AN ADAPTIVE FINITE ELEMENT PROCEDURE FOR ROTORCRAFT AERODYNAMICS AND AEROELASTICITY

By

Mustafa Dindar

A Thesis Submitted to the Graduate

Faculty of Rensselaer Polytechnic Institute

in Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

Major Subject: Aeronautical Engineering

Approved by the
Examining Committee:

[illegible]
Prof. Mark S. Shephard, Thesis Adviser

[illegible]
Prof. Andrew Z. Lemnios, Thesis Adviser

[illegible]
Prof. Joseph E. Flaherty, Member

[illegible]
Prof. Kenneth Jansen, Member

[illegible]
Prof. Brian Thompson, Member

Rensselaer Polytechnic Institute
Troy, New York

November 1998
(For Graduation December 1998)

AN ADAPTIVE FINITE ELEMENT PROCEDURE FOR ROTORCRAFT AERODYNAMICS AND AEROELASTICITY

By

Mustafa Dindar

An Abstract of a Thesis Submitted to the Graduate

Faculty of Rensselaer Polytechnic Institute

in Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

Major Subject: Aeronautical Engineering

The original of the complete thesis is on file
in the Rensselaer Polytechnic Institute Library

Examining Committee:

Prof. Mark S. Shephard, Thesis Adviser
Prof. Andrew Z. Lemnios, Thesis Adviser
Prof. Joseph E. Flaherty, Member
Prof. Kenneth Jansen, Member
Prof. Brian Thompson, Member

Rensselaer Polytechnic Institute
Troy, New York

November 1998
(For Graduation December 1998)

© Copyright 1998
by
Mustafa Dindar
All Rights Reserved

CONTENTS

LIST OF TABLES

LIST OF FIGURES

In memory of

Nihal and Muammer Dindar

ACKNOWLEDGMENT

First, I would like to thank my parents for their sustained support and encouragement throughout my entire education.

I wish to extend my sincere appreciations to my thesis committee members: Prof. Joseph Flaherty, Dr. Kenneth Jansen and Dr. Brian Thompson. Their help, guidance, knowledge and suggestions were very valuable. Especially, Prof. Flaherty's help in providing computer access to MHPCC facilities was a turning point for the success of my research. Dr. Jansen not only provided his unconditional support and help for the advancement of my doctoral studies, but he also exposed his vast knowledge in finite element theory and technology in his classes and in personal discussions. Ken, I thank you for teaching me the basics of Galerkin/least-squares finite element formulation.

I received enormous amount of help from my friends and colleagues in solving the issues of developing a parallel, adaptive finite element flow solver and making large scale computations for helicopter rotor blades. The individuals that I would like to thank are: Dr. Can Ozturan, Dr. Hugues DeCougny, Mr. Mark Beall, Mr. James Teresco, Mr. Ravi Ramamoorthy, Dr. Saikat Dey and all the people I know in SCOREC. Dr. Carlo Bottasso's efforts in developing the initial versions of the finite element analysis code are greatly appreciated.

I would like to thank Dr. David Kenwright and Dr. Roger Strawn of NASA Ames Research Center for their useful discussions, providing the vortex-core detection code and helping me run some of the analysis cases at NAS computer system. David, I thank you very much for your friendship and support. You played a major role in the success of my studies.

I want to thank Mr. Peter Finnigan, Dr. Scott Finn and Dr. William Bachrach for providing support and knowledge during the last three years of my stay at GE Research and Development Center as a contract employee. The financial support that I received from GE for the continuation of my doctoral studies is greatly appreciated.

I would also like to acknowledge the financial support I received from Boeing Helicopters (Mesa, AZ) for the last few months of my stay at RPI. I would also like to thank to Dr. Ahmed Hassan (Boeing Helicopters) for his encouragement and support. .

I want to thank my advisor Prof. Mark Shephard for bringing me to RPI for my doctoral studies and his sustained support for the computer access in SCOREC and elsewhere. Mark, I thank you for your technical knowledge and your constructive criticism.

Finally, I want to extend my deepest appreciations and thanks to my advisor Prof. Andrew Lemnios. Besides teaching me the basics of rotorcraft aerodynamics and related topics, Prof. Lemnios always stood by me, encouraged me to go forward with my ideas and corrected me with patience and understanding.

ABSTRACT

An adaptive, parallel computational fluid dynamics (CFD) technique is developed to compute rotor-blade aerodynamics. For this purpose an error indicator based on interpolation error estimate is formulated and coded into an adaptive finite element framework. It is shown that the error indicator is effective in resolving the global features of the flow-field. Furthermore, for efficiency and problem size considerations, once the interpolation errors are reduced to acceptable levels, the adaptive refinement is done only in regions affected by the vortical flows. To do this, a novel vortex core detection technique is used to capture vortex tubes. The combination of interpolation error estimate and vortex core detection technique proved to be very effective in computing vortical flow-field of rotor blades. Example adaptive, parallel calculations of hovering rotor blades, requiring 1-3 million tetrahedral elements, are presented.

Finally, an extension of the current finite element CFD technology to rotor-blade aeroelasticity calculations is presented. For this purpose a CSD (Computational Structural Dynamics) procedure is coupled with the adaptive finite element CFD solver. Using the CFD-CSD coupling methodology, a hingeless stiff in-plane rotor blade is analyzed for its aeroelastic deformations and vibration frequencies in hover flow conditions. Calculated results are compared against experiments and similar numerical studies.

CHAPTER 1
INTRODUCTION

Ever since Wilbur and Orville Wright showed the world the possibilities of powered flight, man's lifelong dream has become a reality. Today, almost a century after the Flier-I took-off from the sand dunes of Kill Devil Hills (NC), our knowledge in aeronautics and aerospace has lead us even beyond the boundaries of earth's atmosphere.

Fixed-wing aircraft which dominated the civilian and military applications since the beginning of century, slowly yielding the way to other forms of lift generating machinery in the later years of the first powered flight. In particular, the concept of Vertical Take-Off and Landing (VTOL) compelled great deal of attention. Although the sketchy details of a machine that can fly vertically were explored even at the time of Leonardo Da Vinci, the technology that made VTOL flight a reality came only after two decades of the Wright Brothers success. Among thousands of other inventions, Thomas Edison envisioned and experimented with VTOL, realizing that no such machine would be able to fly until the development of a sufficiently lightweight engine. Around 1900, the invention of the internal combustion engine opened the road to airplane flight and started the early developments in helicopter flight.

During 1920's and 1930's the "autogiro" was developed by the Spanish Juan de la Cierva. The autogiro was the first practical use of the VTOL idea. During this period of time the concept of controlled flight and the problems associated with it started to establish. In 1941, Igor Sikorsky built the VS-300, a helicopter with a single three-bladed main rotor and a small anti-torque tail rotor. Lateral and longitudinal control of VS-300 was by the main rotor cyclic, and the directional control was by means of the tail rotor. In the meantime, Lawrence Bell built his model, called Bell 47, with a two-bladed teetering main rotor and a tail rotor using the gyro stabilizer. The Bell Model 47 was the first helicopter to receive an American certificate for air-worthiness in 1947 [4].

In 1950's, an important development was the application of the turboshaft engine to the helicopter, which replaced the reciprocating engine. Kaman Aircraft Company's K-225 helicopter was the first example with turbine power. Since that time the turboshaft engine has become the standard power-plant for most helicopters.

In the second half of the 20th century, the need to design advanced and efficient helicopters also created the need to address some of the important problems of rotorcraft. The transonic flow on the advancing side and the dynamic stall problem on the retreating side of the main rotor, influence of rotor wake on the fuselage, blade-tip-vortex interactions, aeroelastic stability, acoustic noise and vibration are some of the problems that we can mention here for today's advanced rotor-craft (see Figure 1.1). Some of these problems were solved by experimental and analytical methods and this trend still continues today. Nevertheless, with the emergence and growth of the digital computer, numerical methods are being used to help solve aerodynamic, aeroacoustic and aeroelastic problems of advanced rotorcraft.

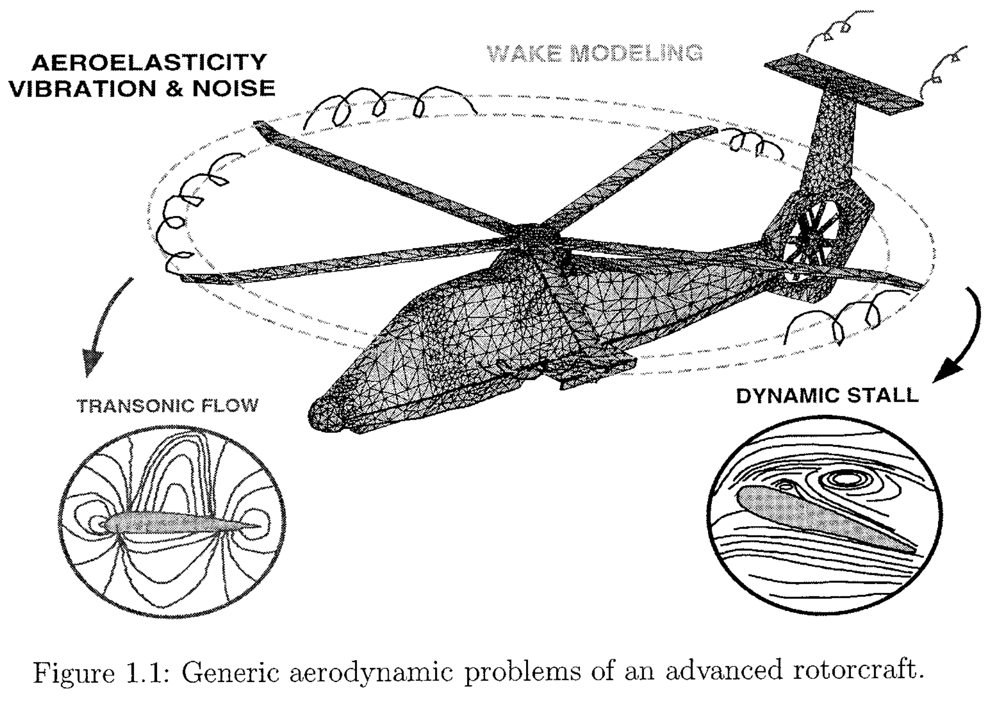

Figure 1.1: Generic aerodynamic problems of an advanced rotorcraft.

1.1 Literature Survey

The performance of a rotary-wing aircraft is often reduced by a host of aerodynamic, acoustic and vibratory problems. Transonic flow, retreating blade stall, tip vortex and wake are some of the major aerodynamic factors that have to be dealt with during the design process of advanced rotorcraft. As stated by F.X. Caradonna [1] "...because of such flow complexity the aerodynamic design of helicopters is traditionally an empirical craft that often relies on experience and test more than on detail analysis." This and many other similar statements motivated us to attack the challenges of rotorcraft aerodynamics and develop methodologies that can be used during the helicopter design process. Steps taken in the past have actually helped today's engineers develop state-of-the-art techniques that take us closer to solving the problems of advanced rotorcraft as accurately as possible. Increasing computer power and our expanding knowledge in combining mathematical theories with advanced numerical methods, have created the research area referred to as computational fluid dynamics (CFD). Over the years, both fixed-wing aircraft and rotary-wing aircraft designs have benefitted from the fast pacing developments of CFD. From an objective point of view, it is this author's belief that the current trend in CFD will likely gain more momentum over the next decade and make today's new computational methods tomorrow's preferred design tools.

The development of CFD methods for rotorcraft aerodynamics can be collected into two groups: Inviscid methods and viscous methods. Historically, the majority of the CFD techniques were developed to solve inviscid flow equations. Limited computer power has been a detrimental factor to select some simplified form of the Navier-Stokes Equations that describe the motion of a true fluid.

Boundary-integral methods have traditionally formed the basis of computational fluid dynamics techniques. These methods relied on the ability to superimpose various elementary solutions of Laplace's Equation, such as the source, the doublet and the vortex. Early works of Hess and Smith [2] also marked the birth of CFD techniques in the beginning of the digital computer era. The procedure of Hess and Smith used a discrete representation of the body surface and the wake (known as panels) and expressed the source, the doublet and the vortex as discrete summations which yielded a system of linear equations that can be solved by either direct or iterative techniques. These methods are usually referred to as "panel methods". Further simplifications of panel methods, known as "lifting-line methods", have also been used widely both in fixed-wing and rotary-wing aerodynamics calculations. In comparison to panel methods, lifting-line methods ignore the thickness (source) effect of wings and rotor-blades. The panel method codes "VSAERO" [5] and "PANAIR" [6] are still favorite computational methods of fixed-wing aircraft industry because of their relative efficiency over Euler or Navier-Stokes flow solvers.

Rotorcraft aerodynamic calculations, and especially hover problems, are very sensitive to the accuracy of the wake prediction. Most panel methods require a specified wake geometry. Because of this, most of the helicopter wake problems were treated by "vortex-lattice-methods" where the velocities were computed using the Biot-Savart law. The Scully vortex-lattice method [7] which is incorporated in the CAMRAD [8] comprehensive rotor analysis code, is the most widely used of such codes. Despite the efficiency and ease of use of both panel and vortex-lattice methods, the empiricism in the wake modeling and the inability of such codes to handle discontinuous solutions, such as shocks, were the primary reasons that motivated the development of methods based on full potential, Euler and Navier-Stokes equations.

Since early 1970's, methods based on full potential equations were developed to handle the transonic flows of fixed and rotary-wing aircraft. In the helicopter industry, the codes ROT22 [9] and FPR [10] were the widely used examples of such developments. However, the relative errors of full potential codes in predicting the shock location and their inability to handle unsteady effects limits their accuracy and load-prediction capability. Later, both full potential and panel methods are coupled with boundary layer methods to improve the prediction of loads. Extensive works of Cebeci [11] in coupling the potential flow codes with various boundary layer methods produced several industrial codes. One major shortcoming of the coupled inviscid-viscous calculations was their inability to handle high angle of attack calculations with large separation regions (e.g, deep dynamic stall).

The importance of the wake in rotorcraft aerodynamics has motivated many researchers since the 1980's to develop "unified flow methods" that capture both the near-field and far-field effects of vortical flows. First examples of unified methods using Euler's equations showed how difficult it is to get good correlation with experiment due to numerical dissipation [12]. It has been noted in these early calculations that the inherent dissipation in the numerical codes caused the wake to dissipate to such an extent that the induced inflow was under-predicted. Most of these initial calculations were done using central-difference schemes on structured meshes. Later on, in an effort to reduce the effects of dissipation, upwind-difference methods were introduced [13]. The main conclusion reached in these preliminary results was that grid density was not sufficient to capture the actual wake formation observed in experiments. Designing less dissipative high-order numerical schemes using sufficiently fine grid to model the hover problem were clearly needed. During the same time, Steinhoff [14] re-examined the full potential solver approach, since it does not have any numerical dissipation problems, and arrived at a new method, called "vorticity embedding", which specifies the shed circulation such that it freely convects through the computational grid. The main idea used in the vorticity-embedding technique was to decompose the total velocity into the sum of potential velocity and the velocity induced by a known vorticity field. The shed wake behind the blade in vorticity-embedding methods were constructed using a process called marker trajectory integration. Even today, the vorticity embedding methods are used extensively to solve both hover and forward flight problems of rotorcraft flows [15] and the results were impressive.

With the increased computer power and storage in 1990's, the development in unified flow methods using both Euler and Navier-Stokes solvers continued. Successful applications of Navier-Stokes equations to rotor-blade hover flows were done by Srinivasan et. al. [16] and Wake et. al. [17]. As an example, the mesh size used in [17] was 918,000 points (a structured finite-difference mesh). Later on, Admed [18] applied the solution technique of [16] to forward flight cases using the over-set grid approach. It has been concluded by Srinivasan et. al. [16] that alternative approaches to solve rotor-blade flows must use adaptive techniques that capture both wake and the tip vortex.

One of the early applications of adaptive solution techniques for rotor-blade flows is due to Strawn et. al. [31]. Reference [31] analyzed a two-bladed rotor system in hover using an unstructured grid that was obtained by tessellating the hexahedral cells of a finite difference grid to linear tetrahedral elements. During the solution process they used vorticity magnitude as the error indicator in the flow-field to resolve the wake and the tip-vortex structures. Although vorticity magnitude can be used as an indicator of wake and vortical flow regions, it does not reflect the error in the solution. To formulate an error indicator we should examine the finite element theory [36]. One possible avenue in designing an error indicator for flow problems is to use the interpolation error estimates [36] [55]. In this thesis, a simple error indicator derived from the interpolation error estimate for linear finite elements is going to be explained and applied to rotor-blade flows in hover.

1.2 Overview

In chapter 2 the Euler's equations in conservation form are introduced. First, a fixed frame of reference form of the Euler's equations are presented, then the rotating frame of reference is explained in its general from, and then the Euler's equations are transformed from fixed frame of reference to rotating frame of reference for the steady-state analysis of rotor blade flows in hover conditions. The Euler's equations are re-written in terms of entropy variables to obtain a symmetric form. A brief explanation why we use this transformation is given.

Chapter 3 is devoted to the explanation of a Galerkin/least-squares finite element formulation of the Euler's equations of previous chapter. The notion of space-time formulation and its functional spaces are introduced and illustrated. A formal definition of Galerkin/least-squares (GLS) finite element formulation is presented. Individual terms of the GLS formulation are stated. Consistency, stability and accuracy of the GLS formulation is mentioned. Furthermore, a model equation (pure advection equation) is used to examine the accuracy and stability of the GLS formulation. For this model problem, an optimal least-squares parameter is obtained and it is shown that GLS formulation is stable. The three-dimensional extensions of both least-squares operator and discontinuity operator are briefly presented.

In chapter 4 an overview of adaptive solution techniques in CFD is presented. After the introduction of full and semi-norm, a general form of local interpolation error is given. Based on the this local interpolation error, an error indicator is developed. This error indicator is the L2-norm of second derivatives of the function of interest. In relation to this error indicator, two methods are presented to calculate the second derivatives on linear finite elements.

In order to show the convergence rates of the error indicator two model problems are presented. First of these two model problems is the re-circulating incompressible flow in square cavity. It is shown using this model problem that quadratic convergence rates are achievable using the error indicator. The second model problem is the compressible conical flows. Results for the compressible model problem has shown that convergence rate was a little less than the quadratic rate. This is probably due to non-linearity of the model problem as well as the adapted mesh being too diffuse for the shock wave.

To increase the efficiency of adaptation procedure, a local adaptation procedure is suggested. In particular, flow regions which are affected by the blade tip vortex are targeted for local mesh adaptation. For this purpose, a novel vortex core detection technique is combined with the error indicator to achieve an efficient adaptation procedure.

In chapter 5, a numerical model for hovering rotor blades is presented. The model relies on one-dimensional momentum theory which is given in this chapter. A combination of momentum-theory and a potential sink is used to represent the flow-field of a hovering rotor system. Computational boundaries of the model problem are defined. Weaknesses of current hovering blade BC's are summarized. One of the weaknesses of the existing hover BC's is that the formulation requires explicit knowledge of thrust coefficient. For rotor systems where the thrust is not know a priori, this creates a problem. Fixing the thrust coefficient also fixes the circulation around the blade to a certain value. To circumvent these problems an iterative inflow-outflow hover BC approach is suggested and implemented. This new boundary condition iteratively changes the magnitude of the inflow-outflow velocities based on the calculated value of the thrust coefficient. This way, the circulation around the blade is not fixed but it rather relaxes and reaches to steady-state with advancing solution steps.

Further enhancements to hovering blade BC's are suggested. One issue that is investigated is the outflow BC in hover. Currently, the outflow boundary for a hovering blade is based on the one-dimensional momentum theory, which states that the slip-stream contraction ratio should be $1/\sqrt{2}$ far below the blade. In practical calculations we place the outer boundaries at a finite distance from the rotor. Therefore, when the outer boundaries are close to the blade, assumption of one-dimensional momentum theory may not hold. That is, the slip-stream contraction ratio may be slightly bigger or smaller than what momentum theory tells us. To tackle this problem an approach based on normal pressure gradient is suggested. This approach looks at the normal pressure gradient at the outer boundary points. Based on the sign of the normal pressure gradient a decision can be made as to whether a point is an inflow boundary or outflow boundary.

In chapter 6, rotor blade flows in hover conditions are studied. First a two-bladed rotor system (Caradonna-Tung) is analyzed. For this case, four levels of mesh adaptation is performed to compute the pressure distributions and the tip vortex. A second rotor blade studied is the UH-60A rotor system. For UH-60A, two hover conditions are analyzed, zero thrust and design thrust. For both cases, the importance of tip vortex resolution is emphasized. At zero thrust, the effect of tip vortex on calculated sectional thrust distribution is less in comparison to design thrust case. For the design thrust case, we found that the calculated sectional thrust distribution is much more accurate with adaptive methods. The calculated torque distributions showed larger discrepancy for the design case because of the lack of viscous effects in Euler's equations.

Finally, a study of the effect of circulation-control on the Caradonna-Tung blade is investigated. For this purpose, the lower surface of the blade is modified such that near the tip of the blade a rectangular patch of area is used as a region where a normal (to the blade surface) flow (transpiration) is injected into the flow-field. The magnitude of the transpiration velocity is chosen to be 20% of the tip speed of the blade. It is found that the transpiration velocity changes the thrust loading of the blade near the tip in comparison to the baseline Caradonna-Tung blade.

In chapter 7 an aeroelastic coupling procedure is suggested to calculate the static deflections of soft rotor blades in hover conditions. To calculate the structural response of the blade, a CSD solver (DYMORE) is coupled with the adaptive CFD solver explained in the earlier sections. To handle the information exchange between the CFD and CSD, a simple dimensional reduction/upgrade approach is developed. A model rotor blade is described and analyzed to calculate its equilibrium deflections in hover. The CFD-CSD coupling procedure is applied to calculate the flapwise, lead-lag and torsional deflections of the blade. The fundamental rotating frequencies of the model blade are calculated and compared with both experiments and similar numerical calculations of others.

Finally, chapter 8 presents an overall discussion of the adaptive hover calculations and CFD-CSD coupling procedure. Recommendations for future enhancements to extend the current analyses to include viscosity effects and be able to do calculations in forward flight conditions are made.

CHAPTER 2
PROBLEM STATEMENT

2.1 Governing Equations

Conservation laws governing the motion of an inviscid flow can be obtained by considering the conservation of mass, momentum and energy in an infinitesimal control volume $(dx_1 \cdot dx_2 \cdot dx_3)$ [19]. To exemplify this derivation process, consider the conservation of mass density, ρ, as shown in Figure 2.1. Stated in the context of differential control volume of Figure 2.1, conservation law requires that the time rate of change of ρ should be equal to the net rate at which mass enters the control volume. Mass enters and leaves the control volume through gross fluid motion. Transport due to such motion is often referred to as *advection.* Following from Figure 2.1, the rate at which mass enters the control volume through surface $dx_2 - dx_3$ may be expressed as $(\rho u_1)dx_2dx_3$, where u_1 is the mass average velocity in x_1 direction. The mass leaving the control volume from the opposite surface may be expressed as

$$\left[(\rho u_1) + \frac{\partial \rho u_1}{\partial x_1}dx_1\right]dx_2dx_3 \tag{2.1}$$

Using similar results for x_2 and x_3 directions, the conservation of mass requirement

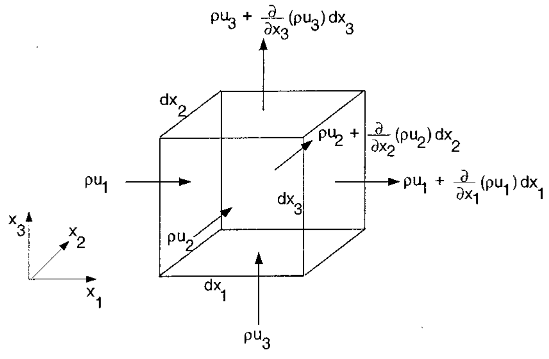
$\rho u_3 + \frac{\partial}{\partial x_3}(\rho u_3)dx_3$
dx_2
$\rho u_2 + \frac{\partial}{\partial x_2}(\rho u_2)dx_2$
ρu_1
$\rho u_1 + \frac{\partial}{\partial x_1}(\rho u_1)dx_1$
dx_3
ρu_2
x_3
x_2
x_1
dx_1
ρu_3

Figure 2.1: Differential control volume $(dx_1 \cdot dx_2 \cdot dx_3)$ for mass conservation.

is written as

$$\frac{\partial \rho}{\partial t} dx_1 dx_2 dx_3 = \underbrace{(\rho u_1) dx_2 dx_3 + (\rho u_2) dx_1 dx_3 + (\rho u_3) dx_1 dx_2}_{\text{mass entering control volume}}$$
$$\underbrace{- \left[\rho u_1 + \frac{\partial \rho u_1}{\partial x_1} dx_1\right] dx_2 dx_3 - \left[\rho u_2 + \frac{\partial \rho u_2}{\partial x_2} dx_2\right] dx_1 dx_3 - \left[\rho u_3 + \frac{\partial \rho u_3}{\partial x_3} dx_3\right] dx_1 dx_2}_{\text{mass leaving control volume}} \tag{2.2}$$

Canceling terms and dividing by $dx_1 dx_2 dx_3$ yields

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_1}{\partial x_1} + \frac{\partial \rho u_2}{\partial x_2} + \frac{\partial \rho u_3}{\partial x_3} = 0. \tag{2.3}$$

Equation 2.3, the *continuity equation*, is the general expression of the overall mass conservation requirement, and it must be satisfied at every point in the flowfield.

The second fundamental law that is pertinent to fluid flow is *Newton's second law of motion*. For a differential control volume, this requirement states that the sum of all the forces acting on the control volume must be equal to the net rate at which momentum leaves the control volume. Two kinds of forces may act on the fluid: *body forces*, which are proportional to volume, and *surface forces*, which are proportional to area [20]. Gravitational, centrifugal, magnetic and electric fields may contribute to the total body force. Whereas the surface forces are due to pressure, p and *viscous stresses*. In this thesis, we are going to be focusing on inviscid flows and ignore the viscosity effects. Therefore, the surface forces are only going to be due to pressure. Furthermore, we will assume that example flow problems that we are going to solve do not have any body forces acting on them.

To use Newton's second law, the fluid momentum fluxes for the control volume must be evaluated. As an example, we will derive the momentum equation for x_1 direction. Following from Figure 2.2, the net rate at which x_1 momentum leaves the control volume is

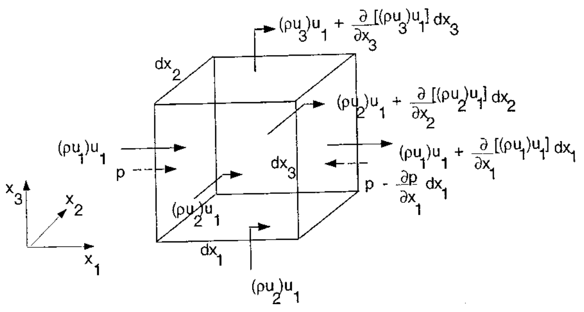
$(\rho u_3)u_1 + \frac{\partial [(\rho u_3)u_1]}{\partial x_3} dx_3$
$(\rho u_2)u_1 + \frac{\partial [(\rho u_2)u_1]}{\partial x_2} dx_2$
$(\rho u_1)u_1 + \frac{\partial [(\rho u_1)u_1]}{\partial x_1} dx_1$
$p - \frac{\partial p}{\partial x_1} dx_1$
$(\rho u_1)u_1$
p
$(\rho u_2)u_1$
$(\rho u_3)u_1$
dx_1
dx_2
dx_3
x_1
x_2
x_3

Figure 2.2: Differential control volume $(dx_1 \cdot dx_2 \cdot dx_3)$ for momentum conservation.

$$\frac{\partial \rho u_1}{\partial t} dx_1 dx_2 dx_3 + \frac{\partial(\rho u_1 u_1)}{\partial x_1} dx_1 dx_2 dx_3 + \frac{\partial(\rho u_2 u_1)}{\partial x_2} dx_1 dx_2 dx_3 + \frac{\partial(\rho u_3 u_1)}{\partial x_3} dx_1 dx_2 dx_3 \tag{2.4}$$

The surface force, which is due to pressure only, is given by

$$-\frac{\partial p}{\partial x_1} dx_1 dx_2 dx_3 \tag{2.5}$$

Note that the pressure originates from an external force on the fluid in the control volume and is therefore a *compressive* force. Finally, equating the rate of change in x_1 momentum (Eq. 2.4) to the fluid forces (Eq. 2.5), we obtain

$$\frac{\partial \rho u_1}{\partial t} + \frac{\partial(\rho u_1^2 + p)}{\partial x_1} + \frac{\partial(\rho u_2 u_1)}{\partial x_2} + \frac{\partial(\rho u_3 u_1)}{\partial x_3} = 0. \tag{2.6}$$

This is the conservation of momentum in x_1 direction and similar expressions can be obtained for x_2 and x_3 directions. The next requirement of conservation laws is energy conservation [20]. The energy per unit mass of the fluid includes the thermal internal energy e_i, the kinetic energy $\frac{1}{2}|\mathbf{u}|^2$, and the potential energy. Ignoring the effect of potential energy, since it is much smaller than both kinetic and thermal energies, the total energy in the fluid is advected through a differential control volume just like it is in mass and momentum conservation cases. Details of derivation of the energy equation can be found in [20].

In general, the governing equations of an inviscid flow are given by the Euler equations. We begin with the Euler equations in conservative quasi-linear form [19]

$$\mathbf{U}_{,t} + \mathbf{F}_{i,i} = 0 \tag{2.7}$$

where,

$$U = \begin{pmatrix} U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{pmatrix} = \rho \begin{pmatrix} 1 \\ u_1 \\ u_2 \\ u_3 \\ e \end{pmatrix} \tag{2.8}$$

$$F_i = \rho u_i \begin{pmatrix} 1 \\ u_1 \\ u_2 \\ u_3 \\ e \end{pmatrix} + p \begin{pmatrix} 0 \\ \delta_{1i} \\ \delta_{2i} \\ \delta_{3i} \\ u_i \end{pmatrix} \tag{2.9}$$

In Equations 2.8-2.9, ρ is mass density; $\mathbf{u} = \{u_1, u_2, u_3\}^T$ is the fluid velocity vector; p is the thermodynamic pressure; e is the total energy per unit mass; and δ_{ij} is the Kronecker delta.

The Euler's equations represent the conservation of mass, momentum and energy in vectorial form. The thermodynamic pressure is related to conservative variables through the constitutive relations, such as ideal gas law [22]:

$$\begin{aligned} p &= \rho R T && (2.10) \\ e_i &= e - \frac{1}{2}|\mathbf{u}|^2 = \mathcal{C}_v T && (2.11) \\ \mathcal{C}_v &= \frac{R}{(\gamma - 1)} && (2.12) \\ p &= (\gamma - 1)(e - \frac{1}{2}|\mathbf{u}|^2) && (2.13) \end{aligned}$$

where $\gamma = \mathcal{C}_p/\mathcal{C}_v$ is the ratio of the specific heats, $\mathcal{C}_p$ and $\mathcal{C}_v$. T is the fluid temperature and R is the gas constant.

2.2 Rotating Frame Analysis

The Euler's Equations given in Section 2.1 are valid in an inertial or "fixed" frame of reference. A rotor blade in hover conditions can be modeled as a steady-state problem in rotating frame of reference. For this purpose, we will now derive the Euler's equation for rotating frame of reference. Consider a frame of reference

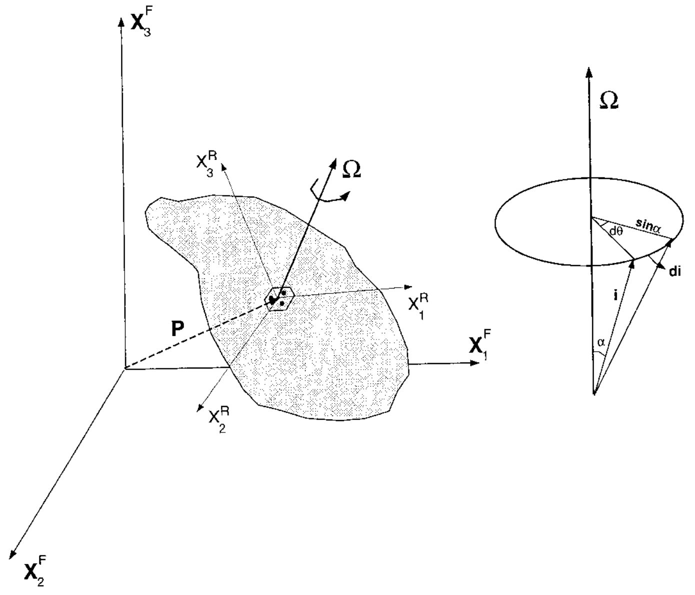

Figure 2.3: Rotating frame of reference.

$\mathbf{X^R}$ rotating at a uniform angular velocity $\vec{\Omega}$ with respect to a fixed frame $\mathbf{X^F}$, as shown in Figure 2.3. Let $\mathbf{P}$ be a vector

$$\mathbf{P} = P_1 i_1 + P_2 i_2 + P_3 i_3. \tag{2.14}$$

To a fixed observer the directions of the rotating unit vectors i_1, i_2 and i_3 change with time. Therefore, the time derivative [21] of the vector $\mathbf{P}$ is

$$\begin{aligned}\left(\frac{d\mathbf{P}}{dt}\right)^F &= \frac{d}{dt}(P_1 i_1 + P_2 i_2 + P_3 i_3), \\ &= \frac{dP_1}{dt} i_1 + \frac{dP_2}{dt} i_2 + \frac{dP_3}{dt} i_3 + P_1 \frac{di_1}{dt} + P_2 \frac{di_2}{dt} + P_3 \frac{di_3}{dt}.\end{aligned} \tag{2.15}$$

The magnitude of the change of $\mathbf{i}$ in time dt is $|d\mathbf{i}| = \sin\alpha d\theta$ (Figure 2.3). Therefore, the magnitude of the rate of change is $|d\mathbf{i}/dt| = \sin\alpha(d\theta/dt) = \vec{\Omega}\sin\alpha$, and the direction of the rate of change is perpendicular to $(\vec{\Omega}, i)$-plane. Thus, $d\mathbf{i}/dt = \vec{\Omega}\times\mathbf{i}$ for any rotating unit vector $\mathbf{i}$ [21]. Using this into Equation 2.15 we obtain

$$\begin{aligned}\left(\frac{d\mathbf{P}}{dt}\right)^F &= \left(\frac{d\mathbf{P}}{dt}\right)^R + P_1\vec{\Omega}\times i_1 + P_2\vec{\Omega}\times i_2 + P_3\vec{\Omega}\times i_3,\\ &= \left(\frac{d\mathbf{P}}{dt}\right)^R + \vec{\Omega}\times\mathbf{P}.\end{aligned} \tag{2.16}$$

Applying Eq. 2.16 to position vector $\mathbf{r}$, and recognizing the fact that $\mathbf{u} = d\mathbf{r}/dt$ we obtain a relation for the velocity vector between "fixed" and "rotating" frame of reference [21]

$$\mathbf{u}^F = \mathbf{u}^R + \vec{\Omega}\times\mathbf{r}. \tag{2.17}$$

Similarly, the accelerations in the two frames are related by [21]

$$\mathbf{a}^F = \mathbf{a}^R + 2\vec{\Omega}\times\mathbf{u}^R + \vec{\Omega}\times(\vec{\Omega}\times\mathbf{r}). \tag{2.18}$$

Now consider a rotating blade with a constant velocity of magnitude Ω and suppose that the $\vec{\Omega}$ is aligned with the x_3^F as shown in Figure 2.4

$$\vec{\Omega} = \Omega x_2^F i - \Omega x_1^F j. \tag{2.19}$$

Using this result, the velocity vector $\mathbf{u^F}$ in fixed-frame of reference takes the following form:

$$\begin{aligned}u_1^F =& u_1^R + \Omega x_2,\\ u_2^F =& u_2^R - \Omega x_1,\\ u_3^F =& u_3^R.\end{aligned} \tag{2.20}$$

For an observer sitting on the blade, the Euler equations 2.7 are still valid, however the velocity vectors have to be interpreted in the rotating frame using Eq. 2.20. For

```
#####
#####
#####
#####
#####
#####       User: adams.scorec.rpi.edu!mdindar
#####
#####      Title: thesis.030
#####
#####    Printed: Tue 16:09 Jan 05, 1999
#####
##### Job number: p1-387
#####
#####
#####
#####
#####
```

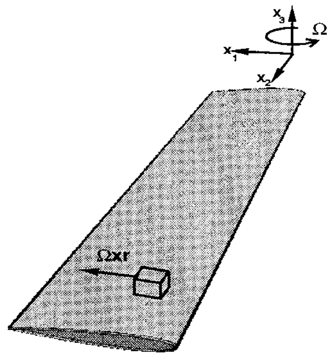

Figure 2.4: A rotating helicopter blade.

example, the continuity equation (Eq. 2.3) becomes

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_1^F}{\partial x_1} + \frac{\partial \rho u_2^F}{\partial x_2} + \frac{\partial \rho u_3^F}{\partial x_3} = 0. \tag{2.21}$$

substituting Eq. 2.20 into Eq. 2.21, we obtain the continuity equation for the rotating frame of reference (with respect to an observer in the fixed frame of reference)

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_1}\left(\rho(u_1^R + \Omega x_2)\right) + \frac{\partial}{\partial x_2}\left(\rho(u_2^R - \Omega x_1)\right) + \frac{\partial \rho u_3^R}{\partial x_3} = 0 \tag{2.22}$$

Similarly, we can extend the same derivation to momentum and energy equations. For brevity, we will derive only the x_1 direction of momentum equation for the rotating frame analysis and then present the results for the vector form of Euler's equations. Substititing the transformed velocity components (Eq. 2.20) into x_1 direction of momentum conservation in Eq. 2.6 we obtain

$$\begin{aligned} &\frac{\partial}{\partial t}\left(\rho(u_1^R + \Omega x_2)\right) + \frac{\partial}{\partial x_1}\left(\rho(u_1^R + \Omega x_2)^2 + p\right) + \\ &\frac{\partial}{\partial x_2}\left(\rho(u_1^R + \Omega x_2)(u_2^R - \Omega x_1)\right) + \frac{\partial}{\partial x_3}\left(\rho(u_1^R + \Omega x_2)u_3^R\right) = 0. \end{aligned} \tag{2.23}$$

Note that, in fixed reference frame, we would like to solve for ρu_1^R, not $\rho(u_1^R + \Omega x_2)$. Therefore, we need to rearrange Eq. 2.23. Expanding Eq. 2.23 and rearranging the terms yields

$$\frac{\partial \rho u_1^R}{\partial t} - \Omega x_2 \underbrace{\left[\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_1^R}{\partial x_1} + \Omega x_2 \frac{\partial \rho}{\partial x_1} + \frac{\partial \rho u_2^R}{\partial x_2} - \Omega x_1 \frac{\partial \rho}{\partial x_2} + \frac{\partial \rho u_3^R}{\partial x_3}\right]}_{\text{Continuity}}$$
$$+ \frac{\partial}{\partial x_1}\left[\rho u_1^R (u_1^R + \Omega x_2) + p\right] + \frac{\partial}{\partial x_2}\left[\rho u_1^R (u_2^R - \Omega x_1)\right] + \frac{\partial}{\partial x_3}\left[\rho u_1^R u_3^R\right]$$
$$= \rho \Omega u_2^F + \rho \Omega^2 x_1. \tag{2.24}$$

We recognize that the second term in Eq. 2.24 is equal to zero since it is the continuity equation (Eq. 2.22). Also we can simplify the right-handside of Eq. 2.24 by using the identity $x_1 = (u_2^F - u_2^R)/\Omega$ given in Eq. 2.20. Finally the the x_1-direction momentum equation in rotating frame of reference is

$$\frac{\partial \rho u_1^R}{\partial t} + \frac{\partial}{\partial x_1}\left(\rho u_1^R u_1^F + p\right) + \frac{\partial}{\partial x_2}\left(\rho u_1^R u_2^F\right) + \frac{\partial}{\partial x_3}\left(\rho u_1^R u_3^R\right) = \rho \Omega u_2^R. \tag{2.25}$$

Simlilar derivations have to be made for x_2, x_3-momentum and energy equations. This is done as follows: First, the velocity components in these equations are replaced by transformed velocity components. Then the terms that are cancelling or adding up to zero are eliminated. Finally, the equations are re-written in a compact form such that common terms are collected under time and space derivatives. Note that only x_1 and x_2-direction momentum equations result in terms that create non-zero entries on the right hand side. For the energy equation the only term that needs to be modified is the velocity term due to kinetic energy. This modification in the energy equation does not generate any source term on the right hand side. After transforming all the equations of conservation law from fixed coordinate system to rotating coordinate system, the Euler's equations may be written in the following form:

$$\frac{\partial U^R}{\partial t} + \frac{\partial F_i^R}{\partial x_i} = \mathcal{R} \tag{2.26}$$

where

$$U^R = \begin{pmatrix} U_1^R \\ U_2^R \\ U_3^R \\ U_4^R \\ U_5^R \end{pmatrix} = \rho \begin{pmatrix} 1 \\ u_1^R \\ u_2^R \\ u_3^R \\ e \end{pmatrix} \tag{2.27}$$

$$F_i^R = \rho u_i^F \begin{pmatrix} 1 \\ u_1^R \\ u_2^R \\ u_3^R \\ e \end{pmatrix} + p \begin{pmatrix} 0 \\ \delta_{1i} \\ \delta_{2i} \\ \delta_{3i} \\ u_i^F \end{pmatrix} \qquad \mathcal{R} = \rho \begin{pmatrix} 0 \\ \Omega u_2^R \\ -\Omega u_1^R \\ 0 \\ 0 \end{pmatrix} \tag{2.28}$$

Note that the source term $\mathcal{R}$ is due to the Coriolis force.

