response to Collective Pitch Rate Variations

In this second test the stability of the model is tested. Essentially the same test was performed as in the above section, but at faster average pitch rates: the blades are feathered from 0°-5° at three different average pitch rates. In the three cases that are considered, the feathering phase is accomplished in 1.0 s, 2.0 s and 3.0 s. The thrust and flap responses are shown in figure 5.2. Again the approximate thrust values are included for comparison.

The thrust for test case 1 is computed by the approximate form and is included in figure 5.2. The results for the test cases are considered to be excellent. The computed blade flap angle for the 1 s case is plotted in figure 5.2, and for case 1 the flap angle as computed by the form:
are included for comparison. The quantity $\gamma$ is the blade Locke number which is the ratio of the aerodynamic to the inertial forces given by $\gamma = \frac{\rho a^b c R^4}{I_b}$. The term $I_b$ is the mass moment of inertia about the flap joint. The quantity $p_f$ is the dimensionless rotational natural flapping frequency which is taken to be $p_f = \sqrt{1 + \frac{K_p}{\Omega^2 I_b} + \frac{e}{R}}$. Examination of the figures show that the model can recover the expected results for both thrust, and coning angle. Note that the discrepancy between the approximate and computed flap angles before time $t = 6.0s$ is because the blade’s motion is not dominated by the aerodynamics before the blades are feathered.

\[
5.14) \quad \beta_f = \frac{3\gamma C_T}{4 p_f^2}
\]
Figure 5.2 Flap and Thrust Response to Moderate Pitch Rate
5.3 Linear Compressible Unsteady Aerodynamics Model

As outlined in chapter 2 the Lieshmann-Beddoes aerodynamics model captures more of the physical effects present in the air flow than the quasi-steady model does. Unsteady aerodynamics, compressibility, and stall are all modeled cleanly and concisely. Moreover, the location of the pitching axis is not required in the model. Hence, the contradictory assumptions that were made regarding the location of the pitching axis in connection with the quasi-steady model are not made here.

The linear, attached flow, portion of the model is conveniently given by the canonical linear system:

\[ x^L = A^L x^L + B^L \begin{bmatrix} \alpha \\ q \end{bmatrix} \]

\[ \begin{bmatrix} C_N \\ C_M \end{bmatrix} = C^L x^L + D^L \begin{bmatrix} \alpha \\ q \end{bmatrix} \]

Unlike the expressions given for the linear quasi-steady model above, this form is in terms of quantities which are directly available, or are easy to calculate, in MBOSS. The pitching moment is to be determined from the nonlinear portion of the aerodynamics model, so it need not be calculated from the above form. This reduces the number of internal states from 8 to 4. States \( x_5^L - x_8^L \) are only associated with the pitching moment and are not coupled with the remaining states. The matrix elements associated with the remaining states are taken directly from [20] and are:

\[ A^L = \text{diag}(a_{11}, a_{22}, a_{33}, a_{44}) \]

where the diagonal matrix entries are given by:
5.18) \[ a_{11} = -2U\beta^2 \beta_i b_i \]
5.19) \[ a_{22} = -2U\beta^2 \beta_i b_i \]
5.20) \[ a_{33} = \frac{1}{K_a T_l} \]
5.21) \[ a_{44} = \frac{1}{K_q T_l} \]

where the parameters \( K_a, K_q \) and \( T_l \) are time constants associated with impulsive loading which are approximations to those given by exact linear theory [20] and are:

5.22) \[ K_a = [(1 - M) + \pi\beta \beta_i M^2 (A_1 b_1 + A_2 b_2)]^{-1} \]
5.23) \[ K_q = [(1 - M) + 2\pi\beta \beta_i M^2 (A_1 b_1 + A_2 b_2)]^{-1} \]
5.24) \[ T_l = c/S^2 \]

The constants \( A_1 b_1 \) and \( b_2 \) are given by Beddoes in [20] and are \( A_1 b_1 = 0.3, A_2 b_2 = 0.7, b_1 = 0.14 \) and \( b_2 = 0.53 \). The reduced \( B^L \) matrix which captures only the normal force terms is:

5.25) \[ B^L = \begin{bmatrix} 1 & 1 & 1 & 0^T \\
0.5 & 0.5 & 0 & 1 \end{bmatrix} \]

The reduced \( C^L \) matrix is simply:

5.26) \[ C^L = \begin{bmatrix} 4\pi U \beta \beta_i A_1 b_1 b_1 / c & 4\pi U \beta \beta_i A_2 b_2 b_2 / c & 4/M a_{33} & 1/M a_{44} \end{bmatrix} \]

and finally the reduced \( D^L \) matrix is given by:

5.27) \[ D^L = \begin{bmatrix} 4/M & 1/M \end{bmatrix} \]
There is one minor numerical difficulty associated with the model. The $A^L$ and $D^L$ matrix entries contain the reciprocal of the Mach number. This causes these matrix elements to approach infinity for very small Mach numbers. During most of the simulation this is not a problem at all, but at the beginning of the rotor spin up phase of the simulation the blade section freestream Mach numbers vanish. Preliminary investigation shows that for Mach numbers greater than 0.05 the model can safely be used, so that a hybrid model suggests itself. The circulatory portion of the quasi-steady model can be used for low Mach number and the Lieshmann model for moderate Mach number. Since the Lieshmann model consists of differential equations consistent set of initial conditions must be found, so that transition between the two models can be accomplished.

