UNSTEADY FLOW SIMULATIONS AROUND COMPLEX GEOMETRIES USING STATIONARY OR ROTATING UNSTRUCTURED GRIDS

A Thesis in
Aerospace Engineering

by
Nilay Sezer-Uzol

© 2006 Nilay Sezer-Uzol

Submitted in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

December 2006
The thesis of Nilay Sezer-Uzol has been reviewed and approved* by the following:

Lyle N. Long  
Professor of Aerospace Engineering  
Thesis Adviser  
Chair of Committee

Philip J. Morris  
Boeing/A. D. Welliver Professor of Aerospace Engineering

Edward C. Smith  
Professor of Aerospace Engineering

Joseph F. Horn  
Associate Professor of Aerospace Engineering

Yousry Azmy  
Professor of Mechanical and Nuclear Engineering

George A. Lesieutre  
Professor of Aerospace Engineering  
Head of the Department of Aerospace Engineering

*Signatures are on file in the Graduate School.
In this research, the computational analysis of three-dimensional, unsteady, separated, vortical flows around complex geometries is studied by using stationary or moving unstructured grids. Two main engineering problems are investigated. The first problem is the unsteady simulation of a ship airwake, where helicopter operations become even more challenging, by using stationary unstructured grids. The second problem is the unsteady simulation of wind turbine rotor flow fields by using moving unstructured grids which are rotating with the whole three-dimensional rigid rotor geometry. The three dimensional, unsteady, parallel, unstructured, finite volume flow solver, PUMA2, is used for the computational fluid dynamics (CFD) simulations considered in this research. The code is modified to have a moving grid capability to perform three-dimensional, time-dependent rotor simulations. An instantaneous log-law wall model for Large Eddy Simulations is also implemented in PUMA2 to investigate the very large Reynolds number flow fields of rotating blades. To verify the code modifications, several sample test cases are also considered. In addition, interdisciplinary studies, which are aiming to provide new tools and insights to the aerospace and wind energy scientific communities, are done during this research by focusing on the coupling of ship airwake CFD simulations with the helicopter flight dynamics and control analysis, the coupling of wind turbine rotor CFD simulations with the aeroacoustic analysis, and the analysis of these time-dependent and large-scale CFD simulations with the help of a computational monitoring, steering and visualization tool, POSSE.
# Table of Contents

List of Tables ................................................................. viii

List of Figures ............................................................... ix

Acknowledgments .............................................................. xix

Chapter 1. INTRODUCTION .................................................. 1

1.1 Background on Unsteady Separated Flows ....................... 3

1.1.1 Ship Airwake ...................................................... 4

1.1.1.1 Previous Ship Airwake Studies ......................... 7

1.1.1.2 Previous Work on Blade Sailing ....................... 11

1.1.1.3 Previous Dynamic Interface Studies ................. 12

1.1.2 Wind Turbine Rotor Flow Fields ............................. 13

1.2 Background on Turbulence Modeling .......................... 19

1.2.1 Direct Numerical Simulations ............................... 19

1.2.2 Large Eddy Simulations ..................................... 20

1.2.3 Reynolds-Averaged Navier-Stokes Solutions ............. 21

1.2.4 Hybrid Methods ............................................... 23

1.2.5 Computational Cost .......................................... 24

1.3 Objectives ............................................................ 28

Chapter 2. Governing Fluid Dynamics Equations ................... 33
Chapter 7. Computational Aeroacoustic Analysis of Wind Turbines

7.6.1 The Ffowcs Williams Hawking Method

7.6.2 Coupling of PUMA2 and PSU-WOPWOP codes

Chapter 8. Conclusions

8.1 Ship Airwake Simulations

8.2 Rotor and Pitching Wing Simulations

8.3 Computational Steering and Visualization

8.4 Wind Turbine Rotor LES Simulations

Appendix. A sample of PUMA2 Input File

References
**List of Tables**

3.1 Characteristics of the several parallel computers used for the computations during this study. ......................................................... 66

4.1 Summary of existing steering and visualization systems. [144] ........ 74

6.1 Drag Coefficients at Re=1.1 ×10^6 for a sphere, experimental results from [1], DES results from [61], [96]. ........................................ 149

7.1 Selected NREL Phase VI experimental cases from Reference [80]. ..... 155

7.2 Selected computational test cases. ............................................. 155

7.3 Computational performance of PUMA2 inviscid simulations on different clusters. ................................................................. 160
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>USS Saipan LHA-2 [197]</td>
<td>7</td>
</tr>
<tr>
<td>1.2</td>
<td>NREL Unsteady Aerodynamics Experiment wind turbine in NASA Ames wind tunnel [184]</td>
<td>17</td>
</tr>
<tr>
<td>1.3</td>
<td>Estimates of (a) the largest Reynolds number, and (b) number of grid cells, $n^3$, that can be solved with available computer power using DNS and LES as a function of year.</td>
<td>26</td>
</tr>
<tr>
<td>1.4</td>
<td>Computational complexity of CFD tasks performed in this study</td>
<td>29</td>
</tr>
<tr>
<td>3.1</td>
<td>A representation of the classical Runge-Kutta method for a model equation $dy/dt = f(t, y)$. [188]</td>
<td>55</td>
</tr>
<tr>
<td>3.2</td>
<td>The law of the wall [226].</td>
<td>62</td>
</tr>
<tr>
<td>3.3</td>
<td>Velocity components at the first cell center away from the wall.</td>
<td>62</td>
</tr>
<tr>
<td>4.1</td>
<td>Schematic of POSSE. [144]</td>
<td>76</td>
</tr>
<tr>
<td>4.2</td>
<td>Simple, complete POSSE (a) client and (b) server applications, written in C++. [144]</td>
<td>79</td>
</tr>
<tr>
<td>4.3</td>
<td>POSSE GUI to connect to the flow solver. [144]</td>
<td>83</td>
</tr>
<tr>
<td>4.4</td>
<td>POSSE PUMA2 client application depicting isosurfaces for a flow solution over the Apache helicopter. [144]</td>
<td>83</td>
</tr>
</tbody>
</table>
4.5 (a) Entropy isosurfaces for a flow solution over the rotor blades (Grid 2) and (b) Flow solution over the landing helicopter amphibious (LHA) ship geometry: entropy isosurfaces and Cp surface contours. [144] 84
4.6 POSSE GUI client for ship airwake simulations [128]. 89
4.7 Time history of w-velocity component at two different points (a) over landing spot 7, and (b) over landing spot 8, for the LHA ship for 30 degree yaw case [128]. 90
4.8 Instantaneous Entropy iso-surface for 30 degree yaw case over the LHA ship [128]. 91
4.9 Red-Blue grayscale anaglyphic image of dominant hairpin vortex [96]. 92
4.10 Instantaneous iso-vorticity lines for landing gear [203]. 92
5.1 Top view of a LHA class ship showing the landing spots on the ship deck [119] 96
5.2 The computational grid used for the LHA ship airwake simulations. 98
5.3 Contours of the X component of velocity (m/s) in a plane 6 meters above the LHA deck for different WOD angles. The solid lines represent the boundary of the ship geometry below. The arrows indicate the direction of wind. 102
5.4 Contours of the Z component of velocity (m/s) in a plane 6 meters above the LHA deck for different WOD angles. The solid lines represent the boundary of the ship geometry below. 103
5.5 Distributions of the time-average and the instantaneous velocity components along the centerline, 6 m above the LHA ship deck for 0° and 30° WOD angles. ................................................................. 104

5.6 Comparison of the instantaneous and the time-averaged CFD velocity magnitude, \((u^2 + v^2 + w^2)^{1/2}\), with the CFD data from Reference [166] along the centerline, 3 m above the LHA ship deck for 0° WOD. .... 105

5.7 Comparison of the instantaneous and the time-averaged CFD velocity components with the experimental data from Reference [166] at Spot 4 (at \(X = -175.87\) m) and 2.23 m above the LHA ship deck for 0° WOD. 106

5.8 Time history of the velocity components in the \(X\), \(Y\), and \(Z\) directions, 6 m above the landing spots 2 and 7 of the LHA for two WOD angles. . 108

5.9 Power Spectral Densities (PSDs) \([m^2/s^2]\) of the velocity components in the \(X\) and \(Z\) directions versus frequency [Hz], 6 m above the two landing spots of the LHA ship for two WOD angles. (Using Hamming window filtering.) .......................................................... 109

5.10 Instantaneous iso-surfaces of vorticity magnitude of 1.0 s\(^{-1}\) around the LHA for 0° and 30° WOD cases. ......................................................... 110

5.11 Snapshots of instantaneous iso-surfaces of vorticity magnitude of 0.8 s\(^{-1}\) around the LHA for 0° and 30° WOD cases. ................................. 111

5.12 Vorticity magnitude [s\(^{-1}\)] contours at \(t = 50\) seconds at several stations along the ship for 0°, and 30° WOD cases. ................................. 112

5.13 Pressure coefficient contours on the ship surface at \(t = 50\) seconds for 0° and 30° WOD cases. ................................................................. 113
5.14 The computational grid used for the LPD-17 ship airwake simulations [192]. 116
5.15 Instantaneous iso-surfaces of vorticity magnitude of $1.0 \, \text{s}^{-1}$ around the LPD-17 for $0^\circ$ and $30^\circ$ WOD cases [192]. 117
5.16 (a) Rectangular volumetric domain of CFD data at the rear deck of LHA for DI simulations. (b) Unstructured mesh on the ship surface and on a slice at $x = -27.432 \, \text{m}$. 120
5.17 Time histories of velocity components ($u, v, w$) of CFD data at a selected point in the DI mesh for $0^\circ$ and $30^\circ$ WOD cases. 121
5.18 Power Spectral Densities (PSDs) of the velocity components non-dimensionalized by the square of the freestream velocity ($V_\infty = 15.43 \, \text{m/s}$) versus frequency [Hz], at a selected point in the DI mesh over landing spot 8 for $0^\circ$ and $30^\circ$ degree WOD cases. (PSDs are not filtered.) 122
5.19 Distribution of the velocity components along the ship on a $XZ$-plane crossing the landing spot 8, and 17 ft above the deck, at $t = 40$ seconds for $0^\circ$ and $30^\circ$ WOD cases. 123
5.20 Distribution of the velocity components across the ship on a $YZ$-plane crossing the landing spot 8, and 17 ft above the deck, at $t = 40$ seconds for $0^\circ$ and $30^\circ$ WOD cases. 123
5.21 Helicopter position (m) w.r.t. ship coordinate system for an approach case. [119] 127
5.22 Helicopter attitude angles: PHI, THETA, PSI [degree] versus time [second], after entering the DI mesh for an approach case, for (a) $0^\circ$, and (b) $30^\circ$ WOD cases. [119] 128
5.23 Pilot inputs: Lateral and Longitudinal cyclic, Collective pitch and Pedal inputs [%] versus time [second], after entering the DI mesh for an approach case, for (a) 0°, and (b) 30° WOD cases. [119] ........................................ 129

6.1 2-bladed rotor geometry created using I-DEAS [188] ............................... 132

6.2 Unstructured tetrahedral grid (Grid B) ................................. 133

6.3 Unstructured mesh on the rectangular rotor blade and on the XY plane (at Z = 0) for two grids. ................................. 134

6.4 Time history of the rotor thrust coefficient (Grid A). ................. 135

6.5 Pressure coefficient distribution: a) spanwise at quarter-chord location and b) chordwise at 80% of radius. ................................. 135

6.6 Iso-surfaces of Mach number (for Grid B) ................................. 136

6.7 Results at the end of 6th revolution for Grid B (a) An iso-surface of vorticity magnitude of 30 s⁻¹ (b) Vorticity magnitude contours on YZ slice at a half chord behind the rotor blade and relative streamlines with respect to the vortex cores. ........................................ 137

6.8 (a) Cylindrical unstructured grid domain around (b) NACA 0015 rectangular half wing with AR = 10. ................................. 139

6.9 Unstructured hexahedral mesh (a) on the wing surface, and (b) on the xy-slice at mid-span ................................. 140

6.10 Variation of the angle of attack with time for a pitching wing according to Equation 6.1 ................................. 141
6.11 Residual versus number of iterations for steady-state computations with SSOR ................................. 142
6.12 (a) Pressure coefficient contours with streamlines, and (b) absolute velocity vectors, at mid-span for steady-state computations with SSOR . 142
6.13 Pressure coefficient contours with streamlines, at mid-span from time-accurate computations at four different instants ......................... 143
6.14 Instantaneous iso-surfaces of (a) Mach number and (b) vorticity magnitude for a pitching finite wing [128] ........................................... 144
6.15 a) Unstructured computational domain around the sphere with clustering in the wake region. b) Unstructured surface mesh of the sphere. [96] . 146
6.16 Pressure coefficient, $Re = 1.14 \times 10^6$. Dotted line- experiments by Achenbach [1], • DES by Constantinescu and Squires [61], × PUMA2 without wall model, □ PUMA2 with wall model (Grid B). [96] ....................... 148
6.17 Skin friction distribution, $Re = 1.14 \times 10^6$. Dotted line- experiments by Achenbach [1], • DES by Constantinescu and Squires [61], ▲ PUMA2 without wall model, □ PUMA2 with wall model (Grid B). [96] ............. 148
6.18 (a) Strong streamwise vortices in wake (b) Omega-shaped structure causing vortex shedding. [96] ................................. 150
7.1 Three-dimensional NREL wind turbine blade geometry generated using ProDesktop ......................................................... 153
7.2 NREL wind turbine rotor blade twist distribution and planform [66] . 153
7.3 Blade-mounted five-hole probe used in NREL Phase VI experiments [80]. 154
7.4 Unstructured tetrahedral grid (Grid A) generated using Gridgen. . . . . 155
7.5 Unstructured tetrahedral Grid A. . . . . . . . . . . . . . . . . . . . . . . 156
7.6 Grid details near the blade surface for Grid B. . . . . . . . . . . . . . 157
7.7 Two bladed wind turbine rotor geometry. . . . . . . . . . . . . . . . . . 158
7.8 Velocity vectors for the wind and the rotating blade at 80% span for the
selected computational cases. . . . . . . . . . . . . . . . . . . . . . . . . . 159
7.9 Permeable surface embedded in the unstructured grid. . . . . . . . . . 159
7.10 Computational performance of PUMA2 inviscid simulations on different
clusters. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 161
7.11 Inviscid Results: Vorticity iso-surface for a) CASE I, b) CASE II, c)
CASE III. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 163
7.12 Snapshots of instantaneous vorticity iso-surfaces for CASE I: Inviscid
results at the end of (a) 1st, (b) 2nd, and (c) 3rd revolutions. . . . . . . . 164
7.13 Inviscid Results: Gauge pressure \((P - P_\infty)\) contours for CASE I at \(t = 1.667\) s (Left: lower surface, Right: upper surface). . . . . . . . . . 169
7.14 Inviscid Results: Gauge pressure \((P - P_\infty)\) contours for CASE II at \(t = 2.500\) s (Left: lower surface, Right: upper surface). . . . . . . . . . 170
7.15 Inviscid Results: Gauge pressure \((P - P_\infty)\) contours for CASE III at \(t = 2.500\) s (Left: lower surface, Right: upper surface). . . . . . . . . . 171
7.16 Inviscid Results: Chordwise pressure coefficients at 5 spanwise blade sta-
tions for a) CASE I, b) CASE II, c) CASE III. Symbols for Instantaneous
results: red square; Averaged results: blue circle, and Experimental data
(average): black diamond. . . . . . . . . . . . . . . . . . . . . . . . . . . . . 172
7.17 Inviscid Results: Time history of the thrust coefficient for a) cases I and II, and b) case III.

7.18 Inviscid Results: Time history of the force coefficients for all three cases in a) \( x \)-direction, and b) in \( z \)-direction (relative to the blade).

7.19 Inviscid Results: Relative velocity magnitude contours with streamlines around the airfoil sections at 5 spanwise blade stations for a) CASE I at \( t = 1.667 \) s, b) CASE II, and c) CASE III at \( t = 2.500 \) s.

7.20 Inviscid Results: Instantaneous relative a) velocity magnitude and b) angle of attack distributions at five spanwise stations along blade 2, for case I at \( t = 1.667 \) s, and for cases II and III at \( t = 2.500 \) s.

7.21 LES Results: Vorticity iso-surface for a) CASE I, b) CASE II, c) CASE III.

7.22 LES Results: Vorticity iso-surface for a) CASE I, b) CASE II, c) CASE III.

7.23 LES Results: Gauge pressure \( (P - P_\infty) \) contours for CASE I at \( t = 4.169 \) s (Left: lower surface, Right: upper surface).

7.24 LES Results: Gauge pressure \( (P - P_\infty) \) contours for CASE II at \( t = 4.169 \) s (Left: lower surface, Right: upper surface).

7.25 LES Results: Gauge pressure \( (P - P_\infty) \) contours for CASE III at \( t = 2.708 \) s (Left: lower surface, Right: upper surface).
7.26 LES Results: Chordwise pressure coefficients at 5 spanwise blade stations for a) CASE I, b) CASE II, c) CASE III. Symbols for Instantaneous results: red square; Averaged results: blue circle, and Experimental data (average): black diamond. ................................................. 184

7.27 LES Results: Relative velocity magnitude contours with streamlines around the airfoil sections at 5 spanwise blade stations for a) CASE I at t = 4.169 s, b) CASE II, and c) CASE III at t = 2.708 s . . . . . . . 185

7.28 Rotor Thrust coefficient for the three LES cases . . . . . . . . . . . . . . 186

7.29 (a) In plane relative velocity magnitude \((u_r^2 + v_r^2)^{1/2}\) at 80% span, (b) velocity profile along the black solid line (at \(X = 0.0215\) m) shown in (a), and (c) comparison of boundary layer profile with the log-law. CASE I at t = 4.169 s. LES results. ................................................. 188

7.30 Wind turbine rotor wake characteristics of the LES results for CASE I at t = 4.169 s.(a) Unstructured mesh on YZ-slice at X = 0 (GRID A) (b) Vorticity magnitude contours and absolute streamlines . . . . . . 192

7.31 Wind turbine rotor wake characteristics of the LES results for CASE I at t = 4.169 s.(a) \(w\) (Z-velocity component) contours on YZ-slice at X = 0 with streamlines in the absolute frame (solid red lines) and streamlines relative to the tip vortex cores (solid black lines). Dashed white line shows the rotor tip diameter. (b) \(v\) (Y-velocity component) contours within the rotor wake on XZ-slice at \(Y = 2.5\) m with the rotor disk area shown as circular solid black line . . . . . . . . . . . . . . . . . . . . . 193
7.32 Comparison of the wake expansion characteristics of the LES results for

CASE I at \( t = 4.169 \) s. ........................................ 194

7.33 Instantaneous pressure contours on the permeable surface embedded in

the unstructured grid. ............................................. 200
Acknowledgments

I would like to thank my adviser Prof. Lyle N. Long for all his guidance and support during this research. I would also like to thank the committee members: Prof. Philip J. Morris, Prof. Edward C. Smith, Assoc. Prof. Joseph F. Horn, Prof. Yousry Azmy, and also Prof. Paul E. Plassmann, for reviewing my work and for their valuable comments. I would like to thank the Pennsylvania State University, the Department of Aerospace Engineering, the Institute for High Performance Computing Applications (IHPCA) (now called the Institute for Computational Science, ICS), and the Rotorcraft Center of Excellence (RCOE) for providing generous resources. I would also like to thank Prof. George A. Lesieutre, Prof. Dennis K. McLaughlin, and Prof. Cengiz Camci for their support during my studies. I would like to thank my friends and colleagues in IHPCA, RCOE and Aerospace Eng. Dept. for their friendship and support. Thanks to following people for working together on various parts of this thesis: Dr. Anirudh Modi, Shailesh Jindal, Dr. Anupam Sharma, Dr. Dooyong Lee, Ankur Gupta, Rui Cheng and Steve Miller. I would also like to thank Dr. Robert P. Hansen for his help on solid modeling and for his friendship.

I would also like to thank all my friends in State College, Pittsburgh, Baltimore and DC, and also in Turkey for their good friendship and support. I want to express my love and deepest gratitude towards my family: Adil, Hulya, Gokay, Aylin, and Hikmet, Muazzez, Bahar, Alper and Alp, and my dear husband Oguz Uzol for their love and support.
Chapter 1

INTRODUCTION

The flow fields encountered in many engineering applications involve complex geometries, and are highly three-dimensional, unsteady, separated and turbulent. Numerical predictions of these complex flow fields are now possible due to the recent progress in computer technology, numerical methods, and turbulence modeling. Therefore, parallel Computational Fluid Dynamics (CFD) simulations now play an increasingly important role in all areas of science and engineering. Although these simulations are still not practical as a design tool because of the costly computational time requirements for time-dependent viscous flows, they are becoming more and more efficient, and they are significant as “numerical experiments” for comprehension and prediction of these flows.

In this research, computational simulations of three-dimensional, unsteady, separated, vortical flows around complex geometries are performed by using stationary or moving unstructured grids. The use of unstructured grids helps to efficiently discretize the flow field around real engineering systems which have complex geometries by allowing efficient cell distribution.

Two main engineering problems are investigated in this study. The first problem is the unsteady simulations of ship airwake flow fields using stationary unstructured grids. These flow fields are particularly challenging to ship-based helicopter operations.
The time-varying nature of the ship airwake may have a significant impact on the helicopter response and pilot control activities. The predictions of airwakes may help in the understanding of helicopter flight dynamics and development of advanced control models.

The second problem is the unsteady simulation of wind turbine rotor flow fields using unstructured moving grids that are rotating with the whole three-dimensional rigid rotor geometry. Accurate predictions of aerodynamic loads and noise are very important and challenging in the design process of wind turbines. These predictions could lead to lighter and more flexible structures, which are necessary to reduce the cost of wind energy, and low noise levels, which are necessary for the public acceptance of wind turbines.

In this chapter, first, the literature related to computational and experimental research on ship airwakes is reviewed in detail. Short reviews on the blade-sailing problem and dynamic interface simulations are also presented. Then, the previous research on wind turbine rotor flow fields is reviewed. Turbulence modeling plays a vital role in the simulations of these complex flows. Various modeling and simulation methodologies have been proposed in the literature for turbulent flow calculations: Direct Numerical Simulations (DNS), Large Eddy Simulations (LES), Reynolds Averaged Navier-Stokes (RANS) solutions, and hybrid methods. These different turbulence modeling approaches are also briefly reviewed. The computational requirements now and in the future for these approaches are also discussed. Finally, the objectives of this thesis are presented.
1.1 Background on Unsteady Separated Flows

Several example flows over simple geometries, which expose the difficulties of simulating complex highly separated turbulent flows, are widely used in the literature mainly for the development of turbulence models and numerical methods. Such example simulations include computations of flow over a backward-facing step [56,59], flow around a square prism [60], flow over and around a cube [242], flow around other bluff bodies such as a circular cylinder [81,82,111,233], a sphere [42,231], and a prolate spheroid [43]. These problems are good models for complex 3-D engineering geometries such as road vehicles [237], landing gears [204], and helicopter fuselages [3,203].

On the other hand, attempts to simulate such complex flows around complex geometries with the help of advancements in computers would help to understand and recognize the difficulties encountered during the simulations (for example, in programming, pre- and post-processing steps) and the further needs in and developments of numerical modelings and methodologies. The flow solutions obtained from such simulations would also allow further interdisciplinary studies such as dynamics and control analyses, aerodynamic noise calculations, or aeroelastic structural analyses.

Two examples of such complex flows around complex engineering geometries are the flow fields of a ship airwake and the flow fields of a wind turbine rotor, both of which are investigated in this thesis.
1.1.1 Ship Airwake

Helicopter shipboard operations present a multitude of challenges. One of the primary factors that make it extremely difficult, and potentially dangerous, to perform helicopter flights to and from a ship deck is the unsteady ship airwake. Other factors that are also important are the possibly adverse weather and sea conditions, ship translation and rotation (oscillating landing spot), small flight decks (for some ships), and poor visibility conditions.

The most challenging rotorcraft/ship dynamic interface problems occur during:

- Engagement and disengagement (run-up and run-down) of the rotor system while the aircraft is on the flight deck (i.e., the blade-sailing phenomena);
- Take-off and landing operations (e.g., launch, departure, approach and recovery);
- Hovering over the moving flight deck (i.e., stationkeeping).

It requires tremendous practice and skill for the pilots to learn to perform these operations. A common practice to ensure safe operation of helicopters from a ship is to find the safe helicopter operating limits (SHOLs) for a ship/helicopter combination. These are usually defined in terms of allowable wind conditions (direction and speed) over the deck and are called wind-over-the deck (WOD) envelopes.

It is difficult to obtain these WOD envelopes using real, full-scale experiments for a variety of reasons. A series of dynamic interface flight tests for all possible combinations of wind speed and azimuth (typically in 5 knot, 15 degree increments) [118] must be performed to establish the envelope. However, these tests are costly, often limited (due to the absence of certain wind conditions and the availability of fleet assets),
and unsafe. This process also depends on subjective pilot ratings. On the other hand, scale-model wind tunnel tests allow control over the wind conditions, and can supply detailed information but are still costly and time consuming. In wind tunnel testing, small size ships (with lower Reynolds number) are used, and the full scale (high) Reynolds number cannot be obtained. In addition, the complexity of the flow field of a rotorcraft during take-off and landing is impossible to reproduce in a wind tunnel. Furthermore, it is both difficult and costly to obtain high quality and complete sets of coupled ship airwake/rotorcraft flow field measurements using both full-scale and scaled-model tests for all of the different WOD conditions for any ship/helicopter combination that could be used to obtain the SHOLs.

A numerical procedure to analyze shipboard operations and to obtain the WOD envelope would be desirable because it is controlled, safer and cheaper. It could also be used to train pilots by coupling with a flight simulator (as a piloted training tool). It can also be valuable as a non-real-time simulation tool for use in engineering and design by helping in the development of advanced flight control systems and in the design processes of future ships. Such numerical simulation tools could include:

- CFD simulations of the unsteady ship airwake,
- A high fidelity flight dynamics model for the helicopter,
- A model for the ship motion,
- A model of a human pilot for analyzing the pilot workload and response.

Understanding and modeling the airwake of a complex ship geometry presents a number of technical challenges. The modeling requires an accurate representation
of the complex ship geometry with superstructures and sharp edges. The flow around complex ship superstructures, including towers, antennae, radar dishes, exhaust stacks, etc, is very difficult to predict. Also, modeling of a moving flight deck because of ship translational speed and random sea motion must be taken into consideration. The effects of the atmospheric environment (atmospheric boundary layer and turbulence) and high winds are important.

The ship airwake is also largely affected by the helicopter rotor and fuselage wakes and vice versa, resulting in complex interactions. The ship airwake flow is highly three-dimensional, unsteady, separated, vortical and turbulent. There are massive regions of flow separation and there is shedding of strong turbulent coherent structures from the ships superstructure and sharp edges. As a result, the ship airwake contains a wide range of spatial and temporal scales, and large, unsteady vortical regions around and behind the superstructure and over the flight deck, which affect the rotor response, fuselage loads, and pilot workload. Turbulence modeling is also important for such simulations. It is a low Mach number flow, however there are complex interactions with the complex helicopter rotor flow during shipboard operations. Hence, accurate modeling and prediction of the highly unsteady airwake of a ship is critical for shipboard operations of rotorcrafts. To accurately resolve all the flow features: turbulence, boundary layer, flow separation and interaction of helicopter flow-field with the ship airwake, in a time-accurate simulation is a big challenge. The magnitude of the unsteadiness demands long time records which, coupled with large grids associated with such geometries, require enormous computational power. Full-scale or wind tunnel experiments, on the other
hand, can easily generate long time records of data, but the associated initial setup cost and time can be substantial [168, 196, 197].

In the following sub-sections, the computational and experimental ship airwake research is reviewed in detail. A short review of the literature on the blade-sailing problem and dynamic interface simulations is also presented. The ship airwake CFD simulations of an LHA (Landing Helicopter Assault) class ship (Figure 1.1) performed in this study are discussed in Chapter 5, and also in References [117–119, 192].

1.1.1.1 Previous Ship Airwake Studies

Understanding and modeling the unsteady, separated and turbulent airwakes of different classes of ships have been the focus of numerous experimental and computational studies.

Experimental investigations include full-scale tests aboard ships such as LHA class ship [166, 168, 245], as well as wind tunnel scaled-model tests, e.g. LHA [48, 166, 168],
DD-963 (a SPRUANCE class destroyer) [98,99], a non-aviation ship model [85,86,174], LPD-17 (Landing Platform Dock) [73,215], a Generic Frigate (GF) [258], a Canadian Patrol Frigate (CPF) [213,260], a Simple Frigate Shape (SFS) [38,255], a La Fayette frigate model [214], and others [132]. In addition to the ship airwake investigations, there have been several full-scale helicopter/ship dynamic interface experiments, e.g., the XV-15 tiltrotor aboard the LPH (Landing Platform Helicopter) class USS TRIPOLI [13], the H-46 Sea Knight helicopter aboard USS GUAM (LPH-9) [14], and others [248]. Recently, wind tunnel tests to study the ship/helicopter interactions were also conducted [121,197,256]. Zan [256] investigated the effects of the wind speed and direction on rotor thrust of a 4-bladed rotor model immersed in a CPF ship airwake. It was shown that the reduced inflow due to the ship airwake can significantly decrease the rotor thrust, therefore impacting the pilot workload and operational envelopes. Lee and Zan [121] performed wind tunnel experiments to investigate the unsteady aerodynamic loading on a rotorless Sea King fuselage in the turbulent airwake of a CPF ship. Silva et al. [196,197] obtained the V-22 tiltrotor airframe force and moment, and detailed velocity field measurements operating over a scaled-model LHA ship.