2.3 Symmetrization Using Entropy Variables

Despite the popularity of conservation variables, [23], [24] and [25] investigated the possibility of using a set of new variables, called "entropy variables" in conservation laws. In this section we will introduce and review the transformation of Euler equations from conservative variables to entropy variables.

Theorem 1 (Mock [25]) *A hyperbolic system of conservation laws possessing a generalized entropy function $\mathcal{H}$ becomes symmetric hyperbolic under change of variables*

$$\mathbf{V}^{\mathbf{T}} = \frac{\partial \mathcal{H}}{\partial \mathbf{U}}. \tag{2.29}$$

The generalized entropy function is defined as [27]

$$\mathcal{H}(\mathbf{U}) = -\rho(s - s_0), \tag{2.30}$$

where

$$s = \ln \frac{p}{\rho^\gamma} \tag{2.31}$$

is the non-dimensional entropy and s_0 is its reference value. Using Theorem 1 and the Gibbs relation

$$ds = \frac{1}{T}\left(de + pd(\frac{1}{\rho})\right) \tag{2.32}$$

we obtain the vector of generalized entropy variables

$$\mathbf{V} = \begin{pmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \end{pmatrix} = \frac{1}{e_i}\begin{pmatrix} -U_5 + e_i(\gamma + 1 - s + s_0) \\ U_2 \\ U_3 \\ U_4 \\ -U_1 \end{pmatrix}, \tag{2.33}$$

where

$$\begin{aligned} s &= \ln(\frac{(\gamma-1)e_i}{U_1^\gamma}) && (2.34) \\ e_i &= U_5 - \frac{U_2^2 + U_3^2 + U_4^2}{2U_1} && (2.35) \end{aligned}$$

The inverse mapping, $\mathbf{V} \rightarrow \mathbf{U}$ is given by

$$\mathbf{U} = e_i \begin{pmatrix} -V_5 \\ V_2 \\ V_3 \\ V_4 \\ 1 - \frac{V_2^2+V_3^2+V_4^2}{2V_5} \end{pmatrix}, \tag{2.36}$$

where

$$s = \gamma - V_1 + \frac{V_2^2 + V_3^2 + V_4^2}{2V_5} \tag{2.37}$$

$$e_i = \left(\frac{\gamma - 1}{(-V_5)^\gamma}\right)^{\frac{1}{\gamma-1}} \exp\left(\frac{-s + s_0}{\gamma - 1}\right). \tag{2.38}$$

Now it remains to apply this transformation to Eq. 2.26. But before doing so, let us write Eq. 2.26 in quasi-linear form:

$$\frac{\partial \mathbf{U}}{\partial t} + A_i \frac{\partial \mathbf{U}}{\partial x_i} = \mathcal{R}, \tag{2.39}$$

where $A_i = \partial F_i / \partial U_i$ is the inviscid flux Jacobian. Applying the transformation given in Eq. 2.29 into Eq. 2.39 we obtain

$$\tilde{A}_0 \frac{\partial \mathbf{V}}{\partial t} + \tilde{A}_i \frac{\partial \mathbf{V}}{\partial x_i} = \tilde{\mathcal{R}}, \tag{2.40}$$

where

$$\tilde{A}_0 = \frac{\partial \mathbf{U}}{\partial \mathbf{V}} \tag{2.41}$$

$$\tilde{A}_i = A_i \tilde{A}_0 \tag{2.42}$$

$$\tilde{\mathcal{R}} = \mathcal{R}. \tag{2.43}$$

The coefficient matrices, $\tilde{A}_0$ and $\tilde{A}_i$, possess the following properties [28]

i) $\tilde{A}_0$ is symmetric positive-definite

ii) $\tilde{A}_i$ is symmetric

CHAPTER 3
FINITE ELEMENT FORMULATION

In this chapter we will introduce space-time Galerkin/least-squares [33] finite element formulation for the solution of Euler's equations.

3.1 Space-Time Galerkin/Least-Squares Formulation

Consider the space-time slab, Q_n for the nth time interval, $I_n =]t_n^+ . t_{n+1}^-[$

$$\begin{aligned} Q_n =& \Omega \times I_n \qquad \text{Interior,} \\ P_n =& \Gamma \times I_n \qquad \text{Boundary.} \end{aligned} \tag{3.1}$$

Note that, the Figure 3.1 illustrates a 2-space, time "3-D" example. Within each

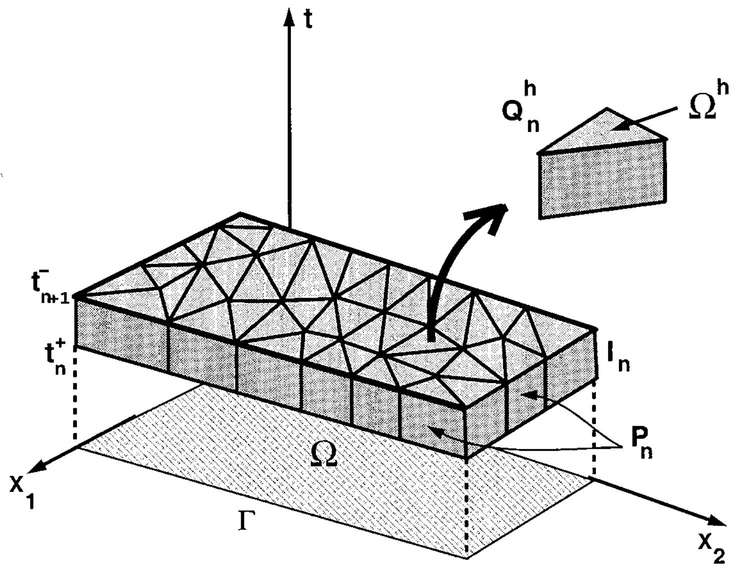

Figure 3.1: A space-time slab.

time-slab, let the domain Ω be partitioned into sub-domains Ω_n^h. Then for each of these space-time finite elements we define the following function spaces:

$$\mathcal{S}^h = \{\mathbf{V}^h | \mathbf{V}^h \in [H^1(Q_n)]^{N_d}, \mathbf{V}^h|_{Q_n^h} \in \left(\mathcal{P}_k(Q_n^h)\right)^{N_d}, q_i(\mathbf{V^h})|_{P_{n}g_i} = g_i(t)\} \quad (3.2)$$

$$\mathcal{V}^h = \{\mathbf{W}^h | \mathbf{W}^h \in [H^1(Q_n)]^{N_d}, \mathbf{W}^h|_{Q_n^h} \in \left(\mathcal{P}_k(Q_n^h)\right)^{N_d}, q_i(\mathbf{W^h})|_{P_{n}g_i} = 0\} \quad (3.3)$$

where $\mathcal{P}_k$ is the kth order interpolation polynomial for approximating the trial solution, V^h and weighting functions, W^h. N_d is the number of degrees of freedom per node, which is 4 for 2D-space and 5 for 3D-space formulation. Also, $q_i : R^{N_d} \mapsto R^i$, $0 \leq i \leq N_d$, are nonlinear boundary condition transformation functions and $g_i(t)$ is the prescribed boundary condition [28].

The finite element functions are considered to be discontinuous between two space-time slabs and the *jump* in time for function $\mathbf{W}$ is denoted by

$$[\![\mathbf{W}(t_n)]\!] \equiv \mathbf{W}(t_n^+) - \mathbf{W}(t_n^-). \quad (3.4)$$

Having defined the finite element space, now we can state our Galerkin/least-squares weighted residual formulation as follows; For each $Q_n, n = 0, \ldots, N-1$, find $\mathbf{V}^h \in \mathcal{S}_n^h$ such that for all $\mathbf{W}^h \in \mathcal{V}_n^h$ the following variational equation is satisfied [33]

$$\begin{aligned}
&\int_{Q_n} \left(-\frac{\partial \mathbf{W}^h}{\partial t} \cdot \mathbf{U}(\mathbf{V}^h) - \frac{\partial \mathbf{W}^h}{\partial x_i} \cdot \mathbf{F}_i(\mathbf{V}^h) \right) dQ + \\
&\int_{\Omega} \mathbf{W}^h(t_n^+) \cdot \left(\mathbf{U}(\mathbf{V}^h(t_{n+1}^-)) - \mathbf{U}(\mathbf{V}^h(t_n^-)) \right) d\Omega + \\
&\sum_{e=1}^{(n_{el})_n} \int_{Q_n^h} \left(\mathcal{L}\mathbf{W}^h \right) \tau \left(\mathcal{L}\mathbf{V}^h - \mathcal{R} \right) dQ^h + \\
&\sum_{e=1}^{(n_{el})_n} \int_{Q_n^h} \nu^h \nabla_\xi \mathbf{W}^h \cdot \mathrm{diag}(\tilde{A}_0) \cdot \nabla_\xi \mathbf{V}^h dQ \\
= &\int_{Q_n} \mathbf{W}^h \cdot \mathcal{R} dQ - \int_{P_n} \mathbf{W}^h \cdot \mathbf{F}_i(\mathbf{V}^h) \cdot n_i dP
\end{aligned} \quad (3.5)$$

Remarks

(1) The first, second and the last integrals in Eq. 3.5 are due to time-discontinuous Galerkin formulation.

(2) All the flux terms in time-discontinuous Galerkin formulation are written in integrated-by-part form, which results in conservation of fluxes under inexact quadrature rules [28].

(3) The second integral in Eq. 3.5 is the result of integration-by-parts of the term $\mathbf{W}^h \partial \mathbf{U}(\mathbf{V}^h)/\partial t$ and is refered to as *jump condition.* We designate the jump condition in short by

$$(\mathbf{W}^h(t_n^+), [\![\mathbf{U}\left(\mathbf{V}^h(t_n)\right)]\!]) = \int_\Omega \mathbf{W}^h(t_n^+) \cdot \left(\mathbf{U}(\mathbf{V}^h(t_{n+1}^-)) - \mathbf{U}(\mathbf{V}^h(t_n^-))\right) d\Omega. \tag{3.6}$$

Equation 3.6 imposes *weakly* the continuity between time t_n^- and t_n^+ [33].

(4) The third integral in Eq. 3.5 is the *least-squares* operator. $\mathcal{L}$ is defined as the compressible Euler differential operator:

$$\mathcal{L} = \tilde{A}_0 \frac{\partial}{\partial t} + \tilde{A}_i \frac{\partial}{\partial x_i} \tag{3.7}$$

The least-squares operator is a stabilization factor to otherwise unstable Galerkin formulation. τ is a $N_d \times N_d$ matrix that will be defined later in this chapter. Note that the least-squares term operates only over element interiors.

(5) The fourth integral is the *discontinuity-capturing* operator. The role of this term is to remove solution oscillations near discontinuities.

Consistency, stability and accuracy are three important features of any finite element formulation. Euler equations are nonlinear hyperbolic partial differential equations and as of yet proof of consistency, stability and accuracy for nonlinear hyperbolic systems do not exist. Nevertheless, proof of consistency, stability and accuracy of linear hyperbolic systems, which may be thought of as analog to nonlinear systems, can be used to gain insight into the capability of a finite element formulation. Proof of consistency, stability and accuracy of Galerkin/least-squares formulation for linear hyberbolic systems can be found in reference [34]. In the next section we will use one-dimensional linear advection equation to analyze stability and accuracy of Galerkin/least-squares formulation and gain more insight into the role of least-squares operator.

3.2 Model Equation

To understand the behavior of the Galerkin/least-squares method better, we can analyze the linear one-dimensional advective equation which can be considered to be as a model to Euler equations. For a given function ϕ, the one dimensional linear scalar advection equation [29] is

$$\frac{\partial \phi}{\partial t} + a\frac{\partial \phi}{\partial x} = 0 \qquad \text{for} \quad x \in [0, L[, \quad t > 0 \tag{3.8}$$

where a is a constant advection speed.

The Galerkin/least-squares formulation of Eq. 3.8 is as follows; For $n = 0, 1, \ldots, N-1$, find $\phi^h \in \mathcal{S}_n^h$ such that for all $w^h \in \mathcal{V}_n^h$, the following variational formulation will be satisfied:

$$\begin{aligned} &\int_{Q_n} \left(-w^h_{,t}\phi^h - w^h_{,x}a\phi^h\right) dxdt + \int_{\Omega} w^h(t_n^+)\left(\phi^h(t_n^+) - \phi^h(t_n^-)\right) dx \\ &+\sum_{e=1}^{n_{el}} \int_{Q_n} \left(w^h_{,t} + aw^h_{,x}\right) \tau \left(\phi^h_{,t} + a\phi^h_{,x}\right) dxdt = 0 \end{aligned} \tag{3.9}$$

where, n is the element number and N is the total number of nodes. We use the following short notation for the Galerkin/least-squares formulation

$$\mathbf{B}_{GLS}\left(w^h, \phi^h\right) = 0 \tag{3.10}$$

$$\mathbf{B}_{GLS}\left(w^h, \phi^h\right) = \mathbf{B}\left(w^h, \phi^h\right) + \sum_{e=1}^{n_{el}} \left(\tau\hat{\mathcal{L}}w^h, \hat{\mathcal{L}}\phi^h\right) \tag{3.11}$$

$$\mathbf{B}\left(w^h, \phi^h\right) = \int_{Q_n} \left(-w^h_{,t}\phi^h - w^h_{,x}a\phi^h\right) dxdt \tag{3.12}$$

$$+ \int_{\Omega} w^h(x, t_{n+1}^+)[\![\phi^h(x,t)]\!] \tag{3.13}$$

where $\hat{\mathcal{L}}$ is defined as

$$\hat{\mathcal{L}} = \frac{\partial}{\partial t} + a\frac{\partial}{\partial x} \tag{3.14}$$

To study the stability and accuracy of Eq. 3.9, we consider *constant-time* and *linear-space* interpolation of ϕ^h and w^h as follows:

$$\phi^h(x,t) = \sum_{b=1}^{2} N_b(x)\phi^h_{b,n+1} \quad ; \quad w^h(x,t) = \sum_{b=1}^{2} N_b(x)w^h_{b,n+1} \tag{3.15}$$

where

$$N_j(x) = \begin{cases} \frac{x-x_{j-1}}{\Delta x} & \text{if } x_{j-1} \le x \le x_j, \\ \frac{x_{j+1}-x}{\Delta x} & \text{if } x_j \le x \le x_{j+1}, \\ 0 & \text{elsewhere.} \end{cases} \tag{3.16}$$

Substituting Eq. 3.16 into Eq. 3.9, a semi-discrete form of Galerkin/least-squares method is

$$\Delta t \int_{x_j}^{x_{j+1}} \left(-(N_j)_{,x} a \sum_{k=1}^{2} N_k {\phi^h}_{k,n+1} \right) dx + \Delta t \int_{x_j}^{x_{j+1}} \tau a^2 N_{j,x} \sum_{k=1}^{2} N_k {\phi^h}_{k,n+1} dx \tag{3.17}$$

$$\int_{x_j}^{x_{j+1}} N_j \sum_{k=1}^{2} N_k {\phi^h}_{k,n+1} dx - \int_{x_j}^{x_{j+1}} N_j \sum_{k=1}^{2} N_k {\phi^h}_{k,n} dx = 0 \tag{3.18}$$

Carrying the integration process and collecting the terms at nodes $j-1, j$, and $j+1$, we obtain the following *algorithmic equation*

$$\boldsymbol{A}{\phi^h}_{j-1,n+1} + \boldsymbol{B}{\phi^h}_{j,n+1} + \boldsymbol{C}{\phi^h}_{j+1,n+1} = \frac{\Delta x}{6}{\phi^h}_{j-1,n} + \frac{2}{3}\Delta x{\phi^h}_{j,n} + \frac{\Delta x}{6}{\phi^h}_{j+1,n}, \tag{3.19}$$

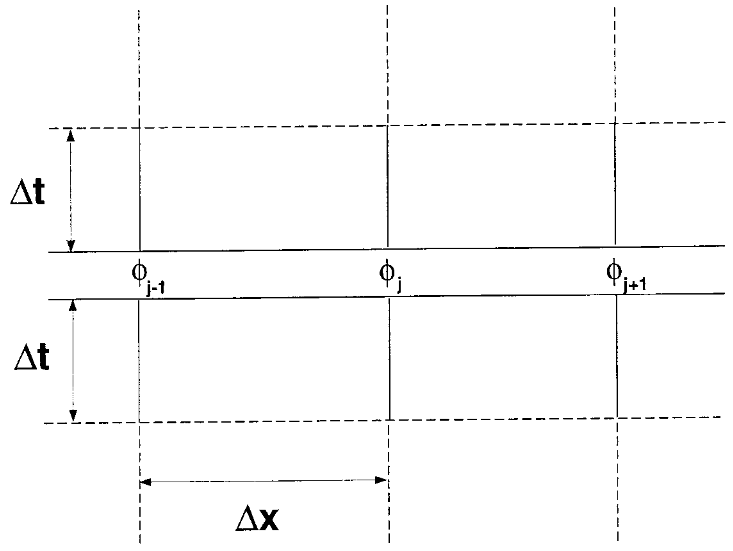

Figure 3.2: Space-time mesh for constant-time linear-space FE approximation.

where

$$\boldsymbol{A} = -\frac{a\Delta t}{2} - \frac{\Delta t\tau a^2}{\Delta x} + \frac{\Delta x}{6} \tag{3.20}$$

$$\boldsymbol{B} = \frac{2\Delta t\tau a^2}{\Delta x} + \frac{2\Delta x}{3} \tag{3.21}$$

$$\boldsymbol{C} = \frac{a\Delta t}{2} - \frac{\Delta t\tau a^2}{\Delta x} + \frac{\Delta x}{6}. \tag{3.22}$$

Now we would like to use Fourier analysis to study the stability and accuracy of Eq. 3.19. For this purpose, let

$$\phi^h{}_{j,n} \equiv e^{\nu^h n\Delta t + \sqrt{-1}Kj}, \tag{3.23}$$

$$\nu_h = -\xi^h + \sqrt{-1}\eta^h, \tag{3.24}$$

where ξ^h and η^h are the *damping* and *frequency* coefficients, respectively. We also define an *amplification* factor in the following form:

$$\zeta^h \equiv e^{\nu^h \Delta t} \tag{3.25}$$

and require that for stability

$$|\zeta^h| \leq 1. \tag{3.26}$$

Substituting the definition in Eq. 3.24 into Eq. 3.19, we obtain the following relation for the damping coefficient

$$\zeta^h = \frac{\frac{2}{3} + \frac{1}{3}\cos(K)}{\frac{2}{3} + \frac{1}{3}\cos(K) + \frac{2a^2\tau\Delta t}{\Delta x^2}\left(1 - \cos(K)\right) + \sqrt{-1}\frac{a\Delta t}{\Delta x}\sin(K)} \tag{3.27}$$

Invoking the stability condition given by Eq. 3.26, we see that Galerkin/least-squares method is *stable* provided that

$$\tau_{opt} = \tau = \frac{\Delta x}{2a} \tag{3.28}$$

Note that the optimal value of τ is obtained by equating Eq. 3.27 to 1 and solving for τ.

Remarks

(1) Note that τ has dimensions of time

$$\tau = \frac{\Delta x}{2a} \rightarrow \frac{[L]}{[L]/[T]} \rightarrow [T] \tag{3.29}$$

(2) For $\tau = 0$, we recover the standard Galerkin formulation. Galerkin method for the pure advection equation is *neutrally stable* and it exhibits high-frequency oscillations.

In order to determine the accuracy of the Galerkin/least-squares formulation given in Eq. 3.9, we will look at the first form of *modified differential equation* as suggested by [26]. To obtain the modified differential equation, we first expand ϕ in Eq. 3.19 into its Taylor series as follows:

$$\begin{aligned}\phi^h_{j\mp1,n\mp1} =& \phi^h_{j,n} \mp \phi^h_{,x}\Delta x \mp \phi^h_{,t}\Delta t + \frac{1}{2}\left(\phi^h_{,xx}\Delta x^2 + \phi^h_{,tt}\Delta t^2\right) \\ &\mp \frac{1}{6}\left(\phi^h_{,xxx}\Delta x^3 + \phi^h_{,ttt}\Delta t^3\right) + \frac{1}{24}\left(\phi^h_{,xxxx}\Delta x^4 + \phi^h_{,tttt}\Delta t^4\right) + \cdots\end{aligned} \tag{3.30}$$

Substituting Eq. 3.30 into Eq. 3.19 and then rearranging the terms such that the original PDE operator $\hat{\mathcal{L}}\phi^h$ is collected on the left hand side of the equation, we obtain the following relation

$$\hat{\mathcal{L}}\phi^h = \left(-\frac{1}{2}\phi^h{}_{,tt}\right)\Delta t - \left(a^2\phi^h{}_{,xx}\right)\tau + \cdots \tag{3.31}$$

This is the first form of modified differential equation for Galerkin/least-squares formulation. It is clear from this relation that the method is $O(\Delta t, \tau)$ accurate.

3.3 Least-Squares Operator

In the previous section we stated the behavior of least-squares parameter $\boldsymbol{\tau}$ for one-dimensional model problem. Although it is easy to obtain the exact form of $\boldsymbol{\tau}$ in one dimension, the problem is much more complicated in two and three dimensional cases. More information about the design of $\boldsymbol{\tau}$ matrix can be found in [28].

To begin with, the least-squares term in Eq. 3.5 is re-written for a space-time element Q_n^h as follows:

$$\int_{Q_n^h} \left(\mathcal{L}\mathbf{W}^h\right)\cdot\boldsymbol{\tau}\left(\mathcal{L}\mathbf{V}^h\right) dQ = \int_{Q_n^h} \hat{\nabla}_\xi \mathbf{W}^h \cdot \hat{\mathbf{K}}_\xi \hat{\nabla}_\xi \mathbf{V}^h dQ, \tag{3.32}$$

where

$$\begin{aligned} \hat{\mathbf{K}}_\xi &= \hat{\mathbf{A}}_\xi \boldsymbol{\tau} \hat{\mathbf{A}}_\xi^T, & (3.33)\\ \hat{\mathbf{A}}_\xi &= \left[\frac{\partial \xi_0}{\partial x_0}\tilde{\mathbf{A}}_0, \frac{\partial \xi_1}{\partial x_i}\tilde{\mathbf{A}}_i, \cdots, \frac{\partial \xi_d}{\partial x_i}\tilde{\mathbf{A}}_i\right]^T. & (3.34)\end{aligned}$$

Note that ξ is the element local coordinate and the $\hat{\nabla}_\xi$ is the gradient operator vector,

$$\hat{\nabla}_\xi = \begin{bmatrix} I \\ \nabla_\xi \\ \nabla_\xi^2 \end{bmatrix}. \tag{3.35}$$

In [28], $\boldsymbol{\tau}$ is designed such that

$$\hat{\mathbf{K}}_\xi = \left(\hat{\mathbf{A}}_\xi \tilde{\mathbf{A}}_0^{-1} \hat{\mathbf{A}}_\xi^T\right)^{\frac{1}{2}}. \tag{3.36}$$

Using Eq. 3.36 and the definition of Jacobian matrices from Eq. 3.35, we write the final form of $\boldsymbol{\tau}$ as follows:

$$\boldsymbol{\tau} = \tilde{\mathbf{A}}_0^{-1} \left(\left(\frac{\partial \xi_0}{\partial x_0}\right)^2 I_m + \left(\frac{\partial \xi_i}{\partial x_j}\frac{\partial \xi_i}{\partial x_k}\right) \mathbf{A_j A_k} \right)^{-\frac{1}{2}}. \tag{3.37}$$

3.4 Discontinuity Capturing Operator

As stated before, the role of the discontinuity-capturing operator is to remove the oscillations near discontinuities. For this purpose, the discontinuity-capturing operator should act in the direction of the gradient. For consistency, it should be proportional to the residual, $\mathcal{L}\mathbf{V^h}$, and for accuracy it should vanish in smooth regions of the flow. In [28], two forms of discontinuity-capturing operator are defined that satisfy the conditions we just mentioned. First, a *linear form*

$$\nu_L^h = \frac{\left|\mathcal{L}^h\right|_\tau}{\left|\mathrm{diag}(\tilde{A}_0)\hat{\nabla}_\xi \mathbf{V}^h\right|_\tau} \tag{3.38}$$

and second, a *quadratic form*

$$\nu_Q^h = 2\frac{\left|\mathcal{L}\mathbf{V}^h\right|_\tau^2}{\left|\hat{\nabla}_\xi \mathbf{V}^h\right|_{\tilde{A}_0}^2} \tag{3.39}$$

In the current implementation of Galerkin/least-squares formulation, both the linear and quadratic forms of discontinuity-capturing operators are implemented. In practical calculations the following logic is used to select the discontinuity-capturing factor ν to minimize the diffusion

$$\nu^h = \min\left(\max\left(\nu_L^h, 0\right), \nu_Q^h\right) \tag{3.40}$$

3.5 Boundary Conditions

In Chapter 2 we talked about the advantages of using *entropy* variables instead of *conservative* variables. In CFD applications the user usually wants to assign boundary conditions in terms of quantities such as velocity, pressure and temperature. From the practical point of view, it is clear that assigning boundary conditions to Eq. 3.5 in terms of entropy variables may not be as intuitive as assigning them in terms of primitive variables. Therefore, we have to establish the proper transformation for boundary conditions such that they are expressed in terms of conservative variables but converted to entropy variables for the solution of the variational formulation.

In three dimensions, consider the following set of primitive variables to be our boundary condition set

$$\begin{bmatrix} \rho \\ \hat{\mathbf{u}} \\ T \\ p \end{bmatrix} = \begin{cases} \text{density} \\ \text{velocity} \\ \text{temperature} \\ \text{pressure} \end{cases} \tag{3.41}$$

where $\hat{\mathbf{u}}$ is the local velocity vector given by

$$\hat{\mathbf{u}} = \begin{bmatrix} u_r \\ u_s \\ u_t \end{bmatrix} = \begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 \\ \beta_1 & \beta_2 & \beta_3 \\ \delta_1 & \delta_2 & \delta_3 \end{bmatrix} \begin{bmatrix} u_1^R \\ u_2^R \\ u_3^R \end{bmatrix} \tag{3.42}$$

where α, β and δ are direction cosines of an orthogonal coordinate system (r, s, t) with respect to the global coordinate system (x_1, x_2, x_3).

Using the transformation $\mathbf{U} \rightarrow \mathbf{V}$, the boundary condition set in 3.41 is written as

$$\begin{bmatrix} \rho \\ u_r \\ u_s \\ u_t \\ T \\ p \end{bmatrix} = \begin{bmatrix} \bar{\gamma}^{\frac{1}{\bar{\gamma}}} \left(-V_5\right)^{\frac{-1}{\bar{\gamma}}} \exp\left(\frac{-1}{\bar{\gamma}} \left(\gamma - V_1 + \frac{V_2^2+V_3^2+V_4^2}{2V_5}\right)\right) \\ \alpha_1 \left(-\frac{V_2}{V_5}\right) + \alpha_2 \left(-\frac{V_3}{V_5}\right) + \alpha_3 \left(-\frac{V_4}{V_5}\right) \\ \beta_1 \left(-\frac{V_2}{V_5}\right) + \beta_2 \left(-\frac{V_3}{V_5}\right) + \beta_3 \left(-\frac{V_4}{V_5}\right) \\ \delta_1 \left(-\frac{V_2}{V_5}\right) + \delta_2 \left(-\frac{V_3}{V_5}\right) + \delta_3 \left(-\frac{V_4}{V_5}\right) \\ \frac{1}{C_v} \left(-\frac{1}{V_5}\right) \\ \bar{\gamma}^{\frac{\gamma}{\bar{\gamma}}} \left(-V_5\right)^{\frac{-\gamma}{\bar{\gamma}}} \exp\left(\frac{-1}{\bar{\gamma}} \left(\gamma - V_1 + \frac{V_2^2+V_3^2+V_4^2}{2V_5}\right)\right) \end{bmatrix} \tag{3.43}$$

where γ is the ratio of specific heats and $\bar{\gamma} = \gamma - 1$.

The natural boundary conditions on the inviscid flux vector $\mathbf{F}_i^R$ is satisfied by calculating the integral,

$$\int_{P_n} \mathbf{W}^h \cdot \left(-\mathbf{F}_i^R(\mathbf{V}^h)\right) n_i dP, \tag{3.44}$$

where n_i is the ith component of the unit outward normal to the space time boundary, P_n. Using the definition of $\mathbf{F}_i^R$ from Eq. 2.28 yields

$$\int_{P_n} \mathbf{W}^h \cdot (-\rho u_n^F \begin{pmatrix} 1 \\ u_1^R \\ u_2^R \\ u_3^R \\ e \end{pmatrix} - p \begin{pmatrix} 0 \\ \delta_{1i} \\ \delta_{2i} \\ \delta_{3i} \\ u_i^F \end{pmatrix})dP \tag{3.45}$$

This boundary integral provides the following boundary conditions:

1) *Normal mass flux* $\rightarrow \rho u_n^R = h^m$

2) *Pressure* $\rightarrow p = h^p$

where h^m and h^p are prescribed data [28].

CHAPTER 4
AN ADAPTIVE SOLUTION TECHNIQUE

4.1 Overview

The ultimate goal of CFD is to approximate the physical flows that we encounter in nature, as accurately as possible using numerical techniques. However, the cost of accuracy of any CFD procedure has to be matched by enough computer power and storage. Practical problems usually are the most complex. The flow-field around a complete aircraft, the internal flow structure of a turbojet engine with several stages and hundreds of blades and passages, the detailed effects of a wake and vortex flow generated by a helicopter rotor system on rotorcraft aerodynamics, aeroacoustic and aeroelasticity problems are some major examples that aeronautical engineers have to solve today and in the future. These large-scale problems require huge amounts of computer power and storage. Despite the steady growth in computer speed and power, complexity of modern problems has made us look for alternative solution procedures.

Adaptive solution procedures are one possible avenue to attack the solution of large-scale CFD problems. Coupled with parallel computing strategies, adaptive solution techniques can be very effective. The main advantage of the parallel-adaptive procedure is its ability to distribute a large problem onto several CPU units, then locally adjust either the resolution of the computational mesh or increase/decrease the degree of the interpolating polynomials used in the variational formulation. The adaptive procedures can be classified into the following four categories: **h**-, **p**-, **hp**- and **r**-adaptive. Both **h**- and **r**-adaptive procedures are based on modifying the computational mesh based on an error estimate to increase/decrease resolution accordingly. Whereas **p**-adaptive scheme modifies the degree of the interpolating polynomials in the finite element formulation again guided by a measure of the computational error. Alternatively, an **hp**-adaptive method is a hybrid of **h**- and **p**-adaptive procedures.

In an **h**-adaptive process, the main idea is to subdivide (refine) the computational mesh into smaller elements to increase the resolution or collapse (de-refine) a group of small elements into one or more larger elements. Again, the decision to refine or de-refine is based on the local measure of the computational error. The areas that have errors higher than some allowable tolerance are refined and the areas that have lower error measures are de-refined. Examples of h-adaptive calculations for CFD problems can be found in [59], [36] and [31].

With respect to **h**-adaptive procedures, a **p**-adaptive method keeps the computational mesh the same but varies the order of the finite element basis locally over the domain. Although examples of **p**- and **hp**-adaptive procedures are scarce for hyperbolic problems [53], succesfull applications are described in [60], [61], [62] for elliptic and parabolic problems.

Finally **r**-adaptive procedure keeps the initial size of a computational mesh the same, but moves the nodes in the computational mesh in such a way that the error is distributed more or less equally in the entire computational domain. Examples of **r**-refinement can be found in [37] and [52].

In this section, we will describe an **h**-adaptive procedure for computing rotor-blade wakes in the framework of parallel Galerkin/least-squares formulation described in chapter 3.

4.2 Preliminaries

Let us consider Ω as an arbitrary open bounded domain in Euclidean space R^3. Also, let $\mathcal{F}$ be a smooth function defined on Ω. The Lebesgue space $L_p(\Omega)$ consists of equivalence classes of $\mathcal{F}$, whose absolute values have p^{th} powers which are Lebesgue-integrable on Ω, where $p \geq 1$ with an L_p-norm of $\mathcal{F}$ given by [37]

$$||\mathcal{F}||_{L_p(\Omega)} = \left(\int_\Omega |\mathcal{F}|^p d\Omega\right)^{\frac{1}{p}} < \infty \tag{4.1}$$

The most commonly used norm of Lebesgue space is the L_2 norm

$$||\mathcal{F}||_{L_2(\Omega)} = \left(\int_\Omega |\mathcal{F}|^2 d\Omega\right)^{\frac{1}{2}} . \tag{4.2}$$

Furthermore, when all weak partial derivatives of $\mathcal{F}$ of order $\leq$ m are in $L_p(\Omega)$, we say that $\mathcal{F}$ belongs to a space denoted by $\mathcal{W}_p^m(\Omega)$ and called the Sobolev space of order m, p on Ω [36]. The norm of $\mathcal{F}$ in Sobolev space $\mathcal{W}_p^m(\Omega)$ is defined as

$$||\mathcal{F}||_{\mathcal{W}_p^m(\Omega)} = \left(\int_\Omega \sum_{|\boldsymbol{\alpha}| \leq m} |D^{\boldsymbol{\alpha}} \mathcal{F}|^p d\Omega \right)^{\frac{1}{p}} \tag{4.3}$$

where $D^{\boldsymbol{\alpha}}$ is a derivative operator of order $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \alpha_3]^T$ and it is defined as

$$D^{\boldsymbol{\alpha}} \equiv \frac{\partial^{\alpha_1+\alpha_2+\alpha_3}}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2} \partial x_3^{\alpha_3}}, \qquad \alpha_1, \alpha_2, \alpha_3 \geq 0. \tag{4.4}$$

Equivalence between Lebesgue and Sobolev norms is given by

$$||\mathcal{F}||_{\mathcal{W}_p^m(\Omega)} = \left(\sum_{|\boldsymbol{\alpha}| \leq m} ||D^{\boldsymbol{\alpha}} \mathcal{F}||^p_{L_p(\Omega)} \right)^{\frac{1}{p}} \tag{4.5}$$

Another frequently used measure is the semi-norm. The semi-norm space consists of those functions whose generalized derivatives of order m are in L_p^m.

$$|\mathcal{F}|_{L_p^m(\Omega)} = \left(\int_\Omega \sum_{|\boldsymbol{\alpha}| = m} |D^{\boldsymbol{\alpha}} \mathcal{F}|^p d\Omega \right)^{\frac{1}{p}} \tag{4.6}$$

For the remainder of this chapter, we will be using the following short notations to denote full and semi-norm

$$||\cdot||_{m,p} \equiv ||\cdot||_{\mathcal{W}_p^m(\Omega)} \tag{4.7}$$

$$|\cdot|_{m,p} \equiv |\cdot|_{L_p^m(\Omega)} \tag{4.8}$$

Theorem 2 *Let Ω^h be an arbitrary convex sub-domain (a finite element) of Ω over which $\mathcal{F}$ is interpolated by a function $\mathcal{F}^h$ which contains complete polynomials of degree k. Then the* **local interpolation error** *in the $\mathcal{W}_p^m(\Omega^h)$ semi-norm is*

$$\begin{aligned} |\mathcal{F}^h - \mathcal{F}|_{m,p} &\leq Ch^\gamma |\mathcal{F}|_{k+1,p} \\ \text{with} \quad \gamma &= \frac{n}{p'} - \frac{n}{p} + k + 1 - m \end{aligned} \tag{4.9}$$

where h is a defined geometric size of Ω^h, n is the Euclidean space dimension of Ω, p' is $p/(p-1)$ and C is a constant independent of $\mathcal{F}$ and h.

Remark

- The interpolation error estimate in Eq. 4.9 is usually restricted to $m = 0, 1$ because typically $\mathcal{F} \in H^1 \cap H^{k+1}$ [55].

As an example, consider a one-dimensional function $f(x)$ defined over Ω and $f^h(x)$ be a finite element approximation of $f(x)$ using a linear interpolating polynomial. Over an element Ω^h, we can write f^h as

$$f^h = f_1(1-\xi) + f_2\xi \tag{4.10}$$

where, $\xi = x/h$ and f_1, f_2 are the nodal values of f^h on Ω^h. We can expand f_2 into its Taylor series around f_1 as follows

$$f_2 = f_1 + \frac{df_1}{d\xi} + \frac{1}{2}\frac{d^2 f_1}{d\xi^2} + \cdots \tag{4.11}$$

Similarly, $f(x)$ can be expanded into its Taylor series around f_1

$$f = f_1 + \frac{df_1}{d\xi}\xi + \frac{1}{2}\frac{d^2 f_1}{d\xi^2}\xi^2 + \cdots \tag{4.12}$$

Using Equations 4.10, 4.11 and 4.12, the truncation error $e = f^h - f$ is computed as

$$|e| \leq \frac{1}{2}(\xi - \xi^2)|\frac{d^2 f_1}{d\xi^2}| \tag{4.13}$$

Or,

$$|e| \leq \frac{1}{2}(\xi - \xi^2)h^2|\frac{d^2 f_1}{dx^2}|. \tag{4.14}$$

Notice the similarity of the terms between Eq. 4.14 and the definition of the interpolation error estimate in Eq. 4.9.

4.3 Error Indicator Based on Interpolation Estimate

Understanding the flow fields with vortical structures and adaptive mesh refinement requires an understanding of finite element approximation. Interpolation errors exist due to finite dimensional space approximation and depend on the order of the finite element basis. Therefore, an error indicator that utilizes derivatives of velocities in a vortical flow field has to be based on the interpolation error of a particular finite element approximation.

Let us seek L_2-norms of errors of the vector field $\vec{u}$ in a finite element procedure which utilizes piecewise linear polynomials in R^3. Using Eq. 4.9, the semi-norm of the interpolation error is given by

$$|\varepsilon^h|_{0,2} \leq Ch^2|\vec{u}^h|_{2,2}. \tag{4.15}$$

In this case, evaluation of the error requires second partial derivatives of $\vec{u}^h$. Let us denote the finite element error indicator by the symbol, θ_i, then a practical error indicator using the L_2-norm of the second derivatives of velocity vector is written as

$$\epsilon_i = h^2\Big(\int_{\Omega^h} \sum_{i+j+k=2} |\frac{\partial^2 \vec{u}_i^h}{\partial x_1^i \partial x_2^j \partial x_3^k}|^2 d\Omega^h\Big)^{1/2}. \tag{4.16}$$

Note that Eq. 4.16 requires the evaluation of second order derivatives which can be achieved by a gradient recovery (smoothing) procedure given as follows,

$$\nabla_r \vec{u}^h = \sum_{\Omega^h \ni P} \frac{\mathcal{V}_{\Omega^h}}{\mathcal{V}} \nabla \vec{u}^h \tag{4.17}$$

Here, Eq. 4.17 constructs a piecewise linear distribution of $\nabla_r \vec{u}$ that is exact for a *quadratic* solution and is $O(h^2)$ in general [38] [39]. Note that, Eq. 4.18 is simply a weighted average of $\nabla \vec{u}$ on a polytope of elements P, where $\mathcal{V}$ and $\mathcal{V}_{\Omega^h}$ are the volume of the polytope and element Ω^h, respectively. After recovering the first derivatives, the second derivatives can be approximated using the derivatives of interpolation functions into $\nabla_r \vec{u}$ on Ω^h.

Another approach to evaluate second derivatives of flow variables on linear elements, is to re-construct the solution data using a quadratic polynomial as a basis function [39],

$$\vec{u}(x,y,z)_0 = \vec{u}_0 + \Delta r^T \nabla \vec{u}_0 + \frac{1}{2}\Delta r^T H_0 \Delta r + O(\Delta r^3) \tag{4.18}$$

where, $\Delta r = x_i - x_0$ and H_0 is the Hessian matrix of second derivatives

$$H_0 = \begin{bmatrix} \vec{u}_{xx} & \vec{u}_{xy} & \vec{u}_{xz} \\ \vec{u}_{yx} & \vec{u}_{yy} & \vec{u}_{yz} \\ \vec{u}_{zx} & \vec{u}_{zy} & \vec{u}_{zz} \end{bmatrix} \tag{4.19}$$

Given the solution data at the vertices of the mesh, we seek to minimize the L_2 norm of the distance between $\vec{u}$ sampled at the vertex v_0 and the element quadrature points. For a polytope of elements surrounding vertex v_0, this operation yields a non-square matrix problem of the form;

$$\begin{bmatrix} L_1^1 & L_1^2 & \cdots & L_1^8 & L_1^9 \\ L_2^1 & L_2^2 & \cdots & L_2^8 & L_2^9 \\ L_3^1 & L_3^2 & \cdots & L_3^8 & L_3^9 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ L_i^1 & L_i^2 & \cdots & L_i^8 & L_i^9 \\ \vdots & \vdots & \cdots & \vdots & \vdots \end{bmatrix} \begin{pmatrix} \vec{u}_x \\ \vec{u}_y \\ \vec{u}_z \\ \vec{u}_{xy} \\ \vdots \\ \vec{u}_{zz} \end{pmatrix} = \begin{pmatrix} \Delta\vec{u}_1 \\ \Delta\vec{u}_2 \\ \Delta\vec{u}_3 \\ \vdots \\ \Delta\vec{u}_i \\ \vdots \end{pmatrix} \tag{4.20}$$

where, L_i is a row vector defined as;

$$L_i = \begin{bmatrix} \Delta x_i & \Delta y_i & \Delta z_i & \frac{1}{2}\Delta x_i^2 & \cdots & \Delta y_i \Delta z_i & \frac{1}{2}\Delta z_i^2 \end{bmatrix} \tag{4.21}$$

and $\Delta\vec{u} = \vec{u}_{\Omega^h} - \vec{u}_0$.