Transition between the two models is best accomplished in an aerodynamic environment in which the two models are comparable. It is clear that during spin-up the quasi-steady circulatory model and the Lieshmann model are not necessarily in agreement. Noncirculatory loading is neglected if one does not include the impulsive terms in the quasi-steady model. During rotor spin-up, however, it is possible that noncirculatory loading may be significant. It is best to make the transition between the models during a portion of the simulation when the impulsive terms are negligible. The transition can safely be made after spin-up, and before the blade feathering phase. During this phase the noncirculatory component of lift is small as the blades are not flapping or pitching. Noting that the Lieshmann states are constant and that the pitch rate is zero the consistent set of initial conditions for the transition is:
Note that the $a_{ii}$ will not be zero. An entirely consistent set of initial conditions can then be found to start the Leishmann model once the rotor spin-up phase is accomplished.

5.3.1 Implementation

A simple algorithm can be designed to implement the hybrid aerodynamics model. This algorithm illustrates the simplicity of the Lieshmann model, and its intrinsic compatibility with multibody dynamics formalism.

1. Before the simulation begins set the initial conditions for the Leishmann states to those given by the expression with angle of attack replaced by the twist angle at the appropriate Gauss point, and the freestream velocity based on the final hub speed.

2. Begin the Simulation. At the current time step enter the applied force portion of the code.
   A. Check to see if the spin up phase has been accomplished.
   B. If it has simply apply the circulatory portion of the quasi-steady aerodynamics model, and return.
   C. If the spin up phase has been accomplished then enter the lieshmann portion of the code.
      a. Compute the velocity of each Gauss point in the twisted frame.
      b. Compute the matrices $A^L, B^L, C^L, D^L$ for each Gauss point.
c Compute the angle of attack, and dimensionless pitch rate.

d Using the current values of the Lieshmann states compute the derivatives of the Lieshmann states from the matrices, angle of attack and dimensionless pitch rate.

e Compute the normal force coefficient and the chord force coefficient. The normal force is given by $C_N \rho U^2/2$, and the chordwise force is given by $C_c U^2 \rho/2$.

f Rotate these forces into the untwisted frame, and replace them equivalent force couple systems acting at the center of mass of the appropriate blade.

g Sum the weighted contributions due to each Gauss point on each blade.

h Rotate the forces into the global frame, add them to the applied force vector, and compute the total thrust vector as above.

D Pass the derivatives of the Leishmann states into the integrator

3 Continue with the simulation.

5.3.2 Validation

The test suite for the linear compressible unsteady model consists of two portions. Firstly, all of the tests that were applied to the quasi-steady model are repeated. Secondly, tests are conducted in order to illustrate several features of unsteady aerodynamics, and also to illustrate some of the shortcomings of the quasi-steady model.
5.3.2.1 Response to Collective Pitch and Pitch Rate Variations

The two tests that were applied to the quasi-steady model were repeated for the Lieshmann model. The thrust results for slow and moderate pitch variations are plotted in figures 5.5 and 5.6. The thrust results are compared to the approximate values computed by the incompressible approximation given above, and also by the compressibility approximation found by application of the Prandtl-Glauret rule, based on tip Mach number, to the incompressible approximation which yields:

\[ 5.29 \]

\[ \theta_{0.75R} = \frac{6C_T \sqrt{1 - M_{rp}^2}}{\sigma \alpha_c^b} \]
Figure 5.3 Unsteady Thrust Due to Increases in Pitch and Pitch Rate
The computed values of thrust fall in between the two approximations as expected. Again the blades are not twisted for this simulation.

5.3.2.2 Response to Sinusoidal Variations in Collective Pitch

In [20] a classical problem is considered in which the blade angle of attack is allowed to vary sinusoidally in time. The test is approximately reproduced here. In the MBOSS models it is impossible to vary the angle of attack directly, but the blade pitch can be given a periodic excitation. For this test the blade pitch was varied according to:

\[ \theta_p = 5^\circ \sin \omega \tau \]

The frequency \( \omega \) was selected such that the reduced frequency \( k_f = \omega c/2U \) at the Gauss point with freestream Mach number \( M = 0.4 \) was equal to \( k = 0.075 \).

The normal force component, was measured at the radial location with a freestream Mach number \( M = 0.4 \). The blade section is initially at a high, stalled, angle of attack. As the blade pitch is reduced the flow re-attaches and the normal force falls into a stable trajectory. The results are typical of the experimental data presented in [20]. The amplitude and phase of the normal response are similar to those observed by Lieshmann. The response here is linear, and the observed hysteretic effects are directly attributable to the unsteady and compressible nature of the flow. It was observed that for lower Mach numbers the length of the minor axis of the ellipse decreased, and for higher Mach numbers the length of the minor axis increased. This is again in agreement with expected results. The results show that the Bisplinghoff notion of "memory effect" in the flow is clearly evident in the model. Models which do not capture this time lag will not show the hysteretic effect.
5.3.2.3 Response to Unsteady Ramp Pitch Variation

An interesting test is used in both [20][21] in which unsteady aerodynamic effects are clearly illustrated. The normal force coefficient at $M = 0.1$ response due to a very rapid ramp increase in angle of attack is computed. As the Mach number is low compressibility effects are not significant in this example, but the flow is very unsteady. The angle of attack is varied from 6°-19° degrees in approximately the amount of time required for the blade to travel 0.63 chord lengths. During the initial phases of the loading the normal force undergoes a large variation due to impulsive loading. The impulsive loading rapidly decays, and circulatory loading begins to build. In [20] computed normal force and pitching moment coefficients are compared with Navier-Stokes results and the correlation is excellent.
This test was approximately reproduced in the current model. The blade pitch was varied from 0°-13° in the same amount of time. It was found that this roughly duplicates Lieshmann's angle of attack variation. It is difficult to exactly duplicate Lieshmann's variation as the blades are allowed to both flap and lag during the test. The results are shown for both the unsteady and quasi-steady models in figure 5.8. The results for the unsteady model are in agreement in both amplitude and phase with those in [20], and as the angle of attack in the simulation reached a maximum value of approximately 22° the correlation is excellent. The quasi-steady model exhibits errors in both amplitude and phase, however. The quasi-steady solution does predict the steady circulatory loading well, however. Amplitude and phase errors in the quasi-steady model are also observed by Lieshmann [28] in connection with a blade vortex interaction problem. The initial impulsive component of the response clearly shows that the impulsive terms are aptly named. It is important to remember that this normal force is used to force the blades. Such forcing may be expected to cause severe numerical problems. None were observed, however.