Computational simulations of ship airwakes provide an attractive alternative and have been performed using different numerical approaches for different ships and WOD conditions. The steady-state Reynolds Averaged Navier-Stokes (RANS) simulations of an airwake were performed using unstructured grids for an LPD ship by Tai [215], for a DD-963 class ship by Tai and Carico [217], for an LHD (Landing Helicopter Deck) ship using an enhanced grid by Tai [216], and using structured grids for an SFS by Reddy et al. [171], and also using hybrid grids for a surface effect ship (SES) by Moctar [137].
Syms [213] explained that the differences between the steady RANS simulations and experimental data might be due to the unsteadiness in the flow, which cannot be handled by such simulations. Also, Bogstad et al. [19] obtained steady-state airwake data for six different ships of the Royal Navy for a helicopter flight simulator by performing inviscid flow simulations. As commented by Zan [257], the incorporation of the time-accurate airwake is important for a high-fidelity flight simulation, and the changes in the wind direction need to be considered for a reasonable validation of CFD.

Guillot and Walker [73] solved the unsteady compressible Euler equations to investigate the unsteady airwake over an LPD-17, and compared the results with the scaled-model wind tunnel measurements. Also, the unsteady airwake and exhaust gas trajectories over highly complex ship superstructures were computed over the DDG-51 Flt-IIA (a SPRUANCE class destroyer) by Landsberg et al. [112] using FAST3D flow solver, and over an LPD-17 by Camelli et al. [31, 32] using Large Eddy Simulations (LES) with a Smagorinsky subgrid-scale stress model. Polsky and Bruner [168], and Polsky [166, 167] performed steady and unsteady simulations of LHA ship airwakes using different numerical methods such as the Monotone Integrated Large Eddy Simulation (MILES) approach as well as $k-\omega$ and shear stress transport (SST) turbulence modeling. It was shown that steady-state CFD calculations were unable to predict the time average of the turbulent flow field, and observed that turbulence modeling added too much dissipation to the calculation since the flux-splitting numerical schemes are very dissipative. Polsky [167] also studied the importance of the grid quality and atmospheric boundary layer modeling for the airwake simulations for beam (90 degrees) winds. The atmospheric boundary layer was modeled as a farstream inflow/outflow boundary condition.
while considering enough grid resolution over the ocean surface plane, and improved the comparisons of the predictions with the full-scale experimental data. Recently, Arunajatesan et al. [8,9] have conducted some preliminary steady/unsteady calculations of the LHA ship airwake using the MILES approach.

In addition to the airwake computations, there have been several computational studies on ship airwake/helicopter rotor or fuselage flow interactions by Zan and Syms [259], Landsberg et al. [113], Tattersall et al. [224], and Wakefield et al. [239]. Arunajatesan et al. [9] calculated the flow field of the AV-8B Harrier outwash for various hover altitudes, and they are also planning to study the interaction of V/STOL (Vertical/Short Take Off and Landing) and ship airwake flow fields.

Previously, at the Pennsylvania State University, Long et al. [126], and Liu and Long [125] used the Non-Linear Disturbance Equations (NLDE) approach [148] (without viscous terms) to solve for the unsteady fluctuations over the generic SFS using both structured and unstructured grids. Modi [140] simulated the steady-state ship airwakes over an SFS, an aircraft carrier CVN-75 and an LHA class ship. Sharma and Long [195] and Sezer-Uzol et al. [192] have performed inviscid airwake simulations over an LPD-17 using parallel flow solver PUMA2. A reasonable match has been observed between the time averaged velocity magnitudes from the experiments and the numerical results. Some of the results are also presented in Chapter 5 for comparison of the airwakes of LPD-17 and LHA ships.
1.1.1.2 Previous Work on Blade Sailing

Blade sailing is one of the most important problems of helicopter shipboard operations, and is often observed when operating a helicopter from a ship deck. In severe conditions blade sailing may result in catastrophic “tunnel-strike” (for tandem rotor configurations) or “tailboom-strike” (for single rotor configurations) which can severely damage the rotor blades and also cost lives. These problems occur before take-off during rotor engagement and after landing during rotor disengagement. During these operations the helicopter rotor is rotating at low speeds and so the centrifugal forces are small. The weak centrifugal force cannot offer enough resistance to the impulsive lift/drag force which can cause large-amplitude oscillations of the blades in longitudinal/lateral direction. The problem is aggravated when operating from a ship deck because of the flow separation from the sharp edges of the mast, the deck and the hangar. The shedding of vortices from the edges characterizes the time variation of the flow over the deck. There is a chance that the shedding frequency will match the angular frequency of the rotor and excite the blades in resonance. This may amplify the deflections to such a level that it strikes the tail boom or fuselage. The vertical wind velocity through the rotor disc is the main contributor to the impulsive forces as it directly alters the angle of attack of the blades.

The problem of blade-sailing has been addressed by many experimental [90, 91, 153–155] and analytical [64,65,91,103,104,106–109,152,154–159,200] investigations. Theoretical analyses tools for transient aeroelastic blade response during engage/disengage operations have been developed for different rotor systems: for teetering and articulated
rotors with flap and droop stops by Newman [157], for articulated and hingeless rotor systems by Geyer and Smith [64, 65] and for multi-bladed gimbaled tiltrotors by Kang and Smith [104], and Smith et al. [200]. Newman [157], Keller and Smith [107], and Keller [106] attempted to model the velocity distribution over the deck using simple linear models. They concluded that such simplified models cannot accurately predict the severe blade sailing observed in practice. This presents the need to obtain the “real” airwake data either experimentally or numerically.

1.1.1.3 Previous Dynamic Interface Studies

Ship/helicopter Dynamic Interface (DI) simulations have been studied [19, 27, 49, 83, 84, 115, 117–119, 130, 133, 177, 178, 214, 223, 236, 245–247, 249, 251] to investigate the pilot workload and response during the shipboard operations, to develop safe WOD flight envelopes and flight trajectories for any ship/helicopter combination, and to develop helicopter flight simulators for pilot training and engineering purposes. In DI simulations, the modeling problems include helicopter flight dynamics, rotor aerodynamics and fuselage loads, flight control systems, ship motion, ship airwake flow (usually as gust penetration models), and a human pilot workload and response model.

As addressed by Healy [84], the simulation of the airwake is the most challenging and computationally intensive task in numerically simulating the ship/helicopter dynamic interface. Healy also highlighted the need for more accurate experimental data on real ships. Recent studies on flight dynamics simulations have used experimental data [214], and/or the numerical data, e.g., the steady-state airwake [19, 115], and the
unsteady airwake \cite{27, 117–119, 251} data from CFD simulations. A stochastic representation of the unsteady airwake may be another alternative \cite{115, 251}. As stated by Zan \cite{257}, the incorporation of the time-accurate airwake is important for a high-fidelity flight simulation. Lee et al. \cite{119} (as described in Chapter 5) performed the LHA ship airwake and DI simulations for the UH-60A Black Hawk helicopter landing, take-off and station-keeping, and showed that the time-varying airwake has significant effect on pilot control activities during shipboard operations.

McKillip et al. \cite{133} described a Dynamic Interface simulation tool in which a real-time, freewake rotor flow field is coupled with panel-based fuselage and ship flow fields, thus considering the interactional aerodynamics of the V-22 tiltrotor operating over an LHA class ship. He et al. \cite{83} performed a similar study for shipboard landing simulation of the CH-46 Sea Knight helicopter.

\subsection{1.1.2 Wind Turbine Rotor Flow Fields}

Wind turbines, which offer the promise of less expensive and clean energy from a renewable energy source: the wind, have unique aerodynamic and aeroacoustic characteristics that make their prediction more challenging in many ways than already complicated problems of helicopter rotors or propellers. In particular, wind turbine blades can experience large changes in angle of attack associated with sudden large gusts, changes in wind direction (wind yaw), atmospheric boundary layer effects, atmospheric turbulence, or interaction with the unsteady wake shed from the tower support on downwind Horizontal Axis Wind Turbines (HAWT). These three dimensional and unsteady blade/inflow/tower wake interactions can result in impulsive loading changes and dynamic stall over
portions of the rotating blades. These influences are described in several reports by the National Renewable Energy Laboratory (NREL) such as Robinson et al. [175]. Furthermore, typically important rotor noise sources, such as steady thickness and loading noise, do not play the same role in large-scale wind turbines, because the blade passage frequency is well below the audible range. Recognition of this fact leads one to consider rotor broadband noise sources as the primary noise source. Turbulence ingestion noise, airfoil self noise, tip-vortex noise, and other broadband noise sources have typically been treated through empirical or semi-empirical methods [5, 6, 25, 69, 88]. In addition, the acceptance of wind turbines by the public depends strongly on achieving low noise levels in application, particularly in European sites. Hence, the accurate predictions of aerodynamic loads and noise are very important and challenging in the design process of wind turbines. These predictions could lead to lighter and more flexible structures, which are necessary to reduce the cost of wind energy.

Three-dimensional flow properties of rotating blades are an essential feature of any wind turbine aerodynamic or aeroacoustic simulation. While important information can be learned from two-dimensional and non-rotating simulations, some aspects of the physics of wind turbine aerodynamics and noise must be obtained from rotating blade simulations. Three-dimensional flow over rotating blades can be significantly different than the flow over a wing, and there can also be dramatic differences between 2-D and 3-D simulations [44,143]. Rotating blades can have significant spanwise (or radial) flow. This radial flow can result in coriolis forces which may induce favorable pressure gradients and delay boundary layer separation [183,208]. Also, of course, the blade speed varies linearly from root to tip. In addition, the three-dimensional wake of a rotating blade can remain
in close proximity to the blade for a long period of time (compared to the wake of a wing). One of the most important and complex flow features of wind turbines (and helicopter rotors) is dynamic stall, which is an unsteady flow phenomena. For wind turbines, dynamic stall occurs because of non-axial, non-uniform inflow due to the rotation of the blades (3-D effects), the unsteady effects of atmospheric turbulence and boundary layer (wind gusts, wind shear), off-axis yaw operations (hence cyclic blade pitch), the interaction of the wind turbine tower with the blades (for downwind configuration).

Using the time-dependent governing equations allows the simulation of a number of important phenomena: broadband noise, incoming atmospheric turbulence and gusts, wind shear (or atmospheric boundary layer). These are important for leading edge noise and tip noise prediction. By incorporating time dependent boundary conditions, either a gust or turbulent incoming flow can be introduced. Chyczewski et al. [39] have done this in the past with jet noise predictions, to simulate turbulence levels inside a nozzle and their effect on the jet shear layers and noise. Also, atmospheric gusts or wind turbine tower shadow effects can be simulated using source terms in the time-accurate, three-dimensional, governing momentum and energy equations. For these reasons, time-accurate, three-dimensional and compressible rotating blade simulations are essential.

Furthermore, the acoustic propagation is of interest at relatively large distances from the wind turbine, and the far-field noise predictions based on the Ffowcs Williams - Hawkings method (FW-H) [57] could be used for this purpose [149]. The calculated three-dimensional mean flow could also provide the basis for detailed unsteady flow simulations based on the Nonlinear Disturbance Equations (NLDE) [148] which in turn
can provide the aerodynamic input needed for the prediction of trailing edge and tip vortex noise [149].

There have been many experimental measurements and numerical computations of wind turbines in the literature to understand the flow field characteristics of rotating blades and to obtain better aerodynamic models for performance analysis and design purposes. Previous NREL Unsteady Aerodynamics Experiments (UAE), (Phase II-V) [79,198], included field tests for a number of wind turbine configurations and wind conditions. The Combined Experiment Rotors (CER) of the UAE were 2 or 3-bladed with untapered, untwisted/twisted planform using the NREL S809 airfoil profile. Recent NREL UAE Phase VI [80] experiments, in which a large-scale wind turbine (Figure 1.2) was examined at the NASA Ames 24.4x36.6 m (80x120 ft) wind tunnel facilities, provides extensive data for controlled wind conditions for validation of numerical computations. The Phase VI, 2-bladed, tapered, twisted, upwind/downwind, stall-regulated, full-scale wind turbine rotor used the same S809 airfoil profile [66]. For HAWT applications, extensive 2-D experimental data sets are available from the Ohio State University [201], Delft University of Technology [173], and Colorado State University [29] for the S809 airfoil. The S809 airfoil is 21% thick with a 1% camber and a leading edge radius of less than 1%. This laminar-flow airfoil from the NREL thick-airfoil family was designed to have a sustained maximum lift, minimal sensitivity of lift to roughness, and low profile drag. It was designed with a maximum lift coefficient of about 1.0, which occurs at around 8 degrees angle of attack.

For design purposes, aerodynamic modeling and performance analysis of wind turbine rotors, similar to helicopter rotors [122], have been mostly based on the 2-D
Fig. 1.2. NREL Unsteady Aerodynamics Experiment wind turbine in NASA Ames wind tunnel [184]

airfoil section database and the blade element momentum (BEM) predictions. For these predictions, improved engineering (aerodynamic) models, which are necessary for unsteady three dimensional and rotational effects, dynamic stall, stall delay and post-stall characteristics, have been investigated using various experimental, especially the NREL UAE, and numerical analyses [76, 122, 183–185, 219]. Some studies investigated the use of vortex panel methods for the aerodynamic analysis of wind turbine rotors [75, 78, 220]. First-principles-based analysis: 3-D Navier-Stokes predictions, which are expensive for routine engineering analysis and design, can be used to improve the engineering approximations inherent in BEM methods [186, 252]. These predictions can model the complexities of 3-D unsteady viscous flow, blade-vortex interactions and tower shadow effects [202]. Previously, 3-D simulations for the NREL blind code comparison [186] were performed for an upwind NREL Phase VI rotor for zero yaw for different wind speed
conditions. Xu and Sankar [252] performed hybrid and full N-S simulations. The hybrid method was a zonal approach using RANS solution with the S-A model in the viscous region, and potential flow solution in the inviscid region with concentrated vortices shed from the blade. Sorensen et al. [202] used an incompressible RANS solver using a $k - \omega$ SST model. The blind comparison has shown that CFD computations are competitive with BEM-type computations for new and unknown rotors [202]. Johansen et al. [97] predicted force coefficients and pressure distributions fairly well from the DES computation of a non-rotating (parked) NREL Phase VI wind turbine blade undergoing pitching motion in the light stall region. The DES results have shown better accuracy for highly separated flows owing to the better resolution of three-dimensional flow structures, compared to the two-equation RANS calculations. Recent numerical computations of wind turbine flow fields for different wind conditions have been performed using different approaches in References [15, 50, 131, 182, 232], and recent investigations for aeroacoustics of wind turbines can be found in References [92, 234, 253].

LES represents large scale motions explicitly and models only smaller scale motions. Therefore for flows with unsteady separation and vortex shedding LES generally gives better results compared to RANS predictions ( [24, 169]). In wind turbines, most important sources of broadband noise are the interaction of inflow turbulence with rotor blades, which generates loading fluctuations, and dominates the spectrum at lower frequencies ($\sim 60$ Hz, [70]); massive flow separation which results in large scale structures that generate also low frequency noise; interaction of the turbulent boundary layer with the trailing edge, which is less dominant at low frequencies and becomes more important at frequencies higher than $\sim 1$ kHz; and trailing edge bluntness, which results in vortex
shedding, and has a confined frequency range (\(\sim 250 \text{ Hz} - 2.5 \text{ kHz}\)). LES has the potential to predict these complex turbulent interactions and resolves energy containing large scale eddies, which contribute significantly to noise radiation. Furthermore, LES allows the representation of a wide range of scales of noise-generating eddies. Hence, LES is a valuable tool in calculating the noise sources at Reynolds numbers of engineering interest ( [240]).

1.2 Background on Turbulence Modeling

Turbulent flow is three-dimensional, unsteady, rotational, and irregular, contrary to laminar flow which is regular and deterministic. A wide range of spatial and temporal scales, which increases with Reynolds number, exists in turbulent flows. The largest length scales (integral scale) are related to the domain size, and the smallest scales (kolmogorov scale) are related to the dissipative eddies where the viscous effects become predominant. Various modeling and simulation methodologies have been proposed in the literature for turbulent flow calculations: Direct Numerical Simulations (DNS), Large Eddy Simulations (LES), Reynolds Averaged Navier-Stokes (RANS) solutions, and hybrid methods.

1.2.1 Direct Numerical Simulations

In Direct Numerical Simulations (DNS) [169], all scales of turbulence are resolved by solving the time-dependent and three-dimensional (i.e., four-dimensional) Navier-Stokes equations directly, and no turbulence modeling is required. DNS is computationally very expensive, especially for high Reynolds-number flows, since the grid size
is required to be of the same order of magnitude as the finest turbulent eddies. The total number of grid points required by DNS is proportional to the Reynolds number as $Re_l^{9/4}$. The only error in the computations is due to the numerical methods [169].

1.2.2 Large Eddy Simulations

In Large Eddy Simulations (LES) [169], the three-dimensional, time-dependent motion of large scales are computed explicitly and a model is used for the effects of small scales. LES does not require as much resolution as DNS, even for LES with near wall resolution where the grid is fine enough to resolve 80% of the energy everywhere including the viscous near wall region [169], therefore the computational cost is much less. In contrast to the large scales, small scales are assumed to be isotropic and universal, hence presumably easy to model. Thus, LES offers a viable tool to better predict and understand the high Reynolds number, turbulent, unsteady, separated flows. Many different approaches have been developed for subgrid-scale modeling such as the classic Smagorinsky model [199], mixed subgrid-scale models [261], and dynamic modeling with any base model which also allows transfer of energy from small scales to large scales, i.e., backscattering [63,147]. Another approach, called Monotone Integrated LES (MILES) [20], uses neither an explicit SGS model nor filtering, but relies on the properties of the specific numerical algorithms to provide an implicit dissipation mechanism. For LES of compressible flows, a density weighted (Favre-filtering) procedure was defined by Speziale et al. [209]. The subgrid modeling for LES of compressible flows are also formulated and discussed by Yoshizawa [254], Kosivic et al. [110] who developed two subgrid models called a nonlinear model and a stretched-vortex model.
In order to perform LES for high Reynolds number wall-bounded flows at reasonable computational cost, wall models have been used to allow fewer cells in the wall layer. The wall models for LES are discussed in detail in References [30, 164, 179, 241]. More information on LES can be found in References [136, 179].

1.2.3 Reynolds-Averaged Navier-Stokes Solutions

The solution of the Reynolds-averaged Navier-Stokes (RANS) equations, which is the most commonly used approach for computation of turbulent flows in engineering applications, computes the mean flow and simply models all scales of turbulence. There have been several approaches for the modeling of turbulence in RANS computations: algebraic models: zero-equation and half-equation models; non-algebraic models: one-equation and two-equation models; and second-order turbulence models: Reynolds Stress Models (RSM).

Some of the zero-equation (algebraic) models, where both the length scale and the time scale are specified by empirical means, are the Cebeci-Smith two-layer model [34], the Baldwin-Lomax two-layer model [12]. These models are the simplest of all turbulence models, easy to implement and included in most fluid dynamics codes. The Reynolds stress tensor is given as a product of an eddy-viscosity and the mean rate of strain tensor. The Boussinesq eddy-viscosity approximation and Prandtl’s mixing length hypothesis are used [169]. The eddy viscosity is, by dimensional analysis, the product of a length scale and a velocity scale. These models are generally applicable to attached flows and the fail to produce accurate results for massively separated flows. An example of half-equation models is the Johnson and King model [100] formulated to improve the predictions of
shock and separated flows. It is a non-equilibrium model accounting for the upstream history of the flow in the calculated eddy-viscosity.

In one-equation models, a model transport equation is solved for the eddy viscosity, and the time scale is built from the turbulent statistics rather than from the mean velocity gradients. The Baldwin-Barth model [11] is a one-equation model which is derived from a two equation \( k - \epsilon \) model [101]. Another widely used one-equation model is the Spalart-Allmaras model [206] which is developed based on dimensional analysis and empirical criteria. These models do not require evaluation of ambiguous length scales. They require numerical solution of only one partial differential equation. In two-equation models, such as the \( k - \epsilon \) model [101], the original \( k - \omega \) model [243,244], and the modified \( k - \omega \) [134] and SST \( k - \omega \) models [135], transport equations are solved for two independent turbulent quantities related to the time and length scales, and the eddy viscosity is calculated using those quantities [169].

The Reynolds Stress Model (RSM) [114,169], called a second-order closure model, solves the modeled Reynolds stress transport equations to directly compute the Reynolds stresses, instead of the eddy viscosity approach used for the other models. The use of RSM results in a closed turbulent flow problem with 16 equations and 16 unknowns: 5 averaged flow equations and 11 turbulent equations (6 Reynolds stress equations, 1 turbulent kinetic energy equation, 1 rate of dissipation of kinetic energy equation, and 3 Reynolds heat flux equations) [169]. This results in high mathematical complexity and computational cost. There are nine constants generated by the modeling processes. These values are determined from several turbulent flow experiments just as the molecular viscosity, thermal conductivity, and specific heats are determined from laminar
flow experiments. There are some improvements yet to be made in modeling of the pressure-strain term, nonisotropic turbulent diffusion, nonisotropic dissipation, near-wall conditions, near-free-surface conditions, and secondary flows. For recent turbulent flow simulations using RSM see References [2, 4]. More information on turbulence modeling can be found on References [36, 51, 169, 244].

1.2.4 Hybrid Methods

The purpose of the hybrid methods is to use the computational advantages of the RANS and LES approaches in a single/coupled solution strategy. The Detached Eddy Simulation (DES) [207, 212] is one of the hybrid RANS/LES methods, which uses a RANS approach in near-wall regions and an LES approach in outer regions away from walls. These regions can be pre-selected explicitly in the computational domain, or can be defined according to the local length scales obtained from the RANS and LES methods during the computations [212].

Another hybrid RANS/LES method is the Flow Simulation Methodology (FSM) [55]. In the FSM method, a contribution function is introduced to allow a consistent transition from DNS to RANS in order to make the proposed approach for LES truly applicable for complex flows. This transition should be time and space dependent and should be based on the instantaneous and local physical resolution, i.e., the local grid size divided by the smallest relevant scale. The computation approaches a direct numerical simulation (DNS) in the fine grid limit, or provides modeling of all scales in the coarse grid limit and thus approaches an unsteady RANS calculation. In between these resolution limits, the
contribution function adjusts the necessary modeling for the unresolved scales while the larger (resolved) scales are computed as in traditional large-eddy simulations (LES) [55].

1.2.5 Computational Cost

The number of grid cells and the resulting computational cost, required to simulate turbulent flow, increase with the Reynolds number at a rate depending on the numerical method used. For example, DNS, which resolves all scales of turbulence, is computationally very expensive because the smallest length scale of the computational grid is required to be on the order of the size of the finest turbulent eddies. If we let \( n \) be the number of grid cells along each dimension of a turbulent flow simulation using a uniform grid, the total number of grid cells required for the three-dimensional simulation will be of order \( n^3 \). For DNS, the number of grid cells required has been estimated [169] to scale with Reynolds number as \( 4.4 \cdot Re_i^{2.25} \), and the total number of time steps, \( m \), scales [169] as roughly \( 23 \cdot n \). For a Reynolds number of \( 10^7 \), this estimate indicates that approximately \( 10^{15} \) grid cells would be required by a DNS simulation. For comparison, the largest case possible in 2003 was about \( 6.9 \cdot 10^{10} \) (4096\(^3\) grid points on the Earth Simulator computing system [102]).

In Large Eddy Simulations (LES), the three-dimensional, time-dependent motion of large scales is computed explicitly and small scale effects are modeled. As a result of the modeling of small scales, LES does not require as much resolution as DNS and, therefore, the computational cost is less than DNS. For LES with no wall function, the number of grid cells required, \( n^3 \), is estimated [169] to scale with Reynolds number as
With the use of wall functions or other numerical methods such as Detached Eddy Simulation (DES), significantly fewer grid cells would be required.

The scaling relations explained above can be used to estimate the largest Reynolds number able to be modeled by DNS and LES on a moderately sized parallel computer as a function of year, as also described in Reference [144]. A commonly used derivative of Moore’s law, that the computational speed of computers will double every 18 months, can be employed. Based on these assumptions, Figure 1.3(a) shows the largest Reynolds number that can be solved with the computing power available with a moderately sized parallel computer using DNS and LES for the same values of $n^3$ and $m$ from the present until the year 2050. This parallel computer model assumes the problem is solved on a dedicated machine with 100 processors running for 10 days. It is assumed that the processor speed doubles every 1.5 years starting from a 4 GHz processor in 2004. For this processor, it is assumed that a sustained performance of 800 Mflops (million floating-point operations per second) can be obtained (based on 2 operations/clock cycle and achieving 10% of peak speed). The total number of operations required to solve the problem is proportional to $m \cdot n^3$ and it is assumed that 1000 operations per grid cell per time step are required. For the results shown, it is assumed that the number of time steps, $m$, required for LES depends on $n$ based on the same relation as for DNS. Given the number of operations available to the simulation as a function of year from this parallel computer model, the largest Reynolds number can be estimated that can be solved by DNS and LES. These estimates are plotted in Figure 1.3(a).

Note that the number of grid cells, $n^3$, will be the same for DNS and LES based on the assumption that the number of time steps required, $m$, is the same for both
Fig. 1.3. Estimates of (a) the largest Reynolds number, and (b) number of grid cells, $n^3$, that can be solved with available computer power using DNS and LES as a function of year.
methods. In Figure 1.3(b), the number of grid cells required as a function of year is plotted for the largest Reynolds number problems shown in Figure 1.3(a). The total memory required for the simulation is proportional to the number of grid cells. If one assumes that approximately $400 \cdot n^3$ bytes of memory (assuming 50 double precision variables) per grid cell is required for the simulations, then the total memory required for these simulations exceeds that available from a single work-station. However, as long as enough memory is available, the required floating-point operations will be the limiting factor in the above calculations. Specifically, for the problem we assume that we can solve in 2004, $n^3$ is $2.3 \cdot 10^9$. This number would require approximately 10 Gbytes of memory per processor on the parallel computer for storing the grid cell information. This memory requirement seems large. However, we note that because the overall computational requirements grow at a higher order (with respect to $n$) than the memory requirements, we can assume that the total memory available to solve future problems will be sufficient. Spalart [205] predicted that it will be possible in 2080 to perform DNS simulation for a chord Reynolds number of $7 \cdot 10^7$ for a wing, which is consistent with our analysis. Therefore, LES, Reynolds Stress methods, or hybrid methods will be essential to do realistic aerospace simulations, now and in the future.

According to the last edition (i.e., 26th edition released on November 2005) of the TOP500 list of the world’s fastest supercomputers [228] (which is released twice a year), the DOE’s IBM BlueGene/L system with 131,072 processors, installed at DOE’s Lawrence Livermore National Laboratory (LLNL) in 2005, is at the No. 1 position. It has achieved a record Linpack benchmark (solving linear system of equations) performance of 280.6 Tflop/s (“teraflops” or trillion floating-point operations per second), which is still the
only system ever to exceed the 100 Tflop/s mark. It also occupied the No. 1 position on the last two TOP500 lists. Whereas, the Earth Simulator with 5120 processors, built by NEC in 2002, which had held the No. 1 position for five consecutive TOP500 lists before being replaced by BlueGene/L, is now at the No. 7 spot with a Linpack performance of 35.86 Tflop/s. In the 26th edition of the TOP500 list, the entry level for the TOP10 exceeds 20 Tflop/s and the entry point for the top 100 moved from 2.026 Tflop/s one year ago to 3.98 Tflop/s. There are 360 systems labelled as clusters, which are connected mostly using Gigabit Ethernet (249 cluster systems) or Myricom’s Myrinet (70 systems), making this the most common architecture in the list.

The characteristics of the available computers used for the CFD simulations of this thesis are presented in Section 3.4. In addition, an interactive and scalable computational steering approach, which is discussed in Chapter 4 and also in References [143,144], has to be adopted for large-scale computations such as these turbulent flow simulations.

1.3 Objectives

The major goal of this research is to investigate unsteady, separated flows around complex engineering geometries through the use of Computational Fluid Dynamics. Different flow problems are studied with increasing computational complexity as shown in Figure 1.4. For all these unsteady simulations, three-dimensional, cell-centered, finite-volume flow solver, PUMA2, is used with unstructured grids. The use of unstructured grids helps to efficiently discretize the flow field around real engineering systems which have complex geometries by allowing efficient cell distribution. These flow problems and associated simulations with their specific objectives are given below:
1) The first problem with minimal computational complexity is an unsteady simulation around a complex ship geometry. This problem has minimal computational complexity because the simulation can be performed as inviscid, due to the fact that the sharp edges of the ships boundaries and the superstructures fix the separation points of the flow, making it reasonably independent of the Reynolds number. Therefore, the objectives of this study can be listed as:

- To simulate the complex time-varying, separated flow around a full-scale LHA ship using inviscid flow approximation.

- To capture the unsteady development of the massively separated flow from sharp edges of blunt bodies, and To investigate the complex flow features that are critical for helicopter shipboard operations.

- To collect long records of time-varying velocity data within the ship airwake.
- To couple the collected velocity time histories with dynamic interface helicopter flight simulation.

2) The second and third problems are an inviscid simulation of a helicopter rotor in hover and a simulation of the flow field around a pitching wing using Navier-Stokes equations with no turbulence modeling (MILES approach), respectively. The rotor simulation is again inviscid as the LHA case described above, but now with an added level of complexity, that is the moving grid. The pitching wing involves the moving grid capability and has the added complexity that the flow is not anymore approximated as being inviscid. The objectives of these test studies are:

- To implement the moving grid capability to PUMA2.

- Test the moving grid capability by simulating the complex unsteady flow field of a hovering helicopter rotor.

- To capture the vortical wake flow field of hovering rotor with reasonable accuracy and compare with experimental data.

- To test the moving grid capability of PUMA2 in solving a viscous flow problem with a pitching wing simulation.

- To use these test cases as sample applications to implement computational steering and monitoring tool POSSE to PUMA2 to improve the handling of large-scale computations using a complete scalable CFD system. As the computational complexity of the selected problems continue to increase, the (three-dimensional and time-dependent) CFD cases become very large in terms of grid size and total simulation time. Therefore, there is a need for such scalable monitoring, steering and visualization systems for
very large-scale parallel CFD simulations because the CFD solution will not fit on a workstation.