Solution of this non-square matrix equation can be obtained by using a least-squares solver such as the Singular Value Decomposition (SVD). Note that, regardless of which method we use to calculate the second derivatives, they are going to be *constant* for linear element. Therefore, Eq. 4.16 can be further simplified and written in the following form

$$\epsilon = h^2\sqrt{V}\Big(|\frac{\partial^2 u}{\partial x^2}|^2 + |\frac{\partial^2 u}{\partial x \partial y}|^2 + |\frac{\partial^2 u}{\partial x \partial z}|^2 + |\frac{\partial^2 u}{\partial y^2}|^2 + |\frac{\partial^2 u}{\partial y \partial z}|^2 + |\frac{\partial^2 u}{\partial z^2}|^2\Big)^{1/2} \tag{4.22}$$

Note that the true interpolation error is $|\varepsilon|_{0,2}$, whereas Eq.4.22 measures $|\varepsilon|_{0,2}/C$. Therefore, unless C is unbounded, the error indicator should measure qualitatively the right convergence rate in the L_2 semi-norm of u and it should be of $O(h^2)$, in general.

4.3.1 Convergence Rates for Model Problems

In this section, we present convergence rate measurements of the error indicator described in the previous section. For this purpose, we use one incompressible and one compressible flow example.

4.3.1.1 Incompressible Flow Example

The two second derivative computation methods described in the previous section are applied to a 2D model problem. This model problem is a recirculating incompressible flow in square cavity [40]. The exact velocity and pressure field for this model problem is given by;

$$\begin{aligned} u(x,y) &= 2(1-x)^2x^2(1-2y)(1-y)y \\ v(x,y) &= -2(1-y)^2y^2(1-2x)(1-x)x \\ p(x,y) &= (1-x)x(1-y)y + \kappa. \end{aligned} \tag{4.23}$$

Where κ is a parameter dependent on the Reynolds number. Figure 4.1 shows the distribution of velocity, pressure and vorticity for the exact solution.

For this model problem, gradient-recovery and quadratic-reconstruction procedures are used to compute the maximum error. The maximum error here is defined

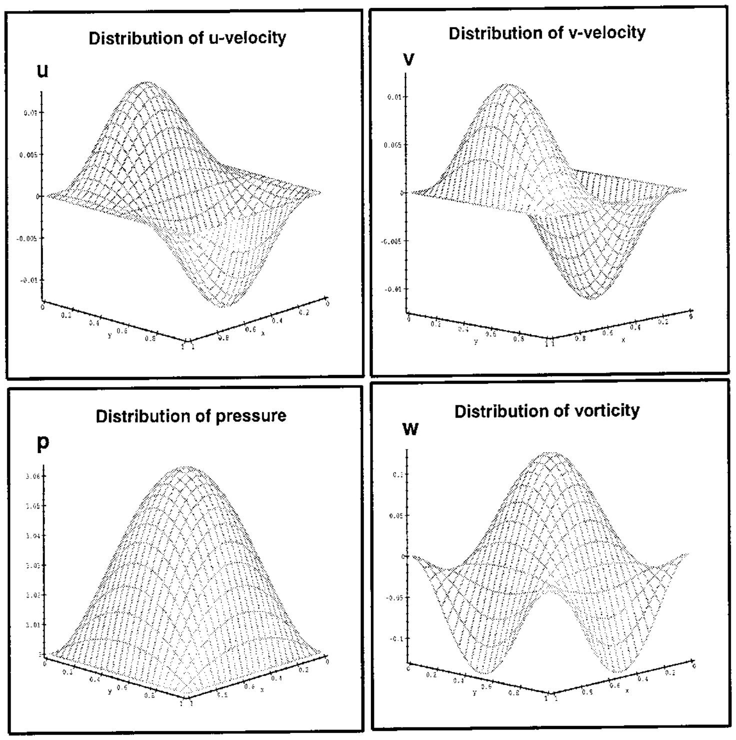

Figure 4.1: Distribution of exact u,v,p and ω for model problem.

to be

$$\epsilon_{max} = Max(|\mathcal{D}\vec{V} - \mathcal{D}\vec{V}^h|) \qquad on \quad \Omega \tag{4.24}$$

where, the operator $\mathcal{D}$ is defined as

$$\mathcal{D}f = \frac{\partial f}{\partial x} + \frac{\partial f}{\partial y} + \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial x \partial y} + \frac{\partial^2 f}{\partial y^2}. \tag{4.25}$$

Figure 4.2 shows the convergence history of the gradient recovery and quadratic re-construction procedures with respect to ideal (quadratic) convergence rate. It is clear from this figure that, quadratic reconstruction method has a superior convergence rate in relation to gradient recovery procedure.

4.3.1.2 Compressible Flow Example

The model problem chosen here is the supersonic flow over a cone. The schematic of this problem is given in Figure 4.3. The problem is axisymmetric and

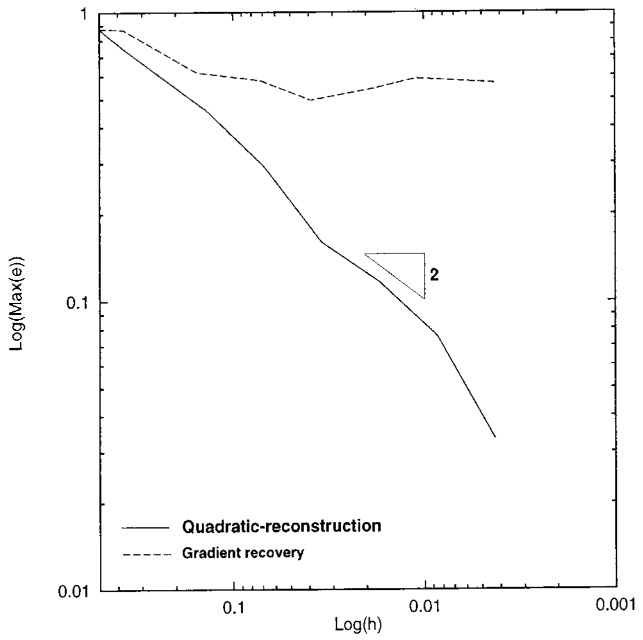

Figure 4.2: Comparison of convergence rates.

the flow variables are constant along constant θ lines, as shown in Figure 4.3. The equations governing the conical supersonic flows are given by the Taylor-Maccoll equations [47] as follows

$$\begin{aligned} \frac{d^2u}{d\theta^2} &= \frac{f(\theta, u, \frac{du}{d\theta})}{g(\theta, u, \frac{du}{d\theta})}, \\ \frac{du}{d\theta} &= v. \end{aligned} \tag{4.26}$$

The right-hand side functions f and g are defined as

$$\begin{aligned} f &= (\gamma-1)\frac{u}{c}\left(1-\frac{u^2}{c^2}\right)+\frac{\gamma-1}{2c}\left(1-\frac{u^2}{c^2}\right)\cot\theta\frac{du}{d\theta}-\frac{\gamma u}{c^3}\left(\frac{du}{d\theta}\right)^2-\frac{\gamma-1}{2c^3}\cot\theta\left(\frac{du}{d\theta}\right)^3 \\ g &= \frac{\gamma+1}{2c^3}\left(\frac{du}{d\theta}\right)^2-\frac{\gamma-1}{2c}\left(1-\frac{u^2}{c^2}\right) \end{aligned} \tag{4.27}$$

where c is the speed of sound and γ is the ratio of specific heats. The numerical solution of equation 4.26 is obtained using a variable coefficient Backward-Difference Formulation (BDF) solver (VODE [48] from NetLib).

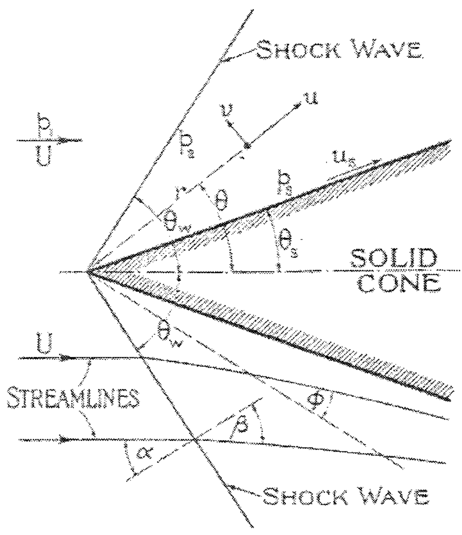

Figure 4.3: Schematics of the supersonic flow over a cone.

Because the problem is axisymmetric, only one quarter of the cone is considered. As an example, the half angle of the cone is set to 10^0 and the free-stream Mach number M_∞ is selected to be 2.0. At these flow-field conditions, both the shock tables from [47] and the numerical solution of the ODE in Eq. 4.26 resulted in a shock angle of 32.26^0. The geometric model, which contains the quarter cone and a box as outer boundaries, is shown in Figure 4.4. An initial mesh with 24,000

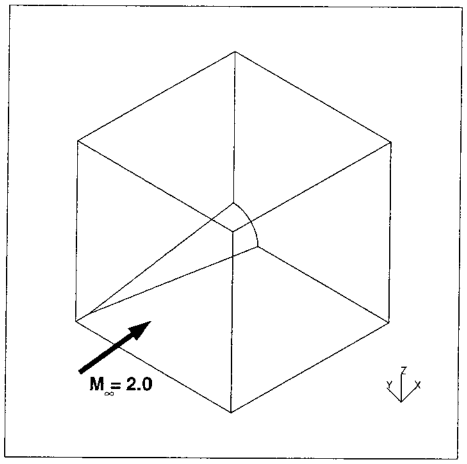

Figure 4.4: A geometric model used for flow over a quarter cone.

elements is generated to start the adaptive calculations. To guide the adaptive refinement procedure, a more meaningful measure of the error indicator is calculated as follows

$$\eta_i = \frac{\epsilon}{\epsilon^{max}} \times 100 \tag{4.28}$$

With this measure, we required at each adaptive step that the refinement is to be done at points where $\eta_i \geq 95\%$. Using this criteria three levels of adaptive refinement are performed. Figure 4.5 shows the progress of the adaptive refinement on the computational mesh (seen from the side). At adaptive level 3, the size of the mesh reached to 1,100,000 tetrahedral elements. It may seem that over 1 million elements for adaptively resolving this problem may be too much. However, we note that all the calculations are done in 3D. In order to make a good judgment, one has to make comparison with the adaptive procedures for the same or similar problems solved in 3D. Another way to make a comparison is to look at similar adaptive problems in 2D. We can make such a comparison with the results of reference [49], where a sub-sonic airfoil (oscillating) is solved adaptively. We note that the characteristic lengths for both the flow over a cone and the sub-sonic airfoil problem of [49] are the same. Although the range of outer boundaries are different, it is noted in [49] that the adaptive refinement is done only near the airfoil surface. Reference [49] reports that for the sub-sonic airfoil problem the initial mesh contained 2486 nodes and the adapted mesh had 8500 nodes. For the flow over a cone, since the flow-field is axisymmetric, we can count the number of nodes on one of the lateral faces of the model and make a comparison with the results of [49]. The maximum number of nodes for the initial mesh (the upper left corner of figure 4.5) is 747 nodes (this is almost 1/3 of the initial mesh of [49]) and the adapted mesh at level number 3 contains (see the lower right corner in figure 4.5) 3834 nodes (this is almost 1/2 of the adapted mesh size of [49]). Therefore, the important fact is that although the error indicator given by Eq. 4.22 is aimed to resolve linear discontinuities (vortex flows), it works quite effectively for resolving nonlinear discontinuities as well.

The calculated velocity field using the adaptive procedure is compared with the solution obtained from the solution of the ODE in Eq.4.26. The absolute accuracy of the ODE solver (MDF) is adjusted to 10^{-10}. Hence the error of the ODE solution

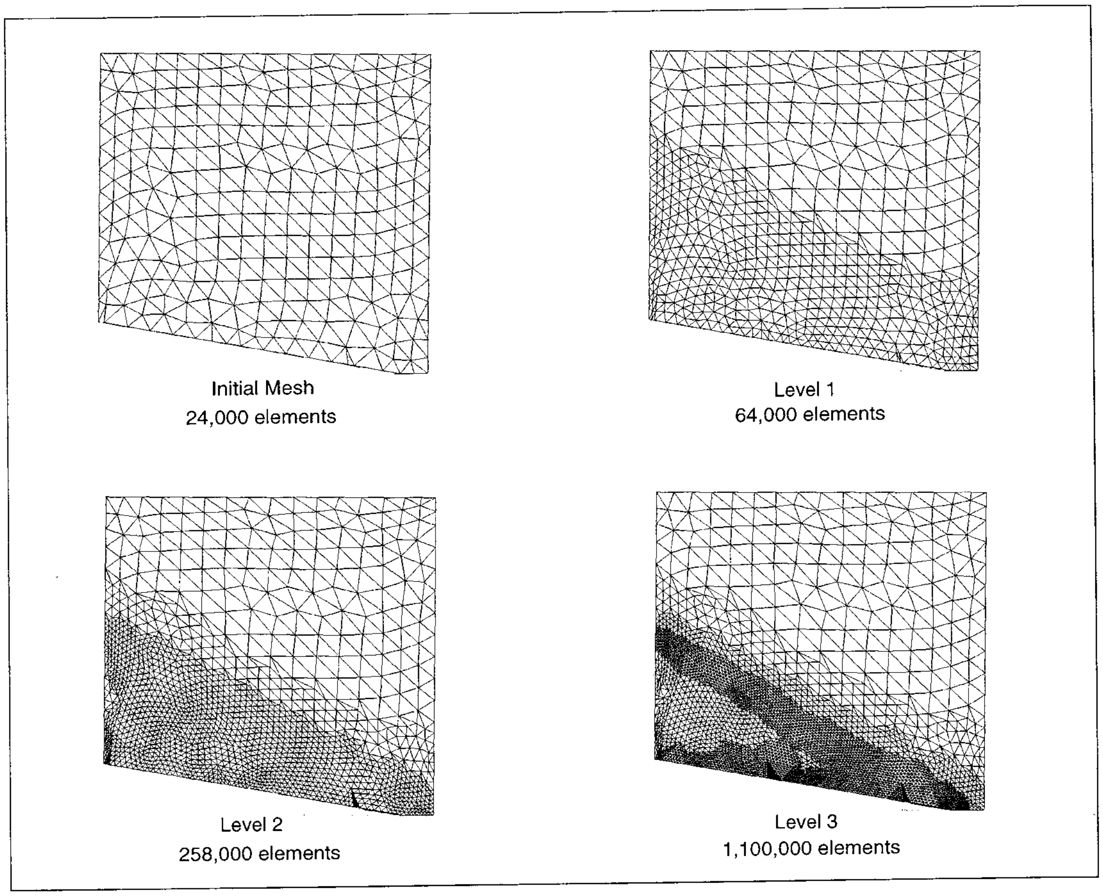

Figure 4.5: Progress of adaptive mesh refinement on the computational mesh.

is negligible compared to solution obtained from PDE. The velocity field between the cone surface and the shock wave is obtained from the solution of Eq. 4.26. Then the following error is measured at every point of the computational mesh:

$$E = \sqrt{|\vec{u}|_0^2 - |\vec{U}^*|_0^2} \tag{4.29}$$

where $\vec{U}^*$ is the solution obtained from the ODE. Later, the maximum of this error measure is plotted against the mesh size h in log-log scale as shown in Figure 4.6. Note that the slope of the convergence-curve in Fig. 4.6 is about 1.8 which is less than the expected quadratic rate. We believe that this is mainly due to the fact that the conical flows have shocks. However, the slow convergence behavior may also be attributed to the mesh refinement being a bit too diffuse. We note that there are better error indicators in the literature [59] that mainly target nonlinear discontinuities. Nevertheless, it has been shown here that the error indicator based on interpolation error estimate can be used to some extent to resolve both linear and nonlinear discontinuities in flow problems.

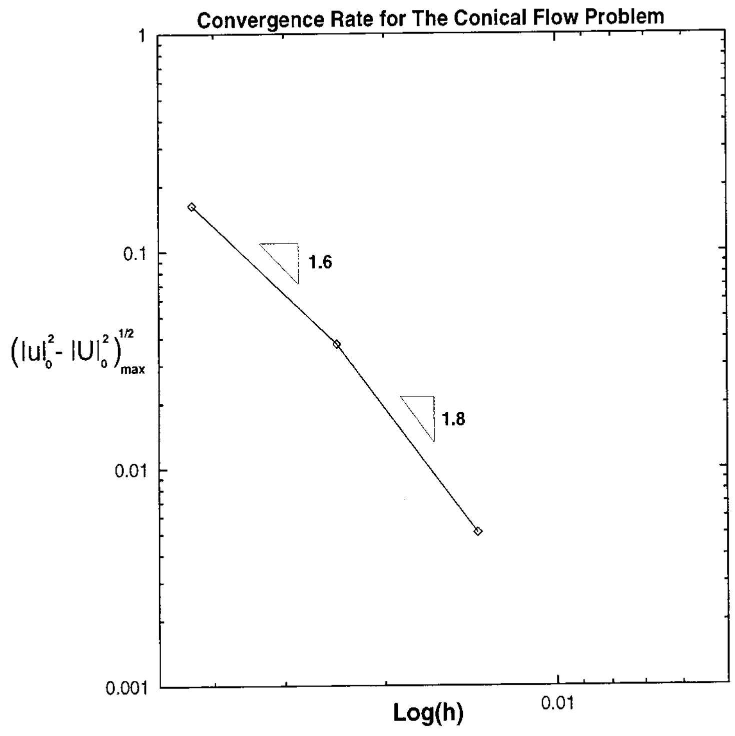

Figure 4.6: Measured convergence rate for the conical flow problem.

4.4 Two-Level Adaptation Procedure for Vortical Flows

So far, the sources of interpolation error have been identified by an error indicator. It is clear that for a finite amount of mesh resolution, there will always be some error in the computational domain. Although, in principle, one could use the error indicator to drive the adaptation to resolve all the features of a flow problem, because of computational efficiency and storage limitations of current computers, this would not be practical. It is desirable to monitor the global accuracy during the adaptive solution procedure and, if permissible, resort to more localized adaptation of the mesh for small-scale features of specific interest.

4.4.1 A Definition for a Vortex Core

From the topological point of view, a vortex core contains features that can be used to distinguish it from other regions of the flow-field. First, a vortex core is a stationary point where flow trajectories spiral in a plane and the vorticity is maximal at the core center. It is possible to define and distinguish a vortex core by looking at the velocity vector field. Let $\vec{u}$ represent the velocity field in and around a vortex core. When $\vec{u}$ is expanded into its Taylor series around the vortex core point P_o, we have

$$\vec{u}(x,y,z) = \vec{u}_0 + \Delta r^T \nabla \vec{u} + \mathcal{O}(\Delta r^3) \tag{4.30}$$

For a real fluid, viscous effects cause the core of a vortex to rotate approximately as if it were a rigid body, hence, on the plane of rotation $\vec{u}_o \rightarrow 0$ at the vortex core [42]. For inviscid flows with the existence of numerical diffusion, a vortex behaves much like it is in a viscous flow. Therefore, we identify a vortex core to be a stationary point. The second term in Eq. 4.30, the velocity-gradient tensor, $\nabla\vec{u}$, has complex eigenvalues if the stationary point is a vortex core. The velocity-gradient tensor can be disassembled into a rotation tensor and a strain-rate tensor. At a vortex core, the rotation tensor dominates over the strain-rate tensor [43]. This approach has been used recently to define and extract flow topology information for scientific visualization [44].

Based on the velocity-gradient tensor,

$$\nabla\vec{u} = \begin{bmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} & \frac{\partial u}{\partial z} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} & \frac{\partial v}{\partial z} \\ \frac{\partial w}{\partial x} & \frac{\partial w}{\partial y} & \frac{\partial w}{\partial z} \end{bmatrix} \tag{4.31}$$

we have three eigenvalues. For these eigenvalues, there could be only two possibilities: (1) all real eigenvalues, and (2) real and complex-conjugate eigenvalues combined. Abraham and Shaw [45] classify the behavior of a dynamic system based on the characteristic eigenvalues. Namely there are three types, (1) nodes, where all the eigenvalues are real and positive, (2) saddle, where all the eigenvalues are real and negative and (3) spiral saddle, where both real and imaginary eigenvalues exist. These three cases are illustrated in Figure 4.7. Clearly a vortex core is said to exist at a stationary point with eigenvalues forming a spiral saddle. It is known [43] that the magnitude of the imaginary part of a complex-conjugate pair represents the strength of the vortex. We should also try to represent the core diameter of a vortex. Since at the core vorticity reaches a maximum, then a possible definition of a core diameter can be made by limiting the magnitude of the vorticity at the outer surface of the vortex tube. In practical applications it is found that the core diameter extends up to a point where the magnitude of the vorticity drops below 10% of the vorticity magnitude at the core of the vortex. Clearly this loose definition can only detect isolated vortex tubes, better definitions should be found for more complex situations.

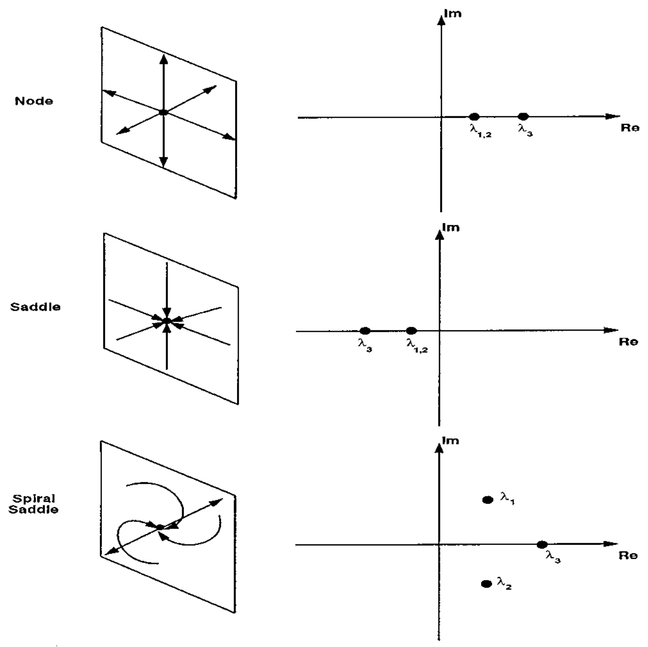

Figure 4.7: Classification of three dimensional critical points.

The topological features of a vortex core can be used to predict the vortex tube in numerical calculations. Numerical implementation of the vortex core detection technique used here follows from [44]. All tetrahedral elements are passed through an eigenvalue test on the velocity-gradient tensor. If the velocity-gradient tensor for the element happens to have both real and imaginary eigenvalues, then that element is flagged as a possible vortex core region. This information is passed to the adaptive refinement procedure so that the tetrahedral element which contains the vortex core can be refined appropriately. This simple implementation of the eigenvalue extraction technique works quite nicely as an error indicator for resolving vortex tubes. We should note here that the eigenvectors that correspond to complex-conjugate eigenvalues span a plane perpendicular to the vortex core, whereas the eigenvector of the real eigenvalue is directed along the trajectory of the vortex tube.

In our numerical examples we use the eigenvalue extraction technique to refine the mesh locally. To do this, the error indicator based on an interpolation error estimate is modified such that,

$$\theta^* = \begin{cases} \theta & \text{if a vortex exists on } \Omega^h \\ 0 & \text{otherwise.} \end{cases} \tag{4.32}$$

The mesh in the vortex tube is refined until the error tolerance drops to an acceptable level so that the vortex core is resolved accurately.

CHAPTER 5
A HOVERING ROTOR-BLADE MODEL

In this chapter we will describe a model for the numerical solution of the hover problem. The boundary conditions that are necessary to apply are explained here.

First we derive the equations for one dimensional momentum theory. Then use the results of momentum theory to establish simple model boundary conditions for hover calculations.

5.1 Momentum Theory

One of the simplest ways to understand the flow topology of a hovering rotor is to use basic momentum theory [3]. For a rotor (see Fig.5.1) which is accelerating a mass of air on a continuous basis, its thrust must equal the mass flow rate, $\dot{m}$, across the rotor disk times the change in flow velocity, $v_2 - v_1$.

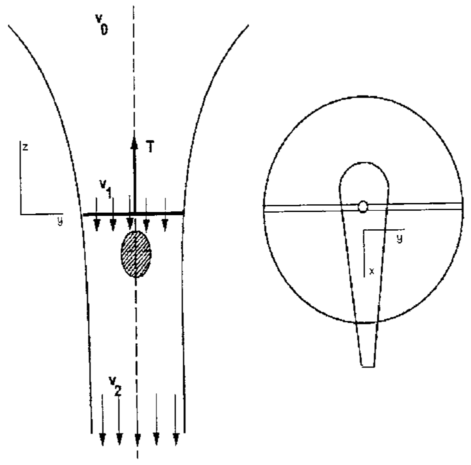

Figure 5.1: Idealized induced velocities for a hovering rotor.

The main assumption in momentum theory is that the induced velocity at the rotor plane be constant. This is possibly the simplest model of a helicopter blade in hover conditions. Higher order models, such as blade element theory [3] with non-uniform induced velocity correction requires a priori knowledge of the lift characteristics of the blade. Furthermore, to establish model boundary conditions of a hovering blade, we only need to know the average flow variables near the blade and at the wake. Therefore, momentum theory with uniform induced flow approximation is a good first-cut approach to achieve this goal.

For a constant induced velocity, the mass flow rate in the plane of rotor is written as

$$\dot{m}_1 = \rho v_1 A \tag{5.1}$$

where ρ is the mass density, v_1 is the *induced velocity* at the rotor plane, and A is the area of the rotor disc (πR^2). Also, the total change in flow velocity can be written as;

$$\Delta v = v_2 - v_0 \qquad \text{or} \qquad \Delta v = v_2 \quad \text{as} \quad z \rightarrow -\infty \tag{5.2}$$

Using these relations into the definition of thrust, we get

$$T = \rho v_1 A v_2 \tag{5.3}$$

Moreover, the potential energy at the rotor disk and kinetic energy below the rotor plane, which again assumes constant down-wash velocity, are written as

$$E_p = \rho {v_1}^2 A v_2, \qquad E_k = \frac{1}{2} \rho v_1 A {v_2}^2 \tag{5.4}$$

Equating E_p and E_k (since the two must match), we obtain a relation for the induced velocities of the rotor system,

$$\rho {v_1}^2 A v_2 = \frac{1}{2} \rho v_1 A {v_2}^2 \quad \rightarrow v_2 = 2v_1 \tag{5.5}$$

That is, for hover the induced velocity far downstream of the rotor plane must be twice the induced velocity at the rotor disc. Inserting this relation into thrust formula given in Eq. 5.3, we finally obtain

$$T = 2\rho {v_1}^2 A \quad \text{or} \quad v_1 = \sqrt{\frac{T}{2\rho A}} \tag{5.6}$$

5.2 Numerical Boundary Conditions at Steady-State Hover

In this section, we will be using the results of momentum theory to formulate the approximate numerical boundary conditions of a hovering blade. But first, let us define the thrust coefficient for a rotor system, that we will be using frequently in this writing. For a rotor system with radius R, rotational speed Ω and a total thrust of T, the thrust coefficient is defined as

$$C_T = \frac{T}{\rho_\infty A(\Omega R)^2} \tag{5.7}$$

where, ρ_∞ is the freestream density, A is the rotor disk area (πR^2) and R is the rotor radius. In addition to total thrust coefficient C_T, we will also be using sectional thrust coefficient, C_t in the later sections. The definition of sectional thrust coefficient is

$$C_t = \frac{dt/dr}{\frac{1}{2}\rho_\infty(\Omega r)^2 c} \tag{5.8}$$

where, t is the sectional thrust of the blade at radius r and c is the chord length.

When viewed from far-field perspective, a hovering rotor may be viewed as a potential sink [16] which attracts the still air from the far-field towards the hub of the rotor (see Fig.5.2).

Following from [16], the magnitude of velocity field, which is induced by a potential sink located at the rotor hub, at a spherical distance $d = \sqrt{x^2 + y^2 + z^2}$ is given as;

$$V_{in} = \frac{M_t}{4}\sqrt{\frac{C_T}{2}}(\frac{R}{d})^2 \tag{5.9}$$

where, M_t is the tip Mach number of the blade. This induced velocity, in fact, is the inflow velocity that penetrates through the computational boundaries around the

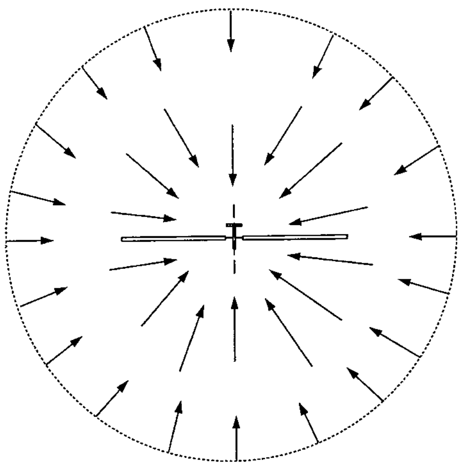

Figure 5.2: Hovering rotor as a potential sink (side view).

rotor plane. Also numerical experience has shown that the dependency to location of the outer boundaries is small if the computational boundaries are truncated at around 3/2 radii above the blade. In addition to this, because of the periodicity of the problem, the computational domain that needs to be solved is 2π/(Number of Blades). Therefore, for a two-bladed rotor, for instance, we need to solve only half of the computational domain.

The formulation of exit velocity, below the rotor plane, follows from the discussion of induced velocities in the previous section. Earlier, we found that the induced velocity at a far-field point below the rotor plane is given by;

$$v_2 = 2v_1 \quad \text{or} \quad v_2 = 2\sqrt{\frac{T}{2\rho A}} \tag{5.10}$$

Using the definition of thrust coefficient into Eq. 5.10, we find the magnitude of the outflow velocity below the rotor plane;

$$V_{out} = 2M_t\sqrt{\frac{C_T}{2}} \tag{5.11}$$

Conservation of Mass: The question of how to choose the location of the outer boundaries is guided by the mass flow conservation principle. For convenience, we will choose the computational domain of a rotating blade as a cylindrical domain as shown in Fig.5.3.

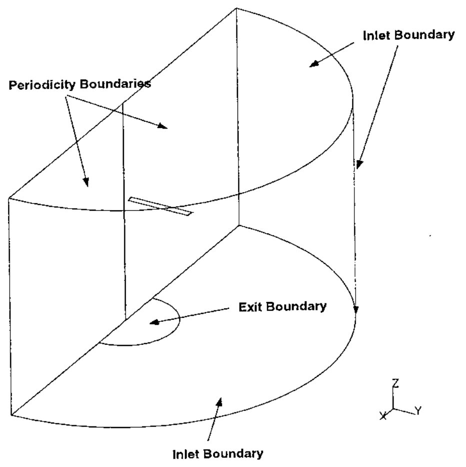

Figure 5.3: Computational domain for a two-bladed hovering rotor.

For a system which is in steady-state equilibrium, the mass flow conservation in the absence of internal mass generation can be written as follows;

$$\int_{A_{in}} \rho_{in} V_{in} dA = \int_{A_{out}} \rho_{out} V_{out} dA \tag{5.12}$$

In order the satisfy the inflow and outflow velocities formulated in the previous section for a given computational domain, we need to integrate the inflow and outflow velocities over the boundaries of the cylindrical domain as shown in Fig.5.4.

Before carrying out the integration, we state all the assumptions that we use in this model as follows:

- The rotor blade is assumed to be a three dimensional potential sink that attracts flow from its surrounding according to Eq. 5.9.
- The outflow velocity below the blade is assumed to be constant (Eq. 5.11) and it is obtained from one dimensional momentum theory.
- The slip-stream contraction ratio is assumed to be $R/\sqrt{2}$ using momentum theory. That is, the outflow surface is a cylindrical area of radius $R_1 = R/\sqrt{2}$.

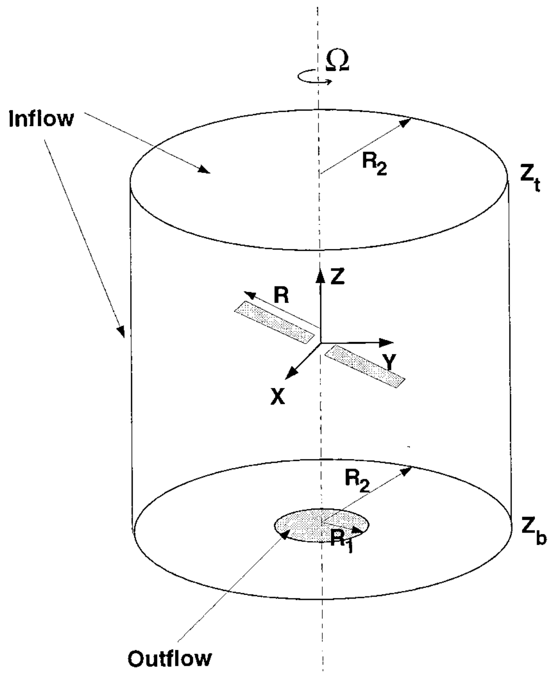

Figure 5.4: Inflow and outflow boundaries for mass flow conservation.

- The location of the outflow boundary is assumed to be $|z_b| = 2R$ below the blade.
- In order to reduce the effects of reflection of boundary conditions, the radius of the cylinder that represents the outer boundaries is assumed to be $R_2 = 3R$.
- The fluid density at the outer boundaries is assumed to be constant.

After stating the assumptions for our hovering blade model, the integration of the inflow velocity, V_{in}, is done as follows: For the top and bottom and lateral surfaces of cylinder the mass inflow is written as

$$\int_{A_{in}} V_{in} dA = \frac{M_t}{4}\sqrt{\frac{C_T}{2}} R^2 \Big(\underbrace{\int_0^{2\pi}\int_0^{R_2} \frac{1}{r^2+z_t^2} r dr d\theta}_{\text{top surface}} + \underbrace{\int_0^{2\pi}\int_0^{R_2} \frac{1}{r^2+z_b^2} r dr d\theta}_{\text{bottom surface}}$$
$$\underbrace{+ R_2 \int_0^{2\pi}\int_{z_b}^{z_t} \frac{1}{R_2^2+z^2} dz d\theta}_{\text{lateral surface}} \Big) \qquad (5.13)$$

Carrying out the integration, we find that the total flow entering into the computational domain is

$$\int_{A_{in}} V_{in} dA =$$
$$\frac{\pi M_t}{2}\sqrt{\frac{C_T}{2}}R^2\Big(\ln\Big[\frac{(R_2^2+z_t^2)(R_2^2+z_b^2)}{z_t^2 z_b^2}\Big] + \arctan(\frac{z_t}{R_2}) - \arctan(\frac{z_b}{R_2})\Big) \tag{5.14}$$

Note that to calculate the inflow at the outer boundaries due to a potential sink, we considered all surfaces of the cylinder. At the bottom surface of the cylinder, the reduction in outflow velocity V_{out} is less then 3% because the inflow velocity V_{in} is superimposed in the outflow region (shaded area in Fig. 5.4). We believe that this is a negligible reduction so that the momentum theory still holds.

The flow leaving the domain from the bottom surface can also be written as

$$\int_{A_{out}} V_{out} dA = \int_0^{2\pi}\int_0^{R1} V_{out} dA = 2\pi M_t \sqrt{\frac{C_T}{2}} R_1^2 \tag{5.15}$$

Finally, the inflow and outflow mass amounts should match by equating the relations 5.14 and 5.15. Making necessary cancellations, we arrive at the following relation

$$4(\frac{R_1}{R})^2 = \frac{1}{2}\ln\Big[\frac{(R_2^2+z_t^2)(R_2^2+z_b^2)}{z_t^2 z_b^2}\Big] + \arctan(\frac{z_t}{R_2}) - \arctan(\frac{z_b}{R_2}) \tag{5.16}$$

Now, using the result of momentum theory from Eq. 5.5, and noting that the mass flux should be constant, it follows that the area at the slipstream is 1/2 of the area at the rotor disk [4]. That is far downstream of the blade $R_1 = R/\sqrt{2}$. This is known as the slipstream contraction ratio in hover. Let us assume that the slipstream contraction ratio of $R/\sqrt{2}$ is attained at an approximate location of $|z_b| = 2R$ below the blade. Also assume that the lateral boundaries are located 3 radii ($R_2 = 3R$) from the center of rotation to avoid reflection of boundary conditions from the outer boundaries. Then using these values into Eq. 5.16 we can solve for $z_t \approx 3/2$. Finally, Eq. 5.16 gives a relation that can be used in determining the dimensions of a cylindrical domain used as outer boundaries of a blade in hover, in order to conserve the mass flux.

Periodic Boundaries: As mentioned earlier, an N bladed rotor system at hover can be solved with domain of size $2\pi/N$. However, this simplification requires the application of periodic boundary condition at the planes as shown in Fig. 5.3. In order to satisfy periodicity, the values of the flow quantities at the periodic boundaries should be reflected around the periodic angle. During the solution process, the flow variables are computed only on one of the symmetry planes (master plane) of the domain at each time step, and the solution on the master plane is copied over the other plane (slave plane) before the next time step. This process has to be repeated at beginning and end of each time step to satisfy the flow periodicity. Also, this process requires that the mesh faces classified on the symmetry planes be perfect reflection each other. This pre-condition should be satisfied during the mesh generation procedure and before solution starts.

5.3 A Consistent Inflow/Outflow Model for Steady-State Hover

One of the immediate shortcomings of the numerical boundary condition of a hovering rotor blade is that the inflow and outflow boundaries are pre-defined. Although the boundary condition and the inflow/outflow boundary arrangements are defined within the guidelines of momentum theory and mass conservation, it is not fully clear at this point that this particular model properly represents an isolated hovering blade problem in free-air conditions. Therefore, the inflow/outflow boundary conditions of a hovering blade have to be handled very carefully. One potential problem that we have observed in our numerical calculations is regarding the high-thrust conditions. From Eq. 5.11 we see that the intensity of the outflow velocity is proportional to the rotor blade thrust. Also, from momentum theory we can deduce that the slip-stream contraction at far below the blade should be $R/\sqrt{2}$, using the Bernoulli's equation [4]. However, since we are truncating the boundaries at a finite distance from the blade, as shown in Figure 5.3, the momentum theory for the outflow velocity may not hold exactly. In fact, in some of the high thrust hover calculations we observed a standing vortex at the outflow boundary because the flow is not exiting the domain smoothly. This situation is rendered with a schematic as shown in Figure 5.5. Another issue regarding the model boundary conditions of a

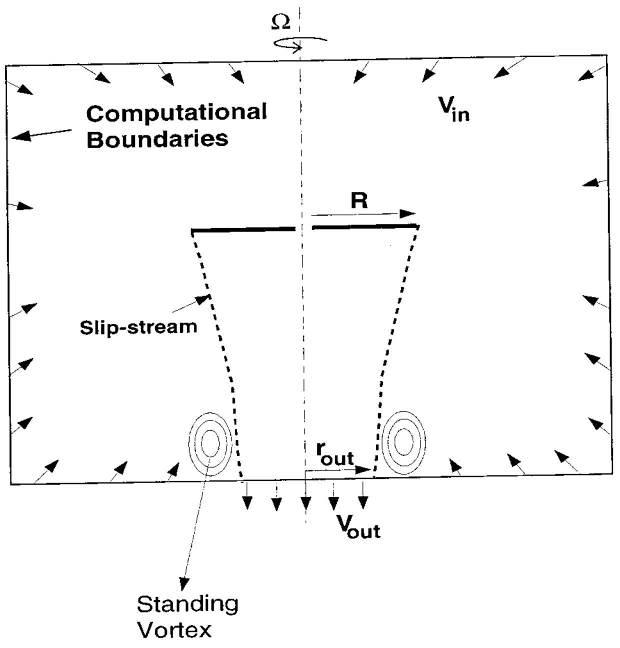

Figure 5.5: Inconsistency in outflow BC at high thrust conditions.

hovering blade is that it requires a priori knowledge of a thrust coefficient, C_T, for the rotor system in order to specify the strength of the potential sink. This requirement creates a problem since the rotor thrust should be calculated and not taken as a given value. Therefore, in this section we will suggest an approach to reduce the aforementioned shortcomings of the model boundary conditions of a hovering rotor blade. Let us start with the unknown rotor thrust issue.

Given a rotor system, we start with an initial thrust coefficient C_T^0 and iteratively update the inflow and outflow conditions such that at the steady-state case we obtain a converged value of final thrust coefficient for the rotor system. Using these guidelines, we suggest the following iterative hover boundary condition update procedure;

Algorithm 1: Iterative procedure for hover thrust

```
Given $C_T^0$ , $M_t$, $\theta$, $\epsilon_{\mathbf{V}}$ and $\epsilon_{C_T}$ for a rotor system
solve the hover problem as follows;

    For $i = 0 \cdots N_{max}$
        Compute inflow/outflow velocity:
        $V_{in}^i = \frac{M_t}{4}\sqrt{\frac{C_T^i}{2}}(\frac{R}{r})^2$ ; $V_{out}^i = 2M_t\sqrt{\frac{C_T^i}{2}}$
        Solve GLS Variational Formulation (Euler's Equations)
        Integrate pressure on blade surface
        $\mathbf{F}^{i+1} = \int_{\Gamma_b} p\mathbf{n}d\Gamma$
        $T^{i+1} = F_{x_3}^{i+1}$ ; $C_T^{i+1} = \frac{T}{\rho_\infty A(\Omega R)^2}$
        if $|C_T^{i+1} - C_T^i| \leq \epsilon_{C_T}$ and $||\mathcal{L}\mathbf{V^h}||_{\mathbf{L_2}} \leq \epsilon_{\mathbf{V}}$ STOP
    EndFor
```

Note that in Algorithm 1, we assumed that the rotor shaft is parallel to x_3 axis, so that we have $T = F_{x_3}$. Also, θ is the rotor collective angle and $\epsilon_{\mathbf{V}}$ and ϵ_{C_T} are stopping criteria for convergence of residual of the finite element solution and thrust coefficient, respectively. This idea was recently applied to a four bladed rotor system of Apache AH-64A helicopter [57]. The collective setting (θ) for this case was 6^o. The calculations are started with an initial $C_T^o = 0.0$ that is far from the experimentally determined value which was 0.0058. Then the calculations are continued with the iterative procedure as described in Algorithm 1 until the difference between calculated and measured values of thrust coefficient drop below 2% of relative error. The convergence of the thrust coefficient for this case is shown in Figure 5.6. With this procedure we show that the CFD solver can be started with an unknown thrust coefficient. An iterative process is used to calculate the thrust during the CFD solution process. The flow calculations are carried until a steady-state thrust coefficient is reached.