The Lieshmann model is more accurate and more general than the quasi-steady model. It captures more physical effects, and can be implemented more cleanly. The location of the pitching axis is not required, so that it is more consistent with the multibody dynamics models than the quasi-steady model is. Also controversial acceleration dependent forces need not be applied. The quasi-steady model will no longer be discussed, therefore, but it was used extensively during the testing phases.
Figure 5.5 Normal Force Due to Rapid Ramp Pitch Variation

5.4 Static Trailing Edge Stall Model

As indicated in chapter 2 the nonlinear portion of the aerodynamics model is simply a correction of the normal force and moment coefficients given by the Leishmann model outlined above. An assumed form is used to determine the chordwise location of the separation point, which is then used to modify the normal force and pitching moment coefficients. Again the forms are given in a manner which is very simple to implement in a multibody dynamics context. Recall the expression for the separation point:

5.31) \[ f_s = \begin{cases} 
1 - 0.3 \exp\left\{ \frac{\alpha - \alpha_1}{S_1} \right\} & \text{if } \alpha \leq \alpha_1 \\ 
0.04 + 0.66 \exp\left\{ \frac{\alpha_1 - \alpha}{S_2} \right\} & \text{if } \alpha > \alpha_1 
\end{cases} \]
This separation point is then used to correct the normal force and to calculate the pitching moment:

\[ C_{N_{nl}} = C_{N_k} \left( \frac{1 + \sqrt{f}}{2} \right)^2 \]

\[ \frac{C_{M_{nl}}}{C_{N_k}} = K_0 + K_1 (1 - f) + K_2 \sin(\pi f^{m}) \]

The parameter \( m \) may be taken to be 2. The other parameters which appear in the expressions are either determined empirically or reconstructed from static airfoil data, and are Mach and airfoil dependent. The parameters \( S_1, S_2 \) and \( \alpha_1 \) are tabulated for different Mach numbers for a NACA-0012 airfoil in [32]. The data for \( M \geq 0.3 \) are taken directly from [32]. Data for \( M < 0.3 \) were apparently unavailable, so these points were obtained by a linear extrapolation. The parameters \( K_0, K_1 \) and \( K_2 \) appear in [32] but are associated with a different assumed moment distribution. The values for these parameters for this assumed moment distribution are given in [36] for \( M = 0.3 \), however. In the implementation these values are used for all Mach numbers. The constant \( K_0 \) is associated with a Mach dependent offset of the aerodynamic center of pressure from the 1/4 chord location. As data are not available for this parameter it is assumed to be zero, hence the aerodynamic center is assumed to act at the 1/4 chord location. The values for the two remaining parameters are \( K_1 = -0.12 \) and \( K_2 = 0.04 \).

In this model dynamic stall effects are neglected. The stall model assumes that stall is initiated when the static stall angle is exceeded. This is somewhat restrictive as vortex
induced lift is **not** modelled. The Lieshmann model can be extended to include dynamic stall, but this requires the addition of 4 more states. If dynamic stall is found to be significant during the tilt-wing conversion additional states should be added to the model.

**5.4.1 Implementation**

The nonlinear portion aerodynamics model is implemented as follows.

1. Enter the Lieshmann model portion of the code.
   1. Once the normal and chord force have been calculated compute the assumed separation point \( f_s \) based on the angle of attack, and Mach number at the current Gauss point.
   2. Compute the modified normal force coefficient.
   3. Compute the modified chord force coefficient.
   4. Compute the moment coefficient, and the pitching moment is \( C_M U^2 \rho/2 \).
   5. Continue with the remainder of the Leishmann model

**5.4.2 Validation**

In the test case the stall model is tested in an essentially static problem. In this test the ability of the Lieshmann model to predict nonlinear compressibility effects is validated, as is the nonlinear models' ability to predict the static stall angle and the loading after stall.

**5.4.2.1 Static Stall Prediction**

In this case the blade is feathered from zero degrees of pitch to 25° of pitch. The computed normal force and pitching moments are plotted versus blade pitch in figures 5-11 and 5-12 at various radial locations. The computed MBOSS output are in good agreement with the data given in [39], but again are not directly comparable as the blades are allowed
to flap and lag. Compressibility effects can also be clearly observed in these curves. The lift-curve slope increases with increased Mach number, the static stall angle decreases, and the maximum normal force also decreases. These, as outlined in chapter 2, are the compressibility effects that the model is expected to capture, and these effects are captured here quite well. The pitching moment response is, shown in figure 5-11 are also in excellent qualitative agreement with those in [39].
Figure 5.6 Nonlinear Normal Force and Pitching Moment
5.5 Chapter Summary

In summary the aerodynamics have been presented, and several tests have been applied to validate their particular implementation. Several test cases are considered which tax both the aerodynamic models and the multibody rotor models. Well known results are recovered for some simple test cases. The nature of the aerodynamic environment which will be encountered during the simulations is reproduced in test cases, and the Leishmann results are excellent. The quasi-steady model is discarded due to observed inaccuracy in its results, however.
6 Induced Inflow Modeling

In chapter 2 the inflow phenomenon is briefly described in the context of simple momentum theory and the necessity for an inflow model is given. The basic approach to induced inflow modelling is summarized here. The velocity of the blade is known with respect to the inertial reference frame, or equivalently with respect to an inertial body of air. The absolute velocities of the blades are directly calculated by MBOSS. In order to compute the blade section angle of attack, however, one needs the velocity of the blade relative to the air. The air velocity that the blade encounters is not inertial, but has a component of due to the effect of induced inflow. Thus the total blade velocity which must be used to compute the angle of attack at a given blade section consists of a component due to the absolute motion of the blade, plus a component due to inflow that accounts for the velocity of the air with respect to the inertial reference frame. In the current chapter the two models that are given in chapter 2 are implemented in MBOSS, and subsequently tested.

