Numerical Simulation and Control of Separated Incompressible Flows

by

Kei Yuen Tang

B.S.E., Aerospace Engineering,
University of Michigan, Ann Arbor, 1994

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

Master of Science in Aeronautics and Astronautics

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1996

Author .....
Department of Aeronautics and Astronautics
May 10, 1996

Certified by. .....
Jaime Peraire
Associate Professor
Thesis Supervisor

Accepted by .....
Professor Harold Y. Wachman
Chairman, Departmental Graduate Committee

Numerical Simulation and Control of Separated Incompressible Flows

by

Kei Yuen Tang

Submitted to the Department of Aeronautics and Astronautics
on May 10, 1996, in partial fulfillment of the
requirements for the degree of
Master of Science in Aeronautics and Astronautics

Abstract

The idea of controlling an unstable or unsteady fluid flow in order to optimize its characteristics in some way (by minimizing the drag on a body, for example) is highly attractive, and has a wide range of potential applications. However, the implementation of this concept is contingent on the availability of a model of the flow which is suitable for control system design. Thus, to date, applications have been restricted to situations where it is possible to derive simple, heuristic models describing the flow behavior and control system action. In many cases of interest though, such models are not available *a-priori*, and one is faced with the prospect of using expensive full-scale numerical simulations in the control design process. In this thesis we first describe the details of development and result for a full Navier-Stokes solver using the Fractional-Step Finite Element method. Then by performing the Proper Orthogonal Decomposition on the data generated by the simulation, a low-order system of ordinary differential equations is developed. Finally, the low-order system is used as the basis for the optimal control system design.

Thesis Supervisor: Jaime Peraire
Title: Associate Professor

Acknowledgments

During the research process, I got much help from many different individuals. The ones that I especially want to acknowledge my sincere appreciation are:

- My advisor, Jaime Peraire, for his exceptional patience and support on my research work. Everytime I dropped by his office for questions; even though he was in the middle of his busy schedule, he would always be willing to be interrupted and spend time with me to discuss the questions that I had and gave me a new way of thinking on how to approach the problem. It was a real pleasure to learn from and work with him in the past year.

- Prof. Will Graham who was visiting from the University of Cambridge, England. Through many useful discussions with him, I learned a deeper understanding on my research.

- My labmates in the CASL and SPPL lab for their company on those long nights and also for their help on many ignorance questions that I had.

- Unified TA's and students, for their efforts to make the Unified engineering class that I am in charge of went very smoothly. Being the head TA of Unified gave me a great opportunity to learn both teaching and administrative skills.

- My friends and my roommate, Tony Ng, for their company and support during the past one and a half year that I have been here at MIT.

- My host family, Donnald, Margaret, Katherine and Kristoper Lee, for their continuing love and support from the beginning when I first came to the U.S. six years ago.

- My beloved parents for their supports on studying here in the U. S. Even though they are far away from me, they always give me good advises and provide me with the best that they have got. Thank you so much, dad and mom.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

1.1.1 Motivation for the Study of Active Flow Control

Control of fluid systems is currently of great interest in many different engineering disciplines. For example, aerodynamicists may want to stabilize the flow over an airfoil in order to achieve a higher lift to drag ratio. In contrast, thermodynamicists may want to destabilize the flow in engines to achieve better mixing. Nowadays, active flow control is attractive in a wide range of scenarios including jet engine flow, bluff body wakes, noise reduction, turbulence and aircraft flows.

To date, active flow control done by the jet propulsion community has been quite successful. Considerable research has been performed to evaluate the ability of controlling the flow passing through the jet engine. By using active feedback control in the inlet and the compressor, operating pressure ratio at which engine stalls was increased, leading to improved engine performance [24].

Preliminary results on active flow control were obtained at the department of Ocean Engineering at the Massachusetts Institute of Technology. Experimental investigation of active vorticity control in bluff body wakes by controlling the motion of a flapping foil was done by Triantafyllou et al [9, 30, 31]. They found that the incoming vortices of the Von Karman vortex street generated by a circular cylinder

were repositioned and their strength was changed by the flapping foil, resulting in new stable patterns downstream from the foil. From analyzing these data, they are hoping to develop an optimal feedback controller which would reduce the drag on the system while extracting energy from the wake.