3) The final problem is the simulation of the flow field around a wind turbine rotor. This case again involves the moving grid capability, and the additional computational complexity is to be able to solve this problem using Large Eddy Simulations. This approach could be used in future studies as a first-principles based prediction of both noise and underlying turbulent flow field of wind turbine applications. The objectives of this part are:

- As a starting point, to obtain inviscid simulations of the wind turbine rotor at three different operating conditions.

- To implement an instantaneous log-law wall model for simulating the boundary layer effects in the near wall regions, to be used in LES simulations, and To test this log-law wall model with LES simulations for a high Reynolds number flow around a sphere.

- To obtain LES results for the wind turbine rotor flow field using the log-law wall model and compare the results with available experimental data.

- To collect time history of pressure on blade surface, or of flow field data on a permeable surface embedded in the unstructured grid for aeroacoustic calculations. LES allows the representation of a wide range of scales of noise-generating eddies. Hence, LES is a valuable tool in calculating the noise sources at Reynolds numbers of engineering interest.
The outline of the thesis is as follows: Chapter 1 gives a general introduction and background on ship airwake and wind turbine flow fields as well as a brief background on turbulence modeling. Chapter 2 presents the governing equations of fluid flow for various flow assumptions. The flow solver and the numerical methods used for the CFD simulations performed are discussed in Chapter 3. An interactive and scalable computational steering approach, coupling of POSSE with PUMA2, and visualization of large-scale computations are discussed in Chapter 4. Chapter 5 presents the details and results of the inviscid simulations of ship airwake. Sample preliminary simulations to verify the code modifications for rotating blades and turbulence are presented in Chapter 6. Chapter 7 presents the inviscid and LES simulations of wind turbine rotor flow fields. Discussions on the coupling of the CFD results with the helicopter simulations, dynamics and control, and the wind turbine aeroacoustics calculations are also presented in Chapters 5 and 7, respectively. Finally, Chapter 8 presents the conclusions.
Chapter 2

Governing Fluid Dynamics Equations

The governing equations of fluid flow are represented by the conservation laws of mass (continuity), momentum (Newton’s second law), and energy (first law of thermodynamics), which are obtained by assuming the fluid as a continuum, and are called Navier-Stokes equations.

The Navier-Stokes (N-S) equations for viscous flows, the governing equations for Large Eddy Simulations (LES) for spatially filtered viscous flows, the Reynolds-averaged Navier-Stokes equations (RANS) for ensemble-averaged viscous flows, and the Euler equations for inviscid flows are presented in this chapter to show the differences between the equations, the flow variables that need to be solved, and the modeling requirements. For easy comparison, all governing equations are given in column vector format, which is also suitable for CFD analysis and programming. Some of the turbulence models (especially the basic models) that are needed for the closure of the governing equations are also presented.

In the Introduction (Chapter 1), discussions of the various modeling and simulation methodologies for turbulent flows have already been presented. In this chapter, the governing equations, including the RANS equations although not used in this thesis, are presented for a more clear and complete mathematical comparison. The flow solver and the numerical methods used in this thesis are discussed in the next chapter 3.
2.1 Navier-Stokes Equations

The governing equations for three-dimensional, unsteady, compressible, viscous fluid flows are the Navier-Stokes equations, which consist of the continuity, momentum and energy equations. The vector form of this system of equations are given below in conservative, integral form for a space volume $V$ bounded by a surface $S$:

$$\frac{\partial}{\partial t} \int_V Q dV + \oint_S (F \cdot \mathbf{n}) dS - \oint_S (F_v \cdot \mathbf{n}) dS = 0$$

(2.1)

$$Q = \begin{cases} 
\rho \\
\rho u_1 \\
\rho u_2 \\
\rho u_3 \\
\rho e_0 
\end{cases}, \quad F_j = \begin{cases} 
\rho u_j \\
\rho u_1 u_j + p \delta_{1j} \\
\rho u_2 u_j + p \delta_{2j} \\
\rho u_3 u_j + p \delta_{3j} \\
\rho h_0 u_j 
\end{cases}, \quad F_{vj} = \begin{cases} 
0 \\
\tau_{1j} \\
\tau_{2j} \\
\tau_{3j} \\
\tau_{ij} - q_j 
\end{cases}$$

where, $Q$ contains the flow field variables in conservative form, $F$ is the inviscid flux vector including pressure variations, and $F_v$ is the viscous flux vector. The index $i, j = 1, 2, 3$ refers to the components in each coordinate direction. The flow field variables defined by $\rho, u_i, p, e_0$ are density, Cartesian velocity components, pressure, and total energy, respectively. The total enthalpy $h_0$ is related to total energy $e_0$ by:

$$h_0 = e_0 + \frac{p}{\rho}$$

(2.2)
together with the following relations:

\[ e_0 = e + \frac{1}{2} u_i u_i, \quad h_0 = h + \frac{1}{2} u_i u_i, \quad h = e + \frac{p}{\rho} \]

where, \( e \) is the internal energy, \( h \) is the enthalpy.

The viscous stress tensor, \( \tau_{ij} \), in the viscous flux vector is a linear function of the velocity gradients for a Newtonian fluid defined by

\[ \tau_{ij} = 2\mu S_{ij} + \lambda \frac{\partial u_k}{\partial x_k} \delta_{ij} \]  \hspace{1cm} (2.3)

where, \( \delta_{ij} \) is the Kronecker delta, and \( S_{ij} \) is the strain-rate tensor:

\[ S_{ij} = \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \]  \hspace{1cm} (2.4)

The coefficient of dynamic viscosity (molecular viscosity), \( \mu \), and the second coefficient of viscosity, \( \lambda \), are related as: \( \kappa = \frac{2}{3} \mu + \lambda \), where \( \kappa \) is the coefficient of bulk viscosity.

Kinematic viscosity is defined as \( \nu = \mu/\rho \). For incompressible flows, or for low Mach number flows, it is usually assumed that \( \kappa = 0 \) and \( \lambda = -\frac{2}{3} \mu \) by Stoke’s hypothesis.

Using Sutherland’s law, the dynamic viscosity is given as a function of temperature, \( T \), as:

\[ \mu = C_1 (T^{3/2})/(T + C_2) \]

where two constants are known for a given gas (e.g., \( C_1 = 1.45 \times 10^{-6} \text{ kg/(msK}^{1/2}) \) and \( C_2 = 110 \text{ K for air and } T \text{ is in degrees Kelvin} \). The heat flux term, \( q_j \), in the energy equation is given by the heat flux in the \( j \)-direction.
by thermal conduction (Fourier law):

\[ q_j = -k \frac{\partial T}{\partial x_j} \quad (2.5) \]

where, \( k \) is the coefficient of thermal conductivity which is related to dynamic viscosity through the Prandtl number: \( Pr = \mu C_p/k \) where \( C_p \) is the specific heat coefficient at constant pressure. In the N-S equations above, body forces and volumetric heating are neglected.

In order to close the system of equations, (5 equations: continuity equation, momentum equations and energy equation, with 6 unknowns: \( (\rho, u_i, T, p) \)), it is necessary to establish relations between the thermodynamic variables \( (p, \rho, T, e, h) \). In aerodynamics, it is assumed that the air behaves as an ideal gas with constant specific heat coefficients (a perfect gas). Then, the perfect gas equation of state \( (p = \rho RT) \) gives the pressure as

\[ p = (\gamma - 1)\rho e = (\gamma - 1)\rho (e_0 - \frac{1}{2}u_iu_i) \quad (2.6) \]

with the use of following relations which exist for a perfect gas:

\[ e = C_v T, \quad h = C_p T, \quad \gamma = \frac{C_p}{C_v}, \quad C_v = \frac{R}{\gamma - 1}, \quad C_p = \frac{\gamma R}{\gamma - 1} \]

where, \( \gamma \) is the ratio of specific heats, and \( R \) is the gas constant. For air at standard conditions, \( R = 287.04 \, m^2/(s^2 \, K) \), \( \gamma = 1.4 \), and \( Pr = 0.72 \).
For a viscous flow, the physical boundary condition at the solid surface (at the wall) is the no slip boundary condition that assumes zero velocity at the wall:

\[ u_1 = u_2 = u_3 = 0 \]

The Navier-Stokes equations are a coupled system of nonlinear equations, and hence are very difficult to solve analytically. To date, there is no general closed-form solution to these equations, except for few simplified specific cases.

### 2.1.1 Navier-Stokes Equations for Moving Solid Geometries

The Navier-Stokes Equations for fluid flow over a moving solid geometry in an inertial reference frame as described in [230] are given by:

\[
\frac{\partial}{\partial t} \int_V Q \, dV + \int_S (F \cdot n) \, dS - \int_S (F_v \cdot n) \, dS = 0
\]  

(2.7)

\[
Q = \begin{cases} 
\rho, & \rho u_1, \\
\rho u_2, & \rho u_3, \\
\rho e_0 & \rho h_0(u_j - b_j) 
\end{cases}, \quad F_j = \begin{cases} 
\rho(u_j - b_j), & \rho u_1(u_j - b_j) + p \delta_{1j}, \\
\rho u_2(u_j - b_j) + p \delta_{2j}, & \rho u_3(u_j - b_j) + p \delta_{3j}, \\
0 & \tau_{ij} - q_j
\end{cases}, \quad F_{v_j} = \begin{cases} 
0, & \tau_{1j}, \\
\tau_{2j}, & \tau_{3j}
\end{cases}
\]
where the inviscid flux term, \( F \), contains the effect of the moving body, through the body/grid velocity \( b(b_1, b_2, b_3) \), while the \( U(u_1, u_2, u_3) \) is the absolute flow velocity with respect to the inertial reference frame.

In this case, for a viscous flow over moving solid geometry, the no slip boundary condition assumes zero relative velocity at the wall:

\[
\begin{align*}
    u_1 - b_1 &= u_2 - b_2 = u_3 - b_3 = 0
\end{align*}
\]

### 2.1.2 Navier-Stokes Equations with Source Terms

The Navier-Stokes Equations with additional momentum source terms can be written as:

\[
\frac{\partial}{\partial t} \int_V QdV + \oint_S (F \cdot n)dS - \oint_S (F_v \cdot n)dS - \int_V sdV = 0 \tag{2.8}
\]

where the column source vector \( s \) may include the time-dependent and spatially varying (distributed) 3-D force per unit volume terms \( f(t, x) \) in the momentum equations and the corresponding work terms \( f(t, x) \cdot u \) in the energy equation as:

\[
s = \begin{cases} 
0 \\
f_1 \\
f_2 \\
f_3 \\
f_i u_i
\end{cases}
\]
The momentum sources can be body force terms such as gravity forces $\rho g$, or any forces depending on the problem considered.

2.1.3 Euler Equations

The inviscid Navier-Stokes equations (when $Re \to \infty$) are defined as the Euler equations, where viscous and heat flux terms in the momentum and energy equations in Equation 2.1 become zero (negligible). Thus, the integral form of the Euler Equations is:

$$\frac{\partial}{\partial t} \int_V Q \, dV + \int_S (F \cdot n) \, dS = 0 \quad (2.9)$$

and together with the perfect gas equation (Equation 2.6), becomes a closed system of equations.

For an inviscid flow, the physical boundary condition at the surface is the tangency condition which states that the flow velocity vector at the wall must be tangent to the wall, i.e. no mass flow normal to the wall:

$$u \cdot n = 0$$

On the other hand, for an inviscid moving body problem, the tangency condition is given for the relative velocity between the flow and the moving body as:

$$(u - b) \cdot n = 0$$
2.2 Governing Equations for Large Eddy Simulations

The governing equations for Large Eddy Simulations are obtained by spatially filtering the Navier-Stokes equations by using a spatially uniform filter function $G$ which commutes with temporal and spatial derivatives:

$$\overline{\phi}(x, t) = \int \phi(x', t) G_\Delta(x - x') dx'$$  \hspace{1cm} (2.10)

As a result of the filtering procedure, the equations are written in terms of the resolved flow variables (resolved scales), $\overline{\phi}$, while the effects of the unresolved scales are represented through the additional terms in the viscous flux vector $F_v$. These additional terms, that appear with the superscript R below, are the residual, or the subgrid-scale (SGS), terms which need to be modeled.

For incompressible flow, the LES equations are:

$$\frac{\partial}{\partial t} \int_V Q dV + \int_S (\mathbf{F} \cdot \mathbf{n}) dS - \int_S (\mathbf{F}_v \cdot \mathbf{n}) dS = 0$$  \hspace{1cm} (2.11)
\[
Q = \begin{bmatrix}
\bar{p} \\
\bar{p} \bar{u}_1 \\
\bar{p} \bar{u}_2 \\
\bar{p} \bar{u}_3 \\
\bar{p} \bar{v}_0
\end{bmatrix},
F_j = \begin{bmatrix}
\bar{p} \bar{u}_j \\
\bar{p} \bar{u}_1 \bar{u}_j + \bar{p} \delta_{1j} \\
\bar{p} \bar{u}_2 \bar{u}_j + \bar{p} \delta_{2j} \\
\bar{p} \bar{u}_3 \bar{u}_j + \bar{p} \delta_{3j} \\
\bar{p} \bar{v}_0 \bar{u}_j
\end{bmatrix},
\]

\[
F_{vj} = \begin{bmatrix}
0 \\
\tau_{1j} - \tau^{R}_{1j} \\
\tau_{2j} - \tau^{R}_{2j} \\
\tau_{3j} - \tau^{R}_{3j} \\
\bar{u}_i \tau_{ij} - \bar{q}_j + \Theta^{R}_j
\end{bmatrix},
\]

\[
\Theta^{R}_j = w^{R}_j - q^{R}_j - k^{R}_j
\]

where, \(\tau_{ij}\) and \(\bar{q}_j\) are the viscous stress tensor and the heat flux vector for the resolved scales, respectively. The SGS terms that require modeling are:

\[
\tau^{R}_{ij} = \bar{p} (\bar{u}_i \bar{u}_j - \bar{u}_i \bar{u}_j)
\]

\[
w^{R}_j = \bar{u}_i \tau_{ij} - \bar{u}_i \tau_{ij}
\]

\[
q^{R}_j = C_p \bar{p} (\bar{T} \bar{u}_j - \bar{T} \bar{u}_j)
\]

\[
k^{R}_j = \frac{1}{2} \bar{p} (\bar{u}_i \bar{u}_j \bar{u}_j - \bar{u}_i \bar{u}_i \bar{u}_j)
\]
where $\tau_{ij}^R$ is the subgrid-scale stresses, and $w_j^R$ is the subgrid viscous work. $q_j^R$ is the subgrid heat fluxes. $k_j^R$ is the convection of subgrid kinetic energy by the subgrid velocity, also called the triple velocity correlation term.

For compressible flow, the LES equations are:

$$\frac{\partial}{\partial t} \int_V Q dV + \oint_S (\mathbf{F} \cdot \mathbf{n}) dS - \oint_S (\mathbf{F}_v \cdot \mathbf{n}) dS - \frac{\partial}{\partial t} \int_V M dV = 0 \quad (2.16)$$

$$Q = \begin{cases} \bar{p} \\ \bar{p} \bar{u}_1 \\ \bar{p} \bar{u}_2 \\ \bar{p} \bar{u}_3 \\ \bar{p} \bar{v}_0 \end{cases}, \quad F_j = \begin{cases} \bar{p} \bar{u}_j \\ \bar{p} \bar{u}_1 \bar{u}_j + \bar{p} \delta_{1j} \\ \bar{p} \bar{u}_2 \bar{u}_j + \bar{p} \delta_{2j} \\ \bar{p} \bar{u}_3 \bar{u}_j + \bar{p} \delta_{3j} \\ \bar{p} \bar{h}_0 \bar{u}_j \end{cases}$$

$$F_{vj} = \begin{cases} -m_j^R \\ \tau_{1j} - \tau_{1j}^R \\ \tau_{2j} - \tau_{2j}^R \\ \tau_{3j} - \tau_{3j}^R \\ \bar{u}_i \bar{v}_{ij} - \bar{q}_j + \Theta_j^R \end{cases}, \quad M = \begin{cases} 0 \\ -m_1^R \\ -m_2^R \\ -m_3^R \\ 0 \end{cases}$$

$$\Theta_j^R = w_j^R - q_j^R - k_j^R$$

where the SGS terms are:

$$m_j^R = \bar{p} \bar{u}_j - \bar{p} \bar{u}_j$$ 

(2.17)
Here, $m^R_j$ is the subgrid mass flux which shows up as a diffusion term in the continuity equation, and as an unsteady source term in the momentum equation. 

For compressible flow, by filtering the N-S equations and also by using the density-weighted (Favre) averaging, $\tilde{\phi} = \frac{\rho \phi}{\rho}$, the additional subgrid mass flux terms disappear and the equations become similar to the incompressible LES equations but in terms of the Favre-averaged flow variables:

$$\frac{\partial}{\partial t} \int_V Q dV + \oint_S (\mathbf{F} \cdot \mathbf{n}) dS - \oint_S (\mathbf{F}_v \cdot \mathbf{n}) dS = 0 \quad (2.22)$$
\[ Q = \begin{array}{c}
\bar{p} \\
\bar{p} \tilde{u}_1 \\
\bar{p} \tilde{u}_2 \\
\bar{p} \tilde{u}_3 \\
\bar{p} \tilde{e}_0
\end{array}, \quad F_j = \begin{array}{c}
\bar{p} \tilde{u}_j \\
\bar{p} \tilde{u}_1 \tilde{u}_j + \bar{p} \delta_{1j} \\
\bar{p} \tilde{u}_2 \tilde{u}_j + \bar{p} \delta_{2j} \\
\bar{p} \tilde{u}_3 \tilde{u}_j + \bar{p} \delta_{3j} \\
\bar{p} \tilde{h}_0 \tilde{u}_j
\end{array}, \quad F_{v,j} = \begin{array}{c}
0 \\
\tilde{\tau}_{1j} - \tau_{1j}^R \\
\tilde{\tau}_{2j} - \tau_{2j}^R \\
\tilde{\tau}_{3j} - \tau_{3j}^R \\
\tilde{u}_i \tilde{\tau}_{ij} - \tilde{q}_j + \Theta_j^R
\end{array}, \quad \Theta_j^R = w_j^R - q_j^R - k_j^R
\]

where the SGS terms are:
\[ \tau_{ij}^R = \rho \tilde{u}_i \tilde{u}_j - \bar{p} \tilde{u}_i \tilde{u}_j \]  
\[ w_j^R = \tilde{u}_i \tilde{\tau}_{ij} - \tilde{u}_i \tilde{\tau}_{ij} \]  
\[ q_j^R = C_p (\rho \tilde{T} \tilde{u}_j - \bar{p} \tilde{T} \tilde{u}_j) \]  
\[ k_j^R = \frac{1}{2} (\rho \tilde{u}_i \tilde{u}_j - \bar{p} \tilde{u}_i \tilde{u}_j) \]

For the incompressible and compressible LES equations, the constitutive relations for pressure, enthalpy, energy, heat fluxes, and viscous stresses are same as for the
N-S equations, but they are functions of resolved flow variables: \((\overline{p}, \overline{u}_i, \overline{p}, \overline{e}_0, \overline{h}_0, \overline{T})\) or \((\overline{p}, \tilde{u}_i, \overline{p}, \tilde{e}_0, \tilde{h}_0, \tilde{T})\).

In the literature, there have been numerous efforts for modeling the SGS terms. The SGS stresses can be modeled using the classical Smagorinsky model which is an eddy-viscosity model described below, or other models such as dynamic Smagorinsky and similarity models [169]. The subgrid heat flux, and the subgrid mass flux for non-Favre averaged compressible LES equations are the terms that need to be modeled. On the other hand, the subgrid viscous work is expected to be small and therefore can be neglected [18] [147]. Also, the triple velocity correlation term is usually assumed to be small [18].

2.2.1 Smagorinsky model

In the classical Smagorinsky model [179], which is an eddy-viscosity model, the SGS stresses and the eddy viscosity \(\mu_T\) are defined as

\[
\tau_{ij}^R = -2\mu_T \overline{S}_{ij}
\]

\[
\mu_T = (C_S\Delta)^2 \overline{p} \sqrt{2\overline{S}_{ij}\overline{S}_{ij}}
\]

where \(C_S\) is a model parameter (Smagorinsky constant) and is typically assigned a value of 0.1 to 0.25. \(\Delta\) is the filter width which is typically a function of the grid resolution. An average \(\Delta\) is often used for nonuniform grids. \(\overline{S}_{ij}\) is the resolved strain-rate tensor.
The model for the SGS heat fluxes can be defined as

$$ q_j^R = \frac{\gamma R \mu_T}{\gamma - 1 Pr_T} \frac{\partial \bar{T}}{\partial x_j} \quad (2.29) $$

where $Pr_T$ is the turbulent Prandtl number and is commonly chosen in the range of 0.3 to 0.5.

For wall bounded flows it is necessary to damp the eddy-viscosity using an appropriate wall damping function since the model parameter does not decrease to zero at the walls.

### 2.3 Reynolds Averaged Navier-Stokes Equations

The Reynolds Averaged Navier-Stokes (RANS) equations are obtained first by decomposing the flow variables into mean and fluctuating parts as $\phi = \bar{\phi} + \phi'$, and then time averaging the equations by

$$ \bar{\phi} = \lim_{T \to \infty} \frac{1}{T} \int_{t}^{t+T} \phi(x, t) dt \quad (2.30) $$

and by using the following relations: $\bar{\phi} = \bar{\phi}$ and $\bar{\phi}' = 0$.

$$ \frac{\partial}{\partial t} \int_V Q dV + \oint_S (\mathbf{F} \cdot \mathbf{n}) dS - \oint_S (\mathbf{F}_v \cdot \mathbf{n}) dS - \frac{\partial}{\partial t} \int_V M dV = 0 \quad (2.31) $$
\[ Q = \begin{cases} \bar{p} \\ \bar{p} \bar{u}_1 \\ \bar{p} \bar{u}_2 \\ \bar{p} \bar{u}_3 \\ \bar{p} \tilde{e}_0 + (\rho' e' + k) \end{cases}, \quad F_j = \begin{cases} \bar{p} \bar{u}_j + \rho' u'_j \\ \bar{p} \bar{u}_1 \bar{u}_j + \bar{p} \delta_{1j} + \bar{u}_1 \rho' u'_j + \rho' u'_1 \bar{u}_j \\ \bar{p} \bar{u}_2 \bar{u}_j + \bar{p} \delta_{2j} + \bar{u}_2 \rho' u'_j + \rho' u'_2 \bar{u}_j \\ \bar{p} \bar{u}_3 \bar{u}_j + \bar{p} \delta_{3j} + \bar{u}_3 \rho' u'_j + \rho' u'_3 \bar{u}_j \\ \bar{p} \bar{u}_0 \bar{u}_j + e_0 \rho' u'_j + (\rho' e' + k) \bar{u}_j \end{cases}, \quad F_{v_j} = \begin{cases} 0 \\ \tau_{1j} - \tau_{1j}^T \\ \tau_{2j} - \tau_{2j}^T \\ \tau_{3j} - \tau_{3j}^T \\ \bar{u}_i \tau_{ij} - \bar{q}_j + \Theta^T_j \end{cases}, \quad M = \begin{cases} 0 \\ -\rho' u'_1 \\ -\rho' u'_2 \\ -\rho' u'_3 \\ -\bar{u}_i \rho' u'_i \end{cases} \]

\[ \Theta^T_j = w_j^T - q_j^T - k_j^T - E_j^T \]

\[ m_j^T = \rho' u'_j \] (2.32)

\[ \tau_{ij}^T = \rho u'_i u'_j \] (2.33)

\[ w_j^T = -\bar{u}_i \tau_{ij}^T + u'_i \tau_{ij}^T \] (2.34)

\[ q_j^T = \rho h' u'_j \] (2.35)

\[ k_j^T = \frac{1}{2} \rho u'_i u'_i \] (2.36)

\[ E_j^T = \bar{u}_i \rho' u'_i \bar{u}_j \] (2.37)
where, \( \tau_{ij} \) and \( \bar{q}_j \) are the viscous stress tensor and the heat flux vector for the time-averaged flow, respectively. In the above equations, \( \tau_{ij}^T \) and \( q_j^T \) are the Reynolds stress tensor and turbulent heat flux vector, respectively, that come from the inviscid momentum fluxes due to turbulent fluctuations. Other additional terms consist of the density-velocity correlations, and triple velocity correlations of the turbulent fluctuations. \( k \) is defined as the kinetic energy of the turbulent fluctuations and shows up as their contributions in the energy equation. All these turbulent terms need to be modeled in order to close the system of equations. Since these equations are very complex and it is difficult to model all the terms, the Favre-averaged compressible RANS equations are usually preferred in the literature.

Using the Favre-averaging, \( \dot{\phi} = \bar{\rho}\dot{\phi}/\bar{\rho} \), and using the decomposition as \( \phi = \dot{\phi} + \phi'' \) with the relations: \( \bar{\dot{\phi}} = \bar{\dot{\phi}} \) and \( \bar{\rho}\dot{\phi}'' = 0 \), the compressible Favre-averaged RANS equations are obtained as:

\[
\frac{\partial}{\partial t} \int_V Q dV + \int_S (\mathbf{F} \cdot \mathbf{n}) dS - \int_S (\mathbf{F}_v \cdot \mathbf{n}) dS = 0
\] (2.39)
\[ Q = \begin{pmatrix} \bar{p} \\ \bar{p} \tilde{u}_1 \\ \bar{p} \tilde{u}_2 \\ \bar{p} \tilde{u}_3 \\ \bar{p} \tilde{e}_0 + k \end{pmatrix}, \quad F_j = \begin{pmatrix} \bar{p} \tilde{u}_j \\ \bar{p} \tilde{u}_1 \tilde{u}_j + \bar{p} \delta_{1j} \\ \bar{p} \tilde{u}_2 \tilde{u}_j + \bar{p} \delta_{2j} \\ \bar{p} \tilde{u}_3 \tilde{u}_j + \bar{p} \delta_{3j} \\ \bar{p} \tilde{h}_0 \tilde{u}_j + k \tilde{u}_j \end{pmatrix}, \]

\[ F_{vj} = \begin{pmatrix} 0 \\ \tilde{r}_{1j} - \tau_{1j} \\ \tilde{r}_{2j} - \tau_{2j} \\ \tilde{r}_{3j} - \tau_{3j} \\ \tilde{u}_i \tilde{r}_{ij} - \tilde{q}_j + \Theta^T_j \end{pmatrix} \]

\[ \Theta^T_j = w^T_j - q^T_j - k^T_j \]

\[ \tau^T_{ij} = \rho u''_i u''_j \]  

\[ w^T_j = -u_i \tau^T_{ij} + u''_i \tau^T_{ij} \]  

\[ q^T_j = \rho h'' u''_j \]  

\[ k^T_j = \frac{1}{2} \rho u''_i u''_j \]  

\[ k = \frac{1}{2} \rho u''_i u''_i \]

For incompressible flow \( \phi' = 0 \) and the difference between the density-averaged and conventional variables vanishes.
The extra terms generated due to the Reynolds averaging process must be modeled in order to close the RANS equation system. This is achieved through a variety of turbulence models, such as algebraic, one equation or two-equation models, which utilize the Boussinesq hypothesis that the stress is proportional the mean strain rate, or the Reynolds Stress Models (RSM) in which the full set of Reynolds stress transport equations have to be solved. More detailed information on turbulence modeling can be found in References [244] [36].
Chapter 3

Flow Solver: PUMA2

The flow solver PUMA2 is a modified version of the computational fluid dynamics (CFD) solver PUMA (Parallel Unstructured Maritime Aerodynamics) [26]. Penn-State researchers have been refining and developing this code since 1997 [139,140,181,187,188, 194,203].

PUMA2 uses a finite volume formulation to solve the integral form of the Euler / Navier-Stokes equations for 3-D, compressible, unsteady or steady-state flows over complex geometries using unstructured grids. Mixed topology unstructured grids composed of tetrahedra, wedges, pyramids and hexahedra are supported. Several time integration and iterative algorithms such as Runge-Kutta, Jacobi and Successive Over-Relaxation (SOR) schemes are available in PUMA2. PUMA2 can be run so as to preserve time accuracy or as a pseudo-unsteady formulation to enhance convergence to steady state. The code is written in ANSI C/C++ using the MPI library for message passing so it can run on parallel computers and clusters. It uses dynamic memory allocation, thus the problem size is limited only by the amount of memory available on the machine. There are several types of boundary conditions available in PUMA2, and more can be added without much difficulty as and when needed. A sample input file for PUMA2 is given in Appendix A.
PUMA2 solves the steady/unsteady governing equations (which have been presented in the previous chapter) on unstructured stationary or moving grids (i.e., rotation about an axis). The moving grid capability, which is discussed in Section 3.1, is implemented in PUMA2 to compute flow around rotating blades.

Large Eddy Simulations (LES) with or without wall models can be performed with PUMA2. Previously, Large Eddy Simulations were performed with a constant Smagorinsky model using very coarse unstructured grids and without using wall models by Souliez [203] and Souliez et al. [204]. A wall model based on an instantaneous logarithmic-law of the wall, which has been implemented in PUMA2 during this study, is discussed in Section 3.2. This wall model has been initially tested with the LES simulation over a sphere by Jindal et al. [95,96], and this test case is also discussed in Section 6.3.

The parallel computations for this thesis are performed mostly on the PC clusters available at The Pennsylvania State University. These clusters and other supercomputers used are described below in Section 3.4. PUMA2 is also compatible with C++ compilers and is coupled with the computational steering and monitoring library, POSSE [138]. The coupling of PUMA2 with POSSE, and the use of the computational steering and visualization for large-scale CFD simulations (time-dependent and three-dimensional) are discussed in Chapter 4.