Secondly, for high thrust conditions we automatically adjust the size of the outflow boundary so that the slip-stream will smoothly leave the computational domain. This will also eliminate the standing-vortex problem at the outflow as shown in Figure 5.5.To do this we suggest again an iterative procedure that starts

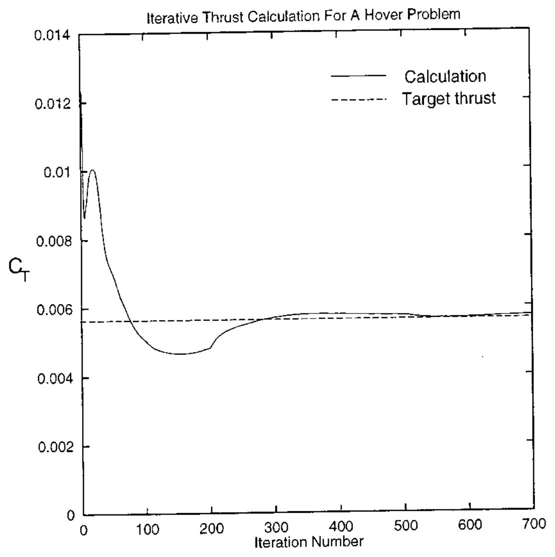

Figure 5.6: Iterative calculation of thrust coefficient.

with an outflow area that is $R/\sqrt{2}$ as suggested by the momentum theory. Then, in the course of computations, we can calculate the normal pressure gradient $\frac{\partial P}{\partial n}$ on the outer boundaries to determine the direction of the flow. If at a point on the boundary $\frac{\partial P}{\partial n} < 0$ that point is an inflow point, otherwise if $\frac{\partial P}{\partial n} > 0$ then that point is an outflow point. These two flow conditions are shown schematically in Figure 5.7. Using the pressure gradient information on the outer boundaries during

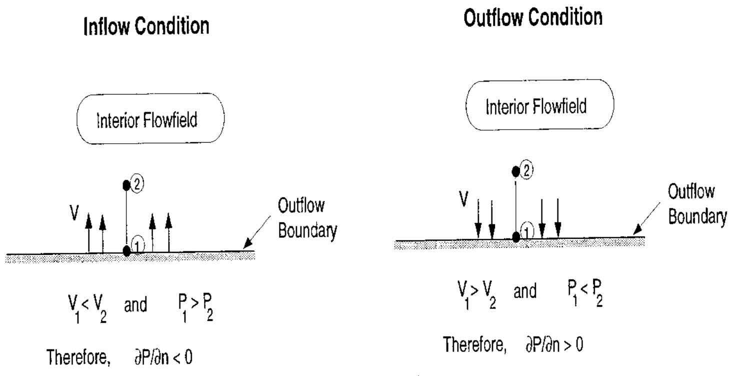
Inflow Condition
Outflow Condition
Interior Flowfield
Interior Flowfield
V
V
2
1
2
1
Outflow Boundary
Outflow Boundary
$V_1 < V_2$ and $P_1 > P_2$
Therefore, $\partial P/\partial n < 0$
$V_1 > V_2$ and $P_1 < P_2$
Therefore, $\partial P/\partial n > 0$

Figure 5.7: Inflow and outflow conditions at the outer boundaries.

the calculations, we can change the boundary conditions automatically. However, while changing the type of the boundary condition at the outer boundaries, we have to make sure that mass conservation is not violated.

CHAPTER 6
HOVER EXAMPLES
Adaptive Calculations

We present parallel, h-adaptive numerical examples using the error indicator presented in the previous chapter. The application problems are helicopter rotor blades in steady-state hover. Two different rotor blade configurations are presented with varying levels of numerical difficulties.

6.1 Carradona-Tung Blade

As a first example, we compare finite element analysis results with test data on a rotor blade tested by Caradonna and Tung [50]. This two bladed rotor uses a NACA0012 airfoil section, it is untwisted and un-tapered. The aspect ratio of the blade is 6. The flow case analyzed for the Caradonna-Tung blade is at a tip Mach number $M_t = 0.439$, thrust coefficient $C_T = 0.00459$, collective setting $\theta_c = 8^o$ and rotor speed of 1250 rpm. For computational efficiency, only one blade of the rotor system is modeled by accounting for cyclic symmetry (periodicity). The surface definition of the blade is generated using a solid modeling system. The computational outer boundaries are approximated by enclosing the blade in a finite radius cylinder. To avoid contamination of boundary conditions, the radius of the cylinder is chosen to be 3 times the blade radius and the blade is placed 1.5 and 2 radii away from top and bottom surfaces of the cylindrical domain, respectively. An initial mesh for this geometry is generated by using an automatic mesh generation program [51]. The initial mesh contains 214,000 linear tetrahedral elements. The geometric model and the initial mesh of the Caradonna-Tung blade are shown in Fig. 6.1. The inflow and outflow boundary conditions of a hovering blade are applied as explained in chapter 3.

Starting from an initial mesh, a series of results have been obtained using the adaptive refinement procedure. The initial mesh is distributed over 16 processors of

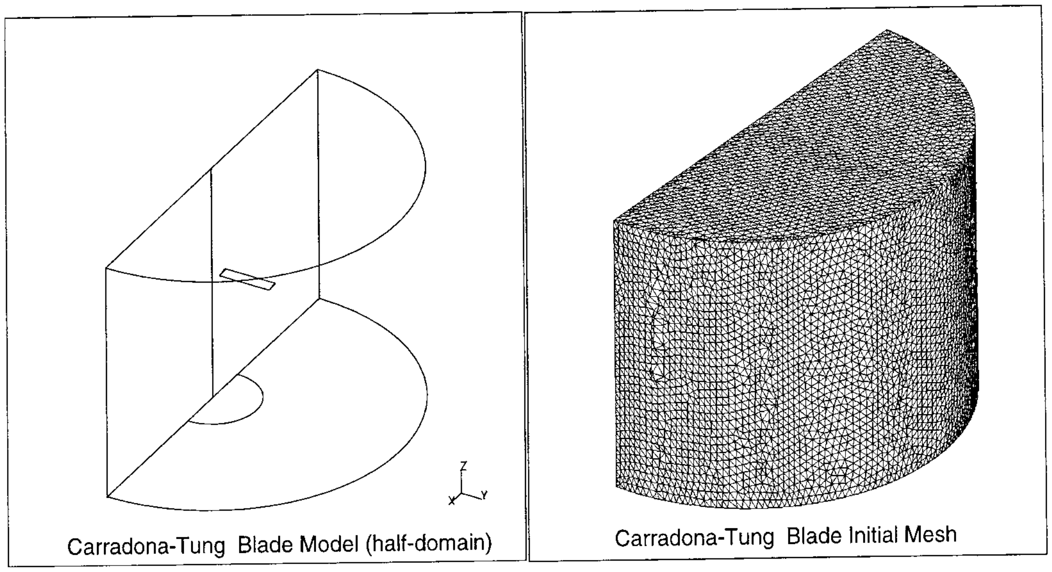
Carradona-Tung Blade Model (half-domain)
Carradona-Tung Blade Initial Mesh
Z
X
Y

Figure 6.1: Geometric model and the initial mesh for Caradonna-Tung blade.

an IBM SP2 parallel machine. At each adaptive step, the solution is interpolated from the previous adaptive level and the program run until the adapted solution reaches steady state. At an average value, the finite element procedure required 200-300 GMRES cycles to converge, where each cycle uses a Krylov space of size 10. A sample convergence history is shown in Fig. 6.2.

One potential problem with running a parallel, h-adaptive, large CFD problem is the available physical memory of each CPU unit and the ability of the finite element framework to use this memory efficiently and effectively. When an adaptive refinement is performed on a given mesh, it is almost certain that some of the processors refine much more than others. For a given physical memory of a CPU unit, which is 128MB for the SP2 machine on which this problem is run, we are limited to create not more than 80,000 tetrahedral elements per processor. In our experience with this and all other example cases reported here, it was not feasible to perform an autonomous mesh refinement step using the initial parallel procedure without manually limiting the number of newly created elements for a given processor. Although some of the processors created no new elements, other processors created large numbers of elements which eventually halted the entire adaptive procedure because of insufficient physical memory. An example of this unsuccess-

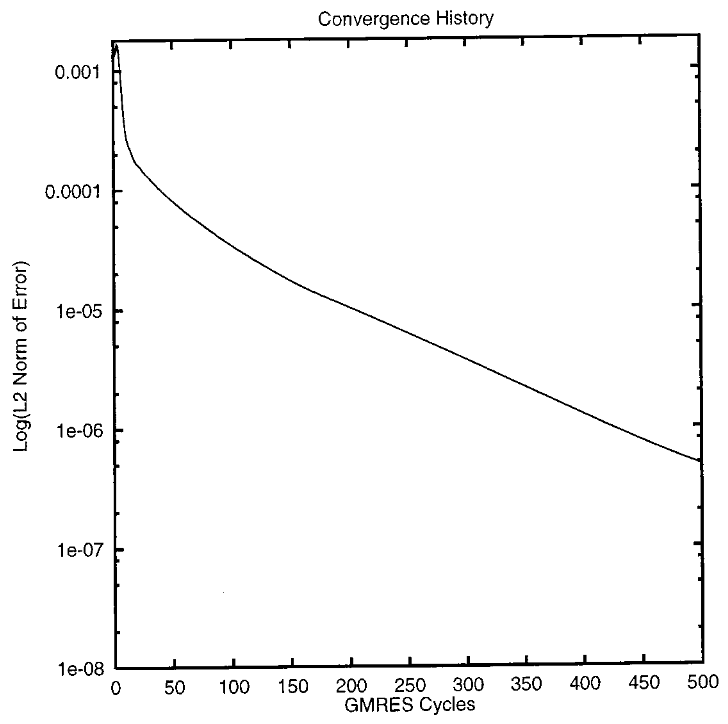

Figure 6.2: A typical convergence history for steady-state hover computations.

ful case is shown in Fig. 6.3, where processors 3,4 and 5 create excessive number of elements than the allowable. To alleviate the unbalanced usage of memory per processor, a predictive load balancing procedure [54] has been incorporated into the adaptive refinement procedure. This predictive load balancing scheme works with the adaptive refinement procedure and distributes tetrahedral elements, before they are split, across processors based on refinement load level, thus reducing the possibility of unbalanced physical memory usage per processor. A successful adaptive run with predictive load balancing is shown in Fig. 6.4. In Fig. 6.4, using the estimated load prior to refinement, the final adapted mesh is distributed equally over the processors. With predictive load balancing, the mesh size per processor is not limited by the available memory of the processor. Since the mesh is balanced before refinement is executed, the problem size becomes limited by the total memory only.

As stated before, the adaptive procedure is performed using an interpolation

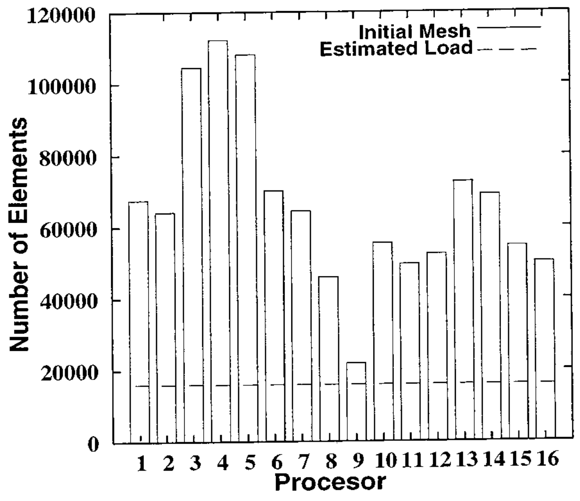

Figure 6.3: Adaptive refinement without predictive load balancing.

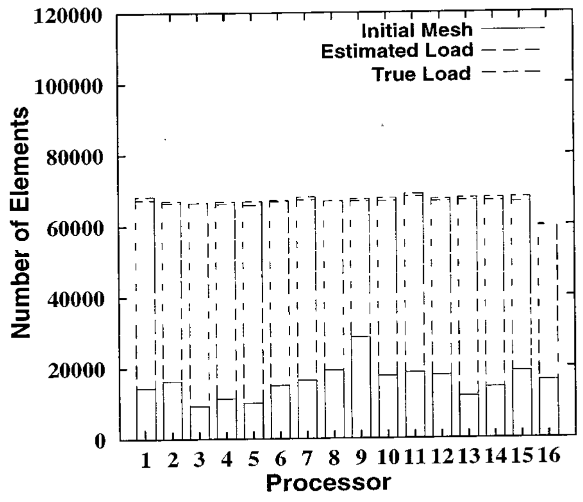

Figure 6.4: Adaptive refinement with predictive load balancing.

error estimate along with a vortex core detection technique. The order of refinement strategies has been selected by looking at the global and local quality of the solution in comparison to experimental data. For example, after two levels of refinement with error indicator based on interpolation estimate, the pressure distribution on the blade surface showed significant improvement with respect to the initial mesh. Then we examined the state of the tip vortex and used the vortex core detection technique to locally refine and enhance the resolution near the tip of the blade. This adaptive process was repeated several times using both error indicators to refine the computational domain globally and locally.

Figure 6.5 shows the changes in mesh resolution on the blade surface with the adaptive refinement levels.

The initial mesh, which contains 214,000 elements is shown as level 0. In adaptive level 1, to reduce the global errors and enhance the quality of solution near the blade, the indicator given in Eq. 4.16 is used to compute the error and refine the mesh. Notice that, in level 1 the mesh resolution on the blade surface is almost doubled. In adaptive level 2, Eq. 4.27 is used to bracket the adaptive refinement to the vicinity of the tip vortex. As shown in Fig. 6.5, the mesh resolution on the blade surface increased only near the leading and trailing edges of the blade tip. This step clearly shows that the up-wash from the lower surface of the blade to the upper surface creates a rotating flow which triggers the vortex core detection process to indicate that this region needs refinement. This adaptive process repeated two more levels to resolve both the tip vortex and flow quality near the leading edge of the blade. At the end of adaptive level 4 the mesh size reached to 2,215,000 tetrahedral elements. Also note that the mesh resolution increased at the leading edge of the blade where flow stagnates. To be able to show the mesh refinement process in the interiors of the computational domain, a faceted slice, which is 15^o behind the trailing edge, is taken from mesh and the progress of the adaptation is shown in Fig. 6.6. It is clear from Fig. 6.6 that for refinement levels 2 and 4, most of the mesh adaptation takes place near the tip vortex.

Computed coefficient of pressure distributions at 4 radial locations are plotted in Figure 6.7. When compared against experimental data, pressure distributions computed by adaptive procedure show noticeable improvement in comparison to initial mesh results. The suction peaks and the upper surface pressure plateau show remarkable improvement with adaptation.

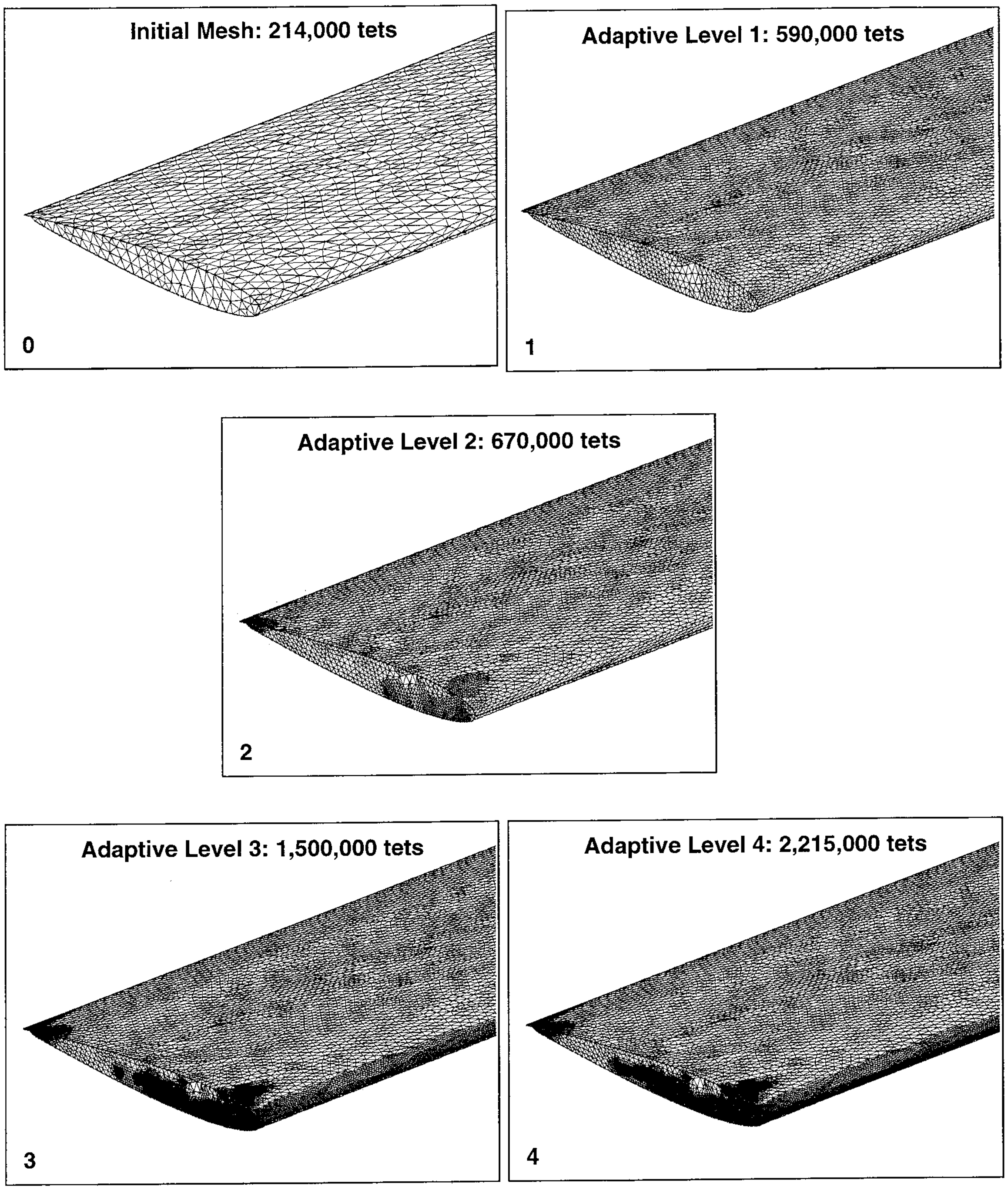
Initial Mesh: 214,000 tets
0
Adaptive Level 1: 590,000 tets
1
Adaptive Level 2: 670,000 tets
2
Adaptive Level 3: 1,500,000 tets
3
Adaptive Level 4: 2,215,000 tets
4

Figure 6.5: Adaptive refinement levels for Caradonna-Tung Blade: surface meshes.

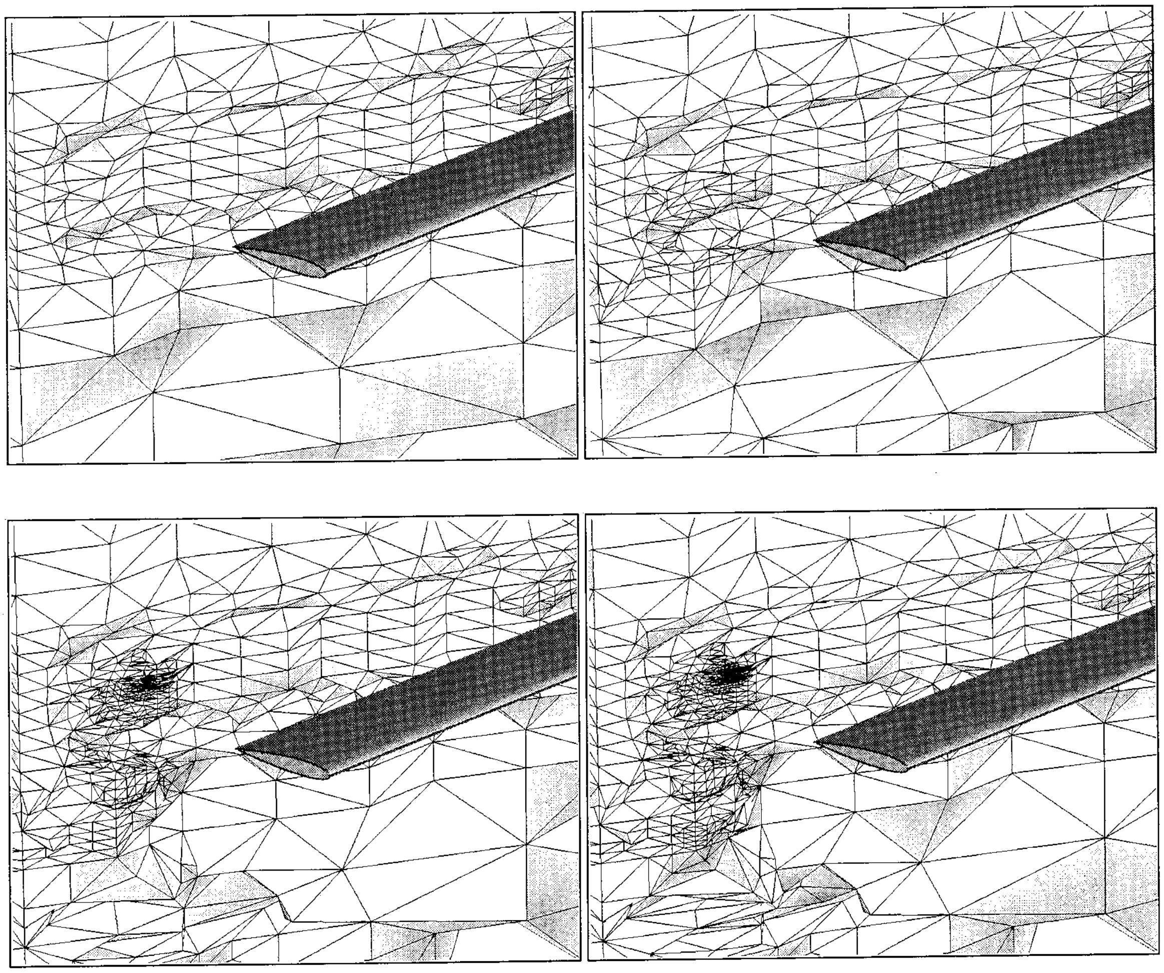

Figure 6.6: Adaptive levels 1 through 4 for the Caradonna-Tung blade: Faceted slice is taken 15^0 behind the TE

Finally, to show the effect of adaptive refinement on the resolution of the tip vortex, the tip vortex structure is rendered for initial mesh, and adaptive levels 2 and 4 in Figure 6.8. Notice that with the adaptation, a full 180^o revolution of the tip vortex is captured successfully.

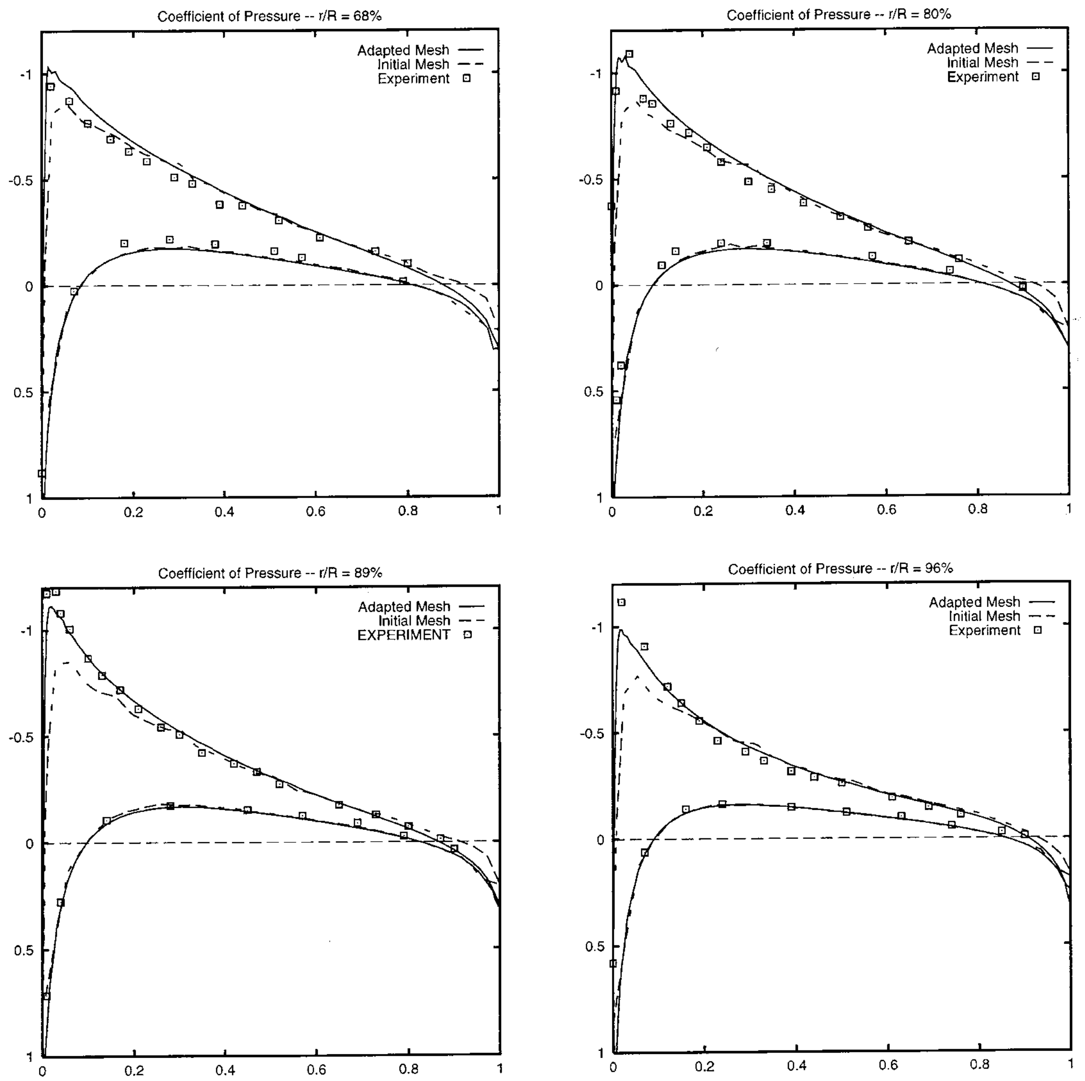

Figure 6.7: Comparison of computed pressure coefficient with experimental data: Caradonna-Tung blade, $M_t = 0.439$, $C_T = 0.00459$ and $\theta_c = 8^o$.

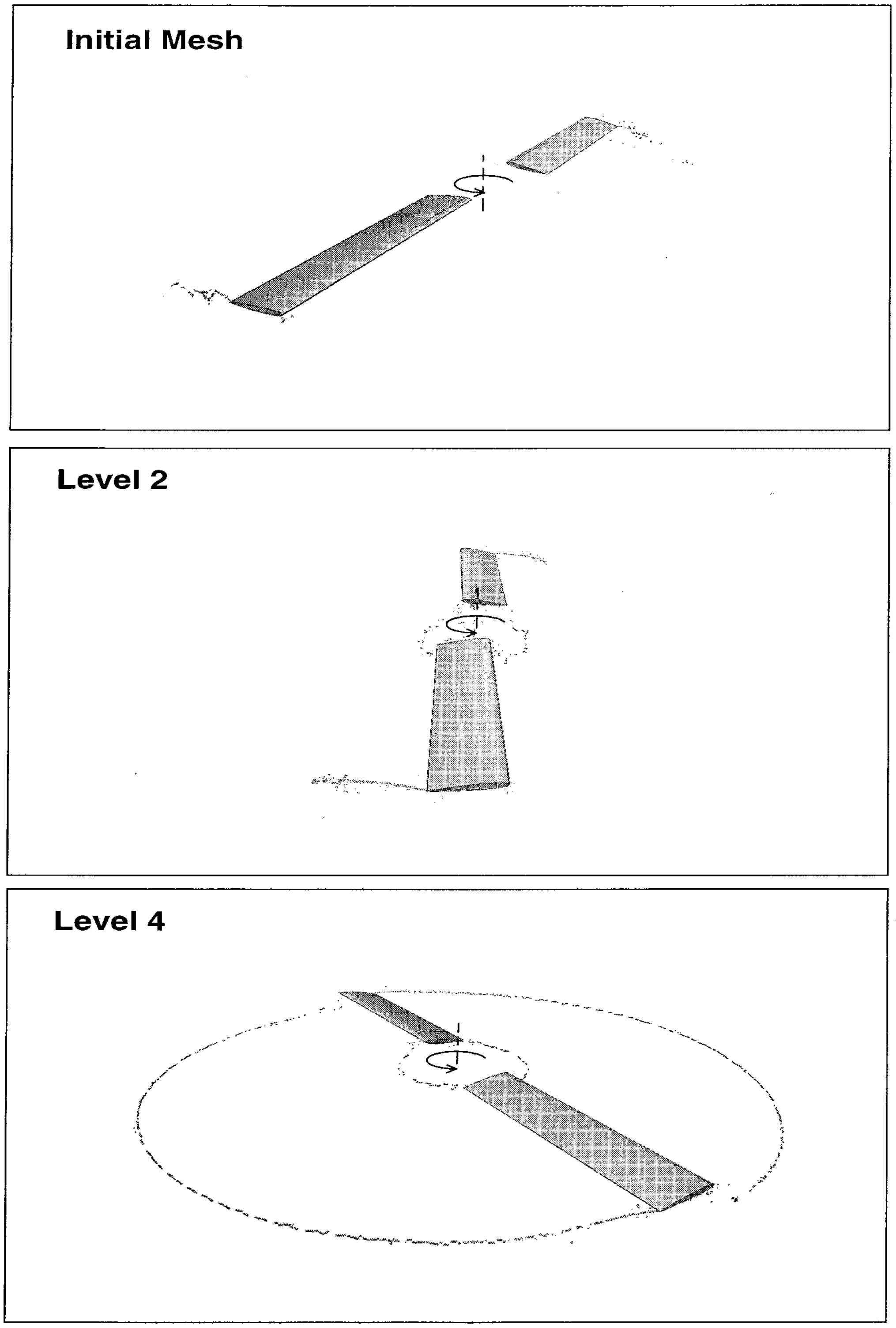

Figure 6.8: Progress of tip vortex geometry with adaptive refinement: Caradonna-Tung Blade.

6.2 UH-60A Blackhawk Blade

As a second case, we selected the four bladed rotor system of the UH-60A Blackhawk. Due to periodicity of the flow-field, only one blade of the rotor system is considered. The geometric dimensions and features of the blade are taken from the scale model (1:5.73) used in the experiment of reference [56]. Two airfoil profiles, SC-1095 and SC-1095R8, are used to construct the surface definition of the blade using a solid modeling system (Fig. 6.9). The built-in twist of the blade, which varies linearly over the first 80% radius of the blade and has a hook-type nonlinear twist near the tip, is included in the model. Also, elastic twist of the blade is incorporated into the model. An example blade pitch distribution, which at this point contains built-in twist, collective, and elastic twist is plotted against non-dimensional radius and compared with experimental measurements in Figure 6.10.

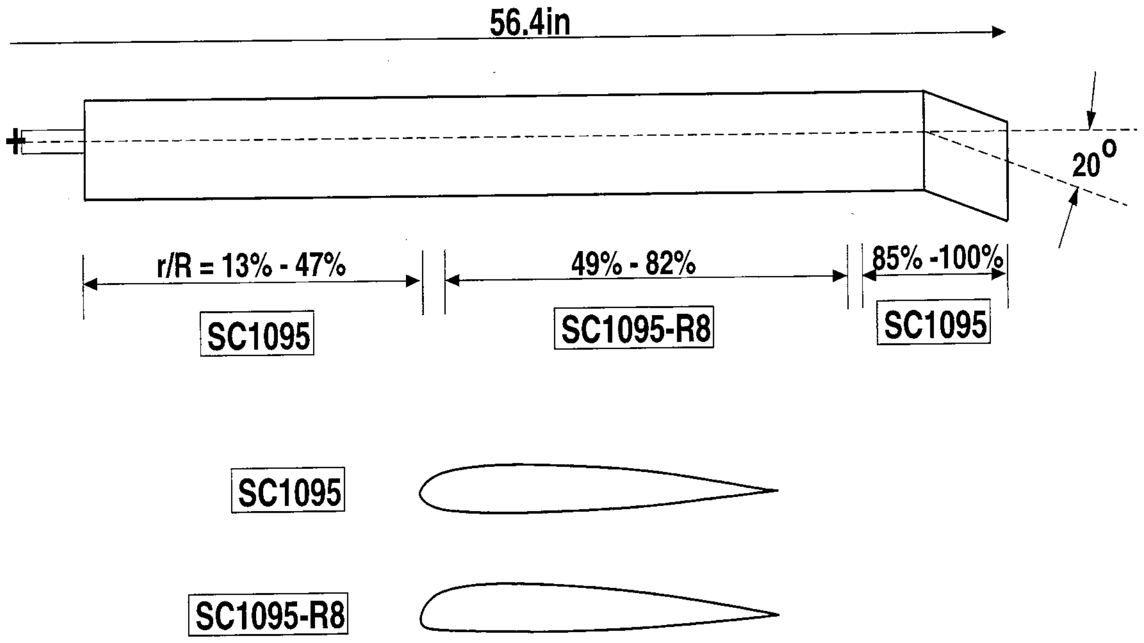

Figure 6.9: UH-60A Blackhawk rotor blade planform.

The flow-field of the UH-60A rotor is interesting due to the swept-tip and nonlinear twist. Two flow-field conditions are chosen to study the adaptive solution procedure of this blade. The first case studied with the UH-60A is at zero thrust and the second case is at high thrust (design) conditions. Both of these cases are

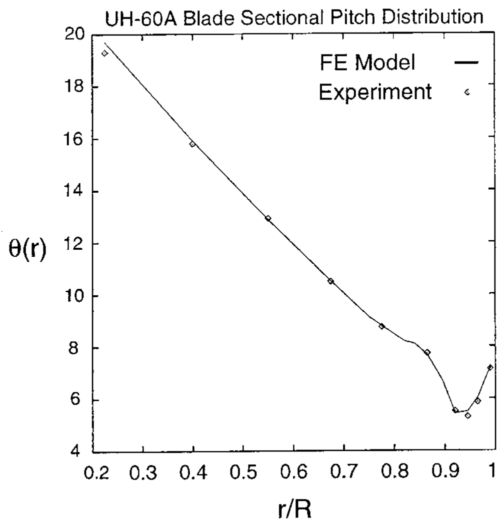

Figure 6.10: UH-60A blade pitch distribution.

reported to have neither separation nor stall in experimental results.

6.2.1 UH-60A at zero thrust

The flow-field conditions used for this case are as follows: tip Mach number $M_t = 0.628$, rotor speed $\Omega = 1427$ rpm, thrust coefficient $C_T/\sigma = 0.0$, collective setting $\theta_{.75} = 0.11^o$ and coning $\beta_0 = -0.20^o$.

Before doing any adaptive solutions, a mesh sensitivity study is done to establish initial element sizes that can be used as a starting point. First, a mesh containing 760,000 tetrahedral elements (see Fig. 6.11), is used to calculate the flow-field. For this mesh the average global element size is about 0.125R and it is more or less uniform throughout the domain with the exception of the blade surface which has a finer (down to 0.001R) mesh resolution than the rest of the domain. On the blade surface, the mesh is clustered near leading edge and trailing edge regions to capture the stagnation pressure correctly. Second, a finer mesh, which is refined near the tip-path-plane of the blade, is generated (see Fig. 6.12). To create this mesh, a slice of the computational domain that encloses the blade is targeted for refinement. The initial size of this second mesh is 1,100,400 tetrahedral elements. The average element size in the domain that is refined manually is about 0.002R. Computed pressure distributions at selected radial locations on the blade surface are compared against experimental data in Figures 6.13 and 6.14. There comparisons reveal that the uniform (coarse) mesh did not capture some of the key aspects of the flow-field as observed in experiment. As anticipated, results with the finer mesh are considerably more accurate than the coarse mesh results. Although the fine mesh does not resolve all the features of the flow-field, it is a good starting point for adaptive refinement. Next, the adaptive procedure is applied to the fine initial grid results to improve the solution further and to locate the vortex structure. With the adaptive procedure, the mesh is refined in two levels, the first level using the interpolation error estimate and the second level using the vortex core detection technique. At this point, the mesh size reached 2,164,704 elements.

Figures 6.15 and 6.16 show computed sectional thrust and torque distributions, respectively. Notice that the adapted grid enhances the accuracy of the thrust distribution in the neighborhood of the inflection point near the tip. The integrated value of C_T/σ for the adapted mesh is 0.00039.

Finally, Figure 6.17 shows the predicted vortex flow structure which is calculated by the vortex core detection technique. Note that there is a vortex tube located between 75%-80% radius of the blade. This vortex tube is independent of the tip vortex. The reason for the existence of a vortex tube at 75% radius is hypothesized

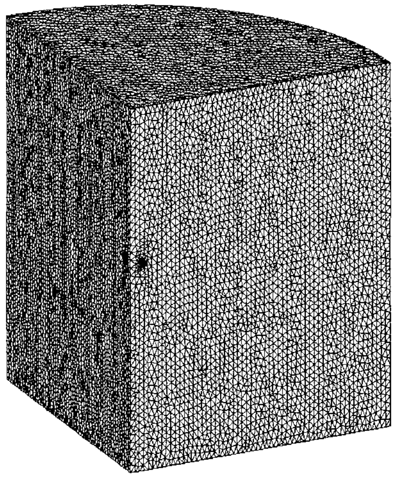
Figure 6.11: Coarse initial mesh for UH-60A blade at zero thrust.

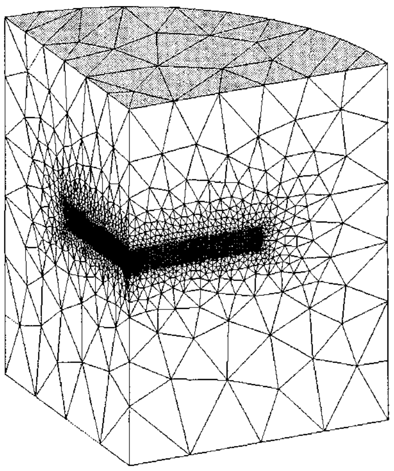

Figure 6.12: Fine initial mesh for the UH-60A blade at zero thrust.

to be the differential change in thrust loading from positive to negative as shown in Figure 6.15. The inboard shed wake, in this case, should have a discontinuity around 80% radial span. Existence of this inboard vortex tube clearly demonstrates a deviation from empirical wake geometry studies [58] where the inboard shed wake is assumed continuous between the root and the tip of the blade. Finally, there is also a vortex tube emanating from the tip of the blade, but it does not seem to have a very strong interaction with the blade tip.

6.2.2 UH-60A at design thrust

The final and the most challenging case analyzed for the UH-60A blade is a design thrust case where the flow conditions are: tip Mach number $M_t = 0.628$, rotor speed $\Omega = 1425$ rpm, thrust coefficient $C_T/\sigma = 0.085$, collective setting $\theta_{.75} = 10.47^o$, coning $\beta_0 = 2.31^o$ and $FM = 0.73$. This particular case offers a stronger tip vortex structure than the zero thrust case.

Similar to the zero thrust case, an initial mesh with 1,200,000 tetrahedral elements is generated. Figures 6.18 and 6.19 show the outer boundary and blade tip close-up views of this mesh, respectively. Notice that the mesh resolution is finer at the outer boundaries in comparison to the zero thrust case (see Fig. 6.12). A

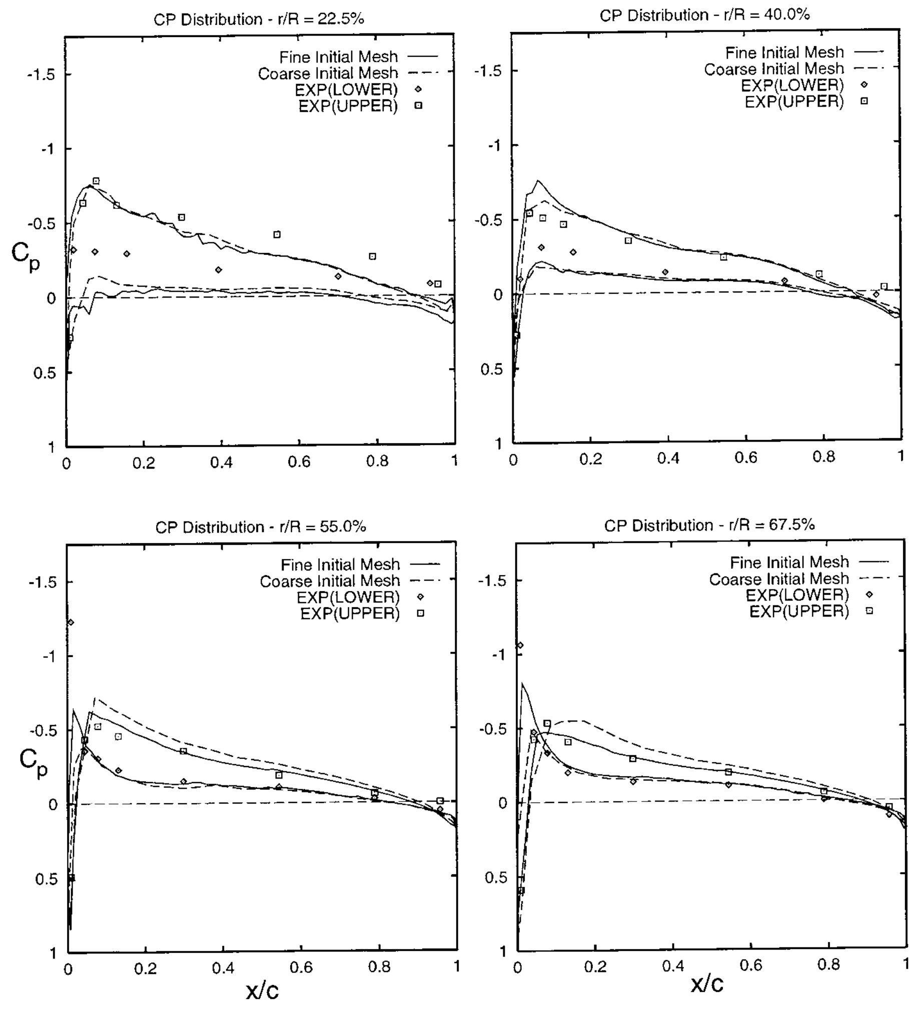

Figure 6.13: Computed coefficient of pressure distributions: $C_T/\sigma = 0.0$.

finer mesh resolution at the outer boundaries is needed to account for the non-zero induced velocity field (due to potential sink) applied as the boundary condition. In the zero thrust case, the induced velocities at the outer boundaries were zero, thus, a fine mesh was not necessary. Also note that we again generated a finer mesh enclosing the rotor disk to capture the initial formation of the blade wake.