6.1 Static Momentum Theory Inflow Model

The key result of simple momentum theory as given in chapter 2 is as follows:

6.1) \[ \lambda_i = \frac{C_T}{2\sqrt{\mu^2 + \lambda^2}} \]

The quantity \( \lambda_i \) is the dimensionless induced velocity. The quantity \( \mu \) is the dimensionless forward flight speed. \( \lambda \) is the dimensionless velocity of the air flow through the rotor disk given by:

6.2) \[ \lambda = \mu \tan \alpha_d + \lambda_i \]
Where $\alpha_d$ is the disk angle of attack. It is clear that the induced velocity is related nonlinearly to the rotor thrust. This equation is not directly invertable as the thrust depends in the inflow. Hence, the aerodynamic loads and the induced velocity must be computed simultaneously using an iterative procedure. For the case of hover the advance ratio $\mu$ is zero, and the inflow expression can be reduced to:

$$\lambda_i = \sqrt{\frac{C_T}{2}}$$

This model does not account for unsteady effects, and, therefore, is of limited applicability to the transient portions of the simulation, for example, during the blade feathering phase. The results given by the inflow model during the transient phase are not expected to be correct, but the model is expected to converge to the correct hover and forward flight solutions after the transients components of the response decay.

6.1.1 Implementation

The implementation couples the momentum theory model with either of the aerodynamics models. The inflow model may be considered an outer loop in the rotor loads prediction, and the aerodynamics model an inner iterative loop. The algorithm is summarized as follows.

1. Enter the rotor loads prediction portion of the code.

   A. Use the value of inflow at the last time step as an initial estimate. Use the inflow as the $z$ component of the inflow vector, assume the other elements are zero.
B Rotate the inflow vector into the global reference frame.

C Subtract the inflow from the inertial velocity vector for each of the Blades.

D Apply one of the aerodynamic models to find the total thrust.

E Compute the Newton-Raphson expressions for the iteration
   a For the case of hover use $f_i = v_i - \sqrt{T_i/2\rho A}$ with Jacobian equal to unity.
   b The case of forward flight use $f_i = T/\pi \rho R^2 - 2v_i \sqrt{V^2 + v_i^2 + 2v_i V \sin \alpha_d}$
      With Jacobian equal to $f_i$

F If the function $f$ is with a tolerance then the process has converged Goto step 2:. Otherwise
   a The updated guess by the Newton-Raphson iteration formula is given
      by $v_{i+1} = v_i - f_i/f_i$, Goto step A: above.

2 The process has converged. Continue with the simulation.

6.1.2 Results

The above algorithm was coupled with the horizontal rotor model and the quasi-steady aerodynamics model. Unfortunately, the results were somewhat less than desired. The algorithm was applied to a simulation in which the hubspeed was set to 25.4 rad/s and the blades feathered from 0°-8° of collective pitch. For very small values of blade pitch the iterative process diverged, and the simulation had to be aborted. Initially it was not clear whether it was the acceleration-dependent force iteration or the inflow iteration which was divergent, but a simple analysis shows that the inflow iteration was the source of the
6.2 Uniform Dynamic Inflow Model

Recall that in chapter 2 a dynamic inflow model was presented in which unsteady effects were included. The induced velocity field during tilt-wing conversion is little understood, so that the full nonuniform model which appears in chapter 2 may not be accurate. The coupling and apparent mass matrices which appear in the model were only validated for trimmed flight conditions. Because the nonuniform model may be inaccurate, and its effects are probably relatively minor compared to the large forward flight speed, the nonuniform inflow components are neglected here. The equations governing the uniform component of the Pitt-Peters inflow model is given in dimensionless form to be:

\[
\frac{d}{d\psi} \lambda_0 + \frac{75\pi}{128} \frac{2}{V_T} \lambda_0 = C_T
\]

The ratio \(128/75\pi\) is an apparent mass term for twisted rotors for untwisted rotors it should be replaced by \(8/3\pi\). The term \(V_T\) is a dimensionless measure of the total flow through the disk and is given by:

\[
V_T = \sqrt{\lambda^2 + v_{x\text{hub}}^2 + v_{y\text{hub}}^2}
\]

with \(\lambda\) defined as in the previous section, however, the induced velocity as calculated by momentum theory is used instead of the induced velocity predicted by the model. The quantities \(v_{x\text{hub}}\) and \(v_{y\text{hub}}\) are the absolute velocities of the hub rotated into the hub body fixed coordinate system.

A physical explanation for the divergence observed in the momentum theory model can be found by noting that the inflow is accelerated to the hover value for a given value of
thrust instantaneously. This cannot occur in reality. Fortunately the dynamic inflow model accounts for the time delay in inflow development. An implementation of the model is presented in the next section.