3.0.1 Finite Volume Formulation

In the finite volume formulation, the integral expressions for conservation of mass, momentum and energy over an arbitrary control volume are solved directly rather than
first being transformed to differential form. In the finite volume formulation, the physical domain is divided into many small control volumes (cells) of arbitrary shape. In this discretization, for a constant control volume, the governing equations 2.1 become

\[
V_i \frac{d}{dt} Q_i + \sum_{j=1}^{N\text{faces}(i)} [(F \cdot nS)_j - (F_v \cdot nS)_j]_i = 0 \quad i = 1, N\text{cells} \quad (3.1)
\]

where \( Q_i \) represents the volume average of the flow variables over the cell \( i \) with volume \( V_i \), and the net inviscid and viscous fluxes is calculated by the summation of the average flux through each face \( j \) of cell \( i \) with face area \( S \).

PUMA2 solves the governing equations in conservation law form, but only the primitive variables are stored in permanent memory. This requires the use of the transformation matrix \( M \):

\[
M = \frac{\partial Q}{\partial q}
\]

\[
Q = \begin{bmatrix} \rho & \rho u & \rho v & \rho w & \rho e_0 \end{bmatrix}^T
\]

\[
q = \begin{bmatrix} \rho & u & v & w & p \end{bmatrix}^T
\]

\[
M = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
\rho & 0 & 0 & 0 & 0 \\
v & 0 & \rho & 0 & 0 \\
w & 0 & 0 & \rho & 0 \\
\frac{u^2 + v^2 + w^2}{2} & \rho u & \rho v & \rho w & \frac{1}{\gamma - 1}
\end{bmatrix}
\]
One of the tasks in using finite volume methods for three dimensional simulations is the evaluation of the cell volume and the area of associated cell surfaces. The details of the cell volume and surface area calculations used in PUMA2 based on the unstructured grids (tetrahedral or hexahedral cells) can be found in Reference [188].

### 3.0.2 Runge-Kutta Time Integration Scheme

Runge-Kutta time integration methods are widely used in numerical solutions of flow fields. The most commonly used Runge-Kutta method is the classical fourth order Runge-Kutta method [94] given as:

\[
Q^1 = Q^n + \frac{\Delta t}{2V_i}R^{n}
\]

\[
Q^2 = Q^n + \frac{\Delta t}{2V_i}R^1
\]

\[
Q^3 = Q^n + \frac{\Delta t}{V_i}R^2
\]

\[
Q^{n+1} = Q^n + \frac{\Delta t}{6V_i}(R^n + 2R^1 + 2R^2 + R^3)
\]

(3.2)

where \( R \) is the residual vector. Every time step is divided into four sub-steps and the right hand side of the governing equation is evaluated four times per time step. This algorithm is fourth-order accurate. However, this method requires all four intermediate solutions in the final step, which is undesirable for large-scale calculations. Figure 3.1 shows a representation of the classical Runge-Kutta method for a model equation \( \frac{dy}{dt} = f(t, y) \).

One of the time integration algorithms of PUMA2 used in this study is the explicit cell-
Fig. 3.1. A representation of the classical Runge-Kutta method for a model equation $\frac{dy}{dt} = f(t,y)$. [188]

centered m-Stage Jameson-style Runge-Kutta algorithm. Unlike the classical Runge-Kutta, this algorithm is only second order accurate in time, but it provides enhanced convergence speed. The algorithm is convenient to program and no intermediate solution needs to be stored. Thus, this style has advantages over the classical form in terms of having less number of operations in the final step and its reduced storage requirements.

\[
\begin{align*}
Q^1 &= Q^n + \frac{\Delta t}{4V_i} R^n \\
Q^2 &= Q^n + \frac{\Delta t}{3V_i} R^1 \\
Q^3 &= Q^n + \frac{\Delta t}{2V_i} R^2 \\
Q^{n+1} &= Q^n + \frac{\Delta t}{V_i} R^3 
\end{align*}
\] (3.3)
Since there is no basic theorem such as Lax’s theorem that one can use to assess the numerical characteristics of the finite volume methods, one has to find the finite difference counterpart of the finite volume method in order to analyze the numerical stability and accuracy of the scheme.

For the classical fourth-order accurate Runge-Kutta scheme, the stability requirement [221] is defined by the Courant-Freidrichs-Lewy (CFL) number as

\[ CFL = \frac{|U| + a}{\Delta x} \Delta t \leq 2\sqrt{2} \]

For the three-dimensional calculations of this study, minimum cell length is used (as \( \Delta x \)) in the CFL number and time step calculations. For unsteady computations, CFL is taken to be less than 1. In addition, for stability of viscous computations, the Von Neumann condition also needs to be satisfied:

\[ VN = \frac{(\nu + \nu_t)\Delta t}{(\Delta x)^2} \leq 0.5 \]

### 3.0.3 Numerical Iterative Methods

Iterative methods are often used for solving discretized partial differential equations. Iterative methods like Jacobi and various Successive Over-Relaxation (SOR) schemes are available in PUMA2 for the time-independent computations.

The Jacobi method is based on solving for every variable locally using the previous known solution of the variables. The Jacobi method is easy to understand and implement, but the convergence is slow. The Gauss-Seidel method is like the Jacobi method, except
that it uses updated values as soon as they are available. In general, if the Jacobi method converges, the Gauss-Seidel method will converge faster than the Jacobi method, though still relatively slowly. The Successive Over-Relaxation method is derived by applying an extrapolation to the Gauss-Seidel method. The extrapolation takes the form of a weighted average between the solution of previous iteration and the solution by Gauss-Seidel iteration by using an extrapolation parameter $\omega$. A value for $\omega$ between 0 and 2 is usually chosen such that the rate of convergence of the iterations to the solution would accelerate. For the optimal choice of $\omega$, SOR may converge faster than Gauss-Seidel by an order of magnitude. If $\omega = 1$, the SOR method simplifies to the Gauss-Seidel method. The symmetric successive over-relaxation (SSOR) method is a forward SOR sweep followed by a backward SOR sweep.

The implicit time marching methods such as these iterative methods allow for much larger CFL numbers to be used when fast convergence to a steady state is needed. However, these algorithms are much more expensive to use in terms of memory required. Detailed descriptions and implementation of these schemes can be found in References [26,203].

3.0.4 Roe’s Flux Difference Splitting Scheme

For the cell-centered finite volume algorithm used in PUMA2, the fluxes on each cell face are obtained using the Roe’s Flux Difference Splitting technique [26,176,221]. Roe’s Flux Difference Splitting scheme [176] is an upwind differencing scheme which takes advantage of the hyperbolicity in the Euler equations. In an upwind formulation for the Euler equations, there are two states on either side of a face ($Q_R$ and $Q_L$) which
must be resolved into a single value for the flux through the face combining the fluxes computed individually from the two states. Roe’s scheme computes an exact solution to an approximate Riemann problem for flux calculations at cell faces in the finite volume approach. The problem of computing the cell face fluxes is viewed as a series of one-dimensional Riemann problems along the direction normal to the control volume faces.

The Roe averaged flux vector at each face is given by

$$ F = \frac{1}{2} [F_L + F_R - [\tilde{A}] (Q_R - Q_L)] $$

(3.4)

where $[\tilde{A}]$ is Jacobian matrix $[A] = \frac{\partial F}{\partial Q}$ evaluated with Roe averaged variables:

$$ \hat{\rho} = R \rho_L = \sqrt{\rho_L \rho_R} $$

$$ \hat{\mathbf{V}} = \frac{\mathbf{V}_L + \mathbf{V}_R R}{1 + R} $$

$$ \hat{h}_0 = \frac{h_{0L} + h_{0R} R}{1 + R} $$

$$ a^2 = (\gamma - 1) \left( \hat{h}_0 - \frac{1}{2} \hat{\mathbf{V}} \cdot \hat{\mathbf{V}} \right) $$

(3.5)

where

$$ R = \sqrt{\rho_R / \rho_L} $$

Equation 3.4 for 1-D Riemann problem in the normal direction can be written as

$$ \mathbf{F} \cdot \mathbf{n} = \frac{1}{2} [\mathbf{F}_L \cdot \mathbf{n} + \mathbf{F}_R \cdot \mathbf{n} - \sum_{i=1}^{3} \hat{\lambda}_i \alpha_i \hat{r}_i] $$
where \( \hat{\lambda}_i = [ \hat{V}_n \quad \hat{V}_n + \hat{a} \quad \hat{V}_n - \hat{a} ]^T \) are the eigenvalues (wave speeds) (\( \hat{V}_n \) is the Roe averaged normal speed) and \( \hat{r}_i \) are the eigenvectors of the Roe matrix \([\hat{A}]\), and \( \alpha_i \) are the coefficients (wave strengths).

### 3.1 Moving Grid Capability

As a part of this study, PUMA2 is modified to solve the governing equations for moving body problems. This moving/rotating grid capability can be used to obtain time-accurate simulations of flow fields around rotating bodies, such as helicopter or wind turbine rotor blades. This is achieved through the use of additional metric terms in the equation that incorporate grid velocities into the flux terms. The flow field is solved directly in the inertial reference frame where the solid body and entire grid are in motion together at a specified rotational speed (i.e., rotation about an axis) through any freestream. Thus, in this formulation, the relative velocities through the cell faces are used in the flux calculations. In addition, in the Roe’s flux calculations, the eigenvalues of the Roe averaged Jacobian matrix include the body/grid velocities \( \mathbf{b} \) as

\[
\hat{\lambda}_i = [ \hat{V}_n - b_n \quad \hat{V}_n - b_n + \hat{a} \quad \hat{V}_n - b_n - \hat{a} ]^T
\]

In the time-accurate moving grid simulations, the solution at each time step is updated with an explicit algorithm that uses a 4-stage Runge-Kutta scheme. Therefore, the grid has to be moved four times per time step to conform to the instantaneous position of the moving body, and it is required to recalculate only the grid velocities at each face center and the face normals for the specified grid motion at each stage.
In addition, the boundary conditions need to use the relative velocities by taking the body/grid velocities into consideration, and the time-step is calculated using the new eigenvalues of the new Roe averaged matrix for the moving grid computations [188].

Simulation of flow fields around two or more solid geometries which can move also with respect to each other, such as flapping or pitching blades, would require the deformation of the grid. This kind of simulation can be achieved by the use of overset grids [35] or adaptive/deforming meshes [161]. Such grids are not supported in PUMA2, however, may be considered in future.

3.2 A Log-law Wall Model

Along the course of this study, a wall model based on logarithmic law of the wall is implemented in PUMA2 [96]. For high Reynolds number turbulent flows, the log-law approach can be used at the first cell away from the solid surface to get the shear stress at the wall instantaneously.

The law of the wall for a boundary layer profile is shown in Figure 3.2 [226]. The logarithmic law of the wall, which is obtained by neglecting all terms in the streamwise momentum equation except the Reynolds-stress gradient, defines the logarithmic velocity profile in the inertial sublayer of the boundary layer profile (called logarithmic law region). The logarithmic law of the wall can be expressed as:

\[
{u}^+ = \frac{\| V_{tangential} \|}{u_r} = \frac{1}{\kappa} \log(y^+) + B.
\]  
(3.6)
where, $y^+ = hu_\tau / \nu$ is the distance to the wall in viscous units, and $\nu$ is the kinematic viscosity. From experiments for a flat plate at higher Reynolds number, the constants $\kappa$ and $B$ are determined to be 0.41 and 5.0 respectively. The profile given in the equation is solved by Newton’s method to obtain the frictional velocity $u_\tau$ which is then used to calculate shear stress as

$$\tau_w = u_\tau^2 \rho.$$  \hfill (3.7)

In the coordinate system of tangential velocity, normal velocity, and their cross product (see Figure 3.3) the wall shear stress tensor $\tau_{mn}$ then can be written as:

$$\begin{pmatrix}
0 & \tau_w & 0 \\
\tau_w & 0 & 0 \\
0 & 0 & 0 \\
\end{pmatrix}$$  \hfill (3.8)

The wall shear stress tensor is then transformed to the inertial coordinate system $(x, y, z)$ using the orthogonal transformation:

$$\tau' = C^T \cdot \tau_{mn} \cdot C$$  \hfill (3.9)
Fig. 3.2. The law of the wall [226].

\[ \frac{U}{U_p} = \text{constant} \]
\[ \frac{U}{U_p} = 2.5 \ln \frac{y^+}{v} + 5 \]

Fig. 3.3. Velocity components at the first cell center away from the wall.

\[ \mathbf{V}_{n} = (\mathbf{V} \cdot \mathbf{n}) \mathbf{n} \]
\[ \mathbf{V}_{t} = \mathbf{V} - \mathbf{V}_{n} \]
\[ h = \mathbf{r} \cdot \mathbf{n} \]
where the rotation matrix $C$ is defined by,

$$
\begin{pmatrix}
    e_{tangent}[i] & e_{tangent}[j] & e_{tangent}[k] \\
    e_{normal}[i] & e_{normal}[j] & e_{normal}[k] \\
    e_{cross}[i] & e_{cross}[j] & e_{cross}[k]
\end{pmatrix}
$$

(3.10)

where $e_{tangent}$, $e_{normal}$ and $e_{cross}$ are unit normal vectors in the tangential velocity, normal velocity and their cross product directions respectively. These shear stress components are then fed back to the outer LES model in the form of proper momentum flux at the wall.

### 3.3 Parallelization in PUMA2

While many different approaches to programming on parallel computers have been proposed over the years, the dominant approach today is to use Fortran or C/C++ with the message-passing interface (MPI). This approach is fairly straightforward for regular grids using domain decomposition, but is more difficult for unstructured grids or implicit codes.

PUMA2 is written in ANSI C/C++ using the Message Passing Interface (MPI) library for message passing so it can run on parallel computers and clusters. PUMA2 uses the Single Program Multiple Data (SPMD) parallelism, i.e., the same code is replicated to each process. One process acts as the master and collects the results from all other processes, thus doing slightly more work than the other processes. A partitioning
algorithm is required to decompose and distribute the unstructured computational domain across several processes. The Gibbs-Poole-Stockmeyer (GPS) algorithm [26] and the METIS Algorithm [105] are two different algorithms that can be used for domain decomposition. GPS is used to minimize communication cost (i.e., time), by having fewer messages of greater length, thus decreasing the number of neighbors per processor. This approach is suitable for parallel computing platforms such as Beowulf clusters with usually high latency. METIS can be used to minimize message length between processors, by increasing the total number of messages of smaller size, thus increasing the number of neighbors per processor. This approach works very well on low-latency computing platforms.

In PUMA2, each compute node reads its own portion of the grid file at startup. Cells are divided among the active compute nodes at runtime based on cell IDs, and only faces associated with local cells are read. Faces on the interface surface between adjacent computational domains are duplicated in both domains. Fluxes through these faces are computed in both domains. The updated flow variables are communicated between domains at every timestep during the flux calculations, which ensures that the computed solution is independent of the number of compute nodes. Communication of the solution across domains is all that is required for first-order spatial accuracy, since $Q_L$ and $Q_R$ are simply cell averages to the first order. If the left and right states are computed to higher-order, then $Q_L$ and $Q_R$ are shared explicitly with all adjacent domains. The fluxes through each face are then computed in each domain to obtain the residual for each local cell.
3.4 Parallel Computers

All the simulations for this study were performed in parallel, mostly on the Linux Beowulf clusters available at The Pennsylvania State University, and also on some of the supercomputers such as NREL, NCSA and NASA clusters. Table 3.1 summarizes the maximum number of compute nodes available, the processor speed, memory and the network connection characteristics of these clusters.

The Linux Beowulf clusters [211] at the The Pennsylvania State University which are mainly used to perform the computations for this study are: COSt effective Computing Array (COCOA, COCOA2, COCOA3), LION-XL and MUFASA. The parallel clusters: COCOA [145] had 25 nodes with dual 400-MHz Pentium-II processors and 512 MB of Random Access Memory (RAM), and COCOA2 [146] had 20 nodes with dual 800-MHz Pentium-III processors and 1 GB of memory. For both clusters, the nodes were connected via Fast-Ethernet (100 Mbit/sec). COCOA3 [40], which was built in 2003, is another parallel Beowulf cluster consisting of 60 nodes with dual 2-GHz Intel Xeon processors. Each node has 2 GB of RAM and the nodes are interconnected by a Gigabit Ethernet network. LION-XL [124] has 128 compute nodes with dual 2.4-GHz Intel P4 processors, 4 GB ECC RAM and Quadrics high-speed interconnect. MUFASA [87] has 85 computational nodes with dual MP2200+ MHz Athlon processors and 1 GB RAM. The nodes are connected both by Ethernet and a high-speed Scali Dolphin network.
Table 3.1. Characteristics of the several parallel computers used for the computations during this study.

<table>
<thead>
<tr>
<th>Computer</th>
<th># of compute nodes</th>
<th>processors</th>
<th>memory</th>
<th>network</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSU COCOA2</td>
<td>20</td>
<td>Dual 800 MHz Intel Pentium III</td>
<td>1 GB of RAM</td>
<td>Ethernet Network</td>
</tr>
<tr>
<td>PSU COCOA3</td>
<td>60</td>
<td>Dual 2 GHz Intel Xeon</td>
<td>2 GB of RAM</td>
<td>Gigabit Ethernet Network</td>
</tr>
<tr>
<td>PSU Lion-xl</td>
<td>128</td>
<td>Dual 2.4 GHz Intel P4</td>
<td>4 GB of ECC RAM</td>
<td>Quadrics high-speed interconnect</td>
</tr>
<tr>
<td>PSU Lion-xm</td>
<td>80</td>
<td>Dual 2.4 GHz AMD Opteron</td>
<td>8 GB of ECC RAM</td>
<td>Force 10 6500 Series Gigabit Ethernet Switch</td>
</tr>
<tr>
<td>PSU Mifasa</td>
<td>168</td>
<td>Dual 3.06 GHz Intel Xeon or Dual 3.2 GHz Intel Xeon</td>
<td>4 GB of ECC RAM</td>
<td>High Speed Myrinet Network</td>
</tr>
<tr>
<td>NREL Lester</td>
<td>81</td>
<td>Dual 2.8 GHz AMD Athlon MP2000+</td>
<td>2 GB Memory (10 nodes) &amp; 1 GB Memory (71 nodes), 4 GB memory</td>
<td>Dothing and fast ethernet networks</td>
</tr>
<tr>
<td>NCSA Tungsten</td>
<td>1280</td>
<td>Dual 3.2 GHz Intel Xeon</td>
<td>3 GB ECC DDR SDRAM memory</td>
<td>Gigabit Ethernet and Myrinet 2000</td>
</tr>
<tr>
<td>NASA/NAS</td>
<td>20 SGI Altix 3700 superclusters, each with 512 processors,</td>
<td>Dual 1.5 GHz Intel Itanium 2</td>
<td>Global shared memory across 512 processors, 1 Gbyte of memory per 512 processors</td>
<td>Interconnect SGI NUMAlink InfiniBand network, 10 gigabit Ethernet</td>
</tr>
</tbody>
</table>
Chapter 4

Computational Steering and Visualization

The development, integration, and testing of a general-purpose computational steering software library with a three-dimensional Navier-Stokes flow solver is described in this Chapter. For this purpose, the Portable Object-oriented Scientific Steering Environment (POSSE) library was used, which can be coupled to any C/C++ simulation code [128,138,139,143,144,191]. This chapter illustrates how to integrate computational steering into a code (such as the CFD flow solver PUMA2), how to monitor the solution while it is being computed, and how to adjust the parameters of the algorithm and simulation during execution. The simulations typically run on a parallel computer, whereas the monitoring and visualization are performed both on the parallel machine and on other computers through a client/server approach. A key advantage of this interactive CFD system is its scalability. Visualization primitives are generated on the parallel computer. This is essential for large-scale simulations, since it is often not possible to post-process the entire flow field on a single computer due to memory and speed constraints. In addition, the visualizations can be displayed using virtual reality (stereographics) facilities (such as CAVEs and RAVEs) to better understand the three-dimensional nature of the flow fields. Therefore, scalable interactive computational steering and monitoring systems are essential. Some of the time-dependent and three-dimensional, large-scale CFD simulations performed in this thesis and performed by other Penn-State researchers are
also discussed here to show the usefulness of POSSE and virtual reality systems. The examples include CFD predictions of flow fields of a ship airwake, a sphere, a landing gear, and a helicopter rotor. The advantages of using object-oriented programming are also discussed.

4.1 Scalable Computational Steering for Visualization/Control of Large-Scale Fluid Dynamics Simulations

Parallel simulations now play an important role in all areas of science and engineering (fluid dynamics, electromagnetics, structural dynamics, materials science, etc.). Fluid dynamics simulations for realistic aerospace engineering applications often require the computations of complex three-dimensional, unsteady, separated and turbulent flows. As the applications for these simulations expand, the demand for their flexibility and utility grows. Interactive computational steering is one way to increase the utility of these high-performance simulations because they facilitate the process of scientific discovery by allowing the scientists to interact with their data. On yet another front, the rapidly increasing power of computers and hardware rendering systems has motivated the creation of visually rich and perceptually realistic Virtual Environment (VE) applications. The combination of the two provides one of the most realistic and powerful simulation tools available to the scientific community.

Whereas a tremendous amount of work has gone into developing parallel Computational Fluid Dynamics (CFD) software, little has been done until the past few years to develop parallel computational steering tools that can be integrated with such parallel CFD software. Without such computational steering systems, the immediate and direct
interaction with simulations is impossible. Instead, visualization of these simulations is mainly limited to the post-processing of data. However, for large-scale, parallel aerospace CFD simulations, it is often not possible to post-process the entire flow field on a single computer due to memory and speed constraints. Large parallel supercomputers can store thousands of times more data than a typical workstation. Also, the interaction with the large-scale data, which is especially important at the early stages of the computations to tune the several flow and input parameters, becomes very difficult and time consuming for such large-scale simulations. Therefore, interactive parallel (and scalable) computational steering and monitoring systems are necessary to perform efficiently large-scale, parallel aerospace CFD simulations. The system described here allows the steering, visualization, and CFD to all be run in a scalable manner. For example, the majority of the work required for the visualization is done in parallel with essentially no communication required. In addition, the computational steering can be a means to make simulations appear more like experiments. That is, one can probe the simulation in arbitrary ways, in real time, in a manner similar to that of an experimentalist. This represents a shift from the old post-processing way of doing large-scale simulations that will make simulations infinitely more useful.

As described in Section 1.2.5, LES, Reynolds Stress methods, or hybrid methods will be essential to do realistic aerospace simulations, now and in the future. Therefore, an interactive and scalable computational steering approach (such as POSSE) has to be adopted for large-scale computations such as these turbulent flow simulations.
4.2 The Computational Monitoring and Steering Library: POSSE

The computational monitoring and steering library POSSE [138] is written in C++ by Modi [139], by the use of advanced object-oriented features, making it powerful while maintaining the ease-of-use by hiding most of the complexities from the user. It has been initially written for the visualization of a wake-vortex simulation of several aircraft in real time [139,142], where the simulations are performed using a vortex-panel method. There is a need for such scalable visualization systems also for very large-scale CFD simulations (time dependent and three-dimensional) because the CFD solution will not fit on a workstation. As a part of this thesis, the parallel flow solver PUMA2 is coupled with the POSSE library. Specifically, the integration of the library into the solver, i.e. instrumenting PUMA2 by defining the steering parameters and steering actions to add steering and monitoring functionality, was performed. Furthermore, guidance and implementation support were provided from a CFD code user perspective in the development of the user interface and other tools such as the iso-surface extraction routine for the PUMA2+POSSE system. The main objective is to obtain “a scalable CFD system” (PUMA2+POSSE) for steering and visualization of large-scale parallel simulations. The technical details of this system are given in this chapter and in [143,144]. Advantages of using this system are discussed and are demonstrated through various CFD test cases here in this chapter as well as in [128,143,144,191].

The POSSE library allows a simulation running on any parallel or serial computer to be monitored and steered remotely from any machine on the network using a simple cross-platform client utility. This library has been used to augment the parallel flow
solver Parallel Unstructured Maritime Aerodynamics-2 (PUMA2), which is written in C/C++ using the message passing interface (MPI) library [160], to obtain an interactive CFD system. This system is being successfully used to monitor and steer several large flow simulations over helicopter and ship geometries, thus providing the user with a fast and simple debugging and analysis mechanism, where the flow and convergence parameters can be changed dynamically without having to stop or restart the simulation. This CFD system, which primarily runs on an in-house Beowulf cluster, the cost-effective computing array-2 (COCOA-2) [127, 141, 146], has been coupled to our Virtual Reality (VR) system, Fakespace RAVE [53], to obtain near real-time visualization of the three-dimensional solution data in stereographic mode. This ability to become immersed in the complex flow solution as it unfolds, using the depth cue of the stereoscopic display and the real-time nature of the computational steering system, opens a whole new dimension to the engineers and scientists interacting with their simulations.

While running a complex parallel flow solver on a high performance computing system, one often experiences several major difficulties in observing computed results. Usually, the simulation severely limits the interaction with the program during the execution and makes the visualization and monitoring slow and cumbersome (if at all possible), especially if it needs to be carried out on a different system, for example, on a specialized graphics workstation for visualization.

For CFD simulations, it is important for the surface contours of flow variables to be computed instantaneously and sent to the visualization client for the user to observe and take appropriate action. This activity is referred to as monitoring, which is defined as the observation of a programs behavior at specified intervals of time during its execution.
On the other hand, the flow variables and/or solver parameters may need to be modified as the solution progresses. Thus, there is a need to modify the simulation based on these factors by manipulating some key characteristics of its algorithm. This activity is referred to as steering, which is defined as the modification of a program's behavior during its execution.

Software tools that support these activities are called computational steering environments. These environments typically operate in three phases: instrumentation, monitoring, and steering. Instrumentation is the phase where the application code is modified to add monitoring functionality. The monitoring phase requires the program to run with some initial input data, the output of which is observed by retrieving important data about the program's state change. Analysis of these data gives more knowledge about the program's activity. During the steering phase, the user modifies the program's behavior (by modifying the input) based on the knowledge gained during the previous phase by applying steering commands, which are injected online, so that the application need not be stopped and restarted.

A significant amount of work has been done on computational steering systems over the past few years. Some of the well-known steering systems are Falcon [71] from the Georgia Institute of Technology, SCIRun [162] from the Scientific Computing and Imaging Research Group at University of Utah, ALICE Memory Snooper [10] from the Argonne National Laboratory, the visualization and application steering environment (VASE) [93] from the University of Illinois, CUMULVS [62] from Oak Ridge National Laboratory, the computational steering environment (CSE) [235] from the Center for
Mathematics and Computer Science in Amsterdam, pV3 [77] from the Massachusetts Institute of Technology, Virtue [193] from the University of Illinois at UrbanaChampaign, and COVISE [41,170] from the University of Stuttgart. A summary of the characteristics of these systems is listed in Table 4.1. These visualization and steering systems provide visualization techniques for displaying the applications input and output data (model exploration). VASE and SCIRun also allow algorithm experimentation by modifying a running programs algorithm and structure. Falcon and Virtue provide only performance optimization for a running application program by changing computational resources. Some of them have a suitable graphical user interface (GUI) for steering and data visualization, and most are based on a client/server model for data extraction and parameter access. The application source code has to be modified manually with program statements (at user-defined breakpoints) by the application developer for all systems except SCIRun. More detailed comparison and description of many steering systems can be found in References [28,150,163,172].

Computational steering has also been attempted by coupling of a two-dimensional solver with the VR environment by Cruz-Neira et al. [46] (also see Reference [67]) at the Electronic Visualization Laboratory (EVL), University of Illinois at Chicago, as early as 1993. They worked on very simple simulations of RayleighTaylor instability and gravitational wave components predicted by Einsteins theory of general relativity. The datasets were relatively small and calculations were simplified by approximating theoretical algorithms and decreasing rendering quality. The steering was implemented by providing a menu to change input parameters of the simulation and then restarting the simulation when a change was detected. The computation was simultaneously performed
<table>
<thead>
<tr>
<th>System</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>VISUAL3 (1991)</td>
<td>Visualization package for three-dimensional, unstructured grid, steady or unsteady data</td>
</tr>
<tr>
<td>VASE (1993)</td>
<td>Visualization through existing packages</td>
</tr>
<tr>
<td></td>
<td>Steering through textual input</td>
</tr>
<tr>
<td></td>
<td>Hybrid control flow and data flow</td>
</tr>
<tr>
<td></td>
<td>Algorithm experiments</td>
</tr>
<tr>
<td>Parallel Visual j (1994)</td>
<td>Parallel version of VISUAL3</td>
</tr>
<tr>
<td></td>
<td>For single parallel PVM simulation</td>
</tr>
<tr>
<td>SCIRun (1995)</td>
<td>Visualization through visualization nodes</td>
</tr>
<tr>
<td></td>
<td>Steering through TCL/TK user interfaces and widgets</td>
</tr>
<tr>
<td></td>
<td>Data flow with feedback loops</td>
</tr>
<tr>
<td></td>
<td>Algorithm experiments</td>
</tr>
<tr>
<td>Falcon (1995)</td>
<td>Two-dimensional visualization</td>
</tr>
<tr>
<td></td>
<td>Steering user interface</td>
</tr>
<tr>
<td></td>
<td>Only for performance optimization</td>
</tr>
<tr>
<td></td>
<td>For thread-based parallel programs, and programs for PVM.</td>
</tr>
<tr>
<td>CSE (1996)</td>
<td>Graphical editor for multidimensional datasets</td>
</tr>
<tr>
<td></td>
<td>Steering GUI</td>
</tr>
<tr>
<td></td>
<td>Client/server</td>
</tr>
<tr>
<td>Collaborative visualization</td>
<td>Interactive collaborative postprocessing</td>
</tr>
<tr>
<td>and simulation environment</td>
<td></td>
</tr>
<tr>
<td>(1993)</td>
<td></td>
</tr>
<tr>
<td>COVISE Virtual Environment (1997)</td>
<td>Visualization user interface such as AVS or SCIRun</td>
</tr>
<tr>
<td></td>
<td>COVER: A virtual reality rendering module, IRIS performer</td>
</tr>
<tr>
<td>CUMULVS (1997)</td>
<td>Interactive visualization through AVS</td>
</tr>
<tr>
<td></td>
<td>Computational steering through textual input and custom TCL/TK GUI</td>
</tr>
<tr>
<td></td>
<td>Client/server</td>
</tr>
<tr>
<td></td>
<td>For single parallel, C, or FORTRAN, PVM simulation</td>
</tr>
<tr>
<td></td>
<td>Performance optimization</td>
</tr>
<tr>
<td>ALICE (1999)</td>
<td>Steering GUI</td>
</tr>
<tr>
<td>Virtue (1999)</td>
<td>MATLAB® interface</td>
</tr>
<tr>
<td>Portable object-oriented</td>
<td>Interactive VR visualization</td>
</tr>
<tr>
<td>scientific steering</td>
<td>Only for performance optimization</td>
</tr>
<tr>
<td>environment (2002)</td>
<td>Steering and visualization</td>
</tr>
<tr>
<td></td>
<td>GUI through cross-platform FLTK, API, and VTK, coupled with Tecplot</td>
</tr>
<tr>
<td></td>
<td>Client/server</td>
</tr>
<tr>
<td></td>
<td>For single C/C++, MPI simulation</td>
</tr>
</tbody>
</table>

Table 4.1. Summary of existing steering and visualization systems. [144]
on a separate machine that was networked with a cave automatic virtual environment (CAVE) [47].