Comparisons of computed pressure distributions with experimental data are given in Figures 6.20, 6.21, and 6.22. The refined mesh results are obtained using the two-level adaptive procedure starting from the initial mesh. During the adaptive solution of this case, the mesh is refined once using the interpolation error estimate

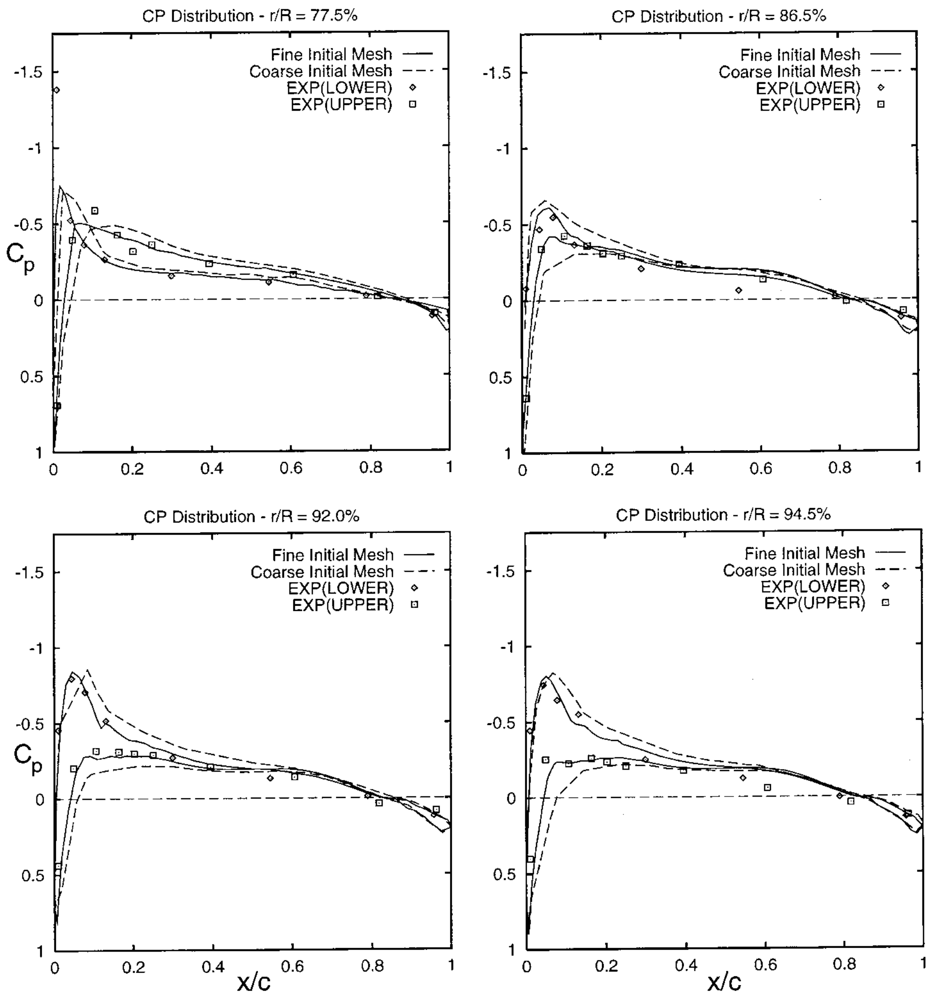

Figure 6.14: Computed coefficient of pressure distributions(cont.): $C_T/\sigma = 0.0$.

and four times using the vortex core detection technique to resolve the global features of the flow-field and adaptively capture the tip vortex. The final refined mesh contains 2,300,000 tetrahedral elements. As seen in Figures 6.20, 6.21, and 6.22, the computed pressure distribution with the refined mesh has an overall better quality in capturing both the leading edge suction peaks and the mid-chord pressure plateau. The pressure distributions at radial stations 94.5%, 96.5% and 99% show noticeable improvement with mesh adaptation. This improvement is primarily due to a more accurate representation of the tip vortex release from the previous blade.

The computed sectional thrust and torque coefficient distributions are pre-

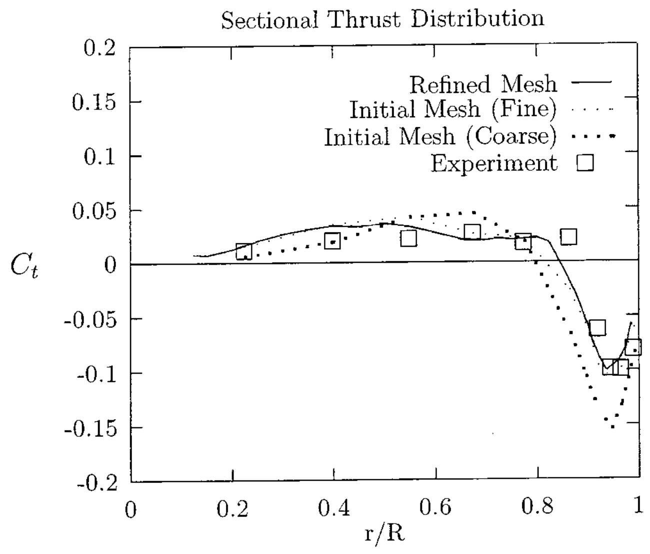

Figure 6.15: Sectional thrust distribution at $C_T/\sigma = 0.0$.

sented in Figures 6.23 and 6.24, respectively. The initial mesh, which could not resolve the tip vortex for the first 90 degrees azimuth, resulted in a poor sectional thrust distribution particularly at the outboard portion of the blade. We emphasize here that all of our attempts to capture the correct distribution of sectional thrust before resolving the tip vortex properly failed. Progressive adaptive refinement steps, using both interpolation error estimate and the vortex core detection technique, resulted in a correct prediction of tip vortex structure. The sectional thrust distribution with the final adapted mesh shows a remarkable improvement with respect to initial mesh results. Therefore, it has been concluded that computing at least the first 90 degrees azimuth travel of the tip vortex is essential for this high thrust UH-60A blade case. Figure 6.24 compares the sectional torque coefficient for the refined and initial meshes against experimental data. Notice that the experimental torque distribution contains both profile and skin friction torque, whereas numerical calculations reflect only the profile torque due to inviscid flow approximation. The integrated values of C_T/σ and figure of merit for the adapted mesh are 0.07946 and 0.72, respectively.

Further investigation of the wake geometry for this high thrust case reveals that the tip vortex, which is released from the previous blade, descends only about

3% blade radius in the axial direction by the time it passes under the next blade. Due to its close proximity to the blade, this strong vortex alters the velocity and pressure field of the blade tip significantly. The computed tip vortex axial and radial displacement plots are given in Figures 6.25 and 6.26, respectively. In Figures 6.25 and 6.26, the dashed-lines represent the generalized wake formulation [58], where the tip vortex geometry is represented by an empirical formula based on experimental observations. Figures 6.27-6.28 shows the progress of mesh adaptation for the tip vortex. The images in these figures depict the UH-60A blade surface and a planar section of the mesh taken at 15 degrees behind the blade. The planar sections in these images reflect the sliced mesh connectivity. Notice that, as the refinement levels progress, the mesh resolution increases at the points on this 15^o azimuthal plane where the tip vortex intersects the plane. We emphasize here that the adaptation procedure combined with the vortex core detection technique is effective. A typical mesh adaptation level using vortex core detection technique increases the mesh size only by 5-7%. This is mainly due to that fact that the adaptation scheme refines the mesh only in the areas where there is an indication of a vortex flow. The vortex core detection technique is also used as a tool to visualize the tip vortex geometry. Figure 6.29 shows the trajectory of the tip vortex around the blade and the velocity

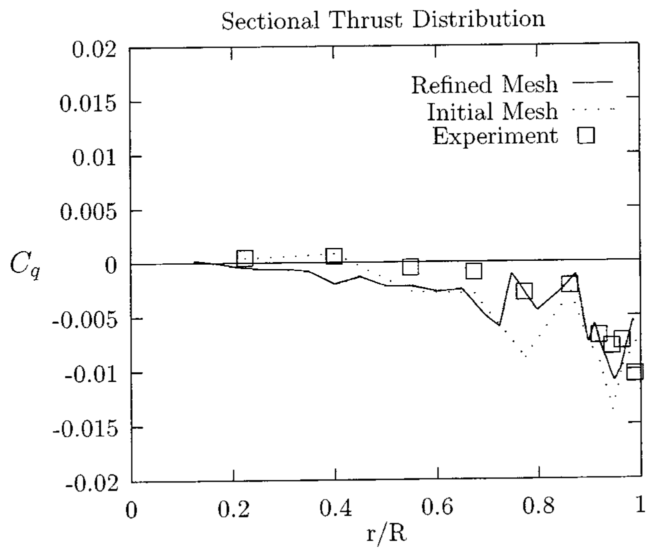

Figure 6.16: Sectional torque distribution at $C_T/\sigma = 0.0$.

vectors on a sectional plane 15^0 behind the blade. Finally, Figure 6.30, shows the progressive adaptation of the tip vortex with increasing refinement levels.

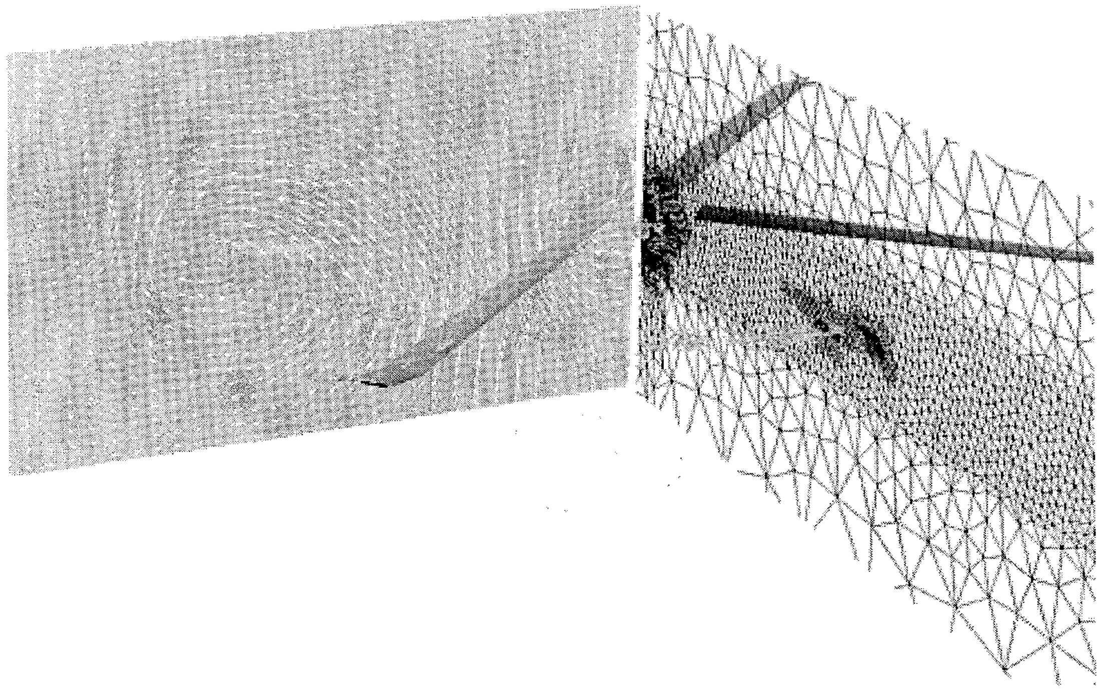

Figure 6.17: Computed vortex flow structure for the UH-60A blade at zero thrust.

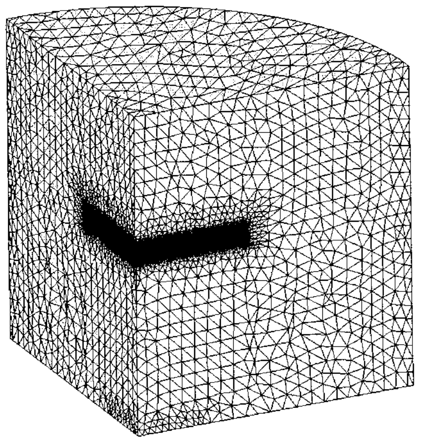

Figure 6.18: Initial mesh for the UH-60A blade outer boundaries: $C_T/\sigma = 0.085$.

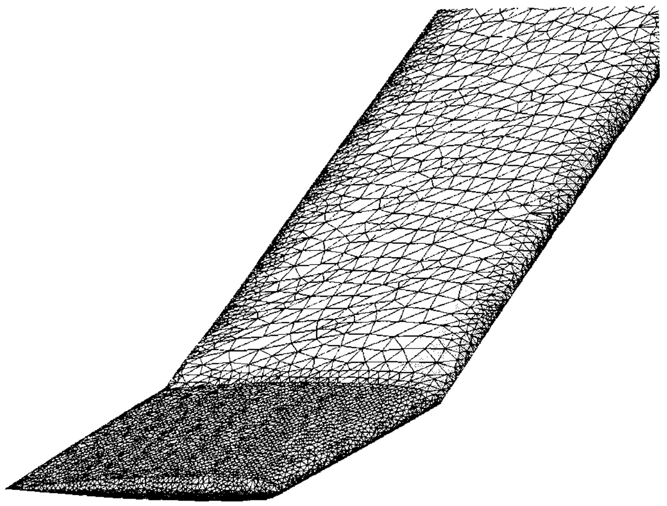

Figure 6.19: Initial mesh for the UH-60A blade tip: $C_T/\sigma = 0.085$.

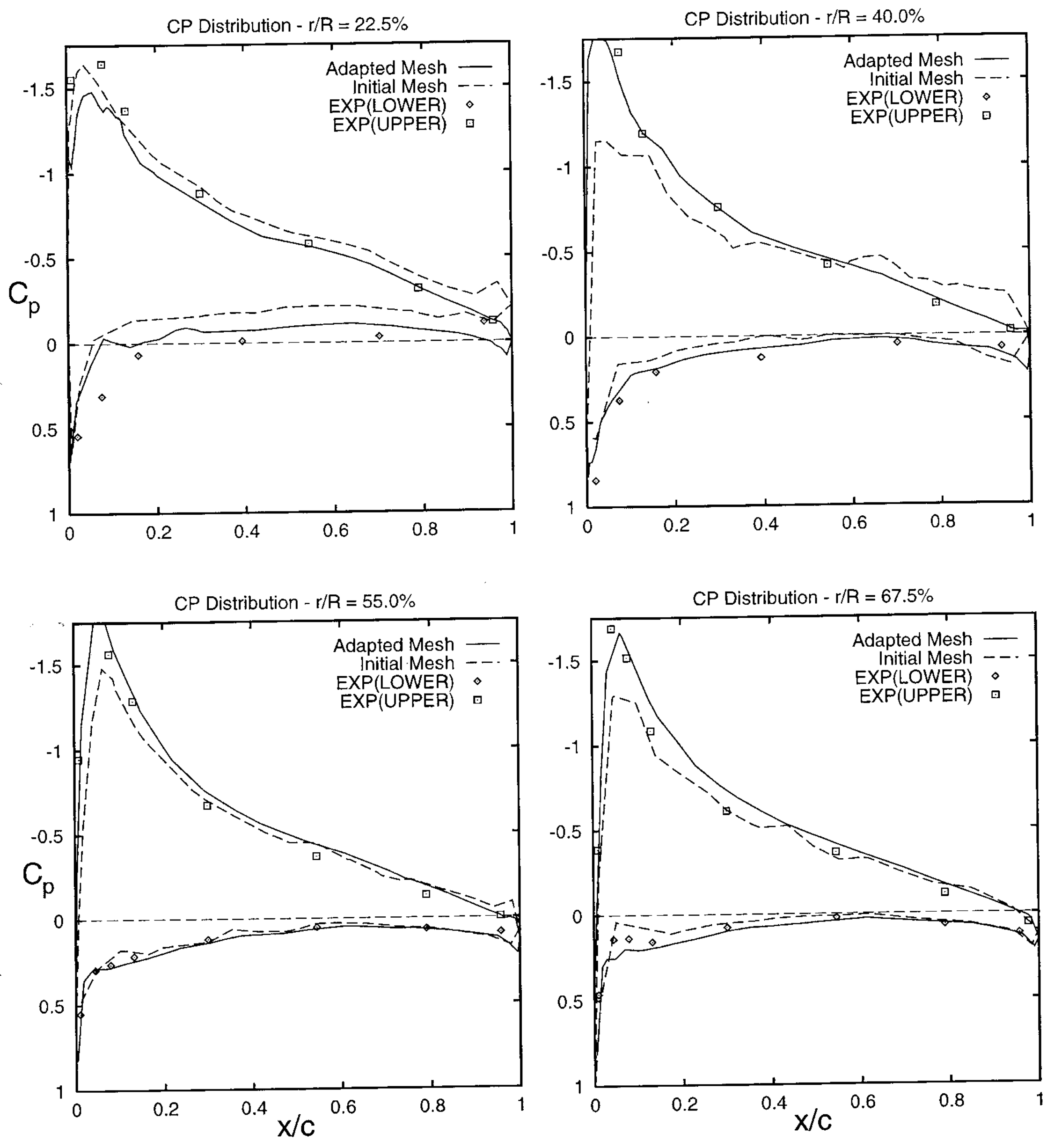

Figure 6.20: Computed coefficient of pressure distributions: $C_T/\sigma = 0.085$.

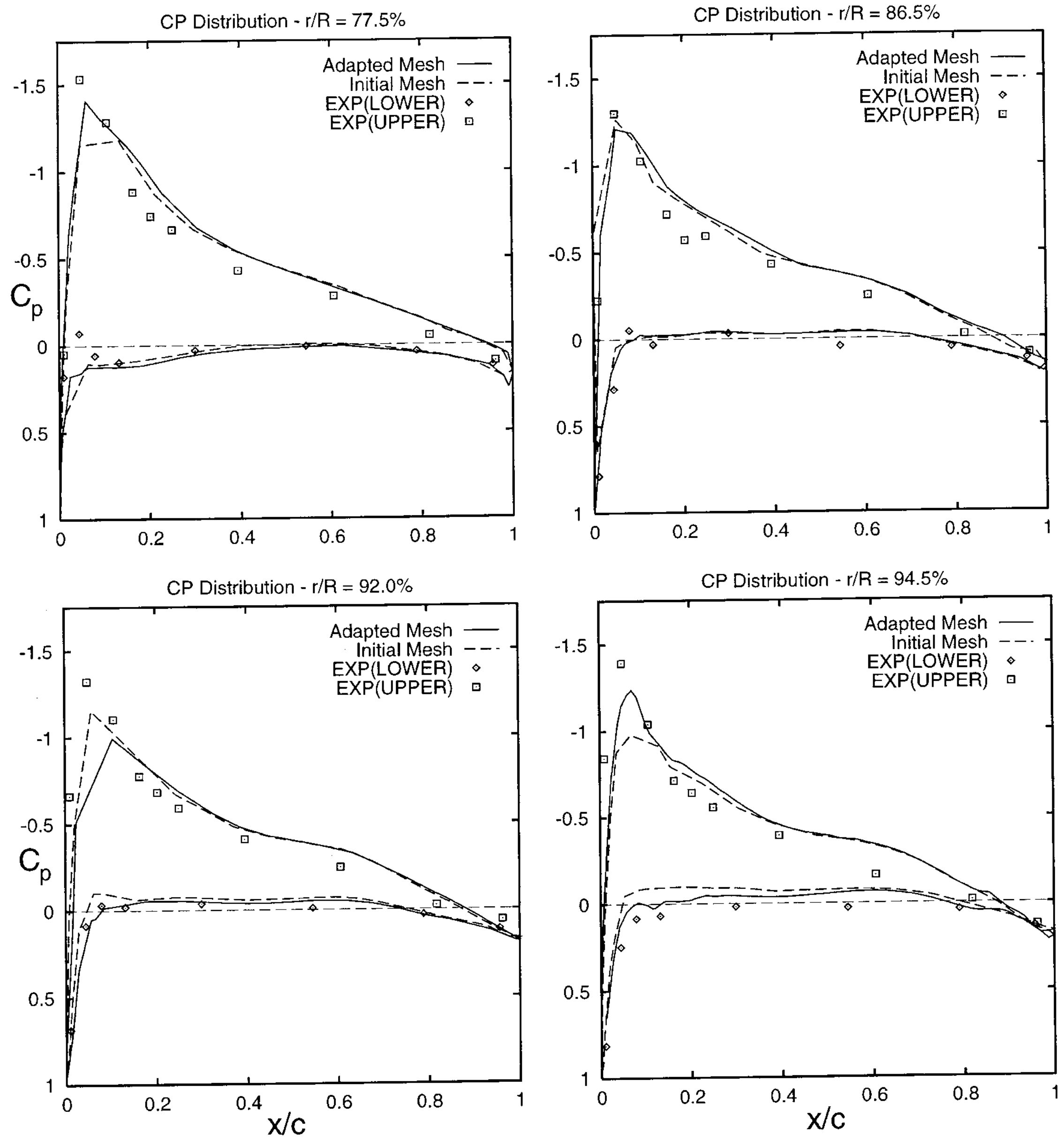

Figure 6.21: Computed coefficient of pressure distributions(cont.): $C_T/\sigma = 0.085$.

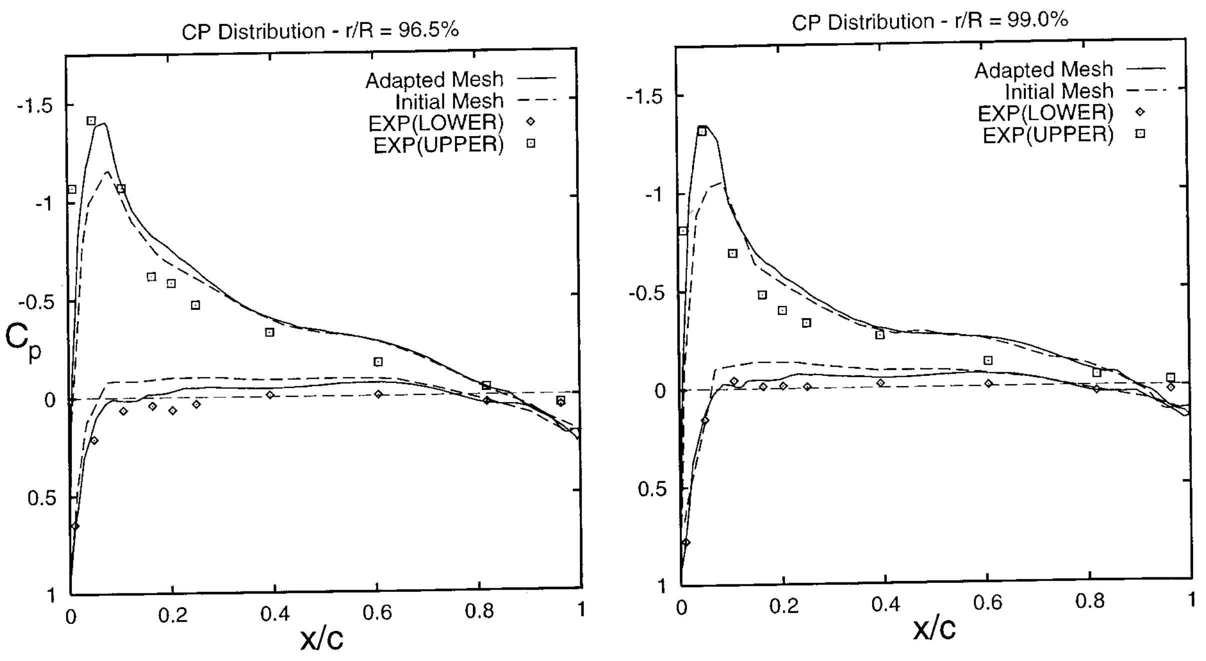

Figure 6.22: Computed coefficient of pressure distributions(cont.): $C_T/\sigma = 0.085$.

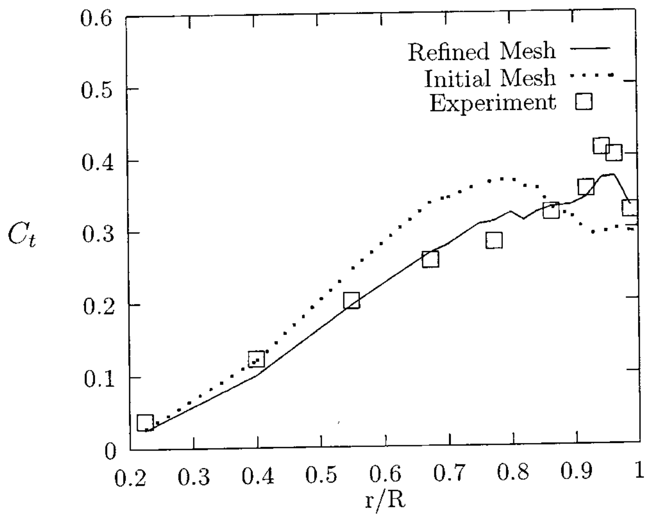

Figure 6.23: Sectional thrust distribution at $C_T/\sigma = 0.085$.

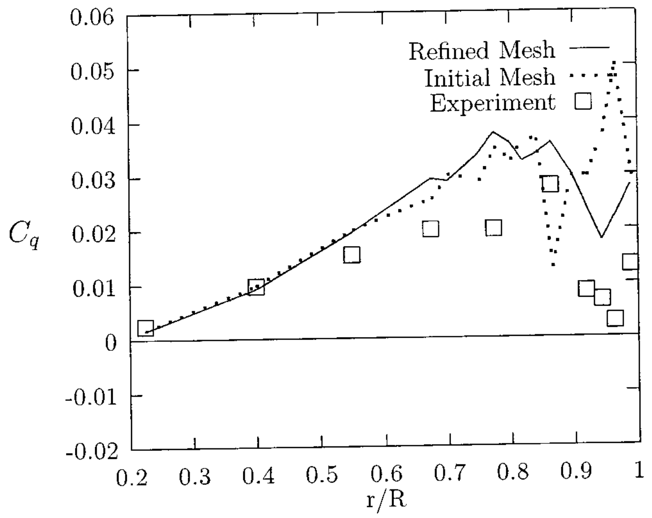

Figure 6.24: Sectional torque distribution at $C_T/\sigma = 0.085$.

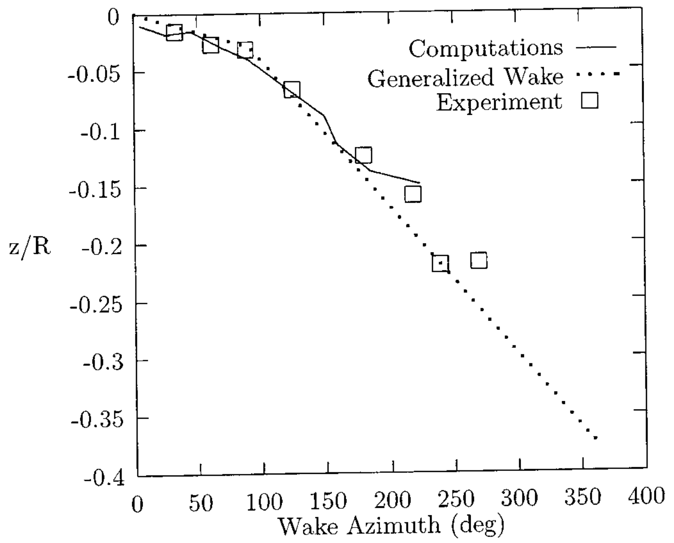

Figure 6.25: Axial displacement of tip vortex: $C_T/\sigma = 0.085$.

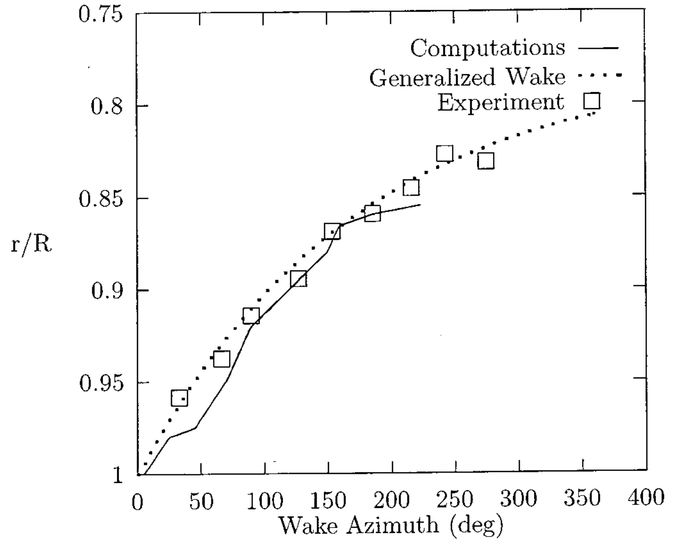

Figure 6.26: Radial displacement of tip vortex: $C_T/\sigma = 0.085$.

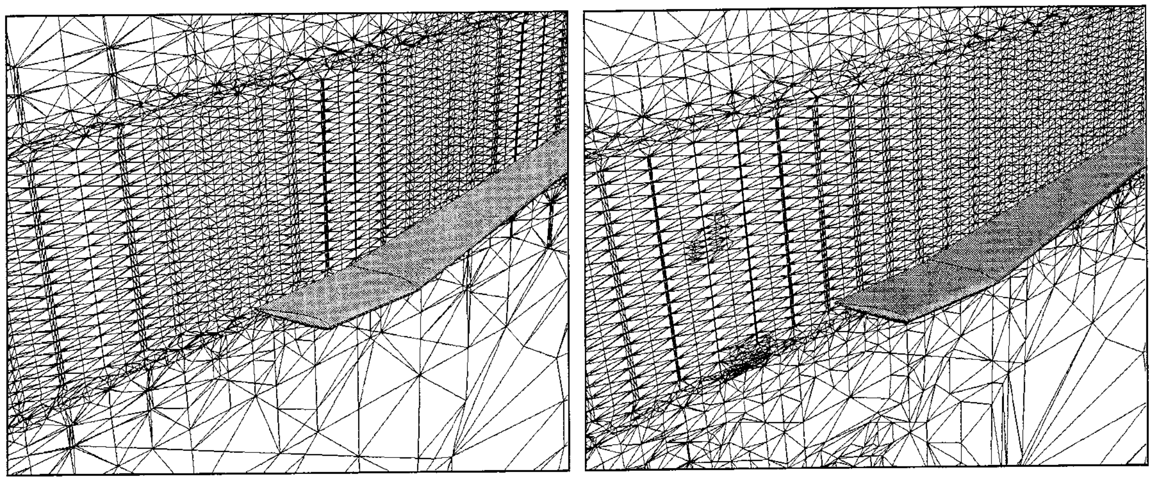

Figure 6.27: Progress of mesh adaption for initial (1,200,000 tets) and adaptive level 1 (1,750,000 tets)

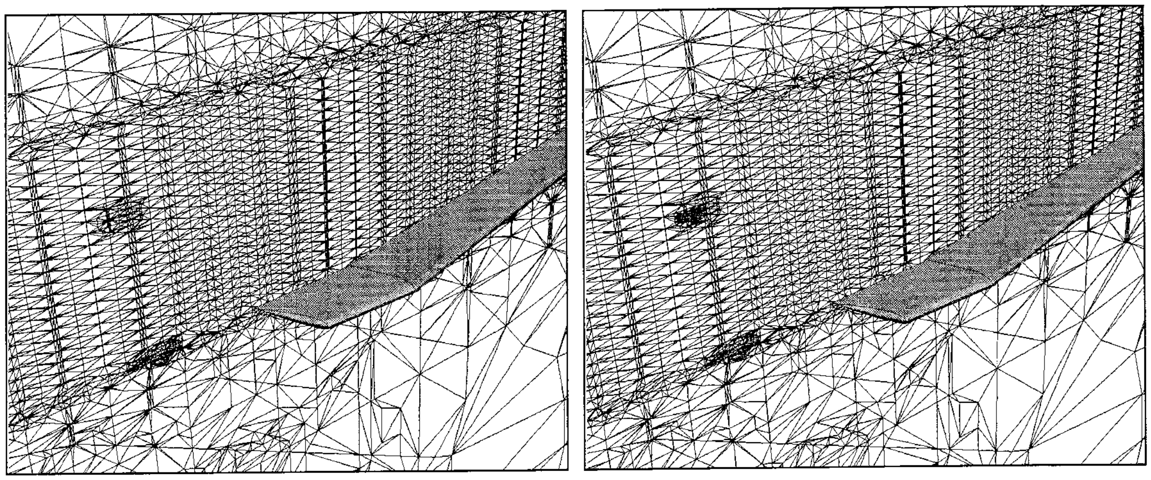

Figure 6.28: Progress of mesh adaption for adaptive level 2 (2,100,000 tets) and adaptive level 3 (2,300,000 tets)

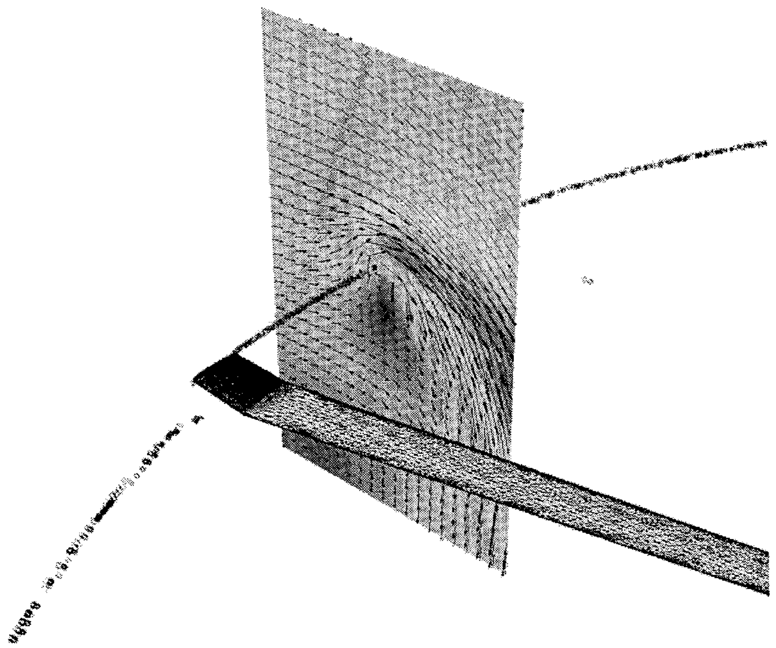

Figure 6.29: Computed tip vortex flow behind the UH-60A blade at $C_T/\sigma = 0.085$.

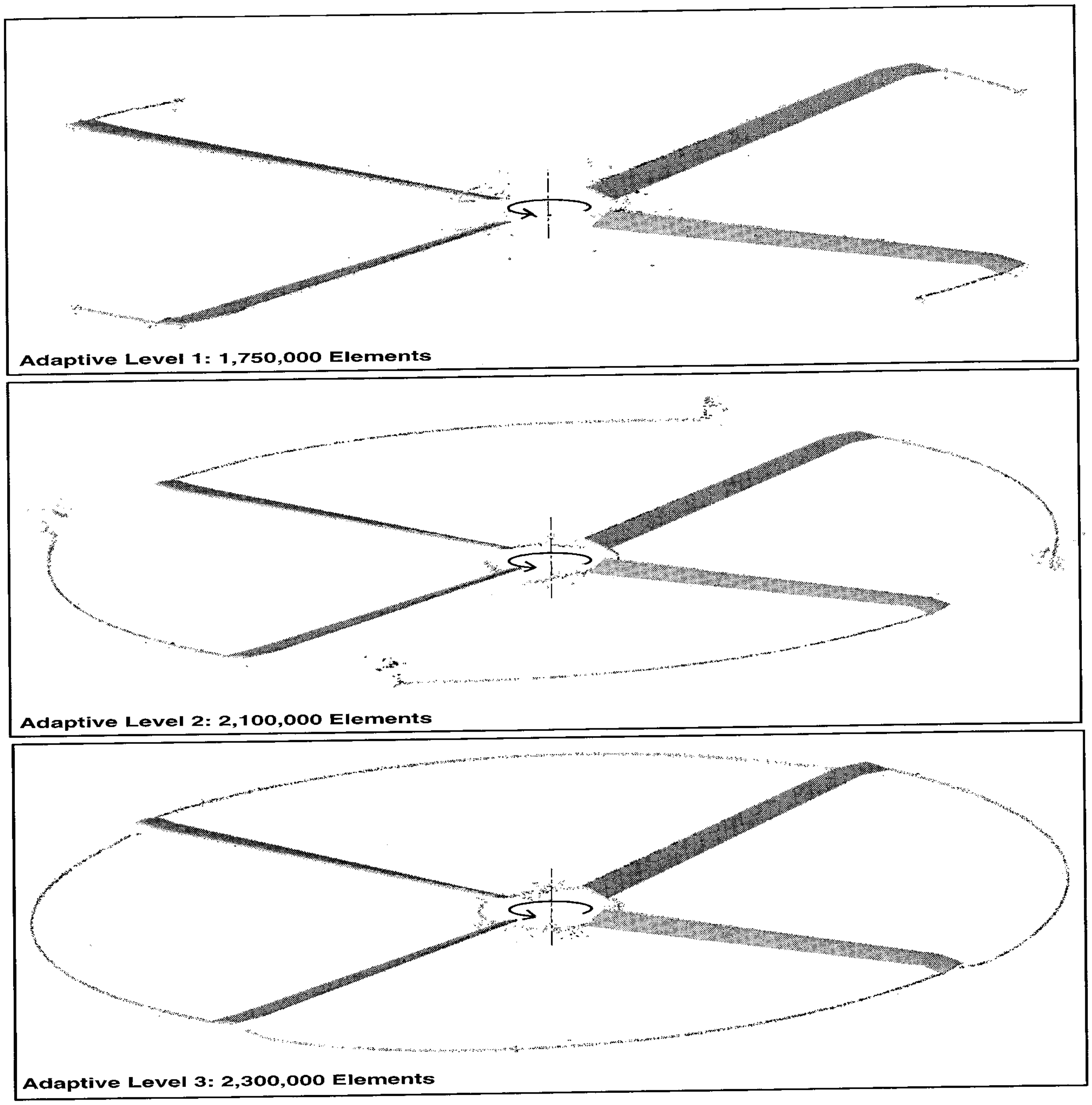
Adaptive Level 1: 1,750,000 Elements
Adaptive Level 2: 2,100,000 Elements
Adaptive Level 3: 2,300,000 Elements

Figure 6.30: Computed tip vortex flow structures for the UH-60A blade at $C_T/\sigma = 0.085$.

6.3 Caradonna-Tung Blade with Transpiration Flow Control

As a last example in this chapter, the Caradonna-Tung [50] blade is modified for transpiration velocity. For this purpose, the original Caraddona-Tung blade lower surface is modified such that a rectangular patch of area is created near the tip of the blade, as shown in Figure 6.31. The size of the rectangular patch is 10%R (span) by 4%R (chord). It is located 60% chord lengths behind the leading edge of the blade and its inboard end starts at 82% radius (see Figure 6.31).

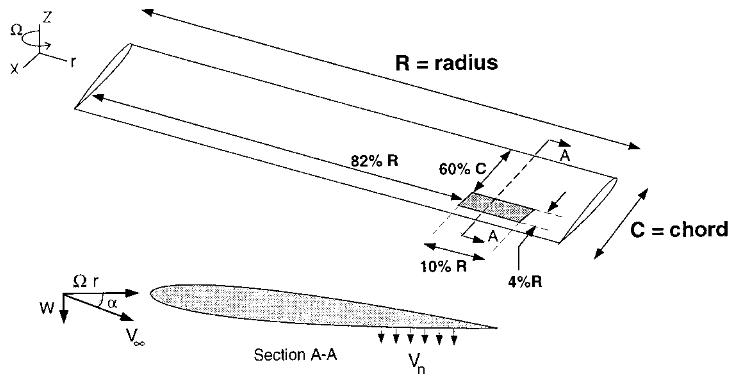

Figure 6.31: Modifed Caradonna-Tung blade for transpiration.