6.2.1 Implementation

The following is the dynamic inflow implementation algorithm:

1. Enter the aerodynamic force section of the code.
   A. Extract the value of the induced velocity from the array returned by the integration routine.
   B. Using the value of the inflow as the z component of the inflow vector and the other two elements equal to zero rotate this vector into the global frame.
   C. Subtract this from the blade velocities, and apply an aerodynamics model
   D. Using the thrust computed in step C: find the momentum theory inflow
   E. Compute \( v_i \), hence \( \dot{v}_i \).
   F. Append \( \dot{v}_i \) to the integration vector.

2. Continue with the simulation.

6.2.2 Validation

The induced inflow model may be validated by some of the methods used to test the aerodynamics models. In the aerodynamics validation, given in the last chapter, some simple tests were made in which blade pitch and pitch rate, are varied and the computed thrust and flap angles are examined. These same two tests can be reconstructed in order to validate the
inflow model.

6.2.2.1 Effect of Collective Pitch Variations

In chapter 5 one of the tests involved using various final pitch settings. Three cases are considered: a case in which the blades are pitched from 0°-5° in 5 s, 0°-6° in 6 s, and 0°-7° in 7 s. The results neglecting inflow are given in chapter 5 and the results including inflow are given in figure 6.1. Here the unsteady Lieshmann model is used for the near wake aerodynamics. In figure 6.1 the total dimensional thrust is plotted versus time. In figure 6.1 the dimensional induced velocity is plotted versus time for the three test cases. Using the computed values of blade pitch and inflow the thrust can be found from the approximate form:

$$\theta_{0.75R} = \frac{6C_T}{\sigma a} - \frac{3 \Omega R}{2 \Omega R}$$

where the flow is assumed to be incompressible. The thrust level computed by this expression is plotted for the 0-5 degree case in figure 6.1 for comparison. The approximate thrust is somewhat less than the computed values due to compressibility effects. This is was observed in chapter 5.

Also the final value of induced inflow for hover may be determined by the expression:

$$v_i = \sqrt{\frac{T}{2pA}}$$

The comparison between the computed inflow, and the momentum theory result is excellent. Note that the thrust is decreased by a factor of approximately 60.0% from the levels observed in chapter 5 which is expected, [27]. The dynamic inflow implementation is in good agreement with the expected value for hover.
Figure 6.1 Inflow and Thrust Response to Pitch
6.2.2.2 Effect of Collective Pitch Rate Variations

In chapter 5 results are given to show the effect of blade pitch on the total thrust. The same test was run with the inflow model, and the same three cases were considered. The dimensional thrust is plotted as a function of time for this case in figure 6.2, and the computed induced velocity is plotted as a function of time in figure 6.2. The induced velocity was compared with the values computed from the momentum theory, and the computed and approximate results are in excellent agreement. The inflow model performs well and can recover momentum theory values for the case of hover. The dynamic inflow algorithm is quite stable with respect to variations in pitch rate. The peaks in the thrust responses are due to the fact that the inflow is lagging the thrust. As the inflow develops after the blades are feathered these peaks decrease. These results clearly show that the inflow and thrust are exhibiting the qualitatively correct phase relationship.
Figure 6.2 Inflow and Thrust Response to Pitch Rate
6.2.2.3 Effect of Moderate Shaft-Tilt

The model must be able to recover correct values of inflow for the case of low speed forward flight. In this simulation the dynamic inflow model was used in conjunction with the Lieshmann model to predict the thrust and inflow levels for forward flight. A steady-state condition of hover with 6.0° of blade pitch and a hub speed of 25.4 rad/s was attained. The shaft was then tilted forward by 10.0° in 1.0 s, and the rotor system was allowed to achieve a steady-state condition of forward flight at about 140 ft/s. The thrust, inflow and forward flight speed are plotted in figures 6.3 and 6.4 as functions of time. The drop in inflow observed in figure 6.3 is expected as the induced velocity should decrease in forward flight. The computed inflow, thrust and forward flight speed are all compatible with momentum theory. Note that the blades are not twisted in this test case. The inflow model can duplicate well known momentum theory results when applied to simple hover and lowspeed forward flight test cases. There is a pronounced once per revolution vibration in the computed flap angle. As the model accelerates the amplitude of the flap vibration increases. Note also that as the model accelerates the thrust drops off. This decrease in thrust is due to the relatively large flow of air through the rotor disk. These are the expected results. The initial transient observed in the flap response is due to the relatively tilting angular acceleration and velocity of the hub. This transient is much less pronounced for when the shaft is tilted more slowly.
Figure 6.3 Inflow and Thrust Response to Moderate Shaft Tilt
Figure 6.4 Flap and Forward Speed Response to Moderate Shaft tilt
6.2.3 Tilt-Wing Application

Now that the basic requirements for helicopter rotor dynamics simulations have been included in the MBOSS model, and important key results have been duplicated, the model may be exercised against a tilt-wing simulation.

The simulation is performed using the horizontal rotor model. Thrust, flap, and inflow responses are provided in figures 6.5 and 6-6. Note that the blade pitch is kept to a constant 6.0° during the full 85.0° shaft tilt phase. As can be seen from the figures the flap degree of freedom undergoes very large transient vibrations during the shaft-tilt phase. The thrust also decreases very rapidly. This is due to the fact that as the rotor disk is accelerated, and simultaneously tilted, ever increasing amounts of air flows through the disk. A similar effect was observed in the previous example. The increases in air flow through the disk reduce the blade section angles of attack until the rotor no longer produces net propulsive thrust. In figure 6-5 the thrust is plotted for differing values of blade twist. The blade twist distribution has only a small effect on the conversion performance of the rotor system. The blade twist does, however, have a large impact on the hover performance. The assumed blade twist distribution, which has negative twist at the outboard locations, is not optimal for the conversion process. In the flap response, depicted in figure 6-6 a fairly large transient vibration can be observed. The vibration due to reduction in rotor thrust during conversion.
Figure 6.5 Inflow and Thrust Response to Tilt-Wing Conversion
Figure 6.6 Flap Response to Tilt-Wing Conversion