Although all of these software packages are powerful, the major drawback of these systems is that they are often too complex, are not object-oriented, and have a steep learning curve. It is not easy to couple these systems with existing scientific codes. It may take a significant amount of time to be productive with these systems, especially for noncomputer scientists. To address these problems, a new lightweight and portable computational steering system based on a client/server programming model has been developed.

The steering software, POSSE, is very general in nature and is based on a simple client/server model. It uses an approach similar to Falcon [71] (an online monitoring and steering toolkit) and ALICE Memory Snooper [10] (an application programming interface designed to help in writing computational steering, monitoring and debugging tools). Falcon was one of the first systems to use the idea of threads and shared memory to serve program data efficiently. POSSE consists of a steering server on the target machine that performs steering and a steering client that provides the user interface and control facilities remotely. The steering server is created as a separate execution thread of the application to which local monitors forward only those registered data (desired program variables, arrays, and/or structures) that are of interest to steering activities. A steering client receives the application run-time information from the application, displays the information to the user, accepts steering commands from the user, and enacts changes that affect the applications execution. Communication between a steering client and server are done via UNIX sockets, and threading is done using portable operating system
Fig. 4.1. Schematic of POSSE. [144]
interface standard (POSIX) threads. POSSE has been written completely in C++, using several of C++s advanced object-oriented features, making it fast and powerful, while hiding most of the complexities from the user. Figure 4.1 shows a schematic of how POSSE can be used. As seen in this Figure, an ongoing scientific simulation is running on a remote Beowulf computing cluster. Any number of remote clients can query/steer registered data simultaneously from the simulation via the DataServer thread. Two clients are shown, a visualization client and a GUI client that provides a simple user interface to all registered simulation data. The visualization code can be used to monitor interactively a dataset at various time intervals, and the GUI code can be used to steer the simulation by changing certain parameters associated with it.

POSSE runs on all Win32 and POSIX compliant UNIX platforms. It deals with byte-ordering and byte-alignment problems internally and also provides an easy way to handle user-defined classes and data structures. It is also multithreaded, supporting several clients simultaneously. It can also be easily incorporated into parallel simulations based on the MPI [160] library. The biggest enhancement of POSSE over existing steering systems is that it is equally powerful, yet extremely easy to use, making augmentation of any existing C/C++ simulation code possible in a matter of hours. It makes extensive use of C++ classes, templates, and polymorphism to keep the user application programming interface (API) elegant and simple to use. Because of its efficient design, POSSE has low computational overhead (averaging less than 1% relative to the computation thread), making it lightweight.

Figure 4.2 shows a simple, yet complete, POSSE client/server programs in C++. As seen in this Figure, registered data on the steering server (marked readwrite) are
protected using binary semaphores when they are being updated in the computational
code. User-defined data structures are handled by a simple user-supplied pack and un-
pack subroutine that calls the POSSE data-packing functions to tackle the byte-ordering
and byte-alignment issues. The programmer need not know anything about the internals
of threads, sockets, or networking to use POSSE effectively.

4.2.1 Coupling of PUMA2 and POSSE

To achieve the interactivity with POSSE, several modifications were made to the
PUMA2 code [143, 144]. The POSSE server component, DataServerMPI, was added
to the main() function of PUMA2. This was done by registering the cell-centered flow
vector \([\rho, u, v, w, p]\) and various important flow parameters in the code. Several new
global variables were added to receive isosurface requests and store resulting isosurfaces.
An isosurface extraction routine also had to be added to PUMA2 [143, 144]. Because
of the use of unstructured mesh data consisting of tetrahedra, a variation of the classic
marching cubes algorithm [129] is used for isosurface extraction. The implementation
is closely related to the marching cube algorithm except that the fundamental sampling
structure here is a tetrahedron [72] instead of a cube. Because this implementation uses
the flow data at the nodes of every tetrahedron, a subroutine to interpolate the flow data
from cell centers to the nodes was also added to PUMA2.

Isosurfaces are important and useful visualization primitives. This tool is rou-
tinely used to visualize quantities such as Mach number, vorticity, and pressure. An
isosurface routine has been written in C++ and coupled to PUMA2 to give real-time
isosurfaces of pressure, Mach number, entropy, etc. These surfaces can be displayed on
Fig. 4.2. Simple, complete POSSE (a) client and (b) server applications, written in C++. [144]
standard monitors or on the RAVE using OpenGL and stereographics. The body/surface grid and the isosurface can also be colored according to some flow variable, for example, pressure. In addition, these isosurfaces are computed in parallel so that the approach is scalable to literally billions of grid points. The drawing of the isosurfaces is performed locally on the client machine, and it does not affect the speed of the simulation being performed on the server or Beowulf cluster.

To compute an isosurface representation, the following steps are performed:

1. Cell-centered flow variables are extrapolated to tetrahedral corner points.

2. For each edge of each tetrahedron, if the value of the isosurface is between the value on the two nodes, then the linear interpolation of the nodes is performed to obtain the location of the isosurface.

3. If the surface cuts through the tetrahedron, then a triangle is obtained (two triangles in some cases).

This approach is scalable to very large grids and thousands of processors. Visualization of the flow field with isosurfaces and surface contours helps users to interact qualitatively with their simulations. It is also possible to get quantitative results in real time with routines coupled with the flow solver.

4.2.2 POSSE GUI Client

A GUI client is used to connect to the computational steering server. Figure 4.3 show a screenshot from the POSSE GUI client application showing PUMA2 registered data. The selected simulation data has to be registered, and the desired actions that
will take place after a change in each of these data have to be defined in the flow solver by the user before the simulation is started. The registered data can be selected, for example, as the freestream flow variables (velocity components, pressure, density, etc.), the parameters for numerical flux and integration schemes, Courant-Friedrichs-Lewy (CFL) number, relaxation parameters, etc., as seen in Figure 4.3. Other solver parameters can also be registered, such as input and output file names, convergence parameters, maximum number of iterations, and a parameter used for writing the flow solution into a file at every specified number of iterations.

With the POSSE GUI client, any of the registered data can be changed to steer the simulation, and/or can be used only for monitoring purposes, for example, observing the residual for the convergence of the solution for steady-state computations. The time required to obtain a converged steady-state solution is usually unknown at the beginning of the simulation. The same can be said about how often the updated solution needs to be written out for time-accurate computations. The ability to control and change the necessary parameters in such cases in the ongoing simulations, without stopping and restarting the simulations, is found to be very helpful. The use of POSSE GUI client for steering and monitoring the desired solver and flow parameters allows the user to save a great amount of time and effort in the initial stages of simulations in terms of debugging and experimenting with the input parameters and also enables access to the solution at any iteration in realtime.

The POSSE PUMA2 client is also used to extract and visualize isosurfaces. The client was written in C++ [139,143,144] with the cross-platform FLTK [227] API for the GUI and the Visualization ToolKit (VTK) [229] for the three-dimensional visualization.
Because VTK is written on top of OpenGL, the resulting application can benefit from the OpenGL hardware acceleration available on most modern graphics chipsets. VTK also supports stereographics, and can thus be used in conjunction with the RAVE [53]. Figure 4.4 shows a screenshot of the VTK output on a separate window. A drop-down menu is provided to choose the flow variable for which isosurfaces are requested. After the numerical value for the isosurface has been selected, a request is sent to the flow solver, which responds by extracting the isosurface for the given flow parameters on each of the processors, then collecting the isosurfaces on the master processor and sending the final isosurface to the client. There are two modes provided for querying isosurfaces. In the default mode, the user queries for isosurfaces while the flow-solver is computing the solution for the next iteration. This mode can be slow if the user wants to query several isosurfaces one after another because the flow solver cannot answer the client requests until the current iteration is over. For greater responsiveness, the user can enable the “query” mode, which temporarily halts the PUMA2 computations, so that the flow-solver can exclusively devote all of its CPU cycles to answering the client isosurface requests without any lag. Whereas several isosurfaces can be layered on top of each other to compare the differences between two iterations by using the default mode, different isosurfaces can be investigated at a single time step in the query mode. There is also an option “get grid” that will download the entire grid and the updated solution file and construct a Tecplot [225] volume grid file for the user to browse locally. Figures 4.4 and 4.5 show isosurfaces for a helicopter fuselage flow [203], a helicopter rotor flow [144], and a ship airwake flow [119], respectively.
Fig. 4.3. POSSE GUI to connect to the flow solver. [144]

Fig. 4.4. POSSE PUMA2 client application depicting isosurfaces for a flow solution over the Apache helicopter. [144]
4.2.3 Scalability and Dimensional Reduction

Two significant advantages arising from the use of POSSE within PUMA2 are those of scalability and dimensional reduction. For an evenly distributed grid, the number of grid faces on each processor of a parallel computation is $N_f/P$. Here, the scalability comes from the extraction of isosurfaces being performed on a parallel machine in a scalable manner rather than the traditional and nonscalable way of consolidating the data into a file and postprocessing it. Thus, the computational time for extraction of an isosurface is $O(N_f/P)$ as compared to the sequential algorithm that takes $O(N_f)$ for the same procedure. The dimensional reduction stems from the data required for the CFD simulation living in higher dimensional space (three-dimensional) than the data that are required for visualization (which are in two-dimensional and one-dimensional space for isosurfaces and sectional flow variable distributions, chord plots, respectively).
The total number of grid faces, $N_f$, is $O(n_f^4)$. On the other hand, the number of triangles in an isosurface obtained from this grid is only $O(n_f^2)$, which is significantly less data by a factor of $O(n_f)$. Scalability and dimensional reduction combine to give an expected $O(n_f^2/P)$ data coming from each processor during the parallel computation of an isosurface.

This approach is also scalable both in space and time. When a time-dependent simulation is monitored, the entire time history can be accessed, whereas it would be prohibitive to store the time history in files that could possibly take tens or even hundreds of gigabytes of disk space even for a moderately complex case. (See Reference [144] for more discussions).

4.3 Visualization

Visualization is a process of transforming raw scientific and engineering data into graphical information as images or animations in order to simplify understanding and analysis of data. The use of a computational steering software such as POSSE enables the interactive visualization and deeper understanding of complex flows over traditional post-processing approaches. Visualizations with POSSE can be displayed using virtual reality facilities (commercial high-end visualization systems such as CAVEs and RAVEs) to better understand the three-dimensional nature of the flow fields [128,191]. Complex fluid flow problems are extremely difficult to understand using conventional two-dimensional graphics. Isosurfaces in three-dimensional space can be used to highlight some flow features, but these really need to be viewed using stereographics. The iso-surface routine coupled to PUMA2 as explained in the previous sections gives real-time iso-surfaces of
flow variables such as pressure, Mach number, entropy, etc. These surfaces can be displayed on standard monitors or on the RAVE using OpenGL and stereographics. Additionally, off-the-shelf graphics packages can also be coupled with POSSE to get an advantage of their powerful features for both qualitative and quantitative visualization and analysis of large-scale CFD simulations.

4.3.1 Software for Virtual Reality Systems

The software architecture, POSSE (shown in Figure 4.1), consists of two software components. The CFD simulation (the generator of simulation data) must have the capacity to act as a data server to remote clients. These remote clients can interact with the simulation data in a passive manner (such as a visualization client) or in an active manner (such as a steering client that changes the state of the simulation as it is executing). The POSSE software can support both the data server and any number of clients.

POSSE has the ability to be integrated with a wide variety of visualization clients capable of supporting VE displays and devices. Today there is a variety of commercial and academic software packages that can be used. One of the most widely used VE display systems is the CAVE and RAVE systems marketed by FakeSpace [53], but originally developed at the EVL. The software library to support these devices is called CAVElibs and is commercially supported by VRCO Inc. [238].

Recently, a great deal of work has been done to develop more portable software systems to support VE devices and displays. These newer systems are academic, not commercially supported, but are more modular in their software implementation and
offer users a greater degree of flexibility in their use. For example, the DIVERSE is a collection of software modules that can be used to implement visualization clients, as an API, or to integrate heterogeneous VR environments [7]. In addition, VRjuggler is a software library that has been developed to support a wide variety of VE devices and has been ported to a variety of architectures [45]. VEs are also now more commonly being supported by commercial visualization systems. For example, Ensight [52] now supports stereo displays and some VE devices. This range of software options allows today’s users a much greater degree of flexibility both in terms of software sophistication and the effort required to develop a VE visualization client. Any of these VE software systems can be used in conjunction with POSSE.

4.3.2 Coupling of the System to Off-the-Shelf Visualization Software

Although it is important to compute and display isosurfaces rapidly, there are many other visualization tasks that are best performed in off-the-shelf graphics packages, such as Tecplot [225], Ensight [52], or Fieldview [58]. It is not practical to replicate the powerful features of these capable software packages in in-house codes; hence, they are used for several visualization tasks in our research group, and coupled with the computational steering system POSSE. Qualitative images are useful for debugging and steering, but often detailed quantitative images are necessary, and these are often performed best in off-the-shelf graphics software. In the CFD system (PUMA2+POSSE), all of the isosurfaces can be optionally filtered through a user-specified Tecplot layout file to create a highly customized visualization display on a separate Tecplot window. For example, for the validation of the flow solvers, the surface pressure coefficient distributions
at several locations need to be compared with the experimental measurements. This can be efficiently done using the Tecplot feature of the POSSE GUI. This is also scalable because the flow field can be processed in parallel, and only a subset of the results need to be sent to the graphics workstation. The key is to use dimensional reduction and not try to postprocess the entire solution on a workstation (which is often not feasible).

4.4 Example Flow Simulations

Example applications of time-dependent and three-dimensional, large-scale, high performance and parallel CFD simulations performed at Penn-State are discussed in this section (and in References [128,143,144,191]) to show the usefulness of POSSE and VR systems.

One of the complex flow simulations performed using the CFD solver PUMA2 is the prediction of the unsteady airwake of an LHA-class ship. The results of the steady-state and time-accurate simulations of the ship airwake, which a helicopter flight dynamics simulation model was interfaced with, are discussed in detail in Chapter 5. The accurate computation of this flow can be coupled to a helicopter flight dynamics simulation model to aid understanding potential safety problems in landing helicopters on this class of ship. POSSE was used in this simulation to observe the flow features at any intermediate iteration and also over different landing spots as the simulation is running on a Beowulf cluster. To be able to probe the flow solution at any point in the computational domain while the simulation is running on COCOA-3, new variables were registered in POSSE as “x_coord, y_coord and z_coord” as seen in Figure 4.6. These are the components of a 1-D array called “data_coord” representing the spatial coordinates.
Fig. 4.6. POSSE GUI client for ship airwake simulations [128].

of a requested point. Also, a routine was written to find the cell in which the requested point is located, and the time history of the flow solution at this cell’s center is written into an output file. The solution could also be obtained at the requested point instead of the cell center by using interpolation schemes similar to the ones used in the iso-surface extraction routine. The Figure 4.7 shows examples of the time history of the w-velocity (vertical velocity component) at two points over landing spots 7 and 8 respectively using this new capability of POSSE. As described in Chapter 5, the flow solution was stored at intervals of 0.1 seconds (every 208 iterations as shown by the symbols in Figure 4.7), however, the data probing capability gave us much more flexibility to investigate the flow solution at any point over the ship deck, and at any intermediate iteration (as shown by red line in Figure 4.7) while the simulation is running. Instantaneous isosurfaces of vorticity magnitude over the LHA ship for different oncoming wind angles are one of the ways to investigate the complex flow features such as separated vortical regions.
Fig. 4.7. Time history of $w$-velocity component at two different points (a) over landing spot 7, and (b) over landing spot 8, for the LHA ship for 30 degree yaw case [128].

on the deck and a complex island wake. For example, an instantaneous iso-surface of entropy is shown in Figure 4.8, that was extracted by the POSSE GUI client (Figure 4.6), over the LHA ship for 30 degree yaw case with 30 knots of relative wind speed. The ship surface and the iso-surface are also colored according to the pressure coefficient. However, these unsteady flow features can be better observed by creating an animation of isosurfaces in time (example animations can be found in Reference [128]). Furthermore, better observations can be made if one can move around and inside the isosurfaces freely such as with an interactive visualization in a VR system. Stereographics displays are useful to visualize and better understand the details of such complex flows. Because the CFD computations take so much wall clock time, the real-time computations of the ship airwake and interface of the airwake with a flight dynamics model in real-time, for example in a helicopter simulator for pilot training purposes, is not yet possible. On the other hand, these large-scale computations could serve as “numerical experiments” to better understand the physics of ship airwake flow fields, and to allow development of advanced flight dynamics and control models.
Another example is the time-accurate simulations of flow over a sphere (also described in Section 6.3) which were carried out at a Reynolds number of $1.14 \cdot 10^6$ with PUMA2 on a Beowulf computing cluster by Jindal et al. [96]. One of the central concerns was to understand the qualitative and quantitative unsteady properties of the wake in these simulations. To aid in this understanding active and passive stereo 3-D animations of iso-surfaces of vorticity magnitude were generated by Jindal [95] by using the computational steering library integrated with PUMA2. These stereographic animations were generated with the commercial software system Ensight [52] using its movie editor, EnVideo2 (this animation can be found in Reference [128]). An anaglyph stereo image from this flow simulation is shown in Figure 4.9. The animations proved extremely useful and confirmed the shedding of hairpin-like structures in these simulations.

Another case is the numerical simulation study of time-accurate flow and noise due to a detailed landing gear configuration using PUMA2 by Souliez et al. [204]. Both
Fig. 4.9. Red-Blue grayscale anaglyphic image of dominant hairpin vortex [96].

Fig. 4.10. Instantaneous isovorticity lines for landing gear [203].
the aerodynamic and acoustic results of this unsteady simulation are discussed in References [203, 204]. Airframe noise (especially on approach to landing) is becoming an important noise source, as fan and jet noise is reduced. The noise due to separated flow around landing gear and flaps can be significant. The landing gear configuration is an especially challenging test case for which massive flow separation occurs [127]. PUMA2 has been combined with a module, which uses the Ffowcs Williams-Hawkings (FW-H) equation with solid and permeable integration surfaces, to compute near- and far-field noise due to landing gear [204]. The time-accurate Monotone Integrated Large Eddy Simulations (MILES) were performed on the PC cluster COCOA2. Figure 4.10 from [203] shows a three-dimensional representation of some vortex filaments shed from various gear components and highlights the impact of the upstream elements wake onto gear elements at downstream locations. Flow separation from the fore wheel and wake impingement on the aft are expected to generate large unsteady pressure fluctuations and, therefore, noise. Unsteady simulations such as these are really four-dimensional (space and time) problems, where complex three-dimensional flow fields change in time. In the future, simulations like these can be studied with stereographics combined with stereo sound.

The interactive CFD system (PUMA2+POSSE) was also used in the time-accurate rotating blade (or pitching wing) simulations performed by using the rotating grid capability of PUMA2. POSSE was useful for monitoring and steering of these simulations during the code development and debugging stages. The iso-surface routine of POSSE was helpful for visualization of the unsteady complex vortical flow features of helicopter
or wind turbine rotors. The extraction of chordwise pressure distributions using the Tecplot feature allowed comparisons with the experimental data ([144]).

Hence, stereographics displays are invaluable for viewing and understanding the complex, three-dimensional nature of flows such as those in ship airwake and rotor wakes, and flows around sphere, helicopter fuselage or landing gear. Design engineers have to use these immersive systems not only to achieve specific results, but to obtain an intuitive feel for these results to enable design optimizations. The use of POSSE in such wide range of large-scale and time-accurate aerospace simulations is important and necessary because it allows interactive, real-time computational steering and visualization in a scalable way.
Chapter 5

Ship Airwake Simulations

The details and results of the computational fluid dynamics (CFD) simulations of the airwake of an LHA-class ship are discussed in this chapter. The simulations are performed by using the flow solver PUMA2 with unstructured grids on a full-scale ship model. The focus is on capturing the massively separated flow from sharp edges of blunt bodies, while ignoring the viscous effects. Unsteady Euler equations (2.9) are solved using the 4-stage Runge-Kutta time integration scheme for the time-accurate ship airwake simulations.

Previous computational and experimental ship airwake research in the literature have been reviewed in the Introduction (Section 1.1.1). Some of the results discussed in this chapter were also presented in References [117–119,192].

In addition, both the steady-state and the time-accurate CFD predictions of the inviscid LHA ship airwake simulations for 0° and 30° WOD cases were also used for the helicopter/ship dynamic interface (DI) simulations, as also described in References [117–119]. Some of the results of the DI simulations are also presented in this chapter.

5.1 Computational Model and Grid

The LHA/LHD class U.S. Navy ships, which resemble small aircraft carriers, are the largest of all amphibious warfare ships. The LHA/LHD ships are capable of
shipboard operations of Vertical/Short Take Off and Landing (V/STOL), Short Take Off Vertical Landing (STOVL) type aircraft such as AV-8B Harrier and F-35 Joint Strike Fighter (JSF), and Vertical Take Off and Landing (VTOL) tiltrotor and Rotary Wing (RW) aircraft such as the MV-22 Osprey tiltrotor, UH-1 Huey, AH-1 Super Cobra, CH-46 Sea Knight, and CH-53 Sea Stallion, SH-60 Sea Hawk helicopters. More details of the Tarawa LHA-class ship can be found from Reference [222].

Figure 5.1 shows a top view of an LHA class ship and the landing spots on the ship deck. The main features of the ship are the ship’s superstructure on the starboard side and the elevator on the port side. There are ten landing spots on the deck as shown in the figure. The landing spot 8 that is just behind the elevator is investigated first, and also presented in References [117–119], as a target spot for shipboard operations. The landing spots 2 and 7 are then considered and the results are presented here and also in Reference [192]. For complex geometries such as the LHA ship, unstructured grids are preferred because the unstructured grids can resolve the complex bodies better without excessive grid size and the grids are relatively easy to generate compared to structured grids. The grid used for the simulations was generated using Gridgen [68]. Since the
time step for time-accurate computations (using an explicit scheme) is determined by the smallest cell in the volume grid, a fairly uniform surface grid was chosen. Clustering is done all around the ship with increasing cell size towards the boundaries of the bounding box, a rectangular computational domain. Hence, the minimum cell size and maximum number of cells were selected during the grid generation process by also considering the total computational time needed for the time-accurate simulations. Although a systematic grid dependence study is not reported here, the grid size considered here, which was selected based on experience, was able to resolve the main flow features as can be seen from the results later in this chapter. In addition, this flow solver has been carefully validated on several different applications using unstructured grids [96,144,203].

The full-scale LHA geometry model used in the simulations is 250 m \( (L = 820 \text{ ft}) \) in length. The ship is approximately 36 m wide \( (W) \), extending from \(-18\) to \(+18\) m in \( Y \). The flight deck height \( (H) \) is approximately 20 m. Figure 5.2(a) shows the surface mesh of the LHA which is composed of almost uniform-size triangles. The rectangular computational domain, which is shown in Figure 5.2(b), has approximately 850K tetrahedral cells with clustering around the ship. The bottom surface of the domain is the waterline \( (Z = 0) \). The computational domain extends approximately \( 2.4 \cdot L \) both in front and behind the ship. The side outer boundaries are \( 6.8 \cdot W \) away from the center plane \( (Y = 0) \). The upper outer boundary is \( 12.4 \cdot H \) away from the waterline. The minimum cell length is about 0.04 m which gives a time step of 0.48 milliseconds for a CFL number of 0.8 (using Mach number scaling).
a) Surface mesh and geometry of the LHA ship

Fig. 5.2. The computational grid used for the LHA ship airwake simulations.
5.2 Results of CFD Simulations

Parallel, time-accurate CFD simulations of the airwake of the full-scale LHA class ship have been performed for several wind over deck (WOD) conditions: a relative uniform steady wind speed of 15.43 m/s (30 knots) and different wind directions of 0, 30, 45, 90, 270 and 315 degrees. Although the effects of the atmospheric boundary layer and atmospheric turbulence are important, they are not included due to the computational time available, but these effects can be included in future simulations. Since the simulations are performed without using any traditional turbulence model, the smallest turbulent scales that are captured in these simulations are of the order of the grid size. The flow is assumed to be inviscid in these simulations and hence the simulations do not capture boundary layer effects. It is prohibitively expensive to do time-accurate simulations for a grid that can resolve the boundary layer. The sharp edges of the ships boundaries and the superstructures fix the separation points of the flow making it reasonably independent of the Reynolds number. Therefore, the inviscid flow approximation can be used for these simulations.

The inviscid computations were performed on PC clusters COCOA-3 and LION-XL (see Section 3.4) using the flow solver PUMA2. A four-stage Runge-Kutta explicit time integration algorithm with Roe’s flux-difference scheme (see Sections 3.0.2 and 3.0.4) was used with CFL numbers of 2.5 and 0.8 for the steady and unsteady computations respectively. The pseudo steady-state computations were performed using local time stepping and initialized with the freestream conditions. The time-accurate computations were started from the pseudo steady-state solution. Since the flow is assumed to be
inviscid, a zero normal velocity (tangency) condition is imposed on the ship surface. The waterline which is the bottom face of the bounding box is also assigned the zero-normal velocity condition. A Riemann boundary condition is used on the other five faces of the bounding box to minimize any reflections into the domain. It required nearly 2000 iterations to simulate 1 second of real flow (which took approximately 1.8 hours on 12 processors of LION-XL).

The time-accurate predictions for five different WOD angles: 0, 45, 90, 270 and 315 degrees, were obtained to investigate the effect of wind direction. The simulations were performed for a total of 18 real-time seconds. At a speed of 15.43 m/s, the flow sweeps through a ship-length (250 m) in about 16.2 seconds. Hence, it is expected that a period of 18 seconds is sufficient to capture the unsteadiness in the flow. The flow properties were sampled every 0.1 seconds, and then time-averaged to obtain a “steady-state” solution. Figures 5.3 and 5.4 present the contour plots of the time averaged velocity data for different WOD angles to show the range of velocity scales of the ship airwake above the deck. Figure 5.3 shows the contours of $X$ component of velocity (the axial velocity component), on an $XY$ plane 6 meters (20 ft) above the deck. The solid lines represent the boundary of the ship below. This plane intersects the superstructure of the ship, hence, that area is left empty in Figures 5.3 and 5.4. This should not be confused with the white color representing the highest contour value in the plots. The front part of the ship, ahead of the superstructure is symmetric about the centerline. The geometric symmetry results in symmetric flow patterns as seen in Figures 5.3(b) and 5.3(c). However, the flow is very different around the superstructure and in the aft of the ship (with respect to the incident wind) for $\text{WOD} = 45^\circ$ and $315^\circ$ because of the
asymmetric location of the superstructure. For WOD = 45°, landing spot 7 is directly in the wake of the superstructure, while for WOD = 315°, it is upstream of the structure and experiences a more steady wind.

Contours of the Z component of velocity (the vertical velocity component) are shown in Figure 5.4. The Z component of velocity is important because it directly affects the angle of attack (and hence the lift) of a blade. The different landing spots experience a wide range of Z velocity for different WOD angles. The spots on the port side of the ship experience very high positive Z velocities (as high as the freestream velocity) for WOD = 315° and 270°, and those on the starboard side for WOD = 45° and 90°. A pilot has to take this into account when trying to land/takeoff a helicopter from these landing pads. Also, since a rotor diameter is very large, the spatial variation of Z velocity (and hence the lift) may result in an unbalanced loading of the rotor and an extra workload for the pilot to compensate for it. Adding to this, the gusty winds and the rolling deck motion make the task more intimidating and hazardous.

In addition, the time-accurate flow simulations were also performed for 0° and 30° WOD cases for a total of 50 seconds of real flow in a relative wind of 15.43 m/s (30 knots). The time-accurate computations were started from the pseudo steady-state solution, and the long time histories were obtained to investigate the unsteady nature of ship airwake. Figure 5.5 shows the velocity components along the centerline, 6 meters above the deck of the LHA ship for 0° and 30° WOD angles. The lines represent the time averaged solution and the symbols represent an instantaneous solution. The time average is calculated over a period of 50 seconds. A comparison of the time averaged data against the data at an instant illustrates the level of fluctuations in the flow field.
Fig. 5.3. Contours of the $X$ component of velocity (m/s) in a plane 6 meters above the LHA deck for different WOD angles. The solid lines represent the boundary of the ship geometry below. The arrows indicate the direction of wind.
Fig. 5.4. Contours of the $Z$ component of velocity (m/s) in a plane 6 meters above the LHA deck for different WOD angles. The solid lines represent the boundary of the ship geometry below.
The large spatial variation in the velocity, especially the high gradients around $X = 150$ m, may result in unsteady loading on a Harrier or V-22 operating from the ship.

The CFD velocity magnitude, $(u^2 + v^2 + w^2)^{1/2}$, and each velocity component of the PUMA2 results are also compared with the computational and the experimental data from Reference [166] in Figures 5.6 and 5.7. The distribution of the instantaneous and time-averaged velocity magnitude along the centerline, 3 m (10 ft) above the ship deck, for the 0 degree case is similar to the instantaneous CFD results of Polsky [166] as shown in Figure 5.6. The distribution of the instantaneous and time-averaged velocity components of PUMA2 results at Spot 4, at 3 m height, for $0^\circ$ WOD case, are in good agreement with the experimental data from Reference [166] as shown in Figure 5.7.