As a test case, we selected the tip Mach number, M_t, to be 0.439 and the collective angle, θ, was set to 8 degrees. A transpiration velocity of magnitude 20% M_t is assumed. This velocity field is a uniform velocity field and it is directed normal to the surface of the blade.

The initial mesh for the Caradonna-tung blade with transpiration contained 280,000 tetrahedral elements. Using this initial mesh, 3 levels of mesh adaptation is performed to capture the vortex structure in the flow field. At adaptive level 3 the mesh size reached to 1,500,000 tetrahedral elements. For the adapted mesh, calculated values of rotor thrust, torque and figure of merit are summarized and compared in Table 6.1.

Calculated coefficient of pressure distributions at several radial stations are compared against the baseline results. The pressure distributions at radial stations

Table 6.1: Comparision of hover performance parameters

Blade	M_t	θ	C_T	C_Q	FMI
Baseline	0.439	8^0	0.004601	0.000250	0.882
Transpiration	0.439	8^0	0.004807	0.000422	0.558

80%, 84%, 87% and 96% are shown in Figure 6.32.

At adaptive level 3, both the tip vortex and the horse-shoe vortex structures behind the blade were resolved to some extent. The tip vortex is resolved for 180^o behind the blade. At this point, the horse-shoe vortex structure due to transpiration extended about 30% chord behind the blade. Figure 6.33 and 6.34 show the computed tip vortex structure with the adaptive procedure. Notice that, the light colored tip vortex is the vortex structure calculated by the baseline Caradonna-Tung blade whereas the dark colored tip vortex structure is the current results with the transpiration. We have observed about a 4-8% more axial displacement in the tip vortex when compared to the baseline case.

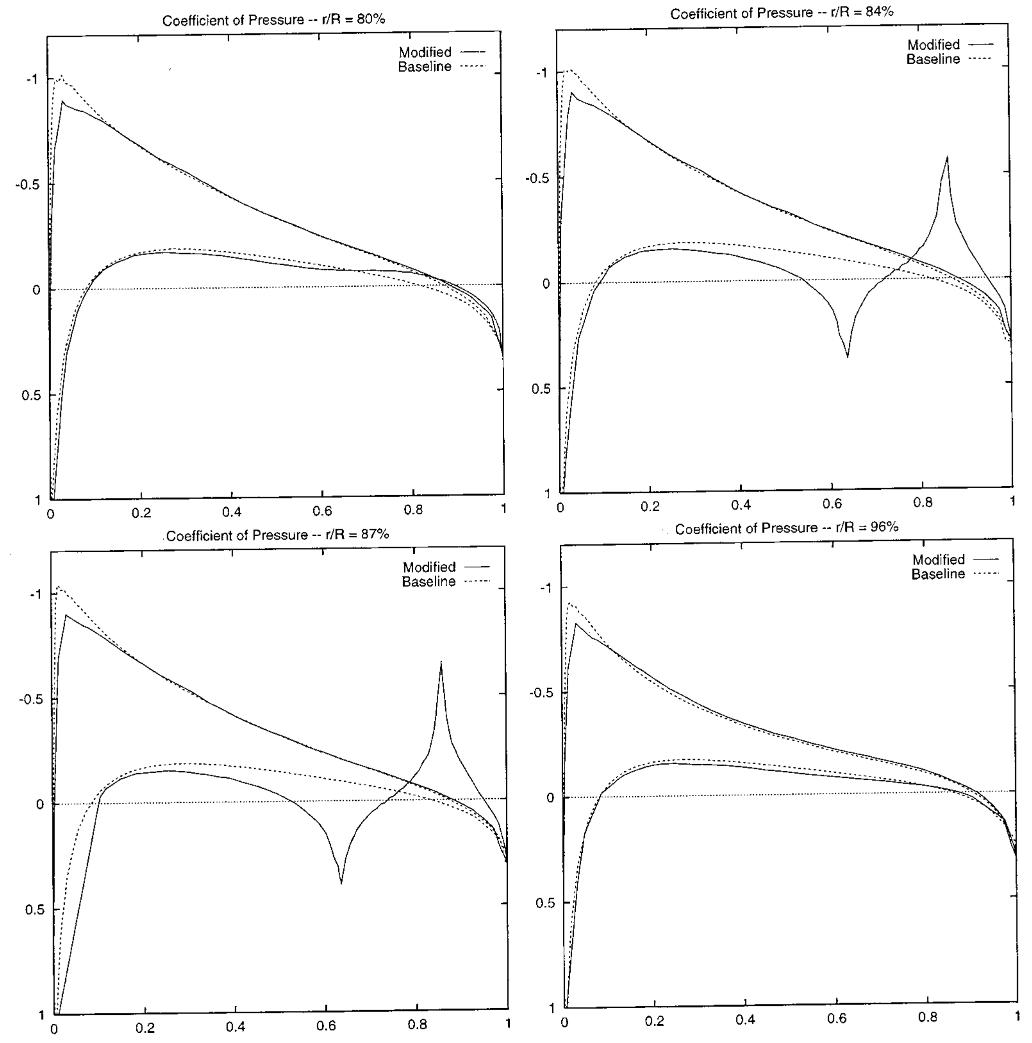

Figure 6.32: Calculated Cp distribution for the Caradonna-tung blade with transpiration: $M_t = 0.439$, $\theta = 8^0$ and $V_{trans} = 0.2M_t$

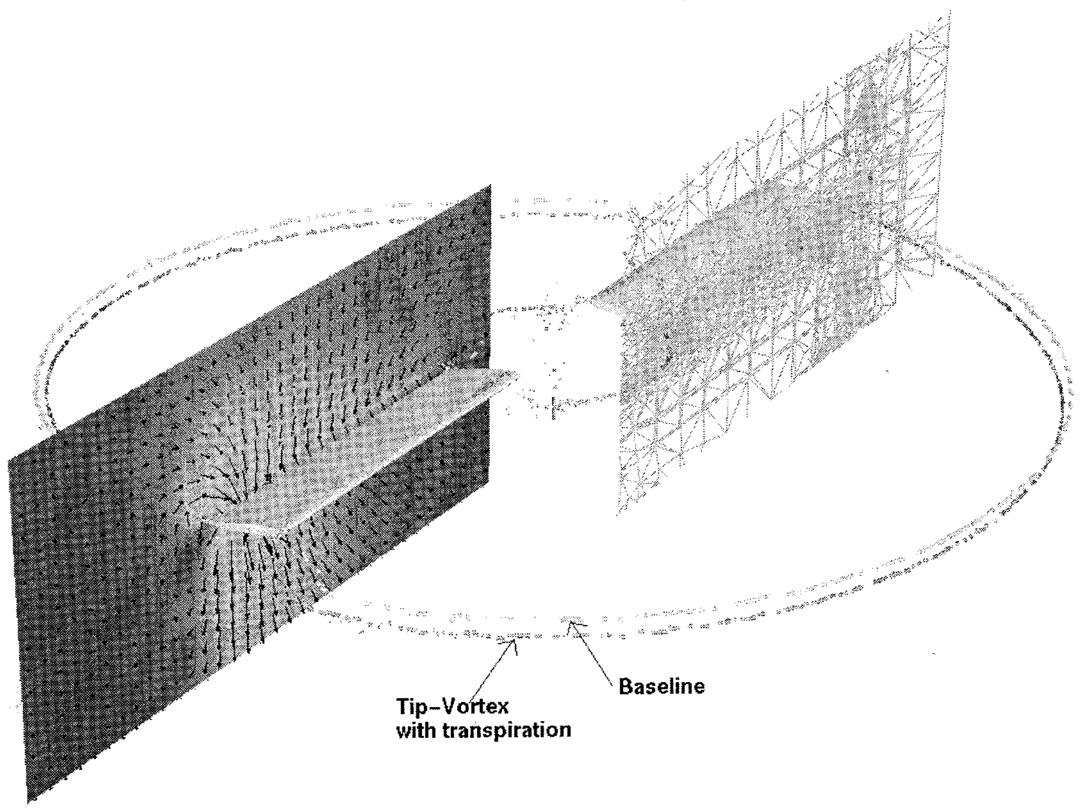

Figure 6.33: Calculated vortex flow field for the Caradonna-tung blade with transpiration

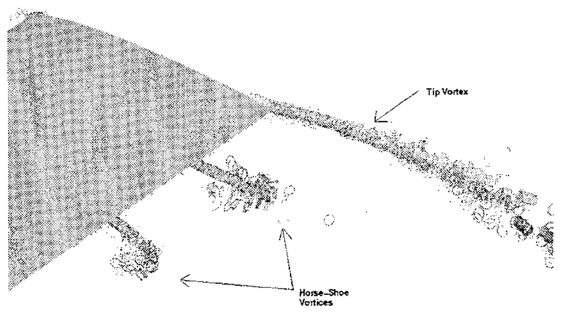

Figure 6.34: Close-up view of the tip-vortex and the horse-shoe vortex behind the TE

CHAPTER 7
AEROELASTIC COUPLING

In this chapter a method to calculate aerodynamics and aeroelastic coupling of rotor blades in hover is presented.

7.1 An Aeroelastic Coupling Procedure in Hover

A rotor blade in hover or in forward flight creates aerodynamic loads. Due to its structural flexibility, the same rotor blade deforms under these aerodynamic loads. Aerodynamic load distribution on a rotor blade can be used to determine the structural deformations. But once the blade deforms, its geometry changes and as a result the aerodynamic loads change too. There is an equilibrium point between aerodynamic loads and structural deformations. At this equilibrium point a deformed shape of the blade is balanced by the aerodynamic loads generated by the deformed geometry of the blade, and vice versa. For this reason, aerodynamic loads and deformations are coupled.

In numerical calculations, one possible approach to address rotor-blade aeroelasticity is to couple CFD calculations with CSD (Computational Structural Dynamics) procedures. There could be two coupling approaches: tight coupling and loose coupling. In a tight CFD-CSD coupling approach both aerodynamic loads and structural deformations are solved simultaneously. Whereas in a loose CFD-CSD coupling approach both aerodynamic loads and structural deformations are solved individually until an equilibrium point between the two is reached. Regardless of the type coupling approach we want to use, the important issue that needs to be addressed is the information exchange between the CFD and the CSD calculations. During a CFD-CSD coupling, the deformations of the blade must be transferred from the CSD calculation to the CFD calculation and aerodynamic loads from the CFD calculations must be transferred from the CFD to the CSD calculation.

The flowchart given in Fig. 7.1, suggests one approach for combining CFD and

CSD for rotor-blade aeroelasticity calculations in hover. The CFD solver starts the calculations for a hovering rotor blade. The CFD calculations are carried out until a converged steady-state solution for the hovering rotor blade is found. Once steady-state is reached, the CFD solver calls the CFD→CSD interface, which calculates the sectional forces and moments required by the CSD solver. Then the calculated aerodynamic force and moment information is sent to CSD solver to start the structural calculations. When the CSD solver is started, the beam-element structural model of the blade is first pre-stressed by rotating the blade at hovering rotational speed Ω. Then, the calculated aerodynamic forces and moments are applied to the blade to obtain the resultant deflections. The deflected shape of the CSD model is then processed by the CSD→CFD interface to from a new three dimensional CFD model. This loose coupling cycle between CFD and CSD analyses is continued until the incremental deflections of the blade drop below a pre-determined tolerance.

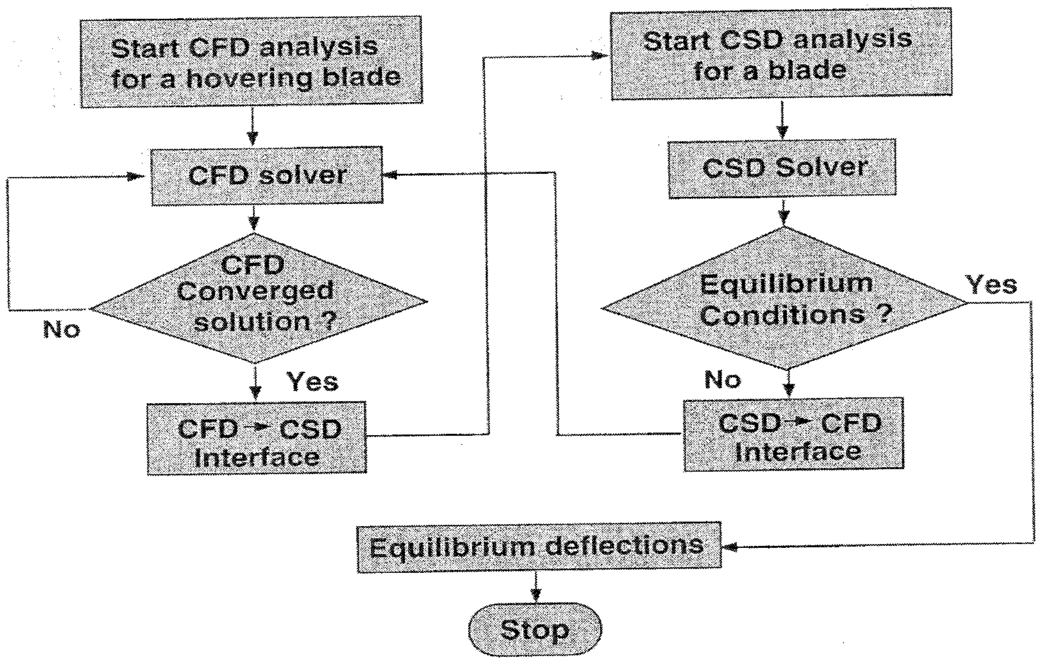

Figure 7.1: Computational flowchart for loose aeroelastic coupling.

The CFD solution procedure introduced in chapter 3 uses complete three-dimensional geometry. For our CSD calculations, we selected the DYMORE [63] finite element analysis code. This CSD analysis code uses a reduced-dimension representation of rotor blades. That is, DYMORE models the rotor blades as one-dimensional beams. More information about the CSD modeling technique and assumptions of DYMORE analysis code can be found in reference [63]. The important issue that needs to be addressed here is that in our CFD-CSD coupling procedure, we have a dimensional difference between the geometric representation of rotor blades in CFD and CSD calculations. This dimensional difference between CFD and CSD models requires specific consideration. These issues are explained next.

7.2 CFD→CSD interface

Consistent transfer of aerodynamic forces and moments between CFD and CSD models requires integration of continuous pressure distribution over the blade surface of the CFD model. Result of this integration is then projected onto the elastic axis where CSD model is situated. As shown in Fig. 7.2, the integration of pressure eventually will be performed on triangular faces.

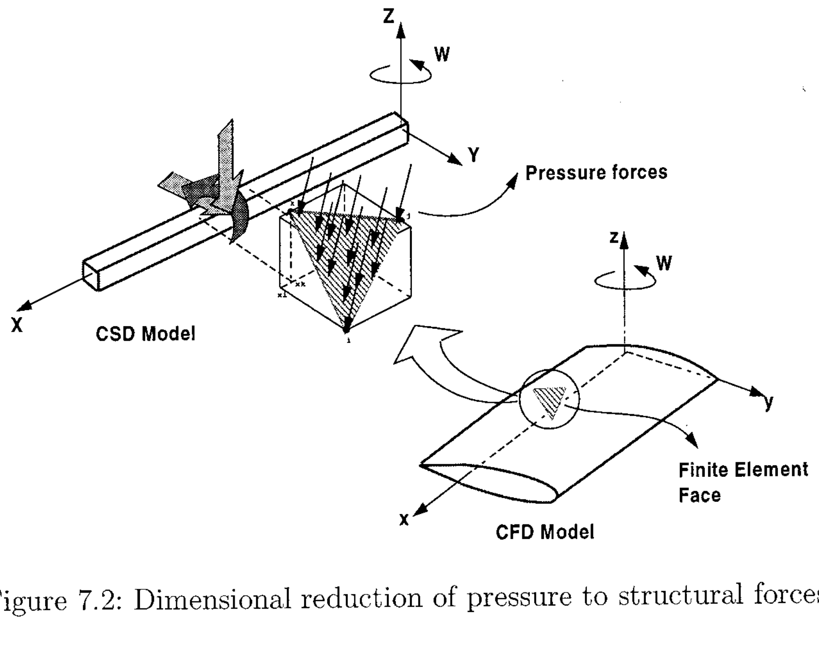

Figure 7.2: Dimensional reduction of pressure to structural forces.

The forces and moments needed by the structural analysis code DYMORE [63] at each radial control point along the blade are normal forces, chordwise forces and pitching moment about the elastic axis. The definition of these forces and moments are illustrated on an airfoil cross-section in Figure 7.3. Given the coefficient of

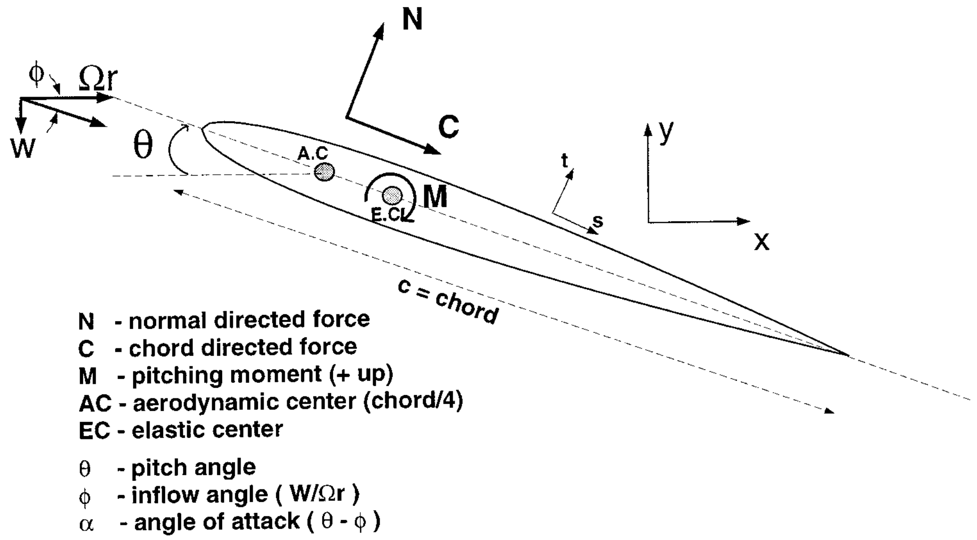
N
C
M
A.C
E.C
ϕ Ωr
W
θ
t
s
y
x
c = chord
N - normal directed force
C - chord directed force
M - pitching moment (+ up)
AC - aerodynamic center (chord/4)
EC - elastic center
θ - pitch angle
ϕ - inflow angle (W/Ωr)
α - angle of attack (θ - ϕ)

Figure 7.3: Sectional forces and moment acting on an airfoil section.

pressure distribution, $C_p(s,r)$ on the blade surface, where s is the local chord-wise direction as designated in Fig. 7.3, we can write down the non-dimensional normal and chord directed force coefficients as follows

$$C_N(r) = \int_0^1 C_p^{\text{lower}}(s,r)d(s/c) - \int_0^1 C_p^{\text{upper}}(s,r)d(s/c) \tag{7.1}$$

$$C_C(r) = \int_0^1 \frac{dt}{ds}C_p^{\text{upper}}(s,r)d(s/c) - \int_0^1 \frac{dt}{ds}C_p^{\text{lower}}(s,r)d(s/c) \tag{7.2}$$

where $\frac{dt}{ds}$ is the local slope of the airfoil surface. Similarly, the pitching moment coefficient about the elastic center s_{EC} is given as

$$C_M(r) = \int_0^1 \left(\frac{s_{EC}}{c} - \frac{s}{c}\right)\left(C_p^{\text{lower}}(s,r) - C_p^{\text{upper}}(s,r)\right)d(s/c) \tag{7.3}$$

Note that a positive value of C_M denotes a nose-up moment.

Alternatively, the dimensional form of normal force N, chord force C, and pitching moment M can be obtained as follows [56]

$$\begin{aligned} N(r) &= \frac{1}{2}C_N(r)c(r)\rho_\infty(\Omega r)^2 \\ C(r) &= \frac{1}{2}C_C(r)c(r)\rho_\infty(\Omega r)^2 \\ M(r) &= \frac{1}{2}C_M(2)c^2(r)\rho_\infty(\Omega r)^2 \end{aligned} \tag{7.4}$$

Note that both forces and moments are obtained in terms of per unit length since they are sectional forces and moments.

7.3 CSD→CFD interface

As discussed earlier, the CSD analysis evaluates the finite deflections of the blade in a one dimensional model space. This deflection and rotation information has to be converted to the three dimensional displacements of the CFD model. This is accomplished by assuming that the cross-sections of the 3D blade are rigid and that the translations and rotations of a CSD station can be applied to the CFD cross section. By using several spanwise stations that are common to both CSD and CFD geometries, it is possible to apply the CSD translations and rotations to the corresponding cross sections of the 3D CFD blade and thereby construct an updated CFD model. This process is shown schematically in Figure 7.4. Once the individual cross sections are transformed geometrically in space then a simple lofting [1] operation [67] generates the deflected surface of the CFD blade.

Once the geometric model for CFD analysis is updated, it remains to either generate new a mesh or update the existing mesh from the undeformed blade to the deformed blade surface. Although the construction of a new mesh for the deformed blade is possible this approach can very time consuming and inefficient. An alternative solution to update the computational mesh is to develop a *mesh-motion* procedure that can move the grid points while maintaining both the mesh validity [65] and quality [66]. Computational time gained through applying mesh-motion in contrast to constructing a new mesh at every time step can be substantial if an unstructured mesh topology is used. In the next section we will describe a general

[1]Lofting is a geometric surface fitting operation over a series of curves

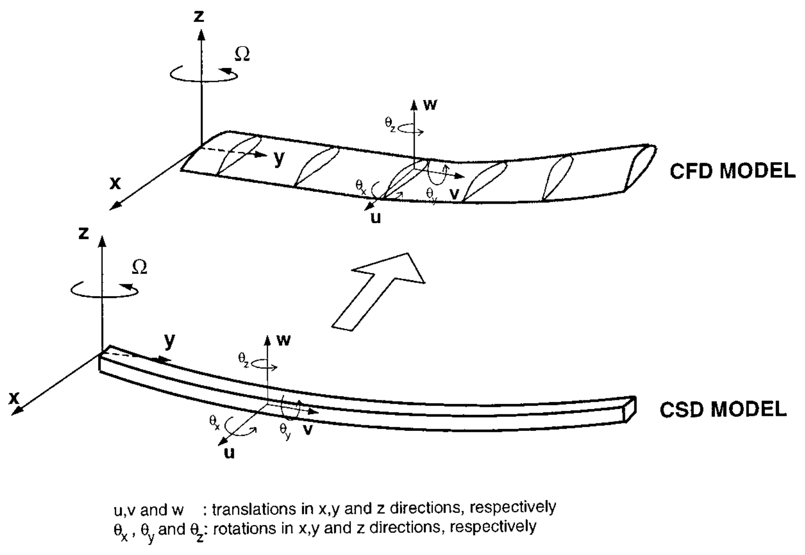
z
Ω
x
y
w
θ_z
θ_x
θ_y
v
u
CFD MODEL
CSD MODEL
u,v and w : translations in x,y and z directions, respectively
θ_x, θ_y and θ_z: rotations in x,y and z directions, respectively

Figure 7.4: Dimensional upgrade of structural deflections to surface deformations.

mesh-motion procedure that can be used in CSD→CFD interface.

7.3.1 A Deforming Mesh-Motion Procedure

In order to start our discussion of a mesh-motion algorithm, we first consider the Laplacian smoothing method given in [64], where a basic node repositioning formula is written as

$$x = \frac{1}{n}\sum_{i=1}^{n} x_i, \quad y = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{7.5}$$

Here, n is the number of points that surrounds the node at (x,y). Note that this formula simply takes the arithmetic averages of n nodes around the node of interest to find its new position. Note that such an operation would be neutral if the mesh is perfectly uniform as shown in Fig. 7.5. Since the elements are regular and all have the same size, application of Laplacian smoothing will not change the location of the point p. We can also view the Laplacian smoothing as a node repositioning method based on the centroid of a group of neighboring nodes. It relocates a point to a new position by simply finding the centroid of the points around it.

If we let (x_0, y_0) be the old coordinates of the point of interest, we can re-write

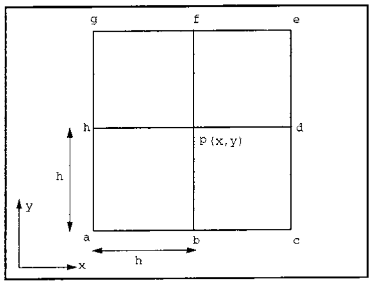

Figure 7.5: An equispaced mesh with quadrilateral elements.

the Laplacian smoothing formula as follows

$$nx - nx_0 = \sum_{i=1}^{n} x_i - \sum_{i=1}^{n} x_0 \tag{7.6}$$

$$ny - ny_0 = \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} y_0 \tag{7.7}$$

or,

$$\Delta x = \frac{1}{n}\sum_{i=1}^{n}(x_i - x_0) \tag{7.8}$$

$$\Delta y = \frac{1}{n}\sum_{i=1}^{n}(y_i - y_0) \tag{7.9}$$

where, Δx and Δy are the difference between old and new coordinate locations as shown in Figure 7.6

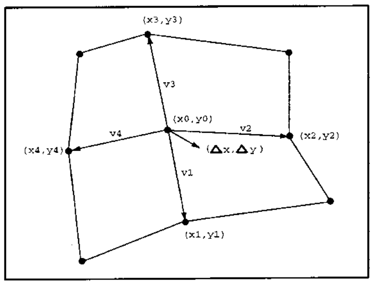

Figure 7.6: Displacement of an interior mesh point.

More generally, in vector form, we can re-write the smoothing equations in the following form

$$\Delta \vec{x} = \frac{\sum_{i=1}^{n} w_i \Delta \vec{x}_i}{\sum_{i=1}^{n} w_i} \tag{7.10}$$

where, w_i's are weights of the averaging process. Note that in the case all the weights are taken as unity, the averaging formula will reduce to the original Laplacian smoothing. Given the displacement vector $\Delta \vec{x}$, the new coordinates of the point can be found by

$$\vec{x} = \vec{x}_0 + \Delta \vec{x} \tag{7.11}$$

A question that needs to be answered at this point is "how should we choose the weights?" The most basic capability to look for in any mesh repositioning operation is the preservation of mesh validity. Therefore, selection of weights should be such that they should avoid elements from becoming invalid. One possibility for this is to choose the weights as functions of element geometry (edge length, element heights or similar quantities). For example we can select the weights such that they are inversely proportional to the edge lengths

$$w_i \propto \frac{1}{l_i} \quad \text{as} \quad l_i \rightarrow 0 \tag{7.12}$$

The idea here is that the edge-lengths are being used to constrain the mesh-motion. As the edge length goes to zero we see that weights will increase infinitely. Introduction of such a penalty to the mesh motion procedure will force mesh points which are very close to each other to move lesser than the rest of the points in the mesh. However, it is possible that all the nodes of a linear tetrahedra collapse onto a plane while maintaining the edge lengths to be greater than zero. In order to avoid this situation either the element volume should be used as weights or the edge length weighted average smoothing has to be followed by a mesh validity check process. The latter approach is used in the mesh-motion procedure described here.

In literature, weighted-average smoothing is also known as "spring analogy" for mesh motion [69]. The reason for this is that the weights, which are inversely proportional to the edge lengths, can be thought as the springs that replace the mesh edges. Using this analogy the weighted-average smoothing formula, given in Eq. 7.10, can be obtained by considering the mesh as a spring network exposed to displacements from its boundaries, and solving for the displacement of the interior node by the force-equilibrium equations in each coordinate direction. Reference [69] looks at this issue from this perspective.

Now consider the weighted-average smoothing formula given by Eq. 7.10. One can easily see that this equation will result in a system of algebraic equations of the following form

$$\begin{aligned}
\Delta\vec{x}_1 - \frac{1}{\beta_1}(w_{i1}\Delta\vec{x}_i + w_{j1}\Delta\vec{x}_j + \cdots + w_{n1}\Delta\vec{x}_n) &= 0 \\
\Delta\vec{x}_2 - \frac{1}{\beta_2}(w_{i2}\Delta\vec{x}_i + w_{j2}\Delta\vec{x}_j + \cdots + w_{n2}\Delta\vec{x}_n) &= 0 \qquad (7.13) \\
& \qquad\qquad (7.14) \\
\Delta\vec{x}_N - \frac{1}{\beta_N}(w_{iN}\Delta\vec{x}_i + w_{jN}\Delta\vec{x}_j + \cdots + w_{nN}\Delta\vec{x}_n) &= 0
\end{aligned}$$

where β_i is the sum of the weights of the nodes surrounding the node i. In matrix form, we can write

$$G(\Delta\vec{x}) = 0 \qquad (7.15)$$

Note that, this is a nonlinear system. Linearization can be done by using Taylor series expansion,

$$G(\Delta\vec{x}^{n+1}) = G(\Delta\vec{x}^n) + \frac{\partial G}{\partial\Delta\vec{x}}p^n \qquad (7.16)$$

where, the incremental solution,p, is given by

$$p^n = \Delta\vec{x}^{n+1} - \Delta\vec{x}^n \qquad (7.17)$$

Here, superscript n denotes the iteration level. Substituting these relations back into the original nonlinear system, we obtain the following relation

$$J^n p^n = -R^n \quad \text{with} \quad J^n = \frac{\partial G}{\partial \Delta \vec{x}} \quad \text{and} R^n = G^n \tag{7.18}$$

The resulting system can be solved by a conjugate gradient-type algorithm, such as GMRES [70]. The other alternative to solve weighted-average smoothing equations is to handle each node point in the mesh one at a time and solve the displacements by an iterative solver. Clearly, such an approach will not only ease the implementation but also will reduce the amount of computer memory and time. Therefore, consider the following form of the smoothing equations

$$\Delta \vec{x}_i^{\,n+1} = \frac{1}{\beta} \sum_{j=1}^{M} w_j \Delta \vec{x}_j^{\,*} \qquad i = 1, 2, \dots, N \tag{7.19}$$

where, asterisk denotes a provisional solution between iterations n and $n+1$. At the end of each iteration, the nodal coordinates of the points can be found by

$$\Delta \vec{x}^{n+1} = \vec{x}^n + \rho \Delta \vec{x}^{n+1} \tag{7.20}$$

where ρ is a relaxation factor that can be adjusted either for under- or over-relaxation. Also different approximation to explicit displacement terms on the RHS will yield different algorithms. Following the geometric slope of the line between it-

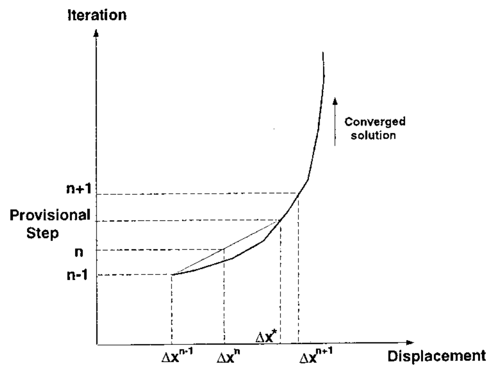

Figure 7.7: Graphical illustration of provisional step for predictor-corrector method.

erations (see Fig.7.7), it can be shown that the intermediate solution can be written as

$$\Delta\vec{x}^{*} = 2.0\Delta\vec{x}^{n} - \Delta\vec{x}^{n-1} \tag{7.21}$$

This choice will translate to a simple predictor-corrector algorithm. If the intermediate solutions are taken only from the iteration level n, then we perform Jacobi iterations,

$$\Delta\vec{x}^{*} = \Delta\vec{x}^{n} \tag{7.22}$$

in this case we do not need to keep the solution from previous two levels of iteration in computer memory. Both predictor-corrector and Jacobi solution methods are implemented in the mesh-motion procedure described here.

7.3.2 Parallel Computing Issues of the Mesh-Motion Procedure

In a parallel computing environment, the computational mesh is distributed over the number of processors being used. That is, each processor works on a part of the mesh. Mesh partitioning is the procedure utilized to distribute a whole mesh over the number of processors available. For example in our parallel calculations, the mesh is partitioned using the recursive bisection [71] algorithm. Both description and the implementation details of recursive bisection algorithm can be found in reference [72]. Figure 7.8 shows schematically the mesh partitioning process. The boundary between two meshes is called a partitioned boundary. The mesh motion

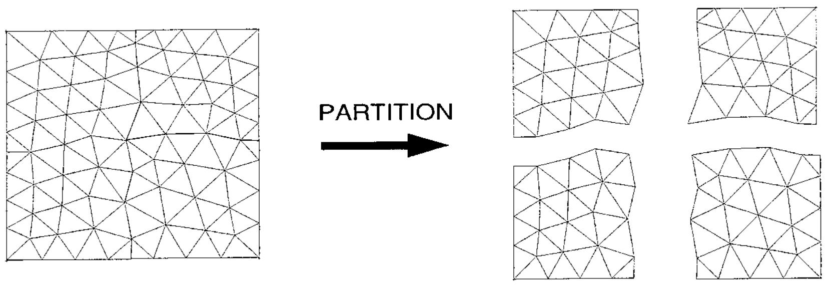

Figure 7.8: A schematic view of mesh partitioning process.

equation requires knowledge of neighboring vertices and their displacements for a given vertex. When we consider a vertex that is located on a partition boundary, it is obvious that the vertex will require information from the adjacent processors to be able to calculate Eq. 7.10. This requirement forces us to apply the mesh motion algorithm in two steps. The first step involves solving the mesh motion equations for all the internal vertices, and the second step is to solve the mesh-motion for the partition boundary vertices. Given an internal vertex which has all of its neighbors in the same partition, the displacements are calculated by

$$\Delta\vec{x_i}^{n+1} = \frac{1}{\beta}\sum_{j=1}^{M} w_j \Delta\vec{x_j}^* \qquad i = 1, 2, \dots, N \tag{7.23}$$

where M is the number of neighboring vertices around vertex i and N is the total number of internal vertices in the partition.

For a vertex, which is located on a partition boundary, Eq. 7.23 can be modified as follows

$$\Delta\vec{x_i}^{n+1} = \frac{\sum_{n=1}^{N_{pb}}(\sum_{j=1}^{M_n}\Delta\vec{x_j}^* w_j)}{\sum_{n=1}^{N_{np}}(\sum_{j=1}^{M_n} w_j)} \tag{7.24}$$

where N_{pb} is the number of adjacent partition boundaries for the vertex i, and M_n is the number of neighboring vertices of vertex i in partition n.

The inter-processor communication process for vertex i which is located on a partition boundary, is explained in Fig.7.9. In Figure 7.9 there are four partitions.

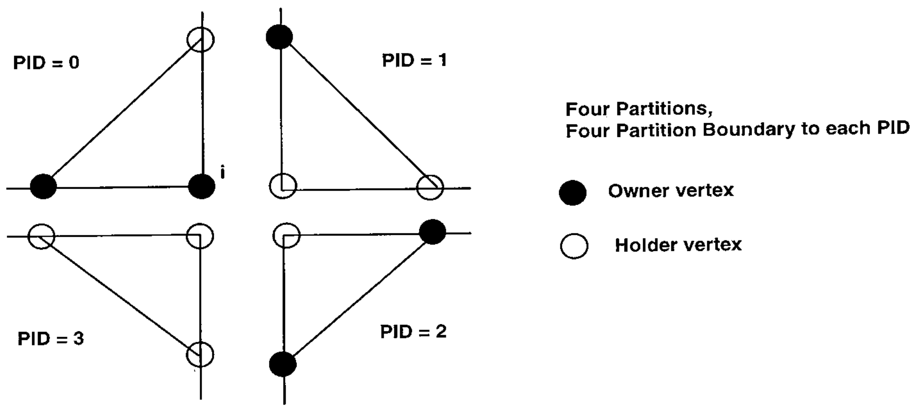

Figure 7.9: Owner-holder relationship for a partition boundary vertex.

In any given partition, vertex i has to communicate with three partition boundaries. The mesh partition operator [72] used in the calculations flags the partition boundary vertices as either *owner* or *holder*. During this flagging process only one vertex is selected as an owner and all other copies of the vertex i are flagged as holder. This distinction process allows us to do calculations on only owner vertex and send the final results to holder vertices in the neighboring processors.

In order to compute the displacements of vertex i, we do the following steps

1) Compute local contributions of displacements using Eq. 7.23

2) Send/Receive the displacements obtained in step 1

3) Sum all the data received from neighboring processors using Eq. 7.24

4) Send calculated displacements from an owner vertex to its neighboring holder vertices

For example, in processor p0 the mesh-motion equation can be written as

$$\Delta\vec{x}_i^{n+1}|_{\mathrm{p0}} = \frac{\sum_{j=1}^{1} \Delta\vec{x}_j^{\,*} w_j}{\sum_{j=1}^{1} w_j}|_{\mathrm{p0}} \tag{7.25}$$

Note that summation is carried over only 1 neighboring vertex. All the holder vertices are skipped from the summation process since their contributions are going to come from the adjacent partition boundaries. Similarly, the processors p1,p2 and p3 are going to carry out the following calculations

$$\begin{gathered}\Delta\vec{x}_i^{n+1}|_{\mathrm{p1}} = \frac{\sum_{j=1}^{1} \Delta\vec{x}_j^{\,*} w_j}{\sum_{j=1}^{1} w_j}|_{\mathrm{p1}}, \quad \Delta\vec{x}_i^{n+1}|_{\mathrm{p2}} = \frac{\sum_{j=1}^{2} \Delta\vec{x}_j^{\,*} w_j}{\sum_{j=1}^{2} w_j}|_{\mathrm{p2}}, \\ \Delta\vec{x}_i^{n+1}|_{\mathrm{p3}} = \frac{\sum_{j=1}^{0} \Delta\vec{x}_j^{\,*} w_j}{\sum_{j=1}^{0} w_j}|_{\mathrm{p3}}\end{gathered} \tag{7.26}$$

Processor 1 carries summation over one vertex, processor 2 two vertices and processor 3 carries no summation at all (see Fig. 7.9). As soon as all the processors finishes their work on the partition boundaries, they receive (or send) two distinct pieces of data form the sender (or receiver) vertex. They are; the dominator and denominator of the displacement equation. For instance, vertex i at partition 0 receives the following info from the surrounding partitions;

$$\begin{array}{lll} \text{From Partition 1 :} & (\sum_{j=1}^{1} \Delta\vec{x_j}^{*} w_j)_{\mathrm{p1}} & (\sum_{j=1}^{1} w_j)_{\mathrm{p1}} \\ \text{From Partition 2 :} & (\sum_{j=1}^{2} \Delta\vec{x_j}^{*} w_j)_{\mathrm{p2}} & (\sum_{j=1}^{2} w_j)_{\mathrm{p2}} \\ \text{From Partition 3 :} & (\sum_{j=1}^{0} \Delta\vec{x_j}^{*} w_j)_{\mathrm{p3}} & (\sum_{j=1}^{0} w_j)_{\mathrm{p3}} \end{array} \tag{7.27}$$

As a result of the data received on processor p0, the displacement at vertex i is calculated by summing all the contributions as follows

$$\Delta\vec{x_i} = \frac{(\sum_{j=1}^{1} \Delta\vec{x_j}^{*} w_j)_{\mathrm{p0}} + (\sum_{j=1}^{1} \Delta\vec{x_j}^{*} w_j)_{\mathrm{p1}} + (\sum_{j=1}^{2} \Delta\vec{x_j}^{*} w_j)_{\mathrm{p2}} + (\sum_{j=1}^{0} \Delta\vec{x_j}^{*} w_j)_{\mathrm{p3}}}{(\sum_{j=1}^{1} w_j)_{\mathrm{p0}} + (\sum_{j=1}^{1} w_j)_{\mathrm{p1}} + (\sum_{j=1}^{2} w_j)_{\mathrm{p2}} + (\sum_{j=1}^{0} w_j)_{\mathrm{p3}}} \tag{7.28}$$

As a last step, the displacements at holder vertices are updated by sending the results of Eq. 7.28 from processor p0 to processors p1, p2 and p3. The parallel mesh motion procedure explained here gives exactly the same results as its serial implementation. The inter-processor message passing for the parallel mesh-motion is obtained by the parallel mesh database library of reference [72].

7.4 A Torsionally Soft Model Blade

In chapter 6, we analyzed several rotor blades in hover using the adaptive CFD procedure explained in chapters 3 and 4. In all those hover calculations, the assumption was that the blades were structurally rigid. Therefore, the deformations due to aerodynamic loads were not taken into account. In reality, we know that the helicopter blades are flexible and will deform under aerodynamic loads. In this section we examine a simple two-bladed rotor that has been used frequently both in experiments [73] and numerical calculations [74] [75] to analyze its aeroelastic deformations in hover. The example calculation uses the CFD-CSD procedure described earlier in this chapter.

The hingeless model blade used has a rectangular planform and is an untwisted and untapered NACA0012 airfoil section. The rotational speed of the blade is 1000rpm. Figure 7.10 shows the schematic of the model blade and the definition of the deformations.

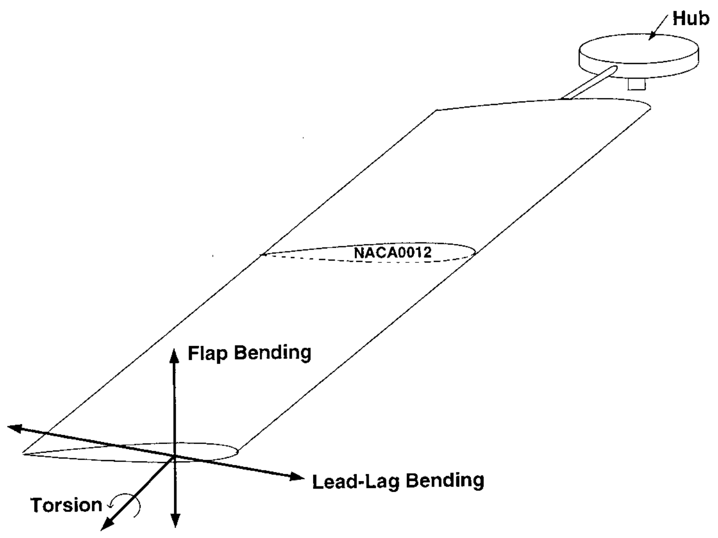

Figure 7.10: Schematic of the model blade [73].