The control of mixing layers [8] and free-shear flow coherent structures [26] are now a very active research area. Also, some promising experimental results by using active control were obtained for both cavity flows and for supersonic screech. Issues on gust alleviation and flutter/buffetting on aircraft flows have many potential applications by using active flow control.

1.1.2 C. F. D. and Flow Control

Most of the successful flow control done to date has been in cases where linear analysis works; however, since many other important flow problems involve non-linear aspects of fluid mechanics, it is desirable to extend applications to non-linear flow problem. However, in those cases, the application of the available theoretical results (mathematical models) is very limited and in most cases not very practical. As a result, most flow control analysis nowadays has been done by using either experimental or computational methods.

In experimental methods, by operating many runs on the setup, suitable data could be obtained and an optimal controller could be developed. However, the costs of setting up and running the experiments for many times can be significant, and the time needed is also of the order of weeks if not months.

In the past two decades, computer speed has been increased by a significant amount and at the same time, the cost for doing computation is getting lower. Also, a more rigorous mathematical background for the numerical methods has been developed and so computations have become more reliable. Thus, the Navier-Stokes equations may now be solved for complex geometries with a relatively short computational time and a low cost. In other words, after a CFD code is developed and verified, many runs can be done in a relatively short time and at a much lower running cost than the experimental method.

In principle, optimal control theory could be used with finite difference or finite element models but they are very expensive since these models do not have the forms that are suitable for control tools. For example, both heuristic models with very simple control laws and "blind" models such as Neural Nets had been investigated by many researchers in the past. However, all of these approaches are problem dependent and often suboptimal. The goal of this thesis research is to show that CFD can be used very effectively to develop flow control strategies not by directly analyzing the solution by some means, but by deriving a low-order Navier-Stokes equation model which is very suitable for control analysis.

The low order model, which is a set of ordinary differential equations, is derived from the numerical simulation by performing a proper orthogonal decomposition on the flow solution generated by the Navier-Stokes solver, thereby generating a set of spatial functions which may be used as a basis for a Galerkin projection of the Navier-Stokes equations. By integrating these equations with respect to time, the uncontrolled flow can be faithfully reproduced and predicted. It is then possible to introduce control into the problem, and use the low order model to predict the effect of the control action.

1.2 Flow Control Considered in This Thesis

In this thesis, we made our effort into controlling the Von Karman vortex street behind a two-dimensional cylinder by rotating the cylinder as a means of actuation. Rotating cylinder flow was chosen as our control test case because it is relatively simple and able to be extended to other real flow applications.

1.3 Overview

This thesis is split into two parts. The first part describes the theoretical background and results of the full incompressible Navier-Stokes simulation. The second part discusses the derivation of the low-order system of ordinary differential equations and

how it was used as a tool for the control system design.

Chapter 2 presents the governing mass and momentum partial differential equations. The arbitrary Lagrangian-Eulerian formulation is also considered because it takes into account the effect of the moving foil in the flow field. Chapter 3 describes the fractional step finite element algorithm which was used to discretize the arbitrary Langangian-Eulerian formulation of the incompressible Navier-Stokes equations.

The discretization of the computational domain by using the unstructured triangulation method from the Felisa system [20] is described in Chapter 4. An algorithm that can treat realistic motions and deformations of any configuration discretized into unstructured triangles is also discussed here.

Chapter 5 gives the solution techniques required to solve the set of algebraic equations $A * x = b$ derived from the governing partial differential equations. Since the matrix A is a sparse matrix, an iterative method (Preconditioned Conjugate Gradient) was used together with the incomplete LU preconditioner.

The results of four test cases including the half cylinder case (inviscid), the full cylinder case (viscous), the rotating cylinder case and the moving cylinder with moving foil case are described in Chapter 6. In Chapter 7, we discuss the reduced order model of the Navier-Stokes equation and how it was used as the basis for developing and analyzing an optimal control model. Conclusion and future development are then given in the last Chapter.