The time histories of velocity components, $(u, v, w)$, 6 m above the two landing spots (that is at $Z = 25.8$ m) for the two WOD angles are shown in Figure 5.8. The
Fig. 5.6. Comparison of the instantaneous and the time-averaged CFD velocity magnitude, \((u^2 + v^2 + w^2)^{1/2}\), with the CFD data from Reference [166] along the centerline, 3 m above the LHA ship deck for 0° WOD.

landing spot 2 (-200 m, -15 m, 25.8 m) is located in front of the deck, and the spot 7 (-90 m, -15 m, 25.8 m) is located in the wake of the superstructure of the LHA ship (for WOD = 30°). At spot 2 (Figure 5.8a), the magnitudes of the velocity components differ with wind direction and speed. For WOD = 30°, it is observed that the magnitude of the Y component of velocity at spot 2 (i.e., -11 m/s) is about 40% higher compared to the Y component of the freestream (i.e., -7.7 m/s). This is most probably due to the coherent oblique vortical structure over this spot, as observed from the vorticity iso-surfaces that will be presented later in this chapter. There is also elevated negative vertical velocity (about 1.5 m/s toward the deck) for the 30° case, whereas for 0° case the vertical velocity is positive and less than 1.0 m/s. These variations in flow field with WOD conditions may influence the loads on the helicopter rotor as well as the fuselage.

Although velocity levels change with WOD conditions over spot 2, the flow is almost steady and the fluctuations are minimal. On the other hand, the fluctuations are much larger for spot 7, especially for WOD = 30° case (Figure 5.8b). Variations in magnitude
Fig. 5.7. Comparison of the instantaneous and the time-averaged CFD velocity components with the experimental data from Reference [166] at Spot 4 (at $X = -175.87$ m) and 2.23 m above the LHA ship deck for $0^\circ$ WOD.
as large as 8 m/s for $X$ component of velocity and as large as 13 m/s for $Z$ component of velocity can be observed. These results clearly show the unsteadiness of the flow field over spot 7, which may significantly affect the flight dynamics of rotorcraft operating over this region as well as the pilot workload.

The Power Spectral Densities (PSDs) \([m^2/s^2]\) of the $X$ and $Z$ components of velocity versus frequency [Hz] are plotted in Figure 5.9. The PSDs are equal to the square of the magnitude of the Fourier Transform of the time-accurate velocity data which is calculated using the FFT function in MATLAB and a Hamming window filtering. Since the fluctuations at the two landing spots differ greatly, the scales used in plotting the PSDs for the two landing spots are different. A few discrete peaks can be observed in the spectra in the frequency range 0-1. For landing spot 2, the peak frequencies are observed around 0.1 and 0.3 Hz, and for landing spot 7, the peak frequency is around 0.5 Hz. This is in general agreement with the results of Polsky [7] where the observed frequencies are in the same range. The strong airwake velocity fluctuations about once every two seconds (0.5 Hz) may present a strong challenge to the pilot in control activities. Since the peaks in the spectra occur at very low frequencies (< 1 Hz), longer time records may be required to better resolve the spectra. In addition, pilot workload is associated with low frequency so it is important to compute longer simulations in the future. It is also possible that PSDs computed from a sufficiently long time record would show much smoother spectra. Wind tunnel experiments can generate long time histories, and those results should be used together with CFD to develop a complete understanding of ship airwake flow fields.
(a) Landing spot 2 (-200 m, -15 m, 25.8 m). Left: WOD = 0°, Right: WOD = 30°

(b) Landing spot 7 (-90 m, -15 m, 25.8 m). Left: WOD = 0°, Right: WOD = 30°

Fig. 5.8. Time history of the velocity components in the X, Y, and Z directions, 6 m above the landing spots 2 and 7 of the LHA for two WOD angles.
(a) Landing spot 2 (-200 m, -15 m, 25.6 m). Left: WOD = 0°, Right WOD = 30°

(b) Landing spot 7 (-90 m, -15 m, 25.6 m). Left: WOD = 0°, Right WOD = 30°

Fig. 5.9. Power Spectral Densities (PSDs) \([m^2/s^2]\) of the velocity components in the \(X\) and \(Z\) directions versus frequency [Hz], 6 m above the two landing spots of the LHA ship for two WOD angles. (Using Hamming window filtering.)
Fig. 5.10. Instantaneous iso-surfaces of vorticity magnitude of 1.0 s$^{-1}$ around the LHA for 0° and 30° WOD cases.
Fig. 5.11. Snapshots of instantaneous iso-surfaces of vorticity magnitude of 0.8 s\(^{-1}\) around the LHA for 0\(^\circ\) and 30\(^\circ\) WOD cases.
Fig. 5.12. Vorticity magnitude [s$^{-1}$] contours at $t = 50$ seconds at several stations along the ship for $0^\circ$, and $30^\circ$ WOD cases.
Fig. 5.13. Pressure coefficient contours on the ship surface at $t = 50$ seconds for $0^\circ$ and $30^\circ$ WOD cases.
Figure 5.10 shows the instantaneous iso-surface of vorticity magnitude of 1.0 s$^{-1}$ around the LHA for 0° and 30° WOD angles. Figure 5.12 shows the vorticity magnitude contours at t = 50 seconds at several stations corresponding to different landing spots along the ship for both WOD cases. The iso-surfaces show the turbulent eddies that are continuously formed around the sharp edges of the ship boundaries and the superstructure. These eddies convect downstream with the wind and are responsible for the unsteady fluctuations observed in the downstream region. These unsteady flow features can be better observed by creating an animation of iso-surfaces in time, and stereographics displays can be useful to visualize and better understand the details of such complex flows [143,144,191]. Figure 5.11 shows the snapshots of iso-vorticity value of 0.8, for 0° (on the left) and 30° (on the right) WOD cases, with 1 second intervals from top to bottom. The complex flow features of the airwake can be identified as deck-edge vortices, bow separation, separated vortical regions on the deck (e.g., burbles between the bow separation and the island for the 0° case), and complex island wake. For the case of WOD = 30°, much more complicated eddy structures can be observed, especially over landing spots 7 and 8 in the island wake. Furthermore, an oblique coherent vortical structure, which is shed from the bow and starboard side edge intersection, is clearly evident over landing spot 2.

Pressure coefficient contours on the ship surface at t = 50 seconds in Figure 5.13 also show the major differences of flow features between both 0° and 30° WOD cases. Low pressure areas correspond to the separated flow regions.
5.3 CFD Simulations of LPD-17 Ship Airwake

Inviscid, steady and unsteady CFD simulations of LPD-17 ship airwake for the same WOD conditions as presented for LHA above, are discussed in detail in Reference [192]. Some previous LPD-17 simulations were presented in References [194, 195]. In this section, some sample results from [192] are presented for comparison of the airwakes of LPD-17 and LHA ships.

The prominent features which mark the geometry of LPD-17 are the two masts one behind the other separated by a distance of 90 m. There are two landing spots on the deck for the shipboard operations of helicopters and V/STOL aircraft. Figure 5.14 shows the surface mesh on the LPD-17 and the bounding box in relation to the size of the ship. The ship length is 200 m, the width of the ship deck is 30 m and the maximum height of the ship is 50 m. The total number of tetrahedra in the volume grid is around 760 K.

Figure 5.15 shows the instantaneous iso-surface of vorticity magnitude of 1.0 s\(^{-1}\) around the LPD-17 for 0° and 30° WOD angles. For \(WOD = 0°\) case, two coherent deck edge vortices at the bow section are clearly visible. It can be observed that the separated vortical structures shed from the front mast impact on the rear mast. This interaction may be the reason for the observed finer eddies in the wake of the rear mast and over the flight deck. For the \(WOD = 30°\) case, the wakes of the front and rear mast get convected separately with minimal interaction as seen in Figure 5.15(b). For this case, the flow field over the flight deck is mostly dominated by the separated vortical structures shed from the starboard side of the deck and the hangar.
a) Surface mesh and geometry of the LPD-17 ship

Fig. 5.14. The computational grid used for the LPD-17 ship airwake simulations [192].
Fig. 5.15. Instantaneous iso-surfaces of vorticity magnitude of 1.0 s\(^{-1}\) around the LPD-17 for 0° and 30° WOD cases [192].
Similar to the LHA results, contour plots of the velocity components give a broad picture of the flow structure above the deck of the LPD-17 ship for different WOD angles. From the power spectra analysis of the $X$ and $Z$ velocity components, multiple peaks such as at 0.2 and 0.5 Hz for both $0^\circ$ and $30^\circ$ WOD cases at both landing spots have been observed. In addition, a good comparison with experimentally observed time averaged velocity field over the LPD-17 deck is obtained. More information can be found in Reference [192].

5.4 Dynamic Interface (DI) Simulations

Both the steady-state and the time-accurate CFD predictions of the inviscid LHA ship airwake simulations for $0^\circ$ and $30^\circ$ WOD cases were also used for the helicopter/ship dynamic interface (DI) simulations, as also described in References [117–119]. In these DI simulations, landing spot 8 was investigated as the target spot for the specified departure and approach operations for the UH-60A helicopter operating off the LHA ship. This represents the same aircraft/ship combination used in the Joint Shipboard Helicopter Integration Process (JSHIP) [178,245]. Three key elements of this DI simulations included: CFD representations of the time varying ship airwake from the PUMA2 flow solver, a high fidelity flight dynamics model of the helicopter based on the GENHEL model, and a feedback control model of the human pilot. The flight dynamics model included improved inflow modeling and gust penetration model to account for both the temporal and spatial variation of the unsteady ship airwake. The pilot control activity was simulated using an optimal control model of the human pilot for the specified approach and departure trajectories.
As explained in the previous section, the time-accurate CFD computations were started from the pseudo steady-state solution, and the long time-histories of total 50 seconds of flow were obtained for the two WOD conditions. Only the last 40 seconds of the time-accurate CFD data stored for every 0.1 seconds was used for the DI simulations. The time-accurate unstructured CFD solutions result in large quantities of time history data that need to be mapped into the DI simulation gust penetration model. Thus, for a given launch or recovery operation, the velocity data can be mapped into a rectangular grid (to allow easy table look up) and stored for only that part of the ship where the aircraft is expected to fly.

The velocity field data for a smaller dynamic interface domain which is located over the rear deck of LHA ship as shown in Figure 5.16(a) was extracted by post-processing the stored flow data. The Figure 5.16(b) illustrates the unstructured mesh on the ship surface, mesh on a $YZ$-plane crossing the landing spot 8, and the structured mesh on the vertical boundary of the DI domain closest to the superstructure of the ship. The DI mesh has $(81 \times 30 \times 23)$ grid points with 1.524 m (5 ft) equal intervals, and it is selected to have the flow data over landing spots 7 and 8. The location of the landing spot 8, which is the selected target spot for the specified approach and departure operations, is also illustrated in this figure by the cutting $YZ$-plane crossing this spot and a circular mark centered 5.182 m (17 ft) above the deck on this plane over the spot. Each CFD flow solution file was 41 MBytes in size, whereas the velocity data (a total of 400 instants of time with 0.1 seconds of interval) required in the DI simulations was only 5.2 MBytes for each instant. The time-varying velocity field data was used in the gust penetration model to find the disturbances at various locations on the helicopter as varying both
Fig. 5.16. (a) Rectangular volumetric domain of CFD data at the rear deck of LHA for DI simulations. (b) Unstructured mesh on the ship surface and on a slice at \( x = -27.432 \) m.

Spatially and temporarily in the DI simulations. Thus, the rotor inflow, fuselage forces were affected by the ship airwake, however, the ship airwake wasn’t affected by the rotor and fuselage wakes.

Figure 5.17 shows time histories of velocity components \((u, v, w)\) of CFD data at a selected point (5.182 m above the deck) in the DI mesh over landing spot 8 for 0 and 30 degree yaw cases. The velocity data at that point, which is located at \((-27.432 \text{ m, -10.668 m, 24.8412 m})\) (-90 ft, -35 ft, 81.5 ft), is extracted from the 40 seconds of real flow solutions with 0.1 seconds of intervals. The Power Spectral Densities (PSDs) of the velocity components non-dimensionalized by the square of the freestream velocity \((V_\infty = 15.43 \text{ m/s})\) versus frequency [Hz] is plotted in Figure 5.18. The power spectrum of the velocity components are calculated using the FFT function in MATLAB without any filtering. It can be seen that CFD predicts the dominant frequencies to be between 0.1-1.0 Hz, which corresponds to Strouhal numbers of 1.6-16.2 calculated using the ship
length and freestream velocity, $St = f \cdot L/V_\infty$. The power spectra of the 30° WOD case generally have higher amplitude than those of the 0° case. This is probably due to the larger and stronger vortical structures present in the wake. For the 30° WOD case, there is strong unsteadiness in the crossflow ($Y$-component) and vertical ($Z$-component) components of the velocity over this spot (Figure 5.17). Also, the fluctuations in the velocity components are much higher than those in 0° WOD case. Flow field differences between the two WOD cases and the complexity of the flow over the ship deck can also be observed by the distribution of the instantaneous velocity components along and across the ship at the selected point 5.182 m (17 ft) above the landing spot 8 in Figures 5.19 and 5.20.

The helicopter/ship dynamic interface simulation has been performed by Lee et al. [117–120] for three different airwake cases (no airwake, steady-state airwake, time-varying airwake) in 0° and 30° WOD conditions by using the CFD results discussed
Fig. 5.18. Power Spectral Densities (PSDs) of the velocity components non-dimensionalized by the square of the freestream velocity ($V_\infty = 15.43$ m/s) versus frequency [Hz], at a selected point in the DI mesh over landing spot 8 for 0° and 30° degree WOD cases. (PSDs are not filtered.)
Fig. 5.19. Distribution of the velocity components along the ship on a $XZ$-plane crossing the landing spot 8, and 17 ft above the deck, at $t = 40$ seconds for $0^\circ$ and $30^\circ$ WOD cases.

Fig. 5.20. Distribution of the velocity components across the ship on a $YZ$-plane crossing the landing spot 8, and 17 ft above the deck, at $t = 40$ seconds for $0^\circ$ and $30^\circ$ WOD cases.
above. In the DI simulations, the ship is assumed to be still with a steady-state wind of 30 knots, and the helicopter enters the DI mesh from the back surface and escapes from it through the front surface according to the calculated approach and departure trajectories. For example, Figure 5.21 shows the helicopter position with respect to the ship coordinate system for an approach case studied by Lee et al. [119]. Typical shipboard approach procedures include all actions that bring the rotorcraft from a point far away from the ship down to a point much closer to the recovery spot [178] [251]. The key parameters for defining the approach profile are the helicopter initial level flight speed, initial altitude, initial distance from the ship, and desired final altitude for stationkeeping. In this simulation, it is assumed that the helicopter approaches the ship from the port side at a 45 degree angle, and then performs a 45 degree left turn to align itself with the longitudinal axis of the ship after it crosses over the deck. This is similar to the trajectory used in the JSHIP study.

From the results of the DI simulations interfaced with the CFD flow data for the two WOD cases [119], the helicopter attitude responses (angles) after entering the DI mesh for the approach case, can be seen in Figure 5.22, with respect to the Earth-fixed coordinate frame, North-East-Down (NED), with the origin at the sea surface directly under the initial position of the helicopter. The pilot stick inputs provided by the optimal control model of the human pilot, which is effectively calculating the control inputs and aircraft attitude required to track the desired trajectory, are also shown in Figure 5.23. The conventions for these control positions are as follows: full left lateral cyclic, full forward longitudinal cyclic, full down collective pitch, and full left pedal correspond to
0%, full right lateral cyclic, full aft longitudinal cyclic, full up collective pitch, and full right pedal correspond to 100%.

It can be observed from these figures that while the steady airwake results (dashed lines) differ only slightly from the results with no airwake (dotted lines), the time-varying airwake (solid lines) results in significant oscillations when the helicopter enters the DI mesh. Compared to the case with no airwake, the differences in the helicopter attitude responses and the pilot control inputs are clearly induced due to the ship airwake. It can also be observed that the oscillations immediately after entering the DI mesh are similar for the steady and time-varying airwake. At this point, the aircraft is still moving with significant velocity so the steady gust field has a time-varying appearance to the aircraft. However, once the aircraft approaches hover, the results indicate the time-varying airwake results in larger oscillations and higher pilot control activity than the steady airwake.

In addition, the results for the 30° WOD case showed significantly larger oscillations in the aircraft attitude responses and pilot control inputs than for the 0° case, particularly when hovering over the ship deck. As discussed above, there is strong unsteadiness in the crossflow (Y-component) and vertical (Z-component) components of the velocity due to bow separation, deck-edge vortices and complex island wake for 30° WOD condition. These effects can be clearly observed from results of aircraft attitude responses and pilot control activities. Similar observations were also made for the departure case studied in Reference [119].

The results of the DI simulations [119] clearly indicated that the time-varying ship airwake, compared to the steady airwake, has a significant impact on aircraft response
and pilot control activity when the helicopter is operating in or near a hover relative to the ship deck (station-keeping) over the selected spot.
Fig. 5.21. Helicopter position (m) w.r.t. ship coordinate system for an approach case. [119]
Fig. 5.22. Helicopter attitude angles: PHI, THETA, PSI [degree] versus time [second], after entering the DI mesh for an approach case, for (a) $0^\circ$, and (b) $30^\circ$ WOD cases. [119]
Fig. 5.23. Pilot inputs: Lateral and Longitudinal cyclic, Collective pitch and Pedal inputs [%] versus time [second], after entering the DI mesh for an approach case, for (a) 0\(^{\circ}\), and (b) 30\(^{\circ}\) WOD cases. [119]
Chapter 6

Preliminary Simulations for
Rotating Blades and Turbulence

Three particular test cases are considered to test the new modifications implemented in PUMA2 which are related to moving grids and wall models.

In order to test the moving grid capability of PUMA2, which has been explained in Section 3.1, two computational test cases are performed: an inviscid simulation of a helicopter rotor in hover [143, 144] (Section 6.1), and a pitching airfoil simulation using MILES approach [149] (Section 6.2). The preliminary results of these simulations are presented here.

An instantaneous logarithmic law (log-law) of the wall is implemented in PUMA2 during this study as explained in Section 3.2. A high Reynolds number Large Eddy Simulation (LES) around a sphere has been performed to test this log-law wall model. Parallel CFD computations for this test case have been done by Jindal et al. [95, 96]. Results of these high Reynolds number simulations are presented here in Section 6.3, and more discussions for both low and high Reynolds number cases can be found in References [95,96].

In addition, the POSSE library, which is integrated into PUMA2 as described in Chapter 4 to obtain a scalable CFD system for large-scale parallel simulations, is tested and used extensively during these preliminary simulations.
6.1 Helicopter Rotor Simulations in Hover

Helicopter rotor simulations in hover have been performed by solving the inviscid Euler equations using the flow solver PUMA2 with moving grids. The computational rigid rotor model and the computational case are selected to simulate an experiment conducted by Caradonna and Tung [33] who performed these experiments as a benchmark test specifically to aid in the development of rotor performance codes. A rectangular, un twisted and untapered two bladed rotor model with NACA 0012 cross section and an aspect ratio of 6 was used in the experiments. Blade pressure measurements and tip vortex surveys (vortex strength and geometry) were obtained for a wide range of tip Mach numbers. Previously, Chen and McCroskey [37], Srinivasan and Baeder [210], Berkman et al. [16], and Boelens et al. [17] have performed computational simulations for this rotor and had good comparisons with the experimental data that were used extensively for the validation and development of rotor aerodynamics and performance codes.

The computational solid rotor geometry which was created using I-DEAS software has two rectangular blades with a NACA 0012 airfoil section and with flat tips (Figure 6.1) [188]. The blades have a chord of 1 m and radius of 6 m, 8 degrees collective pitch angle and 0.5 degrees pre-cone angle. The blade root cutout location is about 1 chord. The 3-D unstructured grids for the two bladed rotor geometry, Figure 6.2, were generated using the Gridgen software. The results for two different grids are presented. The Grid A (Figure 6.3(a)) has 1.3 million tetrahedral cells with nearly homogenous mesh distribution on the surface of rotor blades and also in the wake region. The Grid B (Figure 6.3(b)) has 3.5 million cells, clustered in the leading-edge region and in the
region of tip vortices with increasing cell size toward the outer boundaries. The cylindrical computational domain 6.2 for Grid B extends to 1 radii away from the blade tips and 2 radii down and up from the rotor disk in the vertical direction.

The rotational speed is chosen as 25 rad/s for this computational rotor model with 6 m blade radius to simulate an experimental hover case of 8 degrees pitch with 1250 RPM and tip Mach number of 0.439. The 2-stage Runge-Kutta time integration method with a CFL number of 0.9 and with Roe’s numerical flux scheme is used in the computations. The time-accurate computations for one revolution using an explicit two-stage Runge-Kutta method for Grid A took nearly 15.5 days using 16 processors on the parallel cluster COCOA2. The computations for Grid B has been performed for 6 revolutions (1.508 seconds) in parallel on MUFASA using 32 processors in 44 days.

The thrust coefficient for Grid A for the two-bladed rotor in hover has converged to the experimental value after nearly 3.4 revolutions, as shown in Figure 6.4. The spanwise pressure coefficient distribution at the quarter-chord location and the chordwise pressure
Fig. 6.2. Unstructured tetrahedral grid (Grid B)
Fig. 6.3. Unstructured mesh on the rectangular rotor blade and on the XY plane (at $Z = 0$) for two grids.

coefficient distribution at 80% spanwise station for both grids with the comparison to the experimental data are shown in Figures 6.5 (a) and (b), respectively. The pressure distributions show good agreement with the experimental values. The calculated pressure deficiencies at the leading edge are due to insufficient grid resolution.

Furthermore, the POSSE library has been coupled with PUMA2 and tested during these rotor simulations [143, 144]. Figure 6.5 shows quantitative comparisons of experimental and computational results for the rigid rotor where the plots were obtained by using the Tecplot integration feature of the POSSE GUI [144] as the parallel simulation was being performed on COCOA-2. In addition, Figure 6.6 shows qualitative images of iso-surfaces of Mach number at different simulation times obtained by the POSSE GUI. The grid clustering done in the wake affects the capturing of tip vortices in the rotor wake. Figure 6.7(a) shows an iso-surface of vorticity magnitude of value $30 \, \text{s}^{-1}$ at the end of 6th revolution for Grid B where the wake contraction can be seen. Figure 6.7(b)
Fig. 6.4. Time history of the rotor thrust coefficient (Grid A).

Fig. 6.5. Pressure coefficient distribution: a) spanwise at quarter-chord location and b) chordwise at 80% of radius.
shows the vorticity magnitude contours on $YZ$ slice at a half chord behind the rotor blade and relative streamlines with respect to the vortex cores. In this figure, the rotor blade position (along the $+Y$-axis) is shown with white solid line.

6.2 Pitching Wing Simulations

A Navier-Stokes computation without any turbulence model (i.e., a MILES approach), of a pitching wing has also been performed to test the moving grid capability of PUMA2. Experimental and computational studies of a pitching wing have been usually conducted to investigate the dynamic stall problem, which is important for helicopter and wind turbine rotors, and to allow better modeling of this complex unsteady flow phenomena during design.

When dynamic stall occurs because of the unsteady motion of an airfoil, the flow separation is delayed to a higher effective angle of attack than the static stall angle,
Fig. 6.7. Results at the end of 6th revolution for Grid B (a) An iso-surface of vorticity magnitude of 30 s$^{-1}$ (b) Vorticity magnitude contours on YZ slice at a half chord behind the rotor blade and relative streamlines with respect to the vortex cores.

giving an increase in the maximum lift. When the flow separation occurs at the leading edge of the airfoil, an unstable vortex sheds from the leading edge and then convects over the upper surface. Because of this vortex being over the airfoil surface, the lift increases. On the other hand, because of the convection of the vortex over the upper surface, the center of pressure moves aft, resulting in large nose-down pitching moments on the airfoil and large torsional loads on the blades. A sudden drop of lift and overshoots in drag and pitching moment occur after the vortex is convected into the wake, the flow over the airfoil is totally separated. In addition, large hysteresis in the aerodynamic loads causes reduced aerodynamic damping and various aeroelastic problems on the rotor. Wind-tunnel tests and numerical solutions of oscillating airfoils and wings have been studied in the literature to understand the dynamic stall behavior [51, 122].
The flow solver PUMA2 with moving grids is used to simulate an experiment by Piziali [165]. A 3-D rectangular semi-span wing with a NACA0015 airfoil section and an aspect ratio of 10 (and, a chord of 1 ft) was used in the experiments. The experiments were performed to investigate the dynamic stall characteristics of 2-D and 3-D wings (i.e., with and without the wing tip) undergoing periodic pitching motions. Instantaneous surface pressure data, and cycle-averaged lift, drag and pitching moment coefficients versus angle of attack for the experiments with different parameters are presented in detail in Reference [165]. One of the experimental cases is selected for the CFD computations in this study. For this selected case, freestream velocity is 101.77 m/s with a mean angle of attack of 10.88 degrees. The Mach number is 0.291, and the Reynolds number is $1.9560 \cdot 10^6$. The wing is undergoing a pitching motion about a mean angle of attack defined by

$$angle \ of \ attack(t) = 10.88 + 4.07\sin(Omega \cdot t) \quad (6.1)$$

where, the amplitude of the oscillation is 4.07, the frequency of the oscillation is 4.04 cycles/s, the rotational speed, Omega, is 25.384 rad/s, and the reduced frequency, k, is 0.0038.

A cylindrical grid with hexahedral cells in unstructured format was created using Gridgen (by using hyperbolic extrusion) around a NACA 0015 rectangular half wing with AR = 10 (Figure 6.8) which was created using I-DEAS software. The cells are clustered around the leading and trailing edges, and also in the direction perpendicular to the wing surface as seen in Figure 6.9. A 2-D test case is considered here, therefore,
Fig. 6.8. (a) Cylindrical unstructured grid domain around (b) NACA 0015 rectangular half wing with AR = 10.

the boundaries of the domain end at the tips of the finite-span wing. The hexahedral cells are used for this 2-D case to be able to generate a grid with homogeneous cell distribution along the finite wing span, also in the direction perpendicular to the wing along the span.

Pseudo-steady-state solution is obtained using SSOR numerical method with the CFL number of 40 (and also with CFL slope of 10) and with the under-relaxation factor of 0.8. Figure 6.11 shows the residual versus number of iterations for the steady-state computations. Time-accurate pitching wing computation has been started from a steady solution obtained after 100 iterations. 4-stage Runge-Kutta numerical method with CFL number of 0.8 is used in the computations, and it takes 9 seconds per iteration on 40 processors on COCOA3. The time-accurate computation for the two complete periods of oscillation has been done for the total of 960K iterations in 100 days. Figure 6.10 shows
Fig. 6.9. Unstructured hexahedral mesh (a) on the wing surface, and (b) on the xy-slice at mid-span

the variation of the angle of attack with time (the blue line) for the pitching motion according to Equation 6.1. The constant red line shows the mean angle attack of 10.88 degrees about which the wing pitches. The diamond symbols show the angle of attack of the wing at every 0.005 seconds (every 10000 iterations). The change in angle of attack becomes smaller when the wing gets close to the highest and lowest points of the pitch up and pitch down motions, respectively.

The pressure coefficient contours and streamlines at mid-span section of the wing, and also the velocity vectors around the leading edge for the steady-state solution are shown in Figure 6.12. Sample figures for the time-accurate solutions are presented in Figure 6.13. The pressure coefficient contours with streamlines for the absolute velocity, at the mid-span for four different iterations (30k, 60k, 90k, 110k iterations), which correspond to instantaneous angle of attacks of 12.44, 13.76, 14.65, 14.92 degrees during
the pitch-up motion in the first quadrant of oscillation, are shown in this figure. Complex unsteady vortical flow features of a pitching wing can be observed qualitatively in this figure. Figure 6.14 shows the instantaneous iso-surfaces of Mach number of 0.26 and vorticity magnitude of 50.0 while the wing is pitching down.

6.3 Large Eddy Simulations Around a Sphere: Test Case for the Wall Model

In order to test the performance of the wall model described in Section 3.2, which is implemented as a part of this thesis, a large eddy simulation of unsteady, separated flow around a sphere at a Reynolds number of $1.14 \cdot 10^6$ is performed using a second-order accurate, cell-centered finite volume method using unstructured grids [95, 96]. Classical Smagorinsky subgrid-scale model is used for subgrid-scale modeling. The flow around sphere is particularly chosen because the flow physics is far more complex than its relatively simple shape might suggest. Various investigations [61, 180, 218] have suggested...
Fig. 6.11. Residual versus number of iterations for steady-state computations with SSOR

Fig. 6.12. (a) Pressure coefficient contours with streamlines, and (b) absolute velocity vectors, at mid-span for steady-state computations with SSOR
Fig. 6.13. Pressure coefficient contours with streamlines, at mid-span from time-accurate computations at four different instants.
that over a wide range of Reynolds numbers \((280 < Re < 3.7 \cdot 10^5)\) the flow is characterized by unsteady vortex shedding, with most of the large-scale vorticity originating from the shear layer which separates from the surface. In the moderate to high Reynolds number regime, the flow is far from equilibrium, highly unsteady and experiences transition to turbulence. Simulations of flow over a sphere is a benchmark for techniques used to predict massively separated flows. It belongs to a special class of separated flows where flow separation is not fixed by geometry or other external effects (e.g. wire trips) that are used to force unsteadiness. The computational case is assumed to be fully turbulent, since the turbulence model is active over the entire surface of the sphere. Even with current parallel supercomputers, at such a high Reynolds number it is not feasible to capture the boundary layer in full detail with unstructured tetrahedral grids. Therefore, the wall model is used to model the shear stress at the wall to eliminate the need for very refined meshes near solid walls at high Reynolds numbers.