A beam-element structural model of the blade is analyzed using the DYMORE [63] analysis code. During the structural modeling of this blade, it is assumed that there is no structural damping, no blade root offset, no precone and droop and no pitch flexure. The root-end of the blade is modeled as a cantilever beam. Table 7.1 summarizes the structural properties used for this model blade [73]. These structural properties are used to construct a DYMORE model for CFD-CSD coupling. The DYMORE finite element analysis code has the capability to model rotor blades using linear, quadratic and cubic elements. In our calculations we used cubic elements to construct our CSD models. A mesh dependency study on the CSD model has been conducted to establish the accuracy of the structural calculations. It has been concluded that the results showed no dependency on mesh resolution when a blade is modeled with 4 cubic elements and more. Therefore, the CSD model is constructed using 4 cubic elements and 13 nodes. 21 air-stations are selected along the elastic axis of the blade to transfer load and deformation data between CFD and CSD calculations. Appendix A gives the details of the DYMORE model used in our CFD-CSD coupling.

Table 7.1: Structural parameters of the model blade.

Parameters	Units	Value
Running Mass	(kg/m)	0.341
Elastic Axis	% chord	25.8
Flap Bending Stiffness	N-m^2	17.22
Chord Bending Stiffness	N-m^2	357.8
Torsional Stiffness	N-m^2	5.10
Polar Moment of Inertia	kg-m^2	1.744×10^{-4}

During the calculations, the static analysis option of DYMORE is used to obtain the equilibrium deformations of the blade at each cycle of the CFD-CSD coupling. In the static analysis, DYMORE advances the solution to steady-state by taking pseudo time steps. In our calculations 20 pseudo time steps were taken in CSD analysis to reach steady-state at each static analysis step. The gravity force is taken into account.

7.5 CFD-CSD Coupled Results for the Model Blade

A three dimensional model of the blade [73] is constructed for CFD calculations. The surface of the blade is constructed using 21 airfoil sections from root of the blade to the tip of the blade. These airfoil sections were selected at radial locations which are coincident with the air-station radial locations of the DYMORE CSD model. Surface description of the blade is obtained by lofting a Bezier surface [67] onto the airfoil sections. This model construction process has been coded as C language routine into the CSD$\rightarrow$CFD interface so that during the CFD-CSD coupling a new deformed geometric model can be constructed without having to interrupt the calculations. The collective angle, θ, of the undeformed blade is set to 4^0.

Using the undeformed geometric model of the blade, a mesh consisting of 200,000 tetrahedral elements is generated to start the CFD calculations. Figure 7.11 shows this initial mesh on the surface of the blade. This initial mesh is partitioned onto 8 processors of an IBM-SP2. The CFD calculations are then started at a tip Mach number, $M_t = 0.296$. At the steady-state, the CFD$\rightarrow$CSD interface is called to integrate the aerodynamic loads and moments using the pressure distribution on the blade surface. Aerodynamic loads are then transferred to the main routine of the DYMORE analysis code to start the CSD calculations. Since DYMORE is a serial code, the CSD calculations are carried on one (root) processor. This means that during the CSD calculations 7 out of 8 processors were instructed to wait until DYMORE code finishes its work. However, since one steady-state solution time for DYMORE code was only a small fraction (less than 1/4) of one steady-state step of the CFD solver, the wait time was not long. At each CSD solution case, the calculations are first started with a rotating blade (i.e. centrifugal and gravity loads) without the aerodynamic loads. Then, aerodynamic loads are applied to obtain a steady-state deformation of the blade.

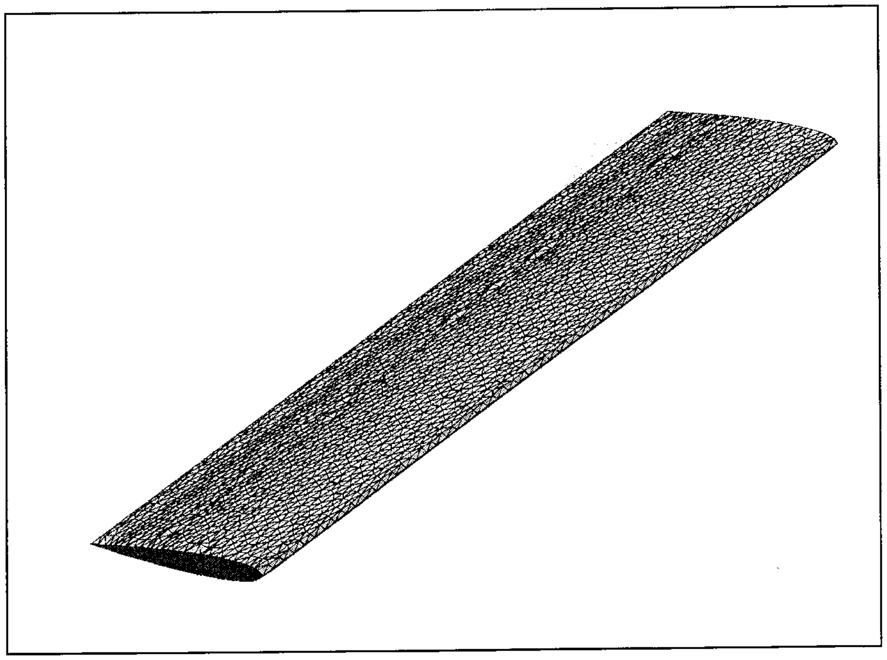

Figure 7.11: Computational mesh of the model blade used in CFD-CSD calculations

Once the deformation information is obtained from CSD analysis, then the CSD$\rightarrow$CFD interface is called to generate a deformed CFD model of the blade.

To do this, the translations and rotations at each air-station are applied to the airfoil sections that are used to build the undeformed geometry of the CFD model. The model construction process is then initiated to build the surface and outer boundaries of the CFD model. At this point, the CFD mesh still belongs to the undeformed geometry of the model. To move the mesh points from the undeformed blade geometry to the deformed geometry, first the mesh vertices classified on the blade surface are projected from the undeformed blade surface to the deformed blade surface. This projection operation is done as follows: Every vertex which is located on the blade surface, has unique parametric coordinates (U,V) which can be obtained from the CAD definition of the surface. As the blade surface deforms, the global coordinates (x,y,z) of a mesh vertex change, however the parametric coordinates of the same vertex remain the same. Using the affine transformation [68] between Cartesian and parametric coordinates, we can then find the new coordinates of a vertex on the deformed blade. This vertex projection process is shown schematically in Figure 7.12. After all the vertices classified on the blade surface are projected

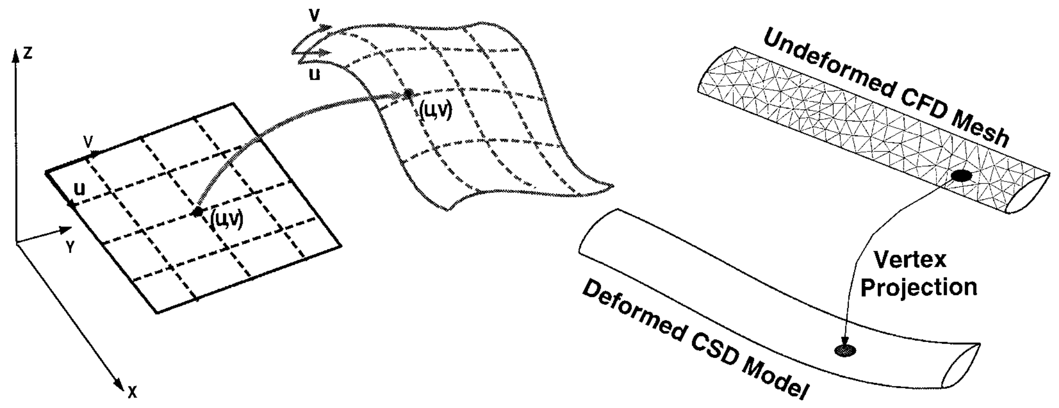

Figure 7.12: Projection of a vertex from undeformed geometry to a deformed geometry.

from undeformed blade surface to deformed blade surface, the relative displacements of these vertices are used as the boundary condition to the mesh-motion procedure described in section 7.2. Using the weighted-average mesh-motion procedure the interior vertices of the mesh are smoothed. Usually, 5-10 predictor-corrector steps are used to successfuly smooth the interior vertices. We have to note that the mesh-motion procedure described in sub sections 7.3.1-7.3.2 cannot take an invalid mesh and make it valid. Therefore, if the mesh becomes invalid after we project the vertices on the blade surface, the mesh-motion procedure cannot be used to smooth interior vertices. To avoid this situation, we monitor the mesh validity (by checking the element volumes) during the vertex projection step. If the displacement of a vertex after projection makes the mesh invalid, then we apply only an increment of displacements. In most cases, the total displacement of a vertex located on the blade surface had to be applied in 4 increments to avoid mesh invalidity. A sample mesh-motion convergence plot is shown in Figure 7.13. Note that, the y-axis in Figure 7.13 is the L2-norm of the maximum displacement of an interior vertex in the mesh-motion process. After a successful mesh-motion procedure, the CFD-CSD

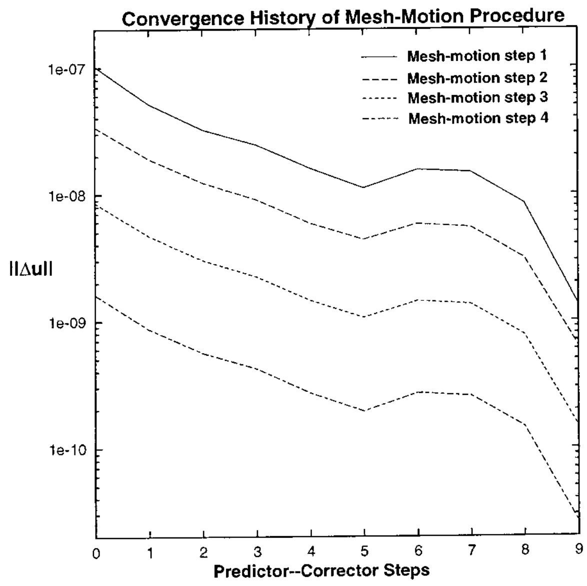

Figure 7.13: A convergence plot of the mesh-motion algorithm.

cycle goes back to the CFD analysis where the finite element analysis is initialized for the deformed blade to calculate another steady-state CFD solution. This CFD-CSD coupling cycle is repeated until equilibrium deflections of the blade are obtained. For the model we performed 4 CFD-CSD cycles and monitored both the deformations and the rotating frequencies of the blade. Figure 7.14 shows the maximum flapwise deformation of the blade versus CFD-CSD cycles.

Table 7.2 gives the first flap, lead-lag, and torsional frequencies computed for

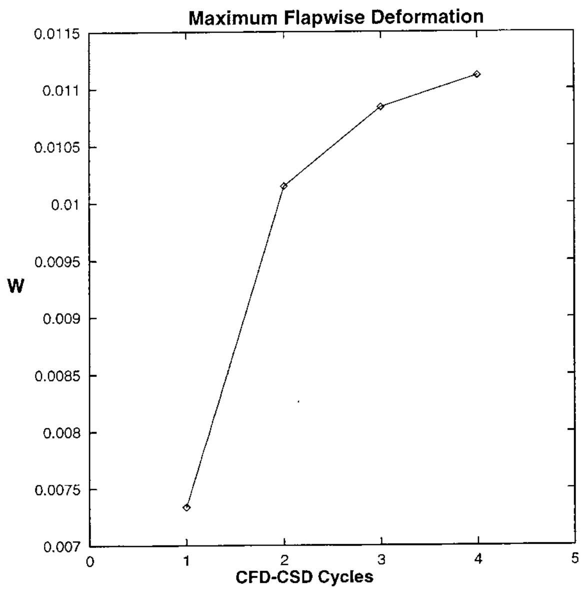

Figure 7.14: Maximum flapwise deformation of the model blade versus CFD-CSD cycle.

the equilibrium deflections of the model blade. The only available experimental data from ref. [73] for this model blade is the lead-lag frequencies. At 4^0 collective, the reported experimental value of lead-lag frequency is 1.516/rev. There is a 3.7% difference between computed and measured lead-lag frequency for this case. Table 7.2 also compares current calculated values of frequencies with the calculations of reference [76]. There is about 2.9% difference between the flap frequency of [76] and the current calculations. Otherwise, both lead-lag and torsional frequencies are in close agreement.

Table 7.2: Fundamental rotating frequencies for the model blade, $\beta_{pc} = 0$ and $\theta = 4^0$

	Flap(1/rev)	Lead-Lag (1/rev)	Torsion (1/rev)
Ref [76]	1.110	1.460	2.950
Current	1.078	1.461	2.967
Experiment [73]	-	1.516	-

Radial distributions of flapwise, lead-lad and torsional deformations are shown in Figures 7.15, 7.16 and 7.17, respectively. It can be seen from these figures that, with the increasing CFD-CSD cycles, the deformations of the blade converge to an equilibrium condition.

Although the aim of this section was to demonstrate the capability and applicability of CFD-CSD coupling procedure, we also have to address the accuracy of the procedure. Table 7.3 compares tip deformations obtained in the current study with the results of reference [74] and [75]. It is believed that the most inaccurate

Table 7.3: Comparison of current tip deformation results with [74] [75]

	Flapwise(w/R)	Lead-Lag (u/R)	Torsion (deg.)
Reference [74]	0.015	0.002	0.5
Reference [75]	0.012	0.0002	0.4
Current	0.011	0.0005	0.1

results are in the torsional deformations due to inaccuracies in pitching moment calculations. The main reasons for the inaccuracies in pitching moment calculations are:

- Insufficient mesh resolution to capture leading-edge and trailing-edge suction peaks
- Changes in location of elastic axis

The major contributions to the pitching moment of an airfoil come from the leading-edge and trailing-edge pressure loadings. Since flow tends to make a stagnation at the leading and trailing-edges, the pressure exhibits a suction peek at these points. The trailing-edge and leading-edge are also the farthest two points from the elastic axis, thus, they have the longest moment arm to the elastic axis. Therefore, an unresolved suction peak at the trailing-edge translates to a lower pitching moment of the airfoil section. For this reason it is very important to adjust the mesh density near the trailing and leading-edges of the rotor blade to capture correct pressure distributions. Adaptive mesh refinement technique described in chapter 4 is one possible solution for this problem.

The second reason for the inaccuracies in pitching moment calculations is due to changes in the location of elastic axis. As the blade deforms, the elastic axis changes its initial location. Since pitching moment is calculated with respect to elastic axis, this changes the computed values of pitching moment also. In our calculations, we assumed a fixed location of the elastic axis.

In summary, we showed a practical application of the CFD-CSD coupling procedure. The calculated flap deflections of the model blade and the lead-lag frequency compares well with the similar numerical calculations and experiment, respectively. However, torsional deformations of the model blade is under-predicted.

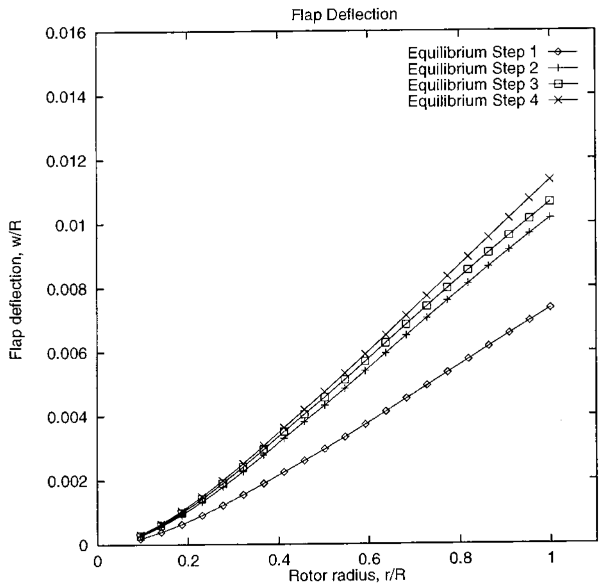

Figure 7.15: Radial distribution of flapwise deformation.

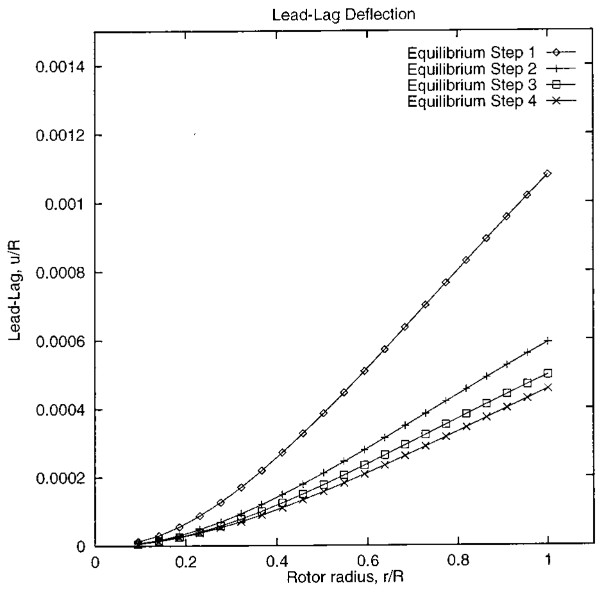

Figure 7.16: Radial distribution of lead-lag deformation.

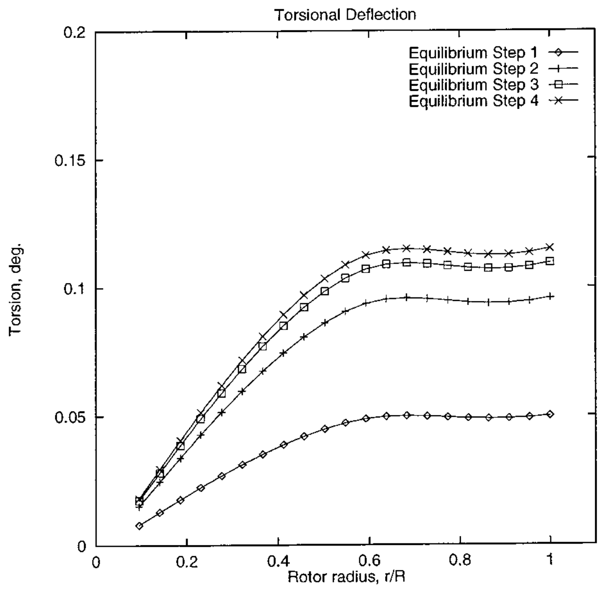

Figure 7.17: Radial distribution of torsional deformation.

CHAPTER 8
CONCLUDING REMARKS AND FUTURE WORK

8.1 Adaptive Hover Calculations

An adaptive refinement procedure has been developed for computing vortical flows encountered in rotor aerodynamics. An error indicator based on interpolation error estimate is formulated and coded into the adaptive finite element framework. It has been shown that the error indicator based on interpolation error estimate is effective in resolving the global features of the flow-field. Along with the first error indicator it has been found that a second error indicator aids in the efficient resolution of small scale features of the flow such as vortex tubes. For this purpose a topology based vortex core detection technique has been used to capture vortex tubes for the rotor blade in hover conditions. It has been shown that the combination of the two error indicators used in refining the flow shows promise for computing rotor-blade flows effectively and efficiently with an acceptable level of user interference during the adaptation procedure.

For the Caradonna-Tung blade, the tip vortex is resolved for 180^o within four levels of adaptive refinement. Overall quality of the computed surface pressure distributions with the adaptation is very good. Especially, using the error indicator formulated in this paper, the leading edge suction peaks are captured successfully.

For the UH-60A rotor blade, a zero thrust and a high thrust hover cases are studied. The effect of tip vortex for the zero thrust UH-60A computations is found to be minimal. However for this case, another bound vortex in strong interaction with the blade is identified at 75-80% radius and it is believed that this vortex tube exists due to an abrupt differential change in thrust loading. The inboard shed wake, in this case, should have a discontinuity around 80% radial span. Existence of this inboard vortex tube clearly demonstrated a deviation from empirical wake geometry studies in which the inboard shed wake was assumed continuous between the root and the tip of the blade. Finally, the sectional thrust and torque distributions computed by the adapted mesh are in good agreement with experimental data.

For the high-thrust case, it is found that the tip vortex interaction is strong and has to be computed as accurately as possible to get good agreement between computed and measured performance characteristics of the UH-60A blade. Progressive adaptive refinement steps, using both interpolation error estimate and the vortex core detection technique, resulted in a correct prediction of tip vortex structure. The sectional thrust distribution with the final adapted mesh showed a remarkable improvement with respect to initial mesh where the tip vortex could not be resolved.

8.1.1 Extension to a Navier-Stokes Solver

In this thesis, we assumed that the viscous forces are not going to be very important for the kind of problems we want to solve. In fact, the hover cases for both Caradonna-Tung blade and the UH-60A blade were selected such that, there were no stall or separation reported by the experiments. We also assumed that the formation of the tip vortex will mainly be driven by the up-wash flow at the tip of the blade. Nevertheless, we know that viscous forces play an important role in accurate drag calculations. Furthermore, for forward flight calculations viscosity has to be taken into account because of the separated and stalled flow region on the retreating blade.

Addition of viscous forces to the current adaptive finite element procedure is not a difficult task. In fact, the finite element code used in this thesis has already have the infrastructure to extend into a Navier-Stokes solver. However, there are two issues that need addressed before upgrading the current inviscid solver to a Navier-Stokes solver. They are:

- Mesh generation for high Reynolds number flows
- Turbulence

When viscous flows are considered, we also have to consider the length scales that need to be resolved during the calculations. Most problems in rotorcraft aerodynamics fall into turbulent flow category, where the chord-based Reynolds number, Re_c is $1-10\times10^6$. In general, to resolve viscous turbulent layers in flow calculations, the mesh resolution normal to solid surfaces should be less than $1/\sqrt{Re_c}$ [77]. For example, the UH-60A blade which we analyzed in chapter 6, is reported to have a Reynolds number between $1 - 1.4 \times 10^6$ [56], based on airfoil chord. The minimum mesh size that we adaptively generated during the inviscid solution of the UH-60A blade was around $0.001R$ or $0.015\tilde{c}$. Where R and $\tilde{c}$ are the rotor radius and rotor nominal chord length, respectively ($R/\tilde{c} \approx 15$). Whereas the length scale (normal to solid surfaces) that needs to be resolved for the UH-60A viscous-turbulent calculations should be $0.0008\tilde{c}$ or $0.00005R$.

An efficient mesh generation procedure that facilitates generation of unstructured grids suitable for high Reynolds number flows has to be used. The length scales in chord-wise and span-wise directions are at least 2-3 orders of magnitude larger than the length scale in surface normal direction. Therefore, for efficiency purposes, the elements can be stretched-out in directions other than the surface normal direction. The methods described in references [78], [79] and [87], address the issues of mesh generation for viscous flows. In our inviscid calculations we adaptively refined the meshes up to 2-3 million tetrahedral elements. Even with the efficient mesh generation procedures for viscous flows, one should expect that the mesh size for viscous calculations would be 2-4 times of inviscid calculations. The most important issues that need to be satisfied for viscous calculations with unstructured grids are mesh quality and available computer resources.

The problem of turbulence has to be addressed and studied for viscous calculations. If Reynolds-Averaged Navier-Stokes (RANS) equations are formulated, then this requires addition of an appropriate turbulence model. Turbulence modeling for aerodynamic flows is a very rich area of research. A turbulence model suitable for unstructured meshes has to be implemented within the GLS finite element formulation. Possible candidates for such turbulence models are one-equation models of reference [80] and [81].

8.1.2 Forward-Flight Calculations

A hovering rotor blade problem can be solved as a steady-state problem in a rotating reference frame. The flow conditions on the blade surface are the same at every azimuth angle. Because of these simplifications, we are able to solve only one blade of a rotor system in steady-state hover conditions.

However, the flow conditions of a rotor blade in forward flight are different from a hovering rotor blade. First of all, in forward flight the rotor system not only rotates around its hub but also translates in the flight direction. Because of this, the flow conditions on the blade are different at every azimuth angle. As a result, the aerodynamic loads and moments of the rotor system are more complex. To trim a rotor system in forward flight additional degrees of freedom have to be added to its rigid body motion. The fundamental rigid body motions of a rotor system in forward flight are *flapping* and *feathering* (pitching). The flapping motion of a rotor blade is provided by the flapping hinge located at hub (see Figure 8.1). The feathering or pitching rigid body motion of the blade is controlled by a *swash plate* as shown in Figure 8.1. Aeroelastic deformations of the blade are not negligible and

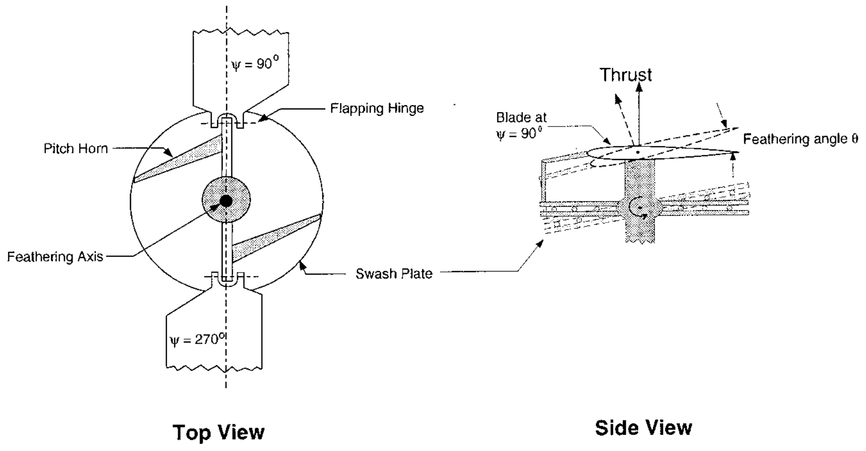

Figure 8.1: A simplified mechanism of rotor blade hub for controlling feathering and flapping

should be accounted for. Aerodynamic loads, aeroelastic deformations and rigid body motions of a rotor system are inter-connected to one another. Therefore, a trimmed forward flight calculation should consider all the ingredients indicated.

This overwhelmingly complex scenario of forward flight may be simplified with certain assumptions. For example, both flapping and feathering rigid body motions can be represented by an infinite Fourier series [3]:

$$\begin{aligned}\beta =&\beta_0 - a_{1s}\cos\psi - b_{1s}\sin\psi - a_{2s}\cos\psi - b_{2s}\sin\psi \cdots - a_{ns}\cos n\psi - b_{ns}\sin n\psi \\ \theta =&\theta_0 + \frac{r}{R}\theta_t - A_{1s}\cos\psi - B_{1s}\sin\psi - A_{2s}\cos\psi - B_{2s}\sin\psi \cdots \\ &- A_{ns}\cos n\psi - B_{ns}\sin n\psi\end{aligned} \tag{8.1}$$

where, ψ is the azimuth angle, β_0 is the pre-cone angle, θ_0 is the average pitch angle, θ_t is the structural twist of the blade and $a_{ns}, b_{ns}, A_{ns}, B_{ns}$, $n = 1, 2, \cdots$, are the coefficients of the Fourier series, respectively. In practice, only the first harmonics of the flapping and feathering motion are included in calculations. The second and higher harmonics represented by the remaining terms of Eq. 8.1 are relatively small and have very little effect on rotor thrust and torque [3]. Therefore, we can write the flapping and feathering equations of a rotor blade in the following form:

$$\begin{aligned}\beta &= \beta_0 - a_{1s}\cos\psi - b_{1s}\sin\psi, \\ \theta &= \theta_0 + \frac{r}{R}\theta_t - A_{1s}\cos\psi - B_{1s}\sin\psi.\end{aligned} \tag{8.2}$$

where the coefficients a_{1s}, b_{1s}, A_{1s} and B_{1s} either have to be calculated in an iterative manner inside the trim loop of forward flight calculations or they can be taken from a flight test data, if available.

A second simplification can be made by assuming the aeroelastic deformations of the rotor blade in hover are smaller than the rigid body motions. This assumption has to be weighted carefully and considered only if the rotor blades are structurally stiff and the flow conditions are at low advance ratios. Nevertheless, with this assumption the CFD-CSD coupling part of the forward-flight calculations can taken out of the trim loop and this is a substantial saving in terms of complexity of the computational procedure.

In summary, one has to be concerned with the following kinematic motions of a rotor system in forward flight:

- Rotational rigid body motion,

- Translational rigid body motion,
- Flapping,
- Feathering.

Aerodynamic calculations of rotor blades in forward flight can be modeled several ways. One possibility is to use a so-called *overset* grid approach [84]. Mainly used by finite-difference type calculations, an overset grid approach is based on decomposing the entire physical domain into several pieces in order to reduce the complexity of the mesh generation process for arbitrary geometric configurations. For example, an overset grid approach models a wing and a fuselage as two separate domains. A computational mesh is created for each domain (i.e., wing and the body) that has an overlap with the neighboring domains. Then the meshes for wing and and the fuselage are placed in a Cartesian mesh that encloses both bodies. After this, the boundary-connectivity of each mesh is determined by using a point-in-a-cell search algorithm [83]. During the flow solution process, mesh connectivity between the domains are used to provide the continuity of flow variables between the neighboring domains. For unsteady flow calculations with rigid body motion, the boundary-connectivity between the domains have to be established at each time step. Successful applications of the overset grid approach to forward flight calculations of rotor blades can be found in references [18] and [82].

Although the overset grid approach simplifies the modeling of a complex geometry and provides the capability to do rigid body motion, it also has certain drawbacks. The first drawback is regarding the overset mesh generation process. When more than one domain is meshed for overset grid calculations, the user has to be very careful to generate meshes for each domain such that there will be an overlap between two the neighboring domains. This requires experienced users to generate meshes and the overall modeling process is time consuming. The second drawback of overset grid approach is about the boundary connectivity of individual domains. As mentioned earlier, the overlap between the two neighboring domains are needed to provide the continuity of the flow variables. However, with this approach the flow quantities (density, velocity and energy) between the two neighboring domains are

C^0 continuous. Therefore, the derivatives of flow variables between the two neighboring domains are discontinuous. For example, in an overset grid approach, the flow velocity is continuous across an overlap boundary, but the vorticity is discontinuous. For rotorcraft aerodynamics, where the accuracy of solution depends on the accurate wake and vortex flow computations, discontinuous vorticity between the domains is a potential reason for inaccurate results.

Another approach to model forward flight of rotor blades is the Arbitrary Lagrangian Eulerian (ALE) formulations. In an ALE approach, once again the domain decomposition idea is utilized to model different pieces of a complex problem. Domains that contain objects with rigid body motion (or structural deformation) are modeled as Lagrangian domains. The Eulerian domain is used to solve the conservation laws for a fluid. Lagrangian and Eulerian domains are then connected to each other via an interface surface to transfer information between the two. For example, using the ALE approach, the forward flight problem can be modeled as follows: A computational domain that encloses the rotor blade is generated and designated as the Eulerian domain. This is the domain where conservation laws for air will be solved. Inside the this Eulerian domain, the rotor is modeled as a Lagrangian object that rotates around its hub. The Eulerian domain and the Lagrangian domain have their own computational mesh, but they share a common interface between each other. While the Eulerian domain is used to calculate the flow-field and pressure loads around the rotor blade (Lagrangian body), the structural deformations of the rotor blade are calculated in the Lagrangian domain. Loads and deformations are exchanged between the two domains during the coupling process. Although the ALE approach has applications in problems like store-separation [59], the author is not aware of any ALE applications for forward flight computations.

It is this author's belief that the forward flight problem of rotorcraft can be solved efficiently and accurately by utilizing the domain decomposition idea of overset grid methods for unstructured grids and the adaptive finite element procedure described in chapters 3 and 4. The main idea here is to decompose the computational domain into stationary and rotating components as shown in Figure 8.2. Here we have two computational domains: The first domain encloses the rotor sys-

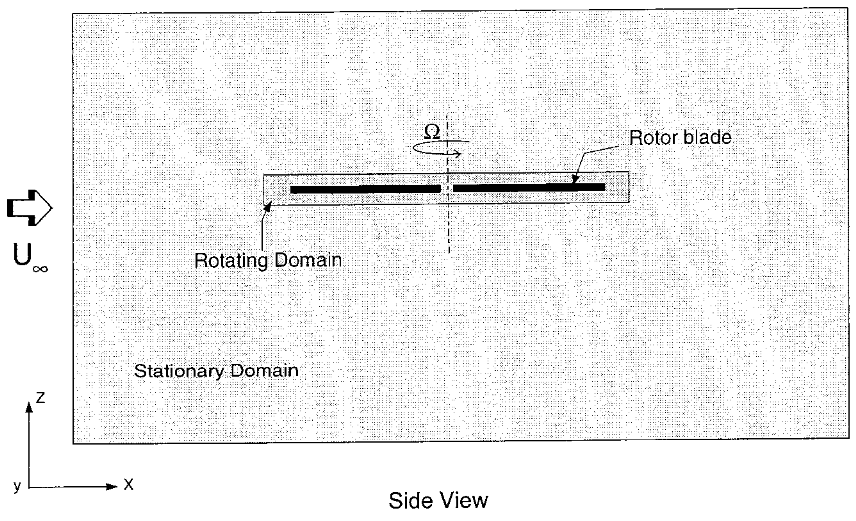

Figure 8.2: A suggested domain decomposition for forward flight calculations

tem and rotates with it, and the second one is a stationary domain which encloses the rotating domain. The rotor blade inside the rotating domain is allowed to flap, pitch and deform freely. The computational mesh inside the rotating domain can be updated by a mesh-motion procedure (e.g., the one described in chapter 7). Thus, as the blade pitches, flaps and deforms the mesh inside the rotating domain, the mesh is moved with the blade. At the same time, the mesh inside the rotating domain is rotated with the rotational speed Ω. The computational mesh for the stationary domain is kept fixed and we have to do something about the interface between the stationary and rotating domains. Finally, the forward motion of the rotor blade is provided by imposing a wind velocity upstream of the stationary domain.

The interface between the two domains, which are in relative motion with respect to one another, can be handled in two ways: In the first case, we can let the computational meshes for fixed and rotating domains to be independent from each other. In this case, the continuity of flow variables across the interface has to be provided by developing special non-conforming elements. An example of this type of interface boundary condition can be found in [37]. A drawback of such an interface is that the finite element procedure has to be modified to allow the flexibility of non-conforming elements. A second way to deal with this interface problem is to use a local mesh modification process to provide the connectivity of the meshes between

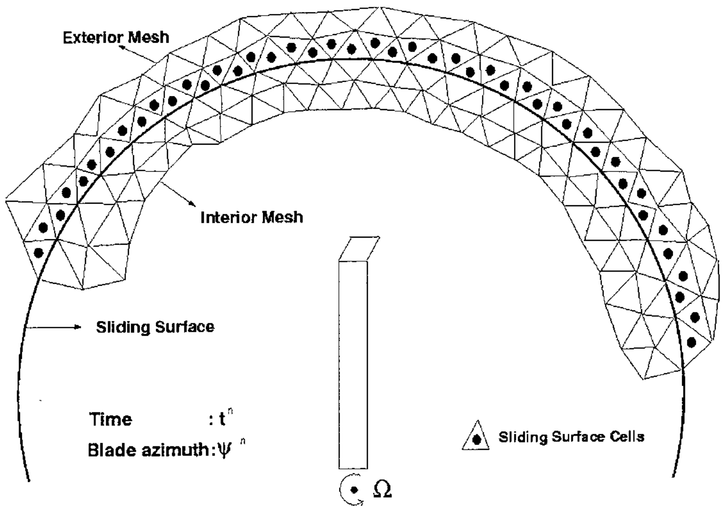

Figure 8.3: A Partial view of mesh at the beginning of the time step n

two domains at all times. What we mean by this is that, the mesh connectivity between the two domains can be updated such that both rotating and stationary domains are tied to each other as if the whole computational domain is covered by a single mesh.

We exemplify the latter choice of interface using the schematics in Figures 8.3-8.6. In Figure 8.3 we are looking at the top view of a rotor blade. The blade is located inside a cylindrical domain and the mesh for this cylindrical domain is rotating with speed Ω. This rotating cylindrical domain is located inside another domain. We are showing only the partial view of the meshes (in 2D) for these two domains to simplify the explanation. At time t^n, the elements adjacent to the sliding interface are tied to each other. We want to advance the flow solution by Δt time step and rotate the blade (and the mesh inside the cylindrical domain) by $\Delta\psi$ azimuth angle. At this point, let us assume we target to modify the elements (marked with black circles) in the stationary domain (see Figure 8.3). As the blade and the mesh inside the cylindrical domain rotates, the connectivity between the meshes for stationary and rotating domains is broken and we end up with a non-conforming mesh at the interface (see Figure 8.4). After this, the elements marked with black circles are removed from the stationary mesh to create a gap as shown in Figure 8.5. Finally, the gap between the two meshes are filled with a mesh generation procedure such that, once again the rotating and the stationary meshes are tied to each other at time step t^{n+1} (see Figure 8.6). This process is repeated for each time step. The space-time Galerkin/least-squares (GLS) finite element procedure explained in chapter 3, is suitable for solving the forward flight calculations of a rotor blade using the "sliding mesh" approach that was just outlined. As explained earlier, the space-time GLS formulation allows the computational mesh to change from one time-slab to the other. Successful applications of space-time GLS finite element formulation to moving boundary problems can be found in [85]. One issue that needs to be addressed at this point is the time-accuracy of the GLS finite element formulation. In steady-state hover calculations, we used constant-in-time formulation for the GLS method. For transient calculations, at least linear-in-time formulation needs to be considered. Because of the moving mesh, the space-time jump term for the variational formulation (see chapter 3) has to be calculated carefully. The jump term imposes the continuity of solution variables between the time slabs. When we consider forward flight calculations, the mesh changes from one time slab to the other. Evaluation of the jump term for transient GLS formulation requires a solution projection procedure between the time-slabs. Reference [85] is a good starting point to understand the issues surrounding the moving meshes and implementation details of the jump condition for space-time GLS method.

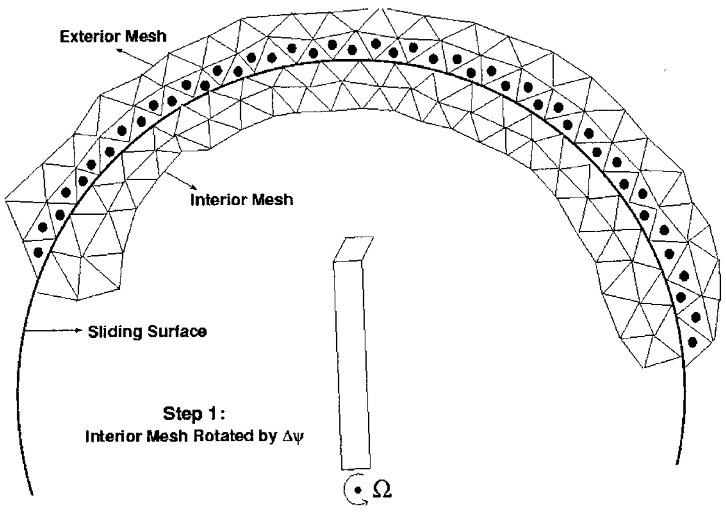

Figure 8.4: Mesh configuration after the blade is rotated by $\Delta\psi$ angle

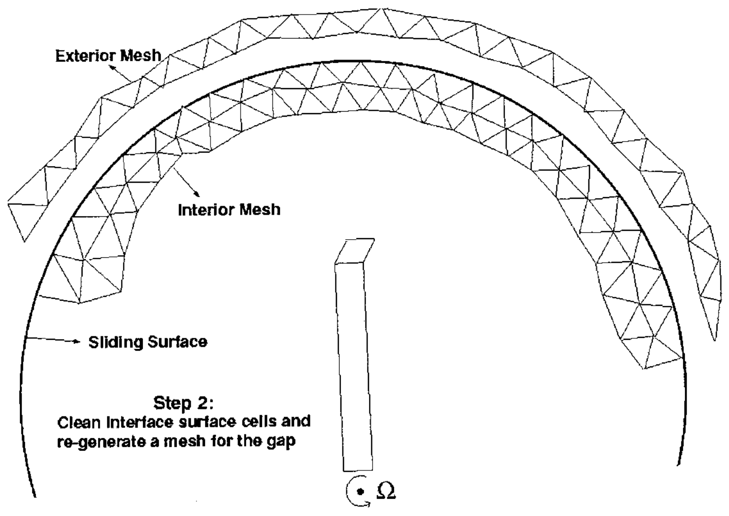
Exterior Mesh
Interior Mesh
Sliding Surface
Step 2:
Clean Interface surface cells and re-generate a mesh for the gap
Ω

Figure 8.5: Local mesh modification to the exterior grid

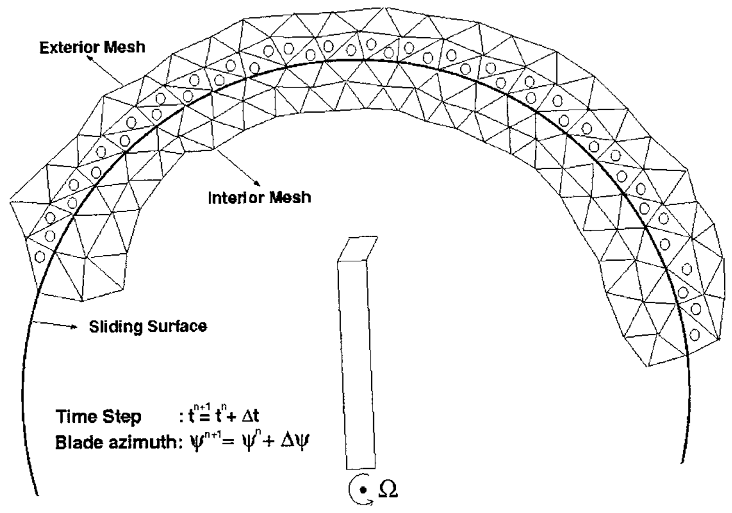
Exterior Mesh
Interior Mesh
Sliding Surface
Time Step : $t^{n+1} = t^n + \Delta t$
Blade azimuth: $\psi^{n+1} = \psi^n + \Delta\psi$
Ω

Figure 8.6: A Partial view of mesh at the beginning of the time step n+1

8.2 CFD-CSD Coupling

In chapter 7, we presented a method to couple aerodynamic loads and structural deformations of a rotor blade in hover. The method uses the adaptive CFD solver explained in chapters 3 and 4 and a beam-element CSD solver. The CFD and the CSD are loosely coupled in this procedure. The necessary interfaces are developed to transfer aerodynamics loads and deformations between the CFD calculation and the CSD calculation. In order to update the CFD mesh for the deforming rotor-blade, a parallel mesh-motion procedure is implemented.