6.3 Chapter Summary

In summary, two inflow models are presented. The static momentum theory model has been found to be unstable during the initial phases of the required flight path, but the dynamic inflow model has been used successfully. The dynamic inflow model is capable of recovering momentum theory results before and after the conversion when applied to hover and conventional forward flight test cases. A first attempt at a full tilt-wing simulation is attempted. It is concluded that the linear blade twist distribution used has no effect after conversion, however, can severely limit hover performance. Also a large vibration is observed in the flap response, and also in the lag response. The amplitude of the vibration
is dependent upon the rotor thrust.
7 Thrust Control System

In chapter 6 a first attempt at a tilt-wing conversion simulation is made. The blade pitch is kept constant during the simulation and various blade twist distributions are used. Large flap and lag vibrations are observed due to a rapid decrease in rotor disk loading during the shaft-tilt phase. The decrease in disk loading is due to a reduction in the blade section angle of attack caused by a large increase in the flow through the rotor disk. This reduction in angle of attack was also observed to severely limit the attainable forward flight speed. One obvious way to maintain disk loading during the conversion phase of the simulation is to feather the blades. The effect of the increase in collective pitch on the flap and lag vibrations is less clear, however. In the current chapter the trim control system that was outlined in chapter 2 will be modified such that increases in collective pitch during the shaft-tilt phase can be realized.

7.1 Derivation

In chapter 2 a trim control system is presented which has been used in rotor calculations successfully in the past [47]. The goal here is not to trim the rotor, however, but to control the thrust by varying the collective pitch. Hence the cyclic pitch components need not be considered. The trim controller given in chapter 2 can, therefore, be reduced to a form that only includes the collective pitch terms. The final reduced form is:

\[ T_{11} \dot{\theta}_e + \dot{\theta}_e = K_{11} A_{p11} \Delta C_T \]

Only the 11 element of the matrices \( T, K \) and \( A_p \) are considered. The derivative here is with respect to rotor time. Converting to actual time, substituting the appropriate terms from [47], expanding the thrust error term, and slightly rearranging the expression yields:
The constraint formulation can, therefore, be used to interface the thrust controller with MBOSS. The second derivative of the pitch joint coordinate can be given by the reduced control law above, and applied as a constraint exactly as outlined in chapter 3.

This simple control law can be expanded for use in the floating model as well. The so-called computed torque technique, extensively used in robotics applications [24], can be used to determine the appropriate commanded thrust level in terms of the errors in desired altitude and velocity, the thrust required to overcome gravity, and drag. The commanded thrust can be expressed in the form used in the computed torque technique. The basic idea is as follows. A desired trajectory is determined, here polynomials are used that are of the form:

7.3) \[ y_c = (t - t_0)^2(a_1 + a_2(t - t_0) + a_3(t - t_0)^2) \]

This form may be differentiated twice to give commanded the acceleration and velocity. The acceleration can be modified in order to overcome the drag on the fuselage and also the force of gravity, so that the commanded acceleration becomes:

7.4) \[ a_c = \ddot{y}_c + \left( \frac{1}{2} C_d \rho A \dot{y}^2 \right) + g \]

Here the nonlinear term is the actual drag on the fuselage, and \( g \) is the acceleration due to gravity. The mass is the total mass of the model. The error between the actual and commanded position and velocity can be determined:
7.5) \[ e = y_e - y \]
and
7.6) \[ \dot{e} = \dot{y}_e - \dot{y} \]

Now the computed torque technique can be used to determine the required commanded thrust level:

7.7) \[ T_e = Ma_e + K_e \dot{e} + K_p e \]

For tilt-wing application the shaft will be tilting. The commanded thrust can be modified to account for this by accounting for the shaft tilt \( T_e' = T_e / \cos(\alpha) \) where \( \alpha \) is the shaft tilt angle.

7.2 Implementation

The basic idea is to simply determine the error between the measured and desired thrust and moments scale the error by appropriate gains. This will determine the second derivatives of the pitch components, and, therefore, the required second derivative to total blade pitch. The constraint formulation can then be used to realize the desired pitch motion.

The implementation is very straightforward, and is as follows.

1. Complete the full rotor loads prediction portion at the current time step.
   A. Enter the joint coordinate constraint portion of the code.
      a. Compute the commanded thrust level using the computed torque form
      b. Using the error between the desired and actual thrust determine the desired pitch joint accelerations from the control law
   B. Continue with the Joint coordinate constraint formulation

2. Compute the accelerations as usual
7.3 Validation

As the stability of this controller is at least questionable a rigorous test procedure was used to validate the implementation. The test procedure has two distinct components. First, the robustness of the algorithm was tested and the gain factor and time constant was chosen. Second the position control portion of the controller is tested.