Jindal et al. [95, 96] has performed several simulations for low speed (with a Mach number of 0.16) viscous flows over a sphere at various Reynolds numbers using
two tetrahedral unstructured grids. The grid A had 900,000 grid cells, and grid B had 2,500,000 cells, which were both created using Gridgen [68]. There is clustering of cells near the surface of sphere and in the wake to capture the gradients as shown in Figure 6.15 where a surface mesh of the sphere is also shown. Inviscid flux calculations are done using a second order accurate Roe’s scheme, and a constant coefficient ($C_{sgs} = 0.1$) Smagorinsky model is used as a subgrid-scale (SGS) modeling. Time integration is done by SSOR (Successive Symmetric Over Relaxation) ($\omega = 1.2$) with local time-stepping till flow reaches a quasi-steady state, then a two-stage Runge-Kutta method is used for time accurate runs. At the outer surface Riemann inflow-outflow conditions are used and on the surface of the sphere a no-slip boundary condition is imposed. For the supercritical flows the log-law of the wall is used to calculate shear flux at the wall. Simulations are initiated with a free-stream condition without any turbulence. These simulations were performed on a Linux Beowulf cluster, Mufasa [87], using 24 processors. For $Re = 1.14 \times 10^6$, the coarser grid case (Grid A) took 14 days (CPU time = 14 \times 24 hrs) for 60 time units ($t \times U_\infty /D$) and the fine grid case (Grid B) took 21 days for 36 time units. For the coarser grid, one time unit was approximately 3000 time steps while for the finer grid case one time unit corresponded to 7500 time steps. Only the results of the high Reynolds number case: the flow about a sphere at a Reynolds number of 1.14106 which is in the supercritical range, are presented here. The results of other cases and more discussions can be found in References [95,96].

The mean pressure coefficients and mean skin friction coefficients are shown in Figures 6.16 and 6.17, respectively. Mean $C_p$ and $C_f$ were obtained by averaging the time accurate runs. The pressure coefficients are in good agreement with the experimental
results of Achenbach [1] and the DES results of Constantinescu and Squires [61]. The value and angular position of minimum $C_p$ is accurately captured. The separation is delayed in a supercritical case compared to a laminar one (not showed here). In the $\theta > 120$ regime, LES predictions are relatively flat, slightly above the data for $120 < \theta < 150$, and then slightly below the data for $\theta > 150$. This mismatch leads to a compensating effect such that drag predictions are reasonably correct. As expected, there is not a marked difference in the pressure coefficients results of PUMA2 with and without wall model since theoretically pressure does not change across a boundary layer. It should be noted that no documentation on experimental error bars is available.

The skin friction measurements are in a qualitative agreement with the experimental measurements [1] and the DES calculations [61]. Experimental measurements suggest that at $\theta = 95$, laminar intermediate separation occurs and is followed by an immediate transition in the shear layer from a laminar to a turbulent flow. Further
downstream turbulent separation occurs at around $\theta = 120$. The standard DES results [61] overpredict the skin friction coefficients while current results (PUMA2 with wall model) underpredict it but are closer to the experimental results of Achenbach [1]. However, DES results [61] had good agreement with experimental skin friction measurements by turning the turbulence modeling on at $\theta = 96^\circ$. For the current results, the under-prediction in $C_f$ might be due to using a standard log-law model without any pressure gradient corrections. It is not possible to implement pressure gradient effects for a complex three-dimensional flow as the flow direction is not known a priori on the body. As expected, the skin friction results of PUMA2 without wall model are close to zero everywhere since the boundary layer is not resolved adequately. The results of skin friction on such a coarse grid without any wall modeling is not justified. Also the experimental results of skin friction distribution of Achenbach are of qualitative character as mentioned in Reference [1]. He stated “In the absence of a better estimate the ‘laminar curve’ was used also for the calibration of turbulent part of boundary layer. The error thus introduced could not be estimated, therefore the results (skin friction) referring to turbulent flow are of a qualitative character.” [1].

The total mean streamwise drag and the contribution due to skin friction drag are presented in Table 6.1 and shows good agreement with experiments as compared to DES calculations by Constantinescu and Squires [61].

Figure 6.18 illustrates that the sphere simulations did capture a similar mechanism as observed by Taneda [218] through his flow visualizations. He observed that a vortex sheet separating from the sphere rolls up into an $\Omega$-shaped structure to form a pair
Fig. 6.16. Pressure coefficient, \( Re = 1.14 \times 10^6 \). Dotted line- experiments by Achenbach [1], • DES by Constantinescu and Squires [61], × PUMA2 without wall model, □ PUMA2 with wall model (Grid B). [96]

Fig. 6.17. Skin friction distribution, \( Re = 1.14 \times 10^6 \). Dotted line- experiments by Achenbach [1], • DES by Constantinescu and Squires [61], ▲ PUMA2 without wall model, □ PUMA2 with wall model (Grid B). [96]
Table 6.1. Drag Coefficients at $\text{Re}=1.1 \times 10^6$ for a sphere, experimental results from [1], DES results from [61]. [96]

<table>
<thead>
<tr>
<th></th>
<th>$C_d$ (total)</th>
<th>$C_f/C_d$ (in %)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Present (fine grid)</td>
<td>0.141</td>
<td>7.2</td>
</tr>
<tr>
<td>Present (coarse grid)</td>
<td>0.149</td>
<td>6.7</td>
</tr>
<tr>
<td>Exp.</td>
<td>0.12-0.14</td>
<td>5-7</td>
</tr>
<tr>
<td>DES</td>
<td>0.084</td>
<td>-</td>
</tr>
</tbody>
</table>

of strong streamwise vortices. The calculations also resolve a shedding of hairpin-like vortex structures which was also observed by Constantinescu [61] using DES.

As a result, for this high Reynolds number simulation in a super-critical regime, there is very good agreement with the experimental measurements in the overall drag and pressure distribution. The application of instantaneous log-law wall model had the effect of markedly improving the skin friction distribution. Though the predicted skin friction is not in excellent agreement with the experimental results by Achenbach [1], his results are only qualitative. In addition the drag coefficients predicted in the super-critical regime are adequate since skin friction contributes a small fraction of drag at higher Reynolds numbers. The ratio of skin-friction drag to total drag is also in good agreement with the experimental results. The vortex structure in the wake was also analyzed and is found to be dominated by shedding of hairpin-like vortices. No empirical parameters were used in the simulations and as compared to standard LES requirement these simulations were done with very few grid points. Strictly, the work might be more appropriate to be classified as Very Large Eddy Simulations (VLES) as 80% of the energy
was not resolved everywhere in computational domain. Also, as unstructured tetrahedral cells were used everywhere in the computational domain, the approach is scalable for any complex geometry. It presents an opportunity to simulate practical flows with high Reynolds number on complex bodies with reasonable accuracy. Although the structured grids scale as well, they often waste cells in the far field.
Chapter 7

Wind Turbine Rotor Simulations

The results of three-dimensional and time-accurate Computational Fluid Dynamics (CFD) simulations of the flow field around the 2-bladed National Renewable Energy Laboratory (NREL) Phase VI Horizontal Axis Wind Turbine (HAWT) rotor are presented in this Chapter. The 3-D, unsteady, parallel, finite volume flow solver, PUMA2, is used for the simulations. The solutions are obtained using unstructured moving grids rotating with the turbine blades. Three different flow cases with different wind speeds and wind yaw angles are investigated: 7 m/s with 0° yaw (pre-stall case I), 7 m/s with 30° yaw (pre-stall, yawed case II), and 15 m/s with 0° yaw (post-stall case III). Both the inviscid and the viscous results are presented and compared. Large Eddy Simulation (LES) methodology with an instantaneous log-law wall model is used for the viscous computations.

These three-dimensional and time-accurate CFD results can be used for the far-field noise predictions based on the Ffowcs Williams - Hawkings method [57], which can provide a first-principles prediction of both the noise and the underlying turbulent flow that generates the noise, in the context of the wind turbine application [149]. The calculated three-dimensional mean flow could also provide the basis for detailed unsteady flow simulations based on the Non-Linear Disturbance Equations (NLDE) [148] which
in turn can provide the aerodynamic input needed for the prediction of trailing edge and
tip vortex noise [149].

7.1 Computational Test Cases

In accordance with the NREL Unsteady Aerodynamics Experiments [66, 80], in
which a large-scale horizontal axis wind turbine was examined at the NASA Ames wind
tunnel facilities, the two-bladed NREL Phase VI wind turbine (baseline) rotor (Figure 1.2) has been considered for the CFD simulations. Blades have the NREL S809
airfoil section from root to tip. Pitch is defined at 75% span and the pitch axis is at
the 30% chord line. A linearly tapered and nonlinearly twisted blade geometry with a
span of 5.029 m and a flat tip, as shown in Figure 7.1, was generated using ProDesktop.
Figure 7.2 shows the blade twist distribution and planform [66]. The blade geometry
used has 0° twist at 75% span (and −2° twist at the tip). The blade has a root chord
of 0.737 m and tip chord of 0.356 m with a taper ratio of ‘2.1’. There is a cylinder with
0.109 m radius which extends from 0.508 m to 0.724 m, and then there is a transition
from the circular section at 0.724 m to the root airfoil section at radius 1.257 m.

In the NREL Phase VI experiments [80], the spherical-tip 5-hole probe shown in
Figure 7.3 provided dynamic pressure, local flow angle, and spanwise flow angle mea-
surements ahead of the blade at a distance of 80% chord at five span locations, 34%,
51%, 67%, 84%, and 91% span. The probes were positioned at an angle nominally 20°
below the chord line to align the probe with the flow under normal operating conditions.
In addition, chordwise blade surface pressure distributions are also measured at five
spanwise stations: 30%, 46.6%, 63.3%, 80%, and 95% span. Table 7.1 shows the selected
Fig. 7.1. Three-dimensional NREL wind turbine blade geometry generated using ProDesktop.

Fig. 7.2. NREL wind turbine rotor blade twist distribution and planform [66].
Fig. 7.3. Blade-mounted five-hole probe used in NREL Phase VI experiments [80].

Experimental cases that may be used for validation and comparison of the computational test cases. Following these experiments, the computational cases are selected as shown in Table 7.2. The rotor blades have $5^\circ$ pitch angle (at 75% span) and $0^\circ$ cone angle for the computational cases considered.

### 7.2 Unstructured Computational Grids

The unstructured grids for the wind turbine simulations were generated using Gridgen software [68]. Through the use of unstructured grids, it is possible to heavily cluster points in regions of interest, while keeping the far-field cells rather coarse. Although the structured grids can also be clustered, they often waste cells in the far field. An unstructured grid (Grid A) as shown in Figures 7.4 and 7.5 has 3.6 million tetrahedral cells clustered around the blades and tip vortices. Figure 7.5 shows the outer boundary, the mesh near the blade surface and the mesh on $x = 0$ and $y = 0$ planes. This grid was used for all the inviscid computations whereas it was used only for case I and case II for the viscous computations.
Table 7.1. Selected NREL Phase VI experimental cases from Reference [80].

<table>
<thead>
<tr>
<th>Test Matrix</th>
<th>Wind speed</th>
<th>Yaw angle</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>H: Upwind baseline case</strong></td>
<td>7 m/s (pre-stall)</td>
<td>-30, 0, 30, and 60 degrees</td>
</tr>
<tr>
<td></td>
<td>10 m/s (onset of stall)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>15 m/s (post stall)</td>
<td></td>
</tr>
<tr>
<td><strong>P: Wake flow visualization</strong></td>
<td>5, 7, 10, 15 m/s</td>
<td>0, -10, -30, -60 degrees</td>
</tr>
<tr>
<td><strong>5: Sweep wind speed case</strong></td>
<td>5 to 25 m/s</td>
<td>0 degrees</td>
</tr>
<tr>
<td></td>
<td>25 to 5 m/s</td>
<td></td>
</tr>
</tbody>
</table>

Table 7.2. Selected computational test cases.

<table>
<thead>
<tr>
<th>Test Cases</th>
<th>Wind speed</th>
<th>Yaw angle</th>
</tr>
</thead>
<tbody>
<tr>
<td>CASE I</td>
<td>7 m/s (pre-stall)</td>
<td>0 degrees</td>
</tr>
<tr>
<td>CASE II</td>
<td>7 m/s (pre-stall)</td>
<td>30 degrees</td>
</tr>
<tr>
<td>CASE III</td>
<td>15 m/s (post stall)</td>
<td>0 degrees</td>
</tr>
</tbody>
</table>

Fig. 7.4. Unstructured tetrahedral grid (Grid A) generated using Gridgen.
Fig. 7.5. Unstructured tetrahedral Grid A.
As case III had a higher wind velocity and separated flow, a finer unstructured grid, called Grid B, with more clustering around the blades is used so that the average $y^+$ value of the first grid points away from the wall is small. This grid had 4.2 million cells in total. Figure 7.6 shows the grid detail near the blade surface for this grid. The maximum $y^+$ value at the first grid point away from the wall for this case was found to be 544. The minimum cell length for this grid is about 0.00017 m. This grid has a finer, clustered cell distribution around the blade compared to the other grid in Figure 7.5. The grid around the wake for this grid was similar to Grid A.

The cylindrical computational domain has a radius of 12.0 m (about the height of turbine tower, i.e. 12.192 m), which extends 4 and 2 rotor radii in the positive (downstream) and negative (upstream) Y-direction, respectively. For the two-bladed wind turbine rotor, the blade at the 12 o’clock position (azimuth angle = 0°) is designated as Blade 2, and the Blade 1 is at 6 o’clock position, to be consistent with the NREL wind turbine experiments (Figure 7.7). The definitions of the yaw and the azimuth
angles used in the computations are also shown in that figure. The blades rotate about the $Y$-axis, in the negative direction at 72 rpm (7.54 rad/sec). Wind speed in the positive $Y$-direction with different yaw angles has been considered for the test computations as seen in Figure 7.8. According to this figure, the relative angles of attack (flow angle with respect to the $X$-axis) of the wind at 80% span are calculated as 12.99°, 10.16° and 26.31° for cases I, II and III, respectively.

The permeable FW-H surface (for noise calculations) was also embedded in the grid system as seen in Figure 7.9 during the grid generation process. This ensures that data can be extracted for noise computations.
Fig. 7.8. Velocity vectors for the wind and the rotating blade at 80% span for the selected computational cases.

Fig. 7.9. Permeable surface embedded in the unstructured grid.
7.3 Parallel Computational Cost

Computational tests and the selected cases were run mainly on the Penn State clusters, LION-XL [124] and MUFASA [87], (also some on the NREL, NCSA and NASA clusters) (see Section 3.4 for details of the clusters). Different numbers of processors were used to better characterize the computational resource requirements and potential throughput on each machine. Table 7.3 shows the computational performance of PUMA2 inviscid computations on different clusters, such as the memory required per compute node, and the number of days required for the computation of one complete revolution of wind turbine rotor for different numbers of processors. The wall time required per iteration versus number of processors is also plotted in Figure 7.10. The inviscid computation on LION-XL using 32 processors takes 6 days of wall-time for one full revolution which requires 90,000 iterations.

<table>
<thead>
<tr>
<th># of processors</th>
<th>memory/node [MB]</th>
<th>LION-xl</th>
<th>MUFASA</th>
<th>NREL cluster Std</th>
<th>NCSA cluster</th>
<th>NASA cluster</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>297</td>
<td>14</td>
<td>42</td>
<td>15</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>32</td>
<td>192</td>
<td>6</td>
<td>12</td>
<td>10</td>
<td>5</td>
<td>5.4</td>
</tr>
<tr>
<td>64</td>
<td>140</td>
<td>-</td>
<td>8</td>
<td>-</td>
<td>2.4</td>
<td>3.5</td>
</tr>
<tr>
<td>128</td>
<td>113</td>
<td>-</td>
<td>5</td>
<td>-</td>
<td>1.7</td>
<td>1.8</td>
</tr>
</tbody>
</table>

Table 7.3. Computational performance of PUMA2 inviscid simulations on different clusters.
Fig. 7.10. Computational performance of PUMA2 inviscid simulations on different clusters.

The LES computational cases were run on the NCSA tungsten supercomputer [151] using 64 processors. In comparison to the inviscid runs, this time the code was optimized and some unwanted communications were removed. The code was compiled using the Intel compiler (version 9.0.026) and ChamPIon/Pro MPI with the following options:-no-gcc -O2 -tpp7. The time required for one revolution for the LES computations for cases I and II was 35 wall-time hours, whereas case III required 190 wall-time hours for a 90 degree rotation of the blades. This was due to the much smaller time step size and more cells in Grid B compared to Grid A. As one full revolution requires 0.833 sec, 104,167 iterations were required per revolution for cases I and II. One full revolution for case III would have required 1,800,000 iterations.
7.4 Results of Inviscid Simulations

The inviscid, time-accurate CFD simulations were performed for the NREL Phase VI turbine rotor using the flow solver PUMA2 with moving grids for three selected computational cases (See Table 7.2). The 4-stage Runge-Kutta numerical time integration method and Roe’s numerical flux scheme was used in the computations. Time-accurate computations were started from the freestream conditions (for a given wind speed). Because the time step for time-accurate computations using an explicit scheme is determined by the smallest cell in the volume grid, the minimum cell size and maximum number of cells were selected during the grid generation process by also considering the total computational time needed for the time-accurate simulations. The simulation time step size was selected as 9.26 microseconds so that the calculated a Courant-Friedrichs-Lewy (CFL) number will be less than 1.0 for the smallest cell size (0.0033 m) in the volume grid. Parallel computations were performed on different Beowulf-clusters as explained in the previous section. The computations for one full revolution (i.e. 0.8333 seconds) required 90000 iterations.

The time accurate flow solution was stored every 30° of rotor azimuth angle in the 1st and 2nd revolution computations for all three cases. The solutions were stored every 1° interval in the 3rd revolution for cases II and III, to be used for the aeroacoustic calculations.

The instantaneous vorticity iso-surfaces, which were obtained with a new utility code written for PUMA2, are shown in Figure 7.11. By post-processing the stored solution data, this utility first scatters the calculated velocity values from cell centers to
Fig. 7.11. Inviscid Results: Vorticity iso-surface for a) CASE I, b) CASE II, c) CASE III.
Fig. 7.12. Snapshots of instantaneous vorticity iso-surfaces for CASE I: Inviscid results at the end of (a) 1st, (b) 2nd, and (c) 3rd revolutions.
face centers, and then calculates the cell velocity gradients, and therefore vorticity, using the face-centered velocities and Gauss’ Theorem. Several observations can be made. First, for case I, the flow is attached (as will be shown later in Figure 7.19), and well defined vortical structures are shed from the blade tips. For case II, the flow is again attached, however the wake is asymmetrical because of the 30° yaw angle. For the higher wind speed case, i.e. case III, the flow is massively separated over the entire blade span. Although the tip vortices can still be depicted, the flow is highly unsteady because of the separation. This is much better observed when time-accurate results are visualized as an animation. In addition, the wake structures are convected at a higher speed compared to cases I and II. The vortical wake of case I is plotted at the end of the second revolution, and it can be seen from Figure 7.12, which shows the snapshots of the time-accurate inviscid results for case I, at the end of 1st, 2nd and 3rd revolutions, respectively, that it continues to develop as the computation is continued for more revolutions. However, the vortices are diffused for cases II and III as shown in Figure 7.11. This may be due to the grid quality and numerical dissipation which need to be studied further. Also, the grid used for all three cases was created considering the 0° yaw cases (see Figure 7.5), so it may not be the best choice for the yawed case II which has an unsymmetric skewed wake structure.

The instantaneous pressure contours on the rotor blade upper and lower surfaces for all three cases are shown in Figures 7.13, 7.14, and 7.15 for both blades 1 and 2. The contours show considerable spanwise pressure variations in addition to the chordwise variations. Figure 7.16 shows the instantaneous and time averaged chordwise pressure coefficient distributions of the inviscid CFD simulations at 30%, 46.6%, 63.3%, 80% and
95% spanwise stations for blade 2. The pressure coefficient is calculated using

\[ C_P = \left( P - P_\infty \right) / \left( \frac{1}{2} \rho_\infty \left( U_\infty^2 + (\Omega r)^2 \right) \right) \]  \hspace{1cm} (7.1)

where, \( U_\infty \) is equal to 7 m/s for cases I and II, and 15 m/s for case III. The instantaneous distributions show the time-accurate CFD solutions at time \( t = 1.667 \) s (i.e. at the end of second revolution) for case I, and at \( t = 2.500 \) s (i.e. at the end of third revolution) for cases II and III. The averaged distributions are obtained by averaging the solutions with \( 30^\circ \) intervals over the 2nd revolution for case I, and by averaging the solutions with \( 1^\circ \) intervals over the 3rd revolution for cases II and III. In Figure 7.16, the results of the 0 yaw cases with 7 m/s and 15 m/s wind speeds are compared with the NREL Phase VI experimental data (from Reference [50]). There is good agreement between the experimental data and the computations for case I. There is more deviation between the experimental data and computations for case III, especially on the blade upper surface. However, the comparisons are still reasonable keeping in mind that the computations are inviscid on a relatively coarse grid. The highest differences between the instantaneous and averaged \( C_P \) distributions are observed for the \( 30^\circ \) yaw case (case II). This occurs due to the cyclic unsteady variation of blade surface pressure as the blade rotates. This is evident from the instantaneous surface pressure distributions on blade 1 (at 6 o’clock position) and blade 2 (at 12 o’clock position), as presented in Figure 7.14, in which the blades are at different azimuthal positions. The unsteady pressure variations also occurs for case III, however, there is no obvious cyclic variation as in case II. Consequently, the difference between the instantaneous and averaged pressure distributions is much
less compared to case II, but still higher than case I which shows minimal level of cyclic variation as expected.

Figure 7.17 shows the time history of the thrust coefficient of time-accurate computations for all inviscid cases, where the thrust coefficient is calculated using

\[ C_T = \frac{T}{0.5 \rho_\infty (\pi R^2) V_\infty^2} \]  

In case I, \( C_T \) starts to reach a steady state value of about 0.52 by the second revolution (corresponding to a thrust of about 1250 N). For case II, a periodic oscillation of the thrust coefficient is observed around an average value of 0.425 (corresponding to a thrust of about 1000 N). For case III, due to the massive separation over the blades, non-periodic oscillations are present (with an average thrust of about 2300 N). The thrust coefficient for the yawed case is lower than the unyawed case as expected. Figure 7.18 shows time history of the force coefficients for all three cases in X- and Z-directions (relative to the blade’s initial position). In this figure, results for case I is presented for the 2nd revolution, and the results for cases II and III are for the 3rd revolution. As seen in the figure, the variations of \( F_x \) and \( F_z \) time histories are similar to the \( C_T \) distributions, however the magnitudes are about 1-2 orders of magnitude smaller.

For helicopter rotors, a parameter called “advance ratio”, which is a ratio of the forward flight speed to the tip speed, is usually defined for various flight conditions. In the case of a wind turbine rotor, if a similar comparison or analogy is to be made for the computational cases considered here, it can be said that the cases I and III have a vertical speed whereas the case II has both forward and vertical speeds. For all of
the computational cases, the rotational speed (so the tip speed) is kept constant as same as the experimental conditions. Case III has an increased vertical speed (15 m/s) compared to case I (7 m/s), and both cases I and III have zero advance ratios. Case II has a decreased vertical speed \(7 \cdot \cos(yaw\ angle) = 6.06\ m/s\) and an advance ratio of \((7 \cdot \sin(yaw\ angle)/\Omega \cdot R = 0.09)\) where \(yaw\ angle = 30^\circ\).

The instantaneous contours of relative velocity magnitude together with streamlines around the airfoil sections at 5 different spanwise stations for blade 2 can be seen in Figure 7.19 for all three computational cases. The flow is attached for both cases I and II, whereas it is massively separated for case III. This separation occurs due to the high angle of attack created by the higher wind speed than the other two cases while the rotational speed of the turbine is kept the same for all cases. Figure 7.20 shows the relative velocity magnitude and relative angle of attack (flow angle with respect to the X-axis) variations along the span of blade 2, obtained at the five-hole probe measurement locations as in the NREL Phase VI experiments (see Figure 7.3). As seen in the figure, the relative angles of attack for case III is about 2 and 3 times higher than those in cases I and II, respectively. The relative angles of attack in case II are lower and the magnitudes of relative velocity are higher than case I as expected from the velocity vector descriptions presented in Figure 7.8.

### 7.5 Results of LES Simulations

Large Eddy Simulations (LES) of flow around the NREL Phase VI Rotor using PUMA2 with moving grids have been performed for the selected three different cases (see Table 7.2). Roe’s scheme was used to compute the inviscid fluxes. Time integration was
Fig. 7.13. Inviscid Results: Gauge pressure \((P - P_\infty)\) contours for CASE I at \(t = 1.667\) s (Left: lower surface, Right: upper surface).
Fig. 7.14. Inviscid Results: Gauge pressure ($P - P_\infty$) contours for CASE II at $t = 2.500$ s (Left: lower surface, Right: upper surface).
Fig. 7.15. Inviscid Results: Gauge pressure \((P - P_\infty)\) contours for CASE III at \(t = 2.500\) s (Left: lower surface, Right: upper surface).
Fig. 7.16. Inviscid Results: Chordwise pressure coefficients at 5 spanwise blade stations for a) CASE I, b) CASE II, c) CASE III. Symbols for Instantaneous results: red square; Averaged results: blue circle, and Experimental data (average): black diamond.
Fig. 7.17. Inviscid Results: Time history of the thrust coefficient for a) cases I and II, and b) case III.

Fig. 7.18. Inviscid Results: Time history of the force coefficients for all three cases in a) $x$-direction, and b) in $z$-direction (relative to the blade).
Inviscid Results: Relative velocity magnitude contours with streamlines around the airfoil sections at 5 spanwise blade stations for a) CASE I at $t = 1.667$ s, b) CASE II, and c) CASE III at $t = 2.500$ s.
performed using a two stage Runge-Kutta method. Riemann Inflow-Outflow boundary conditions were used at the outer surface and no-slip velocity boundary conditions are applied on the surface of the blades. A Smagorinsky model was used to model the sub grid-scale stresses and the value of $C_S$ was taken to be 0.10. The instantaneous log-law, which has been implemented in PUMA2 (see Section 3.2 and was previously tested with the flow over sphere simulations [96], was used to calculate the shear stress at the first grid point away from the blades. This local wall stress is then fed back to the outer LES in the form of proper momentum flux at the wall due to normal diffusion. For the rotating blade simulations, the rotational grid velocities are taken into consideration in the use of wall model.

For Case I, the time-accurate LES computations were started from the inviscid solution at the end of the 2nd revolution (Figure 7.12(b)) and was run for three more revolutions. For the other two cases, the LES computations were started from the inviscid solution at the end of the 3rd revolution (Figures 7.11(b) and (c)), and then case II was
run for two more revolutions and case III was run only for a quarter revolution (90 degrees).

Grid A was used for cases I and II and the maximum $y^+$ value at the first grid point away from the blades was found to be 985 for case I and 1043 for case II. This corresponds to the upper edge of the logarithmic layer. The simulation time step size was selected as 8 microseconds for cases I and II so that the calculated CFL number and the Von Neumann number (for stability of viscous terms) for the smallest cell size will be less than one, and 0.5 respectively.

For case III a different grid, Grid B, was used as the wind speed was higher for this case, and the flow was massively separated. This grid had more clustering near the blades as compared to the other grid. The maximum $y^+$ value at the first grid point away from the blades was 576. The simulation time step size for this case was much less than the previous cases as the smallest cell size was 0.00017 m for this case. The simulation time step for this case was selected as 0.46 microseconds based on the CFL number and the Von Neumann number.

For cases I and II, the full solution at every 5 degree rotation, for one full revolution of the blades was used to obtain the time averaged solution. For case III using Grid B, the full solution at every 1 degree rotation for a 50 degree rotation of the blades was used to obtain the time averaged solution.

Figures 7.21 and 7.22 show the instantaneous vorticity iso-surface plots for cases I, II, and III for three different views. In comparison to the inviscid results (Figure 7.11), it can be seen that the vortices diffuse more in the stream-wise direction. In the inviscid cases, the factors that caused diffusion were the numerical scheme and the unstructured
grid which becomes coarser in the stream-wise direction. For the viscous cases, in addition to the above factors we have viscous diffusion resulting in greater diffusion of the vortices. It is observed that the vortices are more dissipated for case II as compared to the other cases. This could be attributed to the fact that the grid was refined so as to capture the wake shed in the positive Y-direction whereas in this case the wake shed would be inclined at an angle (the yaw angle). Further, as compared to the inviscid result for this case, a distinct vortical structure shed near the middle of the blade span is observed. For case III, we observe that the flow is massively separated over the span of the blades. The flow is highly unsteady and turbulent in this case. We can see blobs of vortical structures shed from the span of the blades. Also, as wind speed is higher in this case, the wake structure repeats itself over a greater length in the wind direction.

Figures 7.23, 7.24, and 7.25 show the instantaneous pressure contours on the rotor blade upper and lower surfaces for all the three cases. As compared to the inviscid results, we can see that the pressure is more irregular and fluctuating throughout the span of the blades. For case II, the sweep effect of the vortex structure shed around the middle of the blade produces the distinct pressure variation seen in Figure 7.24. Similar vorticity “sweep effects” can be seen throughout the span of the blades for case III in Figure 7.25.

Figure 7.26 shows the instantaneous and averaged chordwise pressure coefficient distributions at 30%, 46.6%, 63.3%, 80% and 95% spanwise stations for blade 2. The pressure coefficient is calculated using Equation 7.1. For case III, we see that there is better agreement with experiments as compared to the inviscid results for the 30%, 45%, and 80% locations. For the other spanwise locations, the agreement is not that good.
a) CASE I at $t = 4.169$ s

b) CASE II at $t = 4.169$ s

c) CASE III at $t = 2.708$ s

Fig. 7.21. LES Results: Vorticity iso-surface for a) CASE I, b) CASE II, c) CASE III.
Fig. 7.22. LES Results: Vorticity iso-surface for a) CASE I, b) CASE II, c) CASE III.
on parts of the upper blade surface. Given the fact that a simple Smagorinsky model with a log-law wall model was used, the computational results compare well with the experiments.