Example CFD-CSD coupling results are presented for a model rotor-blade. This model rotor blade corresponds to a stiff in-plane hingeless rotor with dimensionless lead-lag frequency of approximately 1.5. During the CFD-CSD calculations the lead-lag frequency of the model rotor system was calculated to be 1.461. Also, the flapwise, lead-lag and torsional deformations of the model blade is calculated and compared with the similar numerical computations of others.

8.2.1 Enhancements to CFD-CSD Coupling Procedure

In chapter 7, we noted that the calculated torsional deflections of the model rotor blade is under-predicted in comparison to other numerical calculations. The main reason for under-predicted torsional deflections is believed to be resulting from poor pitching moment calculations. The major contributions to the pitching moment of an airfoil come from the leading-edge and trailing-edge pressure loadings. Since flow tends to make a stagnation at the leading and trailing-edges, the pressure exhibits a suction peak at these points. The trailing-edge and leading-edge are also the farthest two points from the elastic axis, thus, they have the longest moment arm to the elastic axis. Therefore, an unresolved suction peak at the trailing-edge translates to a lower pitching moment of the airfoil section. For this reason it is very important to adjust the mesh density near the trailing and leading-edges of the rotor blade to capture correct pressure distributions. Adaptive mesh refinement technique described in chapter 4 is one possible solution for this problem. Therefore, further calculations are needed in CFD-CSD coupling of rotor blades in hover using finer CFD meshes.

The second issue concerning the CFD-CSD coupling is the transient aeroelastic calculations in forward flight. It is more likely that the current loosely coupled CFD-CSD procedure has to be modified for transient rotor blade calculations. A more coordinated coupling between CFD and CSD may be required. In forward flight calculations, obtaining a trimmed rotor solution is the most important issue. A trimmed rotor system requires that all the moments add up to zero. For a trimmed forward flight calculation both rigid body motion and the aeroelastic deformations of the rotor system have to be account for. Aeroelastic deformations of the blade can be obtained from the same CSD procedure that we utilized in our hover calculations. However, in forward flight mode, both CFD and CSD solvers have to be advanced in time simultaneously. Neither CFD nor CSD should lag in time during the solution process. Since aerodynamic loads are affected by structural deformations and vice versa, simultaneous integration of CFD and CSD solutions have to be done accurately. Reference [86] suggest a CFD-CSD coupling scheme for transient problems. In [86] and iterative scheme is used to attain equilibrium per time step of the transient calculations. This means, within a Δt time step, CFD and CSD solvers may be visited more than once to improve the accuracy of overall calculations. Figure 8.7 suggest a possible CFD-CSD coupling strategy in forward flight.

In Figure 8.7, we start the forward flight calculations by first setting the flight conditions: Rotor speed, Ω, advance ratio, μ and the disk angle of attack, α. These are the flight parameters which are going to be set once and remain unchanged during the calculations. Advance ratio is the ratio of forward speed of a helicopter to its main rotor tip speed [3]. After deciding on the flight conditions, we set the flapping and feathering controls for main rotor blades. Flapping and feathering motion of of the blades can be set by using Eq. 8.2. If available, first flapping and feathering Fourier coefficients can be obtained from experimental data, otherwise they have to be adjusted during the calculations to trim the rotor system. Once the initial settings of the blade flapping and feathering are set, then we start CFD and CSD calculations for a rotor system in forward flight. The calculations start at zero degree azimuth, $\psi = 0$, and each time step the blades are rotated by $\Delta\psi$ degrees.

The CFD$\rightarrow$CSD and CSD$\rightarrow$CFD interfaces operate in the same way which they did in hover calculations. They are used to transfer force-moment and deformation data between the flow solver and the structural solver. At each time step of forward flight calculations, if needed, the CFD and CSD solvers can be visited more than once to increase the accuracy of CFD-CSD coupling. This is done by the "sub-cycle" block shown in the flowchart. At the end of a time step, Δt, the azimuth angle is incremented by $\Delta\psi$ and new aerodynamic loads and structural deformations are calculated. When the rotor makes one complete revolution, we check and see if the total moments, M_x and M_y are equal to zero. This is the trim condition for a rotor system. If the rotor system is trimmed, then the forward flight calculations are stopped, otherwise we go back to the beginning of the flowchart and adjust the flapping and feathering controls for a new forward flight calculation.

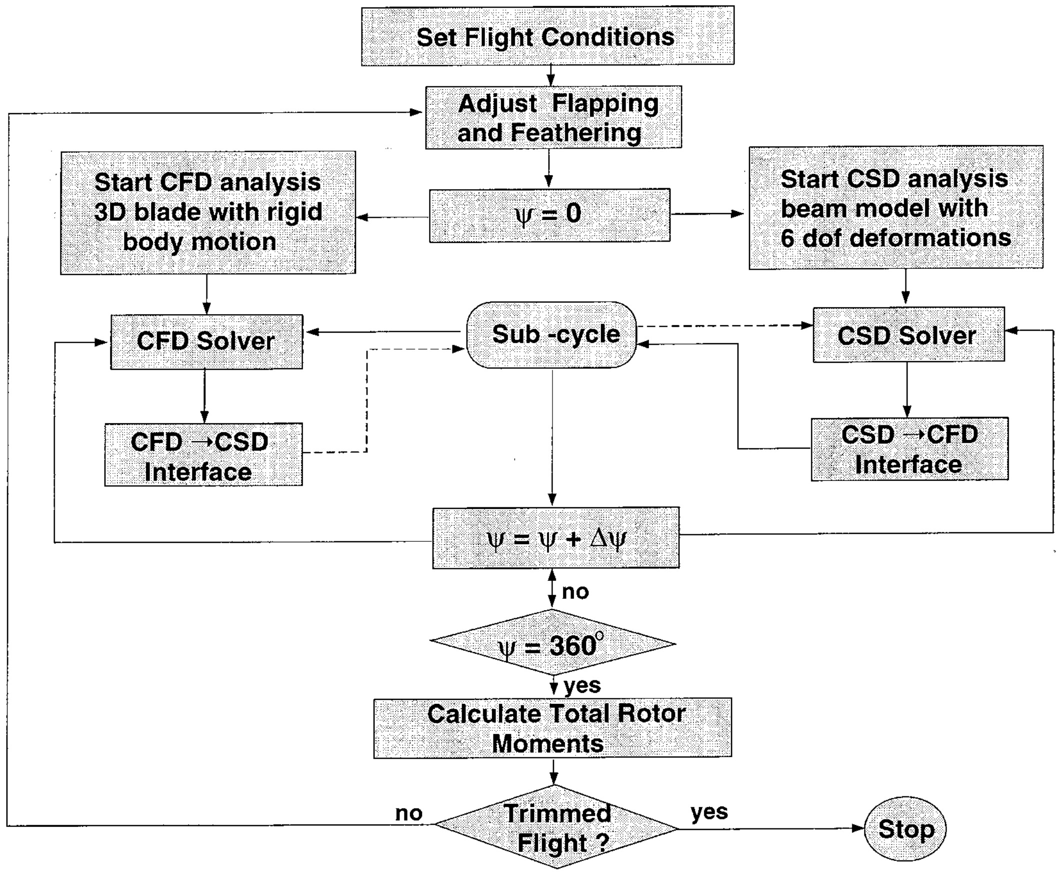
Set Flight Conditions
Adjust Flapping and Feathering
Start CFD analysis 3D blade with rigid body motion
ψ = 0
Start CSD analysis beam model with 6 dof deformations
CFD Solver
Sub -cycle
CSD Solver
CFD →CSD Interface
CSD →CFD Interface
ψ = ψ + Δψ
no
ψ = 360°
yes
Calculate Total Rotor Moments
no
Trimmed Flight ?
yes
Stop

Figure 8.7: CFD-CSD coupling for forward flight calculations

BIBLIOGRAPHY

[1] Caradonna, F.X., "The application of CFD to rotary wing flow problems," AGARD-FDP-VKI Special Course, Aerodynamics of Rotorcraft, Middle East Technical University, Ankara/Turkey, 1990.

[2] Hess J.L., and Smith A.M.O., "Calculation of non-lifting potential flow about arbitrary three-dimensional bodies", Douglas Aircraft Company Report ES40622, March 1962.

[3] Prouty, R.W., "Helicopter performance, stability, and control", Robert E. Krieger Publishing Company, Malabar, FL 1990.

[4] Johnson W., "Helicopter Theory", Dover Publications, NY, 1994.

[5] Maskew B., "Program VSAERO, a computer program for calculating the nonlinear aerodynamic characteristics of arbitrary configurations", Users Manual, NASA CR-166476, Nov. 1982.

[6] Sidwell K., Baruah P., and Bussoletti J., "PANAIR - A computer program predicting subsonic or supersonic linear potential flows about arbitrary configurations using a higher order panel method", Vol. II - Users Manual, NASA CR-3252, May 1980.

[7] Scully M.P., "Computation of helicopter rotor wake geometry and its influence on rotor harmonic loads" MIT report, ASRL TR 178-1, March 1975.

[8] Johnson W, "A comprehensive analytical model of rotorcraft aerodynamics and dynamics. Part 1, Analysis Development", NASA TM-81182, 1980.

[9] Arieli R., and Tauber M.E., "Computation of subsonic and transonic flow about lifting rotor blades", AIAA Paper 79-1667, Aug. 1979.

[10] Strawn R.C., and Caradonna F.X, "Numerical modeling of rotor flows with conservative form of full potential equations", AIAA 24th Aerospace Sciences Meeting, Reno, NV, Jan. 6-9, 1986.

[11] Chang, K.C., Bui, M.N., and Cebeci, T., "The Calculation of Turbulent Wakes", *AIAA Journal*, Vol. 24, pp. 200-201, 1986.

[12] Kroll N., "Computation of the flow fields of propellers and hovering rotors using Euler Equations", 12th European Rotorcraft Forum, Garmisch-Partenkirchen, Germany, Sept. 22-25, 1986.

[13] Chen C.L., and McCroskey W.J., "Numerical simulation of helicopter multi-bladed rotor flow", AIAA Paper 88-0046, 26th Aerospace Sciences Meeting, Reno, NV, Jan. 11-14, 1988.

[14] Steinhoff J.S., and Ramachandran K., "A vortex embedding method for free wake analysis of helicopter rotor blades in hover", 13th European Rotorcraft Forum, Arles, France, 1987.

[15] Ramachandran K., Owen S.J, Caradonna F.X., and Moffit R.C., "Hover performance prediction using CFD", American Helicopter Society 50th Annual Forum, Washington DC, May 11-13, 1994.

[16] Srinivasan, G.R., Baeder, J.D., Obayashi, S., and McCroskey, W.J., "Flowfield of a Lifting rotor in Hover: A Navier-Stokes Simulation," *AIAA Journal*, Vol. 30, No. 10, pp. 2371-2378, Oct. 1992.

[17] Wake B.E., Beader J.D., "Evaluation of Navier-Stokes analysis method for hover performance prediction", American Helicopter Society Aeromechanics Specialist Conference, San Francisco, CA, Jan 19-21, 1994.

[18] Ahmed J., and Duque, E.P.N., "Helicopter Rotor blade Computations in Unsteady Flows Using Moving Embedded Grids," AIAA Paper 94-1922, Jul, 1994.

[19] Zucrow M., and Hoffman J.D., "Gas dynamics", Krieger Pub. Co., Malabar, Fla., 1985.

[20] Incropera F.P., and DeWitt D.P., "Fundamentals of heat and mass transfer", 3rd edition, Wiley, New York, 1990.

[21] Kundu P.K., "Fluid Mechanics", Academic Press, San Diego, 1990.

[22] Wylen G. J., and Sonntag R. E. "Fundamentals of classical thermodynamics" 2nd edition, Wiley, New York, 1973.

[23] S.K. Godunov, "The problem of a generalized solution in the theory of of quasi-linear equations and in gas dynamics", Russ. Math. Surveys, Vol. 17, pp. 145-156 (1962)

[24] A. Harten, "On the symmetric form of system of conservation laws with entropy", *Journal of Computational Physics*, Vol 49, pp. 151-164 (1983)

[25] M. Mock, "Systems of conservation laws of mixed type", *Journal of Differential Equations*, Vol 37, pp. 70-88 (1980)

[26] Warming, R.F., and Hyett, B.J., "The modified equation approach to the stability and accuracy analysis of finite-difference methods", *J. Comput. Physics*, Vol. 14, pp. 159-179, 1974.

[27] Chalot, F., Hugues, T.J.R., and Shakib F., "Symmetrization of conservation laws with entropy for high-temperature hypersonic flows", Computing Systems in Engineering, Vol. 1, pp. 495-521 (1990)

[28] Shakib F., "Finite element analysis of the compressible Euler and Navier-Stokes equations" Ph.D. Thesis, Division of Applied Mechanics, Stanford University, CA (1989)

[29] Hoffmann, K. A., "Computational fluid dynamics for engineers", Engineering Education System, Austin, Tex., 1989.

[30] Srinivasan, G.R., Raghavan, V., Duque, E.P.N., and McCroskey, W.J.,"Flowfield Analysis of Modern Helicopter Rotors in Hover by Navier-Stokes Method,", *AHS Journal*, Vol. 38, No. 3, pp. 3-13, 1993.

[31] Strawn, R.C., and Barth, T.J., "A Finite-Volume Euler Solver For Computing Rotary-Wing Aerodynamics on Unstructured Meshes," AHS 48th Annual Forum Proceedings, pp. 419-428, Jun. 1992.

[32] Duque, E.P.N., Strawn R.C., and Biswas, R, "A Solution Adaptive Structured/Unstructured Overset Grid Flow Solver with Applications to Helicopter Rotor Flows," AIAA Paper, 13th AIAA Applied Aerodynamics Conference, San Diego, CA, 1995.

[33] Hugues, T.J.R., Franca, L.P. and Hulbert, G.M., "A New Finite Element Formulation for Fluid Dynamics: VIII. The Galerkin / Least–Squares Method for Advective–Diffusive Equations," *Journal of Applied Mechanics*, Vol. 73, pp. 173-189, 1989.

[34] Hugues, T.J.R., Franca, L.P. and Mallet, M.., "A New Finite Element Formulation for Fluid Dynamics: VI. Convergence analysis of the generalized SUPG formulation for linear time-dependent multidimensional advective-diffusive systems", *Comput. Methods Appl. Mech. Eng.*, Vol. 63, pp. 97-112, 1987.

[35] Shakib, F., Hugues, T.J.R. and Johan, Z., "A Multi-Element Group Preconditioned GMRES Algorithm for Non-symmetric Systems Arising in Finite Element Analysis," *Computer Methods in Applied Mechanics and Engineering'*, Vol 75, pp. 415-456, 1989.

[36] Oden, J.T., and Reddy, J.N., "An Introduction to the Mathematical Theory of Finite Elements", John Wiley & Sons, Inc., 1976.

[37] Oden, J.T., Strouboulis, T., and Devloo, P, "Adaptive Finite Element Methods for the Analysis of Inviscid Compressible Flow: Part I. Fast Refinement/Unrefinement and Moving Mesh Methods for Unstructured Meshes," Comp. Meth. in App. Mech. and Eng., Vol. 59, pp. 327-362, 1986.

[38] Zienkiewicz, O.C., and Zhu, J.Z., "A simple Error Estimator and Adaptive Procedure for Practical Engineering Analysis," *Int. J. for Num. Meth. in Eng*, Vol. 24, pp. 337-357, 1987.

[39] Goodsell, G., "Pointwise Super-convergence of Gradient for the linear Tetrahedral Element," Num. Meth. for Partial Differential Equations, Vol 10, pp. 651-666, 1994.

[40] Ervin, V., Layton, W., and Maubach J., "A Posteriori Error Estimates for Two-Level Finite Element Method for the Navier-Stokes Equations," Num. Meth. for Partial Differential Equations, Vol 12, pp. 333-346, 1996.

[41] Barth, T.J., and Frederickson, P.O., "Higher Order Solution of the Euler Equations on Unstructured Grids Using Quadratic Reconstruction," AIAA-90-0013, Jan. 8-11, 1990.

[42] Vanyo, J.P., "Rotating Fluids in Engineering and Science,",Butterworth-Hainemann, 1993.

[43] Chong, M.S., Perry, A.E., Cantwell, B.J., "A General Classification of Three-Dimensional Flow Fields," Phys. of Fluids A 2:5, 765, 1990.

[44] Kenwright, D., and Haimes, R., "Vortex Identification - Applications in Aerodynamics," to appear in IEEE/ACM Proc. Visualization '97, Phoenix AZ, October 1997, ACM Press.

[45] Abraham, R.H., Shaw, C.D., "Dynamics - The Geometry of Behavior, Part 2: Chaotic Behavior," Volume 2, Aerial Press, pp. 27-30, 1985.

[46] Dindar, M., Lemnios, A.Z., Shephard, M.S., Flaherty, J.E., and Jansen, K. "An adaptive solution procedure for rotorcraft aerodynamics", AIAA Paper 98-2417, 16th Applied Aerodynamics Conference, Albuquerque, NM, 1998.

[47] Taylor, G.I., and Maccoll, J.W., "The air pressure one a cone moving at high speeds", Proc. Roy. Soc. (London), A139, pp. 278-297, 1963.

[48] Brown P.N., Byrne G.D., and Hindmarsh A.C. "VODE: A Variable Coefficient ODE Solver," SIAM J. Sci. Stat. Comput., Vol.10, 1989.

[49] Webster B.E., Shephard M.S., Rusak Z., and Flaherty J.E. "An automated adaptive time discontinuous finite element method for unsteady compressible airfoil aerodynamics", AIAA Journal, 1992.

[50] Caradonna, F.X., and Tung, C., "Experimental and Analytical Studies of a Model Helicopter Rotor in Hover," NASA TM 81232, 1980.

[51] Georges, M., and Shephard, M.S.,"Automatic Three-Dimensional Mesh Generation by the Finite Octree Technique,", *IJNME*, Vol. 32, No. 4, pp. 709-749, 1991.

[52] Arney, D.C., and Flaherty, J.E., "An adaptive method with mesh moving and local mesh refinement for time-dependent partial differential equations", Department of Computer Science, Rensselaer Polytechnic Institute, 1986.

[53] Devine, K.D., Flaherty, J.E, Loy, R.M., and Wheat S.R., "Parallel Partitioning Strategies for the Adaptive Solution of Conservation Laws", Modeling, Mesh Generation and Adaptive Numerical Methods for Partial Differential Equations, Editors: Babuska, I., Flaherty, J.E., Springer-Verlag, Vol. 75, pp. 215-242, 1995

[54] J. E. Flaherty, R. M. Loy, P. C. Scully, M. S. Shephard, B. K. Szymanski, J. D. Teresco, and L. H. Ziantz, "Load Balancing and Communication Optimization for Parallel Adaptive Finite Element Methods," to appear XVII International Conference of the Chilean Computer Science Society, 1997.

[55] Flaherty, J.E., "Finite Element Analysis", Lecture Notes, Department of Computer Science and Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, 1998.

[56] Lorber, P.F., Stauter, R.C., Pollack, M.J., and Landgrebe, A.J.,"A Comprehensive Hover Test of Airloads and Airflow of an Extensively Instrumented Model Helicopter Rotor", Vol I-V, USAAVSCOM TR 91-D-16E, 1991.

[57] Ahmed, A. H., Personal Communication, The Boeing Company, Mesa, AZ, 1998.

[58] Landgrebe, A.J., "The Wake Geometry of a Hovering Helicopter Rotor and Its Influence on Rotor Performance," 28th Annual National Forum of the AHS, pp. 3-15, May 1972.

[59] Lohner, R. "Mesh Adaptation in Fluid Mechanics", Engineering Fracture Mechanics, Vol.50, pp. 819-847, 1995.

[60] Babuska, I., and Suri, M. "The Optimal convergence Rate of the p-Version of the Finite element Method", *SIAM J. Numerical Analysis*, Vol. 24, pp. 750-776, 1987.

[61] Dey, S., Shephard, M.S. and Flaherty, J.E., "Geometry-Based Issues Associated with p-Version Finite Element Computations", Comput. Meths. Appl. Mech. Engrg, Vol 150, pp. 39-50, 1997.

[62] Aiffa, M., "Adaptive hp-Refinement Methods for Singularly-Perturbed Elliptic and Parabolic Systems," Ph.D. Dissertation, Dept. Math. Sci., Rensselaer Polytechnic Institute, Troy, 1997.

[63] Bauchau, O.A., and Hong, C., "Finite element approach to rotor blade modeling", *J. of the American Helicopter Society*, Vol. 32, pp.60-67, 1987.

[64] Jones, R.E.,"QMESH: A Self-Organizing Mesh Generation Program," SLA-73-1088, Sandia Laboratories, 1979.

[65] Schroeder, W.J., "Geometric Triangulations: With Application to fully automatic 3D Mesh Generation", Ph.D. Thesis, Rensselaer Design Research Center, Rensselaer Polytechnic Institute, May, 1990.

[66] George, P.L., and Borouchaki, H., "Delaunay Triangulation and Meshing - Application to Finite Elements" Hermes, Paris, 1998.

[67] Parasolid "Programmers Reference Manual", Shape Data, 1997.

[68] Modenov, P.S., and Parkhonenko, A.S., "Geometric Transformations" Academic Press, 1966.

[69] Batina, J. T., "Unsteady Euler Airfoil Solutions Using Unstructured Dynamic Meshes,", AIAA Journal, Vol. 28, No. 8, 1990.

[70] Saad, Y., and Schultz, M., SIAM Journal on Sci. and Stat. Comp., Vol. 7, pp. 856-869.

[71] Berger, M.J., and Bokhari, S.H., "A partitioning Strategy for nonuniform Problems on Multiprocessors," IEEE Trans. Comp, C-36(5), pp. 570-580, 1987.

[72] Ozturan C., "Distributed Environment and Load Balancing for Adaptive Unstructured Meshes", PhD Thesis, Rensselaer Polytechnic Institute, 1995.

[73] Sharpe,D.L.,"A Comparison of Theory and Experiment for Aeroelastic Stability of a Hinegless Rotor Model in Hover," Integrated Technology Rotor Assessment Workshop, NASA CP-10007,Jun 1983.

[74] Kwon, O.J., Hodges, D.H., and Sankar, L.N., "Stability of hingeless rotors in hover using three-dimensional unsteady aerodynamics", 45th Annual Forum of the American Helicopter Society, Boston, MA, 1989.

[75] Smith, M.J., "Computational considerations of an Euler/Navier-Stokes aeroelastic method for hovering rotor", AIAA Paper 95-0189, 33rd Aerospace Sciences Meeting and Exhibit, Reno, NV, 1995.

[76] Lim, W.J., and Tan, C.M., "Correlation of 2GCHAS with experimental data", Presented at the ARO-AHS-RPI Fifth International Workshop on Dynamics and Aeroelastic Stability Modeling of Rotorcraft Systems, Troy, NY, Oct. 18-23, 1993.

[77] Cebeci, T., and Bradshaw, P., "Momentum Transfer in Boundary Layers", Hemisphere Publishing Corp., Washington, 1977.

[78] Barth T.J., "Numerical Aspects of Computing Viscous High Reynolds Number Flows on Unstructured Grids", AIAA Paper 91-0721, 29th Aerospace Sciences Meeting, Reno NV, 1991.

[79] Adaptive Mesh Generation for Viscous Flows Using Delaunay Triangulation", ICASE Report No. 88-47, 1988.

[80] Jansen, K., Zdenek, J., and Hugues, T.J.R., "Implementation of a One-Equation Turbulence Model Within a Stabilized Finite Element Formulation of a Symmetric Advective-Diffusive System", Computer Methods in App. Mech. Eng., Vol. 105, pp. 405-433, 1993.

[81] Baldwin, B.S., and Barth, T.J., "A One-Equation Turbulence Transport Model for High Reynolds Number Wall-Bounded Flows", NASA TM-102847, Aug. 1990.

[82] Meakin, R., "Moving Body Overset Grid Methods for Complete Aircraft Tiltrotor Simulations", AIAA Paper 93-3350, 11th Computational Fluid Dynamics Conference, Jul. 1993.

[83] Duque, E.P.N., and Srinivasan, G.R., "Numerical Simulation of a Hovering Rotor Using Embedded Grids", Proceedings of 48th Annual forum and Technology Display, pp. 429-445, Washington, D.C., June 1992.

[84] Buning, P., "OVERFLOW 1.6ap Users Manual", NASA Ames Research Center, Feb. 1995.

[85] Johnson, A.A., and Tezduyar, T.E., "Mesh Update Strategies in Parallel Finite Element Computations of Flow Problems with Moving Boundaries and Interfaces", AHPCRC report 94-018, University of Minnesota, April 1994.

[86] Farhat, C., and Lin, T.Y., "Transient Aeroelastic Computations Using Multiple Moving Frames of Reference", AIAA Paper 90-3053, AIAA 8th Applied Aerodynamics Conference, 1990.

[87] Garimella, R. V. and Webster, B. E. and Shephard M. S., "Automatic Mesh Generation of Complex Configurations including Viscous Boundary Layers", Proceedings of the 10th International Conference on Finite Elements in Fluids, University of Arizona, 1998.

APPENDIX A
The DYMORE model of the example blade used in CFD-CSD calculations

```
**************************************************
* CENTER OF EXCELLENCE IN ROTARY WING TECHNOLOGY *
*        RENSSELAER POLYTECHNIC INSTITUTE        *
*                                                *
*          - - -     D Y M O R E    - - -        *
*                                                *
*    PRE-ANALYSIS OF THE FINITE ELEMENT MODEL    *
**************************************************

{PRF-1}:  ---- Sharpe's blade -- NASA TP 2546



              **********************************
              *   CONTROL PARAMETERS {PRF-2}   *
              **********************************

NUMBER OF NODES  =   13
NUMBER OF TRIADS =    1
NUMBER OF BODIES =    1


ANALYSIS TYPE:    0
[0] =  STATIC ANALYSIS;
[1] = DYNAMIC ANALYSIS: ENERGY PRESERVING SCHEME;
[2] = DYNAMIC ANALYSIS: ENERGY  DECAYING  SCHEME.

NUMBER OF TIME STEPS                  =        20
NUMBER OF   HARMONIC   TIME FUNCTIONS =         0
NUMBER OF USER DEFINED TIME FUNCTIONS =         1


NUMBER OF PRESCRIBED LOADING CASES =      0
NUMBER OF CROSS SECTIONS           =      1
NUMBER OF  SPRING PROPERTY SETS    =      0
SCALING FACTOR FOR THE CONSTRAINTS =  1.00000E+05


--- STRUCTURAL ELEMENTS:
```

```
NUMBER OF          BEAM ELEMENTS =    4
NUMBER OF ADVANCED BEAM ELEMENTS =    0
NUMBER OF RIGID BODIES           =    0
NUMBER OF  FLEXIBLE JOINTS       =    0


--- CONSTRAINT ELEMENTS:

NUMBER OF PRESCRIBED DISPLACEMENTS =    0
NUMBER OF   LINEAR    CONSTRAINTS  =    0
NUMBER OF REVOLUTE  JOINTS  =    0
NUMBER OF UNIVERSAL JOINTS  =    0
NUMBER OF SPHERICAL JOINTS  =    0
NUMBER OF PRISMATIC JOINTS  =    0
NUMBER OF  SLIDING  JOINTS  =    0
NUMBER OF   LINK  ELEMENTS  =    0



--- AERODYNAMIC COMPONENTS:

NUMBER OF LIFTING ROTOR ELEMENTS       =    1
NUMBER OF AERODYNAMIC REFERENCE LINES =    1
NUMBER OF AIRFOILS TABLES              =    0

NUMBER OF   MOMENTUM THEORY   INFLOW ELEMENTS =    0
NUMBER OF GENERALIZED DYNAMIC INFLOW ELEMENTS =    0

NUMBER OF TRIM ELEMENTS                =    0



ROUND-OFF ERROR FOR THIS MACHINE =   1.0E-15


               **********************
               *   TRIADS {GEO-1}   *
               **********************

 TRIAD REFERENCE                      R O T A T I O N   M A T R I X
NUMBER     TRIAD

   1         0     E1 VECTOR =    1.00000E+00   0.00000E+00   0.00000E+00
                   E2 VECTOR =    0.00000E+00   1.00000E+00   0.00000E+00
                   E3 VECTOR =    0.00000E+00   0.00000E+00   1.00000E+00
```

```
                    **********************************
                    *   NODAL COORDINATES {GEO-2}    *
                    **********************************

  NODE   LOCAL AXIS SYSTEM         N O D A L   C O O R D I N A T E S      DEGREES OF FREEDOM CONSTRAINTS
 NUMBER  AT NODE     TRIAD              X1            X2            X3      ID1  ID2  ID3  ID4  ID5  ID6

    1       0          0        1.65000E-02   0.00000E+00   0.00000E+00      1    1    1    1    1    1
    2       0          1        2.62000E-02   0.00000E+00   0.00000E+00      0    0    0    0    0    0
    3       0          1        3.59000E-02   0.00000E+00   0.00000E+00      0    0    0    0    0    0
    4       0          1        4.56000E-02   0.00000E+00   0.00000E+00      0    0    0    0    0    0
    5       0          1        5.15000E-02   0.00000E+00   0.00000E+00      0    0    0    0    0    0
    6       0          1        5.75000E-02   0.00000E+00   0.00000E+00      0    0    0    0    0    0
    7       0          1        6.34000E-02   0.00000E+00   0.00000E+00      0    0    0    0    0    0
    8       0          1        2.13000E-01   0.00000E+00   0.00000E+00      0    0    0    0    0    0
    9       0          1        3.62700E-01   0.00000E+00   0.00000E+00      0    0    0    0    0    0
   10       0          1        5.12400E-01   0.00000E+00   0.00000E+00      0    0    0    0    0    0
   11       0          1        6.62100E-01   0.00000E+00   0.00000E+00      0    0    0    0    0    0
   12       0          1        8.11800E-01   0.00000E+00   0.00000E+00      0    0    0    0    0    0
   13       0          1        1.00000E+00   0.00000E+00   0.00000E+00      0    0    0    0    0    0


                    *****************************************
                    *   BLADE ELEMENT DEFINITION {BLD-1}    *
                    *****************************************

 ELEMENT  BODY    N  O  D  E  S        T  R  I  A  D  S     CROSS - SECTIONS
    NO     NO    N1   N2   N3   N4     V1   V2   V3   V4     S1   S2   S3   S4

    1      1      1    4    2    3      1    1    1    1      1    1    1    1
    2      1      4    7    5    6      1    1    1    1      1    1    1    1
    3      1      7   10    8    9      1    1    1    1      1    1    1    1
    4      1     10   13   11   12      1    1    1    1      1    1    1    1


                    **************************************
                    *   DEGREES OF FREEDOM ASSIGNMENT    *
                    **************************************

    TOTAL NUMBER OF DEGREES OF FREEDOM =      72
```

```
#####
#####
#####
#####
#####
#####       User: adams.scorec.rpi.edu!mdindar
#####
#####      Title: thesis.150
#####
#####    Printed: Tue 17:01 Jan 05, 1999
#####
##### Job number: p1-391
#####
#####
#####
#####
#####
```

```
  NODE  ELASTIC  DEGREES OF FREEDOM ASSIGNED AT THIS NODE
NUMBER  BODY ID        ID1  ID2  ID3  ID4  ID5  ID6

     1        1          0    0    0    0    0    0
     2        1          1    2    3    4    5    6
     3        1          7    8    9   10   11   12
     4        1         13   14   15   16   17   18
     5        1         19   20   21   22   23   24
     6        1         25   26   27   28   29   30
     7        1         31   32   33   34   35   36
     8        1         37   38   39   40   41   42
     9        1         43   44   45   46   47   48
    10        1         49   50   51   52   53   54
    11        1         55   56   57   58   59   60
    12        1         61   62   63   64   65   66
    13        1         67   68   69   70   71   72


                    ************************************
                    *  TIME STEPPING CONTROL {TIM-1}   *
                    ************************************

     *    SIMULATION  INITIAL  TIME =  0.00000E+00
                           FINAL  TIME =  1.00000E+00
                        TIME STEP SIZE =  1.00000E-03
             MAXIMUM NUMBER OF RETRYS =      1
        TIME STEP SIZE SELECTION FLAG =      0 [YES = 1]


     * TIME STEP SIZE SELECTION CONTROL PARAMETERS:

   DESIRED  NORMALIZED  LOCAL  ERROR    =  1.00000E-05; (ALLOWABLE BOUNDS:  2.00000E-06 < LOCAL ERROR <  5.00000E-05
   TIME STEP SIZE BOUNDS                =  1.00000E-06 < TIME STEP SIZE <  1.00000E-02
   REFERENCE ENERGY LEVEL               =  1.00000E+00
   MAXIMUM NUMBER OF TIME STEP REJECTS =     10


                    ******************************************
                    *  USER DEFINED TIME FUNCTIONS {TIM-3}   *
                    ******************************************

 --- FUNCTION ID NUMBER:     1
```

```
        TIME    FUNCTION         TIME    FUNCTION         TIME    FUNCTION
                   VALUE                    VALUE                    VALUE

    0.000E+00   1.000E+00   1.000E+01   1.000E+00



                   *****************************************
                   *   GRAVITY VECTOR DEFINITION {GRV-1}   *
                   *****************************************

      NORM OF GRAVITY VECTOR    =  1.00000E+00
      ITS COMPONENTS ARE GRAV1 =  0.00000E+00   GRAV2 =  0.00000E+00   GRAV3 = -1.00000E+00



                   ************************************
                   *   RIGID BODY ROTATIONS {RIG-1}   *
                   ************************************

    ELASTIC BODY   ABOUT               ANGULAR VELOCITY VECTOR
       NUMBER     NODE NO         OM1          OM2          OM3

         1          1        0.000E+00   0.000E+00   1.047E+02



                   ********************************
                   *   SECTION NO:    1 {CRS-1}   *
                   ********************************



      --- SECTIONAL COEFFICIENT DIRECT INPUT {CRS-3} ---

 ----- Blade Section at root



      STIFFNESS PROPERTIES WITH RESPECT TO CENTROID:

      AXIAL STIFFNESS                          =  1.20000E+08
      BENDING STIFFNESS ABOUT E2 DIRECTION AXIS =  1.72200E+01
      BENDING STIFFNESS ABOUT E3 DIRECTION AXIS =  3.57800E+02
      CROSS BENDING STIFFNESS W.R.T. E2 & E3    =  0.00000E+00
```

```
STIFFNESS PROPERTIES WITH RESPECT TO SHEAR CENTER:

TORSIONAL STIFFNESS                    =  5.10000E+00
SHEAR STIFFNESS IN E2 DIRECTION        =  1.00000E+08
SHEAR STIFFNESS IN E3 DIRECTION        =  1.00000E+08
CROSS SHEAR STIFFNESS W.R.T. E2 & E3 =  0.00000E+00

MASS PROPERTIES WITH RESPECT TO LOCAL AXIS ORIGIN:

MASS                                    =  3.41000E-01
MASS MOMENT OF INERTIA ABOUT THE E1 AXIS =  1.74400E-04
MASS MOMENT OF INERTIA ABOUT THE E2 AXIS =  1.74400E-04
MASS MOMENT OF INERTIA ABOUT THE E3 AXIS =  1.57000E-04


COORDINATES IN LOCAL AXIS SYSTEM:

CENTER OF MASS :  XM2 =  0.00000E+00; XM3 =  0.00000E+00
SHEAR CENTER   :  XK2 =  0.00000E+00; XK3 =  0.00000E+00
CENTROID       :  XC2 =  0.00000E+00; XC3 =  0.00000E+00


    $$$ MASS MOMENT OF INERTIA ABOUT E2 OR/AND E3 AXIS IS ADJUSTED TO ENSURE CONSISTENT MASS MATRIX. $$$


               **********************************
               *   AIRFOIL PROPERTIES {AIR-1}   *
               **********************************


--- AIR PHYSICAL PROPERTIES ---

AIR DENSITY    =  1.108E+00
SPEED OF SOUND =  3.400E+02

FAR FIELD VELOCITY =  5.34649E+01
ITS VECTOR COMPONENTS ARE VINF1 =  9.94353E-01  VINF2 = -5.31189E-03  VINF3 = -1.05995E-01



--- STANDARD AIRFOIL COEFFICIENTS ---

LIFT CURVE SLOPE                        [dCl/da] =  5.730E+00
PROFILE DRAG COEFFICIENT                    [Cd] =  1.800E-02
CAMBER PITCHING MOMENT COEFFICIENT         [Cm0] =  0.000E+00
```

```
QUARTER-CHORD PITCHING MOMENT CURVE SLOPE [dCm/da] =  0.000E+00


                  ************************************************
                  *   LIFTING ROTOR ELEMENT DEFINITION {ROT-1}   *
                  ************************************************



 * ROTOR NUMBER                                :    1
   ROTOR CENTER POINT IS AT NODE               :    1
   ROTOR PLANE IS NORMAL TO THE 3-AXIS OF TRIAD:    1
   NUMBER OF BLADES                            :    1

   ROTOR            RADIUS =  1.00000E+00
   ROTOR ANGULAR  VELOCITY =  0.00000E+00

   ROTOR THRUST          COEFF. =  5.20000E-03
         ROLLING  MOMENT COEFF. =  0.00000E+00
         PITCHING MOMENT COEFF. =  0.00000E+00

   ROTOR ANGLE OF ATTACK (DEGREES) =  6.08448E+00



                  *****************************************************
                  *   AERODYNAMIC REFERENCE LINES DEFINITION {LFN-1}  *
                  *****************************************************


AERO REFERENCE  LIFTING ROTOR    NUMBER OF         AXIS SYSTEM
   LINE NUMBER         NUMBER   AIRSTATIONS   AT NODE     TRIAD

             1              1            21         0         1



                  ************************************
                  *   AIRSTATION DEFINITION {AST-1}  *
                  ************************************


 * AIRSTATIONS ASSOCIATED WITH AERODYNAMIC REFERENCE LINE No:    1

 LOCAL COORDINATE SYSTEM DEFINED AT NODE No:     0, TRIAD No:    1.
 ROOT CUT-OUT LOCATION XR1 =  9.51000E-02 XR2 =  0.00000E+00 XR3 =  0.00000E+00
    TIP       LOCATION XT1 =  1.00000E+00 XT2 =  0.00000E+00 XT3 =  0.00000E+00
```

```
AIRSTATION AT BEAM   TRIAD   LIFT      AIRFOIL        CHORD  COORDINATES IN REFERENCE LINE AXIS SYTEM
    NUMBER ELEMENT  NUMBER  MODEL COEFFICIENTS       LENGTH           X1           X2           X3

         1       3       1      0            0  8.98600E-02  9.51000E-02  0.00000E+00  0.00000E+00
         2       3       1      0            0  8.98600E-02  1.40345E-01  0.00000E+00  0.00000E+00
         3       3       1      0            0  8.98600E-02  1.85590E-01  0.00000E+00  0.00000E+00
         4       3       1      0            0  8.98600E-02  2.30835E-01  0.00000E+00  0.00000E+00
         5       3       1      0            0  8.98600E-02  2.76080E-01  0.00000E+00  0.00000E+00
         6       3       1      0            0  8.98600E-02  3.21325E-01  0.00000E+00  0.00000E+00
         7       3       1      0            0  8.98600E-02  3.66570E-01  0.00000E+00  0.00000E+00
         8       3       1      0            0  8.98600E-02  4.11815E-01  0.00000E+00  0.00000E+00
         9       3       1      0            0  8.98600E-02  4.57060E-01  0.00000E+00  0.00000E+00
        10       3       1      0            0  8.98600E-02  5.02305E-01  0.00000E+00  0.00000E+00
        11       4       1      0            0  8.98600E-02  5.47550E-01  0.00000E+00  0.00000E+00
        12       4       1      0            0  8.98600E-02  5.92795E-01  0.00000E+00  0.00000E+00
        13       4       1      0            0  8.98600E-02  6.38040E-01  0.00000E+00  0.00000E+00
        14       4       1      0            0  8.98600E-02  6.83285E-01  0.00000E+00  0.00000E+00
        15       4       1      0            0  8.98600E-02  7.28530E-01  0.00000E+00  0.00000E+00
        16       4       1      0            0  8.98600E-02  7.73775E-01  0.00000E+00  0.00000E+00
        17       4       1      0            0  8.98600E-02  8.19020E-01  0.00000E+00  0.00000E+00
        18       4       1      0            0  8.98600E-02  8.64265E-01  0.00000E+00  0.00000E+00
        19       4       1      0            0  8.98600E-02  9.09510E-01  0.00000E+00  0.00000E+00
        20       4       1      0            0  8.98600E-02  9.54755E-01  0.00000E+00  0.00000E+00
        21       4       1      0            0  8.98600E-02  1.00000E+00  0.00000E+00  0.00000E+00
```