Chapter 2

Governing Equations and Boundary Conditions

2.1 Incompressible Navier-Stokes Equations

In this section we describe the governing equations and boundary conditions for a two-dimensional, incompressible, viscous, flow of a Newtonian fluid for a stationary control volume. For a moving control volume, it is necessary to use the arbitrary Langangian-Eulerian governing equations which take into account the velocity of the moving control surfaces. This formulation will be discussed in the next section.

In differential form, a viscous, incompressible flow is governed by both the conservation of mass equation :

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0 \tag{2.1}$$

and the momentum equations :

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + \frac{1}{\rho}\frac{\partial p}{\partial x} = \nu(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}) \tag{2.2}$$

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + \frac{1}{\rho}\frac{\partial p}{\partial y} = \nu(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}) \tag{2.3}$$

This set of governing equations is written in the 2-D Cartesian coordinate system (x, y), using primitive variables : u, v which are the x- and y- components of velocity,

and p which denotes the pressure. The kinematic viscosity, ν, and density, ρ, are assumed to be constant with respect to both time and spatial domain.

2.2 Arbitrary Langangian-Eulerian Formulation

The governing equations described in the last section are suitable for a flow problem with a fixed control volume. However, these governing equations have to be modified for a moving control volume. By using the arbitrary Langangian-Eulerian formulation, the motion of the control surfaces is taken into account by fluxes going into and out of the control volume. Details on the formulation and applications can be found in the literature [11, 10, 19, 25].

In Cartesian coordinates, let the x- and y- components of velocity of the moving control volume to be w_1 and w_2 respectively. The conservation of mass for that control volume now becomes:

$$\frac{\partial(u-w_1)}{\partial x}+\frac{\partial(v-w_2)}{\partial y}=0 \tag{2.4}$$

and the conservation of momentum equation becomes :

$$\frac{\partial u}{\partial t}+(u-w_1)\frac{\partial u}{\partial x}+(v-w_2)\frac{\partial u}{\partial y}+\frac{1}{\rho}\frac{\partial p}{\partial x}=\nu\,(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}) \tag{2.5}$$

$$\frac{\partial v}{\partial t}+(u-w_1)\frac{\partial v}{\partial x}+(v-w_2)\frac{\partial v}{\partial y}+\frac{1}{\rho}\frac{\partial p}{\partial y}=\nu\,(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}) \tag{2.6}$$

Notice that the momentum equation can also be expressed in a compact form given in equation (2.7) below:

$$\frac{\partial \mathbf{u}}{\partial t}+\mathbf{A}\,\frac{\partial \mathbf{u}}{\partial x}+\mathbf{B}\,\frac{\partial \mathbf{u}}{\partial y}+\frac{1}{\rho}\,\nabla p=\nu\,\nabla^2\mathbf{u} \tag{2.7}$$

where $\mathbf{u}=(u,v)$, and $\mathbf{A}$, $\mathbf{B}$ are matrices defined as :

$$\mathbf{A}=\begin{pmatrix} u-w_1 & 0 \\ 0 & u-w_1 \end{pmatrix};\quad \mathbf{B}=\begin{pmatrix} v-w_2 & 0 \\ 0 & v-w_2 \end{pmatrix}$$

By using these governing equations, a flow with moving boundaries may be accu-

rately represented.

2.3 Boundary Conditions

The incompressible Navier-Stokes equations have to be solved in a bounded domain, $\Omega \subset \Re$, together with velocity and pressure boundary conditions (BC) for all the outer and inner boundary points. Assume that the boundary can be decomposed in two complementary parts, Γ_g and Γ_h, such that $\Gamma = \Gamma_g \cup \Gamma_h$ and $\Gamma_g \cap \Gamma_h = \emptyset$.

The BC are of the Dirichlet type on Γ_g:

$$\mathbf{u}(x, y) = \mathbf{g}(x, y) \; on \; \Gamma_g \tag{2.8}$$

with $\mathbf{g}$ a given vectorial function defined on Γ_g. Examples are no-slip conditions at solid walls, $u = v = 0$, or imposed velocity profiles at a channel inlet, $u = u_{in}$ and $v = 0$.