The instantaneous contours of relative velocity magnitude together with streamlines around the airfoil sections at 5 different span-wise stations for blade 2 can be seen in Figure 7.27 for all the three computational cases. The flow is attached for both case I and II, whereas it is massively separated for case III. As expected, in comparison to the inviscid plots, the relative velocities are lower specifically near the leading edge and the trailing edge of the airfoil due to viscous effects.

Figure 7.28 shows the time history of the thrust coefficient, $C_T$, of the LES computations for all cases. The thrust coefficient was computed for each blade and then multiplied by the number of blades (2) to provide a rough estimate of the aerodynamic thrust on the rotor. In highly yawed conditions, the assumption that the blades experience equal loads is clearly incorrect. But these estimates do provide the general trends as compared to the non-yaw cases in the uniform flow. In case I, $C_T$ starts to reach a steady state value of about 0.47 by the second revolution (corresponding to a thrust of about 1120 N). For case II, a periodic oscillation of the thrust coefficient is observed around an average value (for two revolutions) of 0.37 (corresponding to a total thrust of about 882 N). For case III, due to the massive separation over the blades, non-periodic oscillations are expected. But, as the simulation for case III was done for only a 90 degree rotation of the blades, nothing can be said positively about the average thrust value. However, Figure 7.28 shows that the thrust coefficient for this case after remaining constant at roughly 0.2 (corresponding to a thrust of 2190 N) is starting to decrease towards the end
Fig. 7.23. LES Results: Gauge pressure \((P - P_\infty)\) contours for CASE I at \(t = 4.169\) s (Left: lower surface, Right: upper surface).
Fig. 7.24. LES Results: Gauge pressure \((P - P_\infty)\) contours for CASE II at \(t = 4.169\) s (Left: lower surface, Right: upper surface).
Fig. 7.25. LES Results: Gauge pressure \((P - P_\infty)\) contours for CASE III at \(t = 2.708\) s (Left: lower surface, Right: upper surface).
Fig. 7.26. LES Results: Chordwise pressure coefficients at 5 spanwise blade stations for a) CASE I, b) CASE II, c) CASE III. Symbols for Instantaneous results: red square; Averaged results: blue circle, and Experimental data (average): black diamond.
Fig. 7.27. LES Results: Relative velocity magnitude contours with streamlines around the airfoil sections at 5 spanwise blade stations for a) CASE I at \( t = 4.169 \) s, b) CASE II, and c) CASE III at \( t = 2.708 \) s
of the 90 degree rotation. As expected, the thrust coefficient for the yawed case (case II) is lower than the unyawed case (case I). The rotor thrust for case I is reported to be about 1060 N by predictions in Reference [66] which is close to the value obtained by computation. For case III, experiments [66] report a value of about 1725 N. It is expected that if the simulation for a full revolution of the blades is performed for case III, the computational thrust would approach this value.

Figure 7.29 shows a comparison of a sample boundary layer profile obtained from the LES results for CASE I at $t = 4.169$ sec with the log-law obtained from the wall-model described in Section 3.2. This velocity profile is extracted along the solid black line shown in Figure 7.29a, which is a cross-section at 80% span location showing the in-plane relative velocity magnitude contours (i.e. $(u_r^2 + v_r^2)^{1/2}$). The wall-model is applied below the first grid point. Figure 7.29c shows that the computed velocity profile matches
well with the log-law below about $y^+ = 600$, where the first grid point is approximately located.

Figures 7.30, 7.31 and 7.32 show the wind turbine rotor wake characteristics of the LES results for CASE I at $t = 4.169$ s. The position of the tip vortices within the wake with respect to the blade can be seen in Figure 7.30b, which presents the vorticity magnitude contours as well as the streamlines in the absolute frame of reference on the $YZ$-slice at $X = 0$ (Figure 7.30a). The vortex cores are visible as the locations where the vorticity magnitude is maximized, and these locations are determined quantitatively, as described in detail below. The expansion of the wake is also visible by the divergence of the absolute streamlines as they pass through the wind turbine. This expansion is also investigated in detail and compared to existing expansion models, as also described below.

Figure 7.31a shows the contours of $w$ velocity component (i.e. velocity along $z$-direction) again on $YZ$-slice at $X = 0$, focusing on the wake region behind the upper blade of the wind turbine. The tip vortices are clearly marked with side-by-side positive and negative $w$ distributions. Using this figure, the tip vortex core locations are determined as the locations where the $w = 0$ contours intersect with the maximum values of the vorticity magnitudes presented in Figure 7.30(b). The streamlines relative to these vortex core locations are also plotted in Figure 7.31(a) (shown as solid black lines), which are obtained by using the $w$ and $(v - v_{conv})$ as the in-plane velocity vector components, where $v_{conv}$ is the convection velocity of the vortex cores in $Y$-direction. The white dashed line in this figure shows the tip diameter of the wind turbine rotor. Comparing the vortex core locations with respect to this line shows that the shown three
Fig. 7.29. (a) In plane relative velocity magnitude \[ (u_r^2 + v_r^2)^{1/2} \] at 80% span, (b) velocity profile along the black solid line (at \( X = 0.0215 \) m) shown in (a), and (c) comparison of boundary layer profile with the log-law. CASE I at \( t = 4.169 \) s. LES results.
vortex cores are all above this line, with the distance between the core and the dashed line increasing from 1st to 2nd vortex and then slightly decreasing in the 3rd vortex. The absolute streamlines are also shown in this figure as a reference (solid red lines), again illustrating the wake expansion through the rotor. Figure 7.31b presents $v$ ($Y$-velocity component) contours within the rotor wake on $XZ$-slice at $Y = 2.5$ m. The rotor disk area is marked with the solid black line. Except near the center of the disk area the $v$-velocities are lower compared the freestream value of 7 m/s. Near the center, the levels are higher, due to the effect of the root vortices.

Figure 7.32 shows the comparison of the wake expansion characteristics of the LES results with the one obtained from a wake expansion model given in Reference [250]. The computed absolute streamlines clearly show the wake is expanding through the turbine rotor. Near the tip of the rotor blade, the streamlines become more wiggly due to the effect of the tip vortices. The cores of these vortices, which were determined as explained above, are also marked in this figure as blue circular symbols. The wake expansion model [250], shown as the dashed grey line in Figure 7.32, is given by:

\[
\frac{R_{\text{wake}}}{R} = \left(1 - \frac{E q}{(1 + q^2)^{1/2}}\right)^{-1/2} \tag{7.3}
\]

where $q = y/R$, $R$ is the rotor radius and $y$ is the downstream distance, and $E$ is given by

\[
E = a(\lambda - 1)/(1 - a) \tag{7.4}
\]
where, $a$ is the axial induction factor at the rotor disk and can be approximated from the thrust coefficient:

$$C_T = 4a(1 - a) \text{ for } C_T < 0.64 \text{ and } \lambda = 2$$

which is given by the actuator disk momentum theory where $\lambda \cdot a$ is the axial induction factor at the far wake. The value of $a$ obtained using the thrust coefficient (i.e., $C_T = 0.47$) for the current wind turbine for case I for the LES results is 0.14. The green triangles mark the locations of the estimated wake radius using the LES results. These estimations are obtained using the the average $v$ velocities within a region in the wake coinciding with the rotor disk area as shown in Figure 7.31b, at four different $XZ$-slices at $Y = 2.5, 5, 7.5, 10$ m, and then calculating the mass flow rate using these average $v$ values within these areas and comparing them with the mass flow rate at the freestream within an area again corresponding to the rotor disk area. Since the mass flow rates within the wake region are lower compared to the freestream due to the lower values of the average $v$ velocities, the wake needs to expand in order to accommodate the incoming mass flow. This area expansion is calculated and the obtained wake radii are plotted as the green triangles in Figure 7.32. When we compare the LES results, i.e. both the absolute streamlines and the estimated wake radii, with the wake expansion model, the results are consistent with each other. The expansion in the case of LES seems to be less than the one predicted by the model when we get further away from the blade within the wake. This might be due to the strong root vortices shed from the rotor. These vortices cause an increase in the average $v$ velocity values (see Figure 7.31(b)) and hence the
mass flow rate, which results in a decrease in the rotor wake radii. Also, the numerical
dissipation due to the computational method as well as the coarser grid after \( Y = 15 \) m
might have an effect on the wake expansion. Also, further time-accurate computations
for more revolutions might be necessary to see the effect of convergence. These points
need further detailed analysis and investigation.

Schreck and Robinson [184] analyzed the blade surface pressure and local inflow
data from the NREL Unsteady Aerodynamics Experiment to characterize the dynamic
stall vortex generated on the blade during yawed operations. It was observed that the
highly three dimensional and complex vortical flow field responded systematically to
alterations in the wind speed and turbine yaw angle. The changes in the flow field due
to the variations in the wind speed and yaw angle were observed also from the results of
the current study as presented in Figure 7.27, discussed in detail above. Leishman [122]
presented the challenges in modeling the unsteady aerodynamics of wind turbines, and
the significance and modeling of dynamic stall for the wind turbines. Hence, the three-
dimensional and time-accurate CFD simulations of the rotating wind turbine rotor blades
are important and necessary in understanding the three-dimensional and complex nature
of the rotor flow fields.

7.6 Computational Aeroacoustic Analysis of Wind Turbines

The noise radiation from the wind turbine rotating blades can be predicted from
the unsteady loading generated by a combination of the rotating blade aerodynamics
and the airfoil self-noise prediction tools. The Ffowcs Williams-Hawkings (FW-H) equa-
tion [57] can be utilized to predict the noise in the vicinity of the wind turbine and can
Fig. 7.30. Wind turbine rotor wake characteristics of the LES results for CASE I at $t = 4.169$ s. (a) Unstructured mesh on $YZ$-slice at $X = 0$ (GRID A) (b) Vorticity magnitude contours and absolute streamlines
Fig. 7.31. Wind turbine rotor wake characteristics of the LES results for CASE I at $t = 4.169$ s. (a) $w$ ($Z$-velocity component) contours on $YZ$-slice at $X = 0$ with streamlines in the absolute frame (solid red lines) and streamlines relative to the tip vortex cores (solid black lines). Dashed white line shows the rotor tip diameter. (b) $v$ ($Y$-velocity component) contours within the rotor wake on $XZ$-slice at $Y = 2.5$ m with the rotor disk area shown as circular solid black line.
Fig. 7.32. Comparison of the wake expansion characteristics of the LES results for CASE I at $t = 4.169$ s.
also provide the initial condition for long-range propagation. Thus, the three-dimensional
and time-accurate CFD results can be used for the far-field noise predictions based on
the FW-H method, which can provide a first-principles prediction of both the noise and
the underlying turbulent flow that generates the noise, in the context of the wind turbine
application [149].

### 7.6.1 The Ffowcs Williams Hawkings Method

The FW-H equation [57] is the most general form of the Lighthill acoustic anal-
ogy [123] and is appropriate for the prediction of rotating blade noise both in the time and
frequency domains. The FW-H equation is an exact rearrangement of the Navier-Stokes
equations into the form of an inhomogeneous wave equation with surface (monopole and
dipole) and volume (quadrupole) source terms. The FW-H equation can be written [149]
as

\[
\frac{1}{c^2} \frac{\partial^2}{\partial t^2} - \frac{\partial^2}{\partial x_i^2} p'(x, t) = \frac{\partial}{\partial t} \left\{ \left[ \rho_o v_n + \rho (u_n - v_n) \right] \delta(f) \right\}
- \frac{\partial}{\partial x_i} \left\{ \left[ \Delta P_{ij} \hat{n}_j + \rho u_i (u_n - v_n) \right] \delta(f) \right\}
+ \frac{\partial^2}{\partial x_i \partial x_j} \left[ T_{ij} H(f) \right] \tag{7.5}
\]

where \( \rho_o \) is the density of the undisturbed fluid, \( u_n \) is the fluid velocity in the direction
normal to the integration surface (defined by \( f = 0 \)), \( v_n \) is the normal velocity of
the integration surface, \( \Delta P_{ij} \) is the jump in the compressive stress tensor across the
integration surface (\( \Delta P_{ij} = p' \delta_{ij} \) for an inviscid fluid), and \( T_{ij} \) is the Lighthill stress
tensor. The surface sources, identified by the presence of the Dirac delta function \( \delta(f) \),
can be specified either on a surface corresponding to the actual blade surface or a fictitious data surface that contains all of the desired physical noise sources [23]. Because the fictitious surface is located in the flow field, this form of the FW-H equation is usually called a porous or permeable surface FW-H formulation. In either permeable or actual blade surface application, the source strength is determined \textit{a priori} through a detailed computation of the flow field.

Farassat’s Formulation 1A of the FW-H equation [54,149] can be written as:

$$p'(\vec{x}, t) = p'_T(\vec{x}, t) + p'_L(\vec{x}, t)$$ \hspace{1cm} (7.6)

where \( p' \) is the acoustic pressure, \( \vec{x} \) is the observer position, \( t \) is the observer time, the subscripts \( T \) and \( L \) correspond to thickness (monopole) and loading (dipole) components, respectively, and where:

$$4\pi p'_T(\vec{x}, t) = \int_{f=0} \left[ \frac{\rho_0 (\dot{U}_n + U_n)}{r(1 - M_r)^2} \right] \text{ret} dS + \int_{f=0} \left[ \frac{\rho_0 U_n \left( r \dot{M}_r + c \left( M_r + M^2 \right) \right)}{r^2 (1 - M_r)^3} \right] \text{ret} dS$$ \hspace{1cm} (7.7)

and

$$4\pi p'_L(\vec{x}, t) = \frac{1}{c} \int_{f=0} \left[ \frac{\dot{L}_r}{r(1 - M_r)^2} \right] \text{ret} dS + \int_{f=0} \left[ \frac{L_r - L_M}{r^2 (1 - M_r)^2} \right] \text{ret} dS$$

$$+ \frac{1}{c} \int_{f=0} \left[ \frac{L_r \left( r \dot{M}_r + c \left( M_r + M^2 \right) \right)}{r^2 (1 - M_r)^3} \right] \text{ret} dS$$ \hspace{1cm} (7.8)
The dot over a variable implies source-time differentiation of that variable, and a subscript $r$ or $n$ indicates a dot product of the vector with the unit vector in the radiation direction, $\hat{r}$, or outward surface normal direction, $\hat{n}$, respectively. The moving surface considered in the integration is defined by the function $f(\bar{x}, t) = 0$, $\bar{M}$ is the local surface velocity vector divided by the freestream sound speed. The subscript $ret$ denotes that the integrand is evaluated at the retarded time. The vector components and are defined as:

$$U_i = \left[ 1 - \frac{\rho}{\rho_0} \right] v_i + \frac{\rho}{\rho_0} u_i$$  \hspace{1cm} (7.9)$$

$$L_i = P_{ij} \hat{n}_j + \rho u_i (u_n - v_n)$$ \hspace{1cm} (7.10)$$

When the surface is coincident with the actual blade surface these reduce to $U_i = v_i$ and $L_i = P_{ij} \hat{n}_j = l_i$ where the surface force components are determined from the unsteady CFD computations. For more information, see Reference [149].

### 7.6.2 Coupling of PUMA2 and PSU-WOPWOP codes

The underlying philosophy of the Computational Aeroacoustics (CAA) analysis of wind turbines [149] is the realization that a single full three-dimensional LES simulation of a full wind turbine flow, which includes the flow details in sufficient detail to allow all noise mechanisms to be simulated, is computationally too expensive: at least for the foreseeable future. However, the three-dimensional flow properties of rotating blades are an essential feature of any wind turbine aerodynamic or aeroacoustic simulation.
Furthermore, the acoustic propagation is of interest at relatively large distances from the wind turbine, so it is impractical to use the CFD simulation alone to propagate the radiated sound. Thus, the two centerpieces of the approach are the full three-dimensional simulations of rotating blades and the far-field noise predictions based on the FW-H method. For this purpose, the coupling of the CFD code PUMA2 and the aeroacoustic code PSU-WOPWOP has been done during this study.

WOPWOP, which has been developed by Brentner [21], is a series of rotor noise prediction codes to predict the noise from helicopter rotors and tiltrotors. These codes are typically used to predict the thickness and loading noise of helicopter rotors, and can be applied directly to compute the rotational noise of wind turbines. The FW-H solver, PSU-WOPWOP [22], uses a permeable surface implementation of Farassat’s Formulation 1A of the FW-H equation (Equation 7.6). The integrals in equations (7.7) and (7.8) are evaluated numerically by subdividing the integration surface into panels. Each panel is treated as if it were acoustically compact; hence, the size of the panels is related to the minimum wavelength that can be resolved. Similarly, the temporal resolution of the input data is related to the highest frequency that can be computed by the numerical method. It is possible to assume that individual panels or groups of panels account for acoustic signals that are not correlated. When this is done the acoustic pressure from each group of panels is added on a \((p')^2\) basis before the spectrum is computed. This process has been used for landing gear noise prediction [204] and may prove useful for the computation of wind turbine broadband noise. PSU-WOPWOP can support both face- and node-centered unstructured data formats. Faces with any number of nodes (up to an arbitrary limit of 10) are supported, and different face types can be used in the
definition of a single surface. The integration procedure uses the centroid of the faces as the location of the areas and forces for the unstructured data.

PUMA2 is an unstructured finite volume CFD flow solver described in Chapter 3. From the wind turbine rotor flow field simulations, the time accurate inviscid flow solutions are stored every 1° interval in the 3rd revolution for cases II and III, to be used for the aeroacoustic calculations. The total size of the stored inviscid solution files for one full revolution with 1° interval is about 60 GB. (The file size of the Grid A is about 325 MB, the solution file for this grid is 170 MB, volume Tecplot file is 90 MB, and blade surface Tecplot file is 800 KB). In addition, for the yawed case II of LES computations, the blade surface pressure data has been stored for every 0.1 degrees. Therefore, the CFD simulations performed for both inviscid and LES cases, require huge storage space, and additional wall-time for post-processing and analysis. Several utility codes have been written for post-processing of the stored flow solution from PUMA2 to obtain the necessary data required for PSU-WOPWOP. The time-accurate blade surface pressure data at the face centers (centers of each unstructured triangular faces on the blade surface), and also the information of the normal, center and area of each surface face at the initial blade positions, which are all required for the aeroacoustic calculations, can be extracted from the stored flow solutions by post-processing or directly from the PUMA2 computations. Similarly, the flow data can easily be extracted on the permeable FW-H surface as seen in Figure 7.9, which was embedded in the grid system during the grid generation process, either by post-processing the stored solution files, or directly from the computations as the code runs. Figure 7.33 shows example of instantaneous pressure contours on the permeable surface embedded in the unstructured grid, which will be
input to PSU-WOPWOP. This permeable surface is a closed outer surface of a cylinder that surrounds both of the turbine blades, and it is also composed of unstructured triangles. When the required flow data on the blade surface or permeable surface can be extracted from the CFD computations as the flow solver runs, the whole flow solution may not need to be stored as often as it is done for case II of LES simulations.
In this thesis, unsteady, separated flows around complex engineering geometries are investigated through the use of Computational Fluid Dynamics. Different flow problems are studied with increasing computational complexity:

1) Unsteady inviscid simulation of the airwake around a full-scale complex ship geometry (LHA) using stationary unstructured grids.

2) Test cases for moving grid capability: unsteady inviscid simulation of a helicopter rotor in hover, and unsteady viscous simulation of the flow around a pitching wing (using MILES approach).

3) Large Eddy Simulation of a wind turbine rotor flow field using rotating unstructured grids.

The three dimensional, unsteady, parallel, unstructured, cell-centered, finite volume flow solver PUMA2 has been used for all CFD simulations. The unstructured grids around the complex geometries were generated using the Gridgen software.

8.1 Ship Airwake Simulations

Both steady and time-accurate inviscid CFD simulations of the airwake of an LHA class ship were performed using PUMA2 flow solver with unstructured grids on a full-scale ship model [117–119,192]. A comprehensive study of the effect of WOD angle
on the airwake was performed. The time accurate runs were performed for WOD angles of 0, 30, 45, 90, 270 and 315 degrees. The time averaged solutions were also analyzed to investigate the characteristics of the wake.

The unsteady nature of the wake was highlighted and a quantitative measure of the unsteadiness was presented. The CFD results showed significant time-varying flow effects over the selected landing spots. The time accurate CFD results showed significant temporal and spatial variation that may be responsible for additional pilot workload. A general agreement of the dominant shedding frequencies for LHA simulations with previous results is observed. In addition, a reasonable comparison with the experimentally observed velocity field over the LHA deck is obtained.

The long time histories (for a period of about 50 seconds) are also obtained for the WOD angles of 0 and 30 degrees to investigate the airwake over the selected landing spot and to be used in the dynamic interface simulations to model a UH-60A operating off an LHA class ship. The results from the CFD simulations have been successfully interfaced with the flight dynamics simulations to estimate pilot workload due to complex unsteady airwake [117–119]. Thus, a fundamental high fidelity simulation testing tool has been developed to provide insight into the helicopter/ship dynamic interface problem.

The results clearly indicate that the time-varying airwake has a significant impact on aircraft response and pilot control activity when the aircraft is flown for specified approach and departure trajectories. The differences are most notable when the helicopter is operating in or near a hover relative to the ship deck (stationkeeping). In the past, gust models for fixed-wing aircraft simulation have often used a stationary or frozen field model. This is adequate when the aircraft is moving at a significant forward speed.
However, the model clearly breaks down as airspeed approaches zero. The same appears to be true of helicopters operating in turbulent ship airwake. The time-varying nature of the ship airwake becomes dominant as the helicopter approaches hover. And, the 30° WOD condition showed a substantial increase in pilot workload.

The use of time-accurate ship airwake data was found to present some practical implementation difficulties, in that the method requires that the simulation handle large quantities of data. For every grid point a set of time history data must be stored for each component of velocity. Memory storage can become an issue, particularly if the simulations are to be run in real-time, in which case accessing data from disk storage may not be feasible. It was helpful to select a subset of the flow field when performing the simulations in which the landing spot is known. However, for real-time simulations the pilot might want to access different deck spots during the same simulation run. The use of stochastic airwake model (based on the time-accurate CFD results) might be an attractive alternative [89, 116]. However, to develop such models, the unsteady flow information is still needed for different WOD conditions and for different locations on the ship.

It is quite apparent that the time-varying ship airwake is very significant for helicopter/ship dynamic interface testing. These ship airwake effects can increase pilot workload and possibly degrade handling qualities during shipboard launch and recovery operations.

Future work for ship airwake simulations may include modeling of a moving flight deck because of random sea motion. The effects of the atmospheric environment (atmospheric boundary layer and turbulence) and high winds are important, and these can
be modeled using boundary conditions or using momentum source terms (for gust and turbulence) in the governing equations. The complex interaction of ship airwake and helicopter rotor and fuselage wakes could be investigated as future work. This can be done by modeling the rotor with momentum source terms.

8.2 Rotor and Pitching Wing Simulations

The moving grid capability is successfully implemented to PUMA2, and two sample flow simulations are performed for a helicopter rotor in hover and a pitching wing. The pressure distributions from the time-dependent, inviscid helicopter rotor simulations with rotating unstructured grids showed good agreement with the experimental values. The grid refinements around the rotor blades and in the wake closed to the helical tip vortex simulations improved the capturing of vortical flow features of the rotor wake. The contraction of the wake and the tip vortices are visualized. Future simulations for rotors could include LES with wall models.

The results of the MILES simulation of a pitching wing using unstructured rotating grids with hexahedral cells showed the complex unsteady vortical flow features of a pitching wing qualitatively. Further analysis of the flow data may be performed in future studies to investigate this highly three-dimensional complex flow. Computations for more revolutions with better grid resolutions or using wall models could be performed in future.
8.3 Computational Steering and Visualization

POSSE has been successfully coupled to the C/C++ based parallel CFD code, PUMA2, to interact with and visualize the large-scale parallel CFD simulations in a scalable way while the simulations are running on the parallel computers [128,143,144,191]. POSSE, a general-purpose computational steering and monitoring library, is a software system with a simple user interface. Scientists and engineers often need an easy-to-use software system such as POSSE. The advanced object-oriented features of C++ have given POSSE an edge over existing steering libraries written in older languages. It is very important to have libraries such as POSSE to visualize large-scale parallel simulations. Large parallel aerospace simulations need scalable visualization solutions such as these. Benefits of scalability and dimensional reduction arising from this approach were imperative in the development of this computational steering system.

POSSE has been successfully tested with the results of the rotor flow field computation in hover and is compared with experimental measurements using the Tecplot integration feature of the POSSE GUI. In addition, several complex flow problems have been discussed that are extremely difficult to understand using conventional two-dimensional graphics. Isosurfaces in three-dimensional space can be used to highlight some flow features, but these really need to be viewed using stereographics. For four-dimensional problems that change in both space and time, it will be essential to use stereographics and virtual reality systems to understand the time-dependent nature of complex flow features.
At a more basic level, the ability to interact and visualize a complex solution as it unfolds, and the real-time nature of the computational steering system, opens a whole new dimension to the scientist and engineer for interacting with their simulations. POSSE is the best interactive computational steering and monitoring approach, and people who are performing such large-scale parallel aerospace simulations could use it or a similar approach.

8.4 Wind Turbine Rotor LES Simulations

Time-accurate inviscid and LES simulations were performed for the 2-bladed NREL Phase VI wind turbine rotor using the 3-D, unsteady, parallel, finite volume flow solver, PUMA2, with rotating unstructured tetrahedral grids [149, 190]. Three different flow cases with different wind speeds and wind yaw angles were investigated. For the higher wind speed case, a grid with more clustering near the blade compared to the inviscid grid was used in the LES simulations. The LES simulations were performed for the quiescent air cases by using classical Smagorinsky model to model the sub grid-scale stresses. The instantaneous log-law wall model is implemented successfully to approximate the boundary layer effects near wall regions and applied in these simulations.

The log-law was tested with the LES simulations of flow over a sphere [96]. The application of instantaneous log-law wall model had the effect of markedly improving the skin friction distribution for the high-Reynolds number flow solution around a sphere. This presented an opportunity to simulate practical flows with high Reynolds number on complex bodies with reasonable accuracy. For the rotating blade simulations, the rotational grid velocities are taken into consideration in the use of wall model.
Results from the inviscid and LES simulations for the three rotor cases and comparisons with the experimental data were presented. The results show that the flow is attached for the pre-stall cases (I and II) with 0° and 30° yaw angles, with the latter having an asymmetrical wake structure, whereas there is massive separation over the entire blade span in the post-stall case (III), which has a higher wind speed of 15 m/s. Comparisons of sectional pressure coefficient distributions with experimental data show good agreement. Considerable spanwise pressure variations, in addition to the chordwise variations, are also observed in all three cases. The difference between the inviscid and the viscous computational cases has been discussed. Comparisons with the experiments show that the thrust coefficients from the LES results are better predicted compared to the inviscid computations but they are still overpredicted 6-25% depending on the inlet flow condition (for cases I and III).

The present unsteady moving grid simulations with the PUMA2 code will provide a basis for the simulations of more complex cases to investigate the interactions due to the turbine tower shadow effects for the downwind configurations as in Reference [74], and the interactions due to atmospheric gust, turbulence and atmospheric shear layer. Also, further modifications are being performed on PUMA2 in order to improve the code’s computational efficiency for LES simulations.

These three-dimensional and time-accurate CFD results can be used for the far-field noise predictions based on the Ffowcs Williams - Hawkings method [57], which can provide a first-principles prediction of both the noise and the underlying turbulent flow that generates the noise, in the context of the wind turbine application [149]. The calculated three-dimensional mean flow could also provide the basis for detailed unsteady
flow simulations based on the Nonlinear Disturbance Equations (NLDE) [148] which in
turn can provide the aerodynamic input needed for the prediction of trailing edge and
tip vortex noise [149].

Future work for wind turbine simulations could include detailed simulations of
tower shadow, atmospheric boundary layer and turbulence effects by using appropriate
boundary conditions or momentum source terms. Aeroacoustic calculations using the
time-accurate LES results remain as future work.
Appendix

A sample of PUMA2 Input File

The standard input file has been changed, to take into account the use of the rotating grid capability with a flag of "RotorSim", and "Omega" showing the rotational speed.

```
Gamma  R   rhoRef Vref  muRef
1.4  287.04 1.0  1.0  1.0
rho_inf u_inf  v_inf  w_inf  p_inf  xyzMult  fsMult
1.225 0.0 7.0 0.0 101325.0 1.0 1.0
mu_inf  T_Suth  PrLam  PrTurb
1.7894e-5 110.0 0.72 0.9
numIts  maxMinutes  wrRestart  wrResid  relResid  absResid
5000  200000  100  5  1.0e-6  1.0e-16
CFLconst  CFLslope
0.8  0.0
localStepping  spatial_order  inviscid_flux  integ_scheme  limiter
0  2  "roe"  "runge-kutta"  "none"
innerIters  innerTol  omega  stages  K  commScheme
50  0.1  1.0  4  5.0e+2  "delta q"
gridName  restartFrom  restartTo  residName
"grid.sg.gps"  "rotor.rst"  "rotor.rst"  "rotor.rsd"
implicitBCs  viscousModel  artDiss  VIS-2  VIS-4  M_CR  Kexact
0  "inviscid"  0  0.25  0.25  0.00  0
Num_surfaces  initialCond
3  0
Hybrid_method  InitialFlow  RotorSim  Omega
0  0  1  7.54
surface  BC_type  rho  u  v  w  p
1  21
2  21
3  5
```
References


[123] Lighthill, M. J.


229


234


Vita

Nilay Sezer-Uzol was born on August 16, 1976 in Balikesir, Turkey. She graduated from Ankara Fen Lisesi (Ankara Science High School) in June 1993. She received her Bachelor of Science in Aeronautical Engineering from the Middle East Technical University, Ankara, Turkey in June 1998. She then received her Master of Science in Aerospace Engineering from the Pennsylvania State University in May 2001. She has been a graduate student in the PhD program of Department of Aerospace Engineering at the Pennsylvania State University since August 2001.