7.3.1 Robustness Test

The test suite that was selected was intended to test the controller under highly stressful and unrealistic flight conditions. A "bang-bang" control test was used in which the controller was to achieve commanded thrust levels as shown in figure 7.1. The rotor hub speed was 25.4 rad/s for the simulation. The joint coordinate constraints are used to feather the blades to an initial collective pitch of 8.0°. At time \( t = 6.25 \) s the thrust controller is turned on. The computed collective pitch is plotted in figure 7.1, and the computed rotor thrust in figure 7.1. The control system was able to achieve the commanded thrust levels. Crisp stable performance was found for the gain \( \kappa = 17.0 \) and time constant \( \tau = 2\pi \). The position and velocity feedback gains were set to zero for this test. In the computed thrust one can observe a sudden spike just as the controller is turned on. This spike is due impulsive loading of the type observed in chapter 5. Blade section angle of attack changes rapidly during this initial phase, and, hence, causes a dramatic increase in the impulsive loading. The overall performance of the controller is satisfactory and the results shown in [47] show similar convergence rates for much less demanding problems.
Figure 7.1 Thrust and Pitch Results for Controller Test 1
7.3.2 Altitude Control Test

The floating model was used to validate the altitude control capability of the thrust control system. The specified flight path is such that the helicopter takes off and $t = 3.25$, and reaches a steady state of hover at 200.0 ft in 5.0. Here the computed torque technique is used to command the vertical position of the aircraft. The gains that were used in this simulation are $K_p = 500.0$ and $K_p = 500.0$. The calculated thrust, collective pitch, and altitude are given in figures 7.2. With these gains the hover condition can be achieved with a tolerable overshoot. The results show that this simple thrust control law can be used to control the aircraft.
Figure 7.2 Thrust, Pitch, and Altitude Results for Controller Test 2
7.4 Tilt-Wing Application

The thrust controller was coupled to the horizontal model and applied to the tilt-wing simulation that was undertaken in chapter 6. The blades are feathered to 10.0° of collective using the constraint algorithm in order to achieve steady-state vertical flight conditions. Again the shaft is tilted from 0°-85° of tilt in 10 s, but the thrust controller is used feather the blades during the shaft tilt in order to maintain desired thrust levels. For a desired flight speed of 450.0 ft/s the drag on the fuselage is approximately 18,000 lbf using the data give in the ABRA users guide and a flat plate drag coefficient of 0.88. This was, therefore, the commanded thrust level.

The flap, lag, pitch, and thrust response are plotted in figures 7.3-7.6. Note that the thrust controller can maintain nearly constant thrust during the tilt phase. The flap and lag degrees of freedom still undergo vibration, but in the chapter 6 simulations the flap decreased, and in the current simulations the flap increases. This indicates that thrust control can be used to control the flap vibrations. As can be seen the required collective is quite high, approximately 45.0°. The final value of the forward flight speed was found to be equal to the desired 450.0 ft/s.
Figure 7.3 Pitch Response During Tilt-Wing Conversion
Figure 7.4 Flap Response During Tilt-Wing Conversion
Figure 7.5 Lag Response During Tilt-Wing Conversion
Figure 7.6 Forward Flight Speed Response During Tilt-Wing Conversion
7.5 Chapter Summary

In summary a very simple thrust controller can be used to maintain a commanded thrust levels during the tilt-wing conversion. The transient flap and lag vibrations observed in chapter 6 changed by controlling rotor disk loading during conversion. Also the thrust controller can be used to achieve desired forward flight speeds. The stability of the thrust controller is adequate for the tilt-wing conversion process. There are, however, unmodelled dynamics associated with the tilt process. Their incorporation into the control law would probably improve its performance.
8 Tilt-Wing Conversion Results and Conclusions

All aspects of the tilt-wing model have been validated in the previous chapters. The horizontal model has been used to accomplish some tilt-wing conversion simulations, and the flying model has been exercised. It has been clearly shown that the multibody model, aerodynamics, inflow and thrust controller are working properly. A full tilt-wing conversion simulation was undertaken with the flying model coupled with models that were presented in the previous chapters. In the current chapter the some of the results of the final tilt-wing conversion simulation are presented. Conclusions are drawn regarding multibody rotor modeling, and tilt-wing conversion.

8.1 Tilt-Wing Conversion Results

In the current section the some of the output from the final full tilt-wing conversion simulation is presented. The simulation flight sequence is summarized as follows. The rotor is spun up to a final hub speed of 25.4 rad/s in the first 5.0 s. Once the system has stabilized, the modified thrust controller is used to fly the model to a commanded altitude of 200 ft in 5.0 s. The flight trajectory is generated by a polynomial as was discussed in chapter 7. Once the desired altitude is realized, the model is allowed to hover until the system has stabilized. The conversion process is split into two distinct parts. During the first 70° degrees of tilt the airflow over the wing is assumed to be stalled (recall nonlinear aerodynamics are included on the wing). The controller is designed to attempt to maintain constant altitude during this first phase of the simulation. During the second phase of the tilt phase the controller is designed to accelerate the model to the desired flight speed of 320 ft/s using a polynomial for the flight speed variation. The total tilt time is 10.0 s.
The computed altitude and forward flight speed during the entire simulation are shown in figures 8-1 and 8-2. Figure 8-1 shows that the rotor generates enough thrust for the model to take-off at approximately 6.0 s. The model then climbs to the commanded altitude to 200.0 ft and hovers. During the first phase of the conversion the controller maintains the commanded altitude very well. At the time denoted by A in figure 8-1, the tilting wing has tilted through 70°, and the controller begins to command the desired flight speed. At position B in figure 8-1 the conversion process is complete.
Figure 8.2 Forward Flight Speed

Figure 8-2 shows the computed forward flight speed. The algebraic sign is negative because of the sign convention used in the simulation. At time 10.0 s (position A) the model begins to move forward at a constant 20.0 ft/s. This is because the rotor is generates a net pitching moment which accelerates the base slightly. It can be seen that once conversion
is undertaken at position B in figure 8-2 the model begins to accelerate quite dramatically. The conversion process terminates at position C in the figure. From figure 8-1 it can be seen that the controller maintains the commanded altitude during conversion quite well. Finally, figure 8-2 shows that the model achieves the commanded final forward flight speed of 320 ft/s after the conversion with only slight steady-state error. Note that the addition of an integral term in the controller will eliminate this error. An integral term will also help eliminate the error noted in the forward flight speed after position A in figure 8-2.
Figure 8.3 shows the computed thrust during the simulation. At position A in the figure the controller begins to attempt the take-off phase of the simulation. The thrust undergoes a wide variation as the model is accelerated. At position B the conversion process
begins and large increase in required thrust is observed as the shaft is tilted. A position C
the controller begins to command the desired forward flight speed, and at position D the
desired forward flight speed is realized.