On Γ_h, derivative type (Neumann) BC can be specified. The specific form of these BC depends on the weak weighted residual formulation of the governing equations. In this solver, they are written as follows:

$$\sigma \cdot \mathbf{n} = \mathbf{h}(x, y) \; on \; \Gamma_h \tag{2.9}$$

with $\mathbf{n} = (n_x, n_y)$ the unit outward normal on Γ_h, and $\mathbf{h}$ a given vectorial function defined on Γ_h. The traction force tensor, σ, is given by:

$$\sigma = \begin{bmatrix} -p + \mu\frac{\partial u}{\partial x} & \mu\frac{\partial u}{\partial y} \\ \mu\frac{\partial v}{\partial x} & -p + \mu\frac{\partial v}{\partial y} \end{bmatrix} \tag{2.10}$$

with, μ, the dynamic viscosity. For example, one could apply zero traction forces, $\sigma \cdot \mathbf{n} = 0$ at the inlet and outlet of a channel in the x-direction, $n_x = 1$ and $n_y = 0$.

In the discussion above we generalized the boundary conditions to any flow field

described by using the governing equations mentioned in the last section. However, more specific BC will be discussed in the test cases chapter with regard to the actual examples considered in this thesis.

Chapter 3

Finite Element Solution Algorithm

3.1 Finite Element Method

The Finite Element Method (FEM) is a general description of many variant methods existing for numerical discretization, all of which have some characteristic procedures in common. Therefore, it is actually more appropriate to speak about Finite Element Techniques instead. In the remainder of this section we will describe the history and the general procedures for the finite element techniques.

The Finite Element Techniques originated from the field of structural analysis as a result of many years of research, mainly between 1940 and 1960. The concept of 'elements' can be traced back to the techniques used in stress calculations, whereby a structure was subdivided into small sub-structures of various shapes and re-assembled after each 'element' had been analyzed. After having been applied with great success to a variety of problems in linear and non-linear structural mechanics it soon appeared that the method could also be used to solve continuous field problems (Zienkiewicz and Cheung, 1965). From then on, the finite element method was used as a general approximation method for the numerical solution of physical problems described by field equations in continuous media. In the past two decades this method has become popular for researchers working in the field of computational fluid dynamics. An

excellent introduction to the subject can be found in [22].

The first step for the FEM is to discretize a computational domain by subdivision of the continuum into elements of arbitrary shape and size. Since any polygonal structure with rectilinear or curved sides can finally be reduced to triangular and quadrilateral figures that later are the basis for the space subdivision. Within each element a certain number of points are defined, which can be positioned along the straight sides or inside the element. These nodes will be the points where the numerical value of the unknown functions, and eventually their derivatives, will have to be determined. The total number of unknowns at the nodes, function values and eventually their derivatives are called the degrees of freedom of the numerical problem, or nodal values.

Secondly, the field variables are approximated by linear combinations of known basis functions (also called shape, interpolation or trial functions). In standard finite element methods the interpolation functions are chosen to be locally defined polynomials within each element, being zero outside a considered element. A detailed description of the piecewise linear trial function can be found in the next section.

Finally, the most essential and particular step of the finite element approximation is the definition of an integral formulation of the physical problem equivalent to the field equations to be solved. The most popular method is the weak formulation, or method of weighted residuals. The details of this formulation for our incompressible Navier-Stokes equations can be found in the last section of this chapter.

In conclusion, the FEM can be seen as a tool to transform a continuum problem governed by a set of PDE's into a system of non-linear algebraic equations which can be solved by using a computer.

3.2 Linear Piecewise Trial Function

Our flow problem is described in two-dimensional Cartesian coordinates and so it consists of three function values u, v and p on each point (x,y) of the domain, Ω, and its boundary, Γ. As in general no analytical solutions u(x,y), v(x,y), p(x,y) can

be obtained, and so the flow equations are discretized and the approximate solution function $\hat{u}(x,y)$,$\hat{v}(x,y)$ and $\hat{p}(x,y)$ are being searched for. A finite number of points in the domain is selected and the unknowns of the problem are defined to be the velocity components and the pressure at these points. In this solver, the unknowns are located at the vertices $[n_i]_{i=1,3}$ of the triangular elements. As a result, there are 9 unknowns