\[ \text{Figure 8.4 Flap Response} \]
Figure 8-4 shows the computed flap response during the simulation. At position A the controller begins to attempt the take-off phase of the simulation. The rapid increase in the flap angle at this point is due to the large increase in thrust observed in figure 8-3. At position B the conversion phase begins, and the amplitude of the flap response increases as the model accelerates. The frequency of the flap response is 1.0/rev as expected. At position C the model has achieved the commanded forward flight speed, and the flap angle undergoes a constant amplitude 1/rev vibration which is bounded by the two dashed lines. The vibration is due to the fact that the wing is only tilted forward by 85.0°. There is a region in which the flap angle is negative during the conversion. The reason for the negative flap will become clear below. The relationship between forward flight speed and flap angle is clearly shown in the figure: as the model accelerates the amplitude of the flap vibration increases. The lag degree of freedom exhibits a similar vibration as was observed in the previous chapter.
Figure 8.5 Required Collective Pitch

Figure 8-5 shows the computed collective pitch during the simulation. At position A the thrust controller begins to attempt the take-off phase of the simulation, and the required increase in pitch is observed. At position B the conversion process begins, and the expected increase in collective is observed. At position C the model reaches the desired forward
flight speed, and the pitch reaches a constant value.

Figure 8.6 Required Tilt-Torque

Figure 8-6 depicts the torque which must be applied to tilt the rotor (tilt-torque). The key feature of this curve is shown by position A in the figure. At this point the flow on the airfoil re-attaches, and the pitching moment acting on the airfoil decreases dramatically.
This is reflected in the computed torque. Clearly, this effect is significant and the conversion process should be designed to eliminate it. Note that an abrupt change in torque is not due to the commanded tilt angle. Figure 8-7 below shows the commanded tilt acceleration during the conversion. The change in tilt-torque at position A cannot be attributed to this curve. The abrupt change in the tilt-torque curve at position B, however, is clearly caused by the discontinuity in tilt acceleration at 25.0 s. This discontinuity is not shown in figure 8-7, but it was part of the simulation. When the tilt phase is accomplished the commanded acceleration jumps from -0.18 rad/s/s to 0.0 rad/s/s.
Figure 8.7 Tilt Acceleration

Figure 8-8 shows the normal force coefficients at various radial locations during the conversion process. Before the conversion the normal force coefficients are positive for the inboard radial locations, and are negative for the outboard locations (recall from chapter 4 that Gauss point 1 is the most outboard and Gauss point 8 is the most inboard). This is
due to the -10° twist distribution used in the blade model. After the conversion, the inboard radial locations are observed to have negative angles of attack, and Gauss point 8 has stalled. Careful optimization of the blade twist distribution may overcome the negative loading and stalling at the inboard locations. From the amplitude of the response it is clear that unsteadiness is important at the inboard radial locations after the conversion. During the early portions of the conversion, however, unsteadiness is more important at the outboard radial locations. From figure 5-6 it can be seen that Gauss point 1 is very close to its maximum static normal force coefficient, hence dynamic stall is important at the far outboard radial locations. Recall from figure 8-4 that the flap response was negative for a small interval of time during the conversion. It can be seen from figure 8-8 that the negative normal force observed at Gauss point 1 during the conversion is causing the negative flap. The negative flap, therefore, may be eliminated by another twist distribution.
8.2 Conclusions

Multibody packages can be effectively used to simulate advanced rotorcraft dynamics. In particular, the transient dynamic response of a rotor system can be effectively captured using a multibody model. Multibody packages can be extended to include sophisticated
aerodynamics models as well as inflow models. Flight control systems can be included into the multibody models as well. Well known textbook results can be recovered with the multibody model, and the tilt-wing conversion results seem reasonable. It should also be noted that the MBOSS implementation of the joint coordinate method provided stable results over a wide range of problems. The implementation is also quite efficient as the simulation results presented in this chapter were computed in about 2 hrs of SUN SPARC CPU time.

There are several preliminary conclusions that may be drawn regarding the conversion process itself. The blade twist distribution may be carefully optimized in order to minimize negative disk loading and stall effects. This may also have the effect of maintaining positive flap angles during the conversion. Unsteady aerodynamic effects are important during the conversion at the outer radial blade locations, and are important at the inboard locations after the conversion. The flap response can be effected by the thrust controller, and it seems possible to optimize the flap response by controlling the thrust optimally. Stall effects on the tilting wing effect the tilting-torque significantly. A higher performance airfoil may help, but the forward velocity of the model should be minimized until the flow over the tilt-wing has attached.

There are several interesting topics that can be explored with the current model. During the conversion the cyclic pitch can be employed to generate potentially beneficial pitching moments. These pitching moments may help to reduce the amount of tilting torque that must be applied. The effect of the cyclic pitch on the stability of the aircraft is unclear, however. The current model can be used to design a cyclic pitch controller, and study its effects of the stability of the model. The model can readily be extended, by adding another
rotor system, to study the handling properties of the twin-rotor tilt-wing configuration. It can be used to validate yaw control laws, and optimize such controllers. The model can also be used to help optimize blade twist and planform.
List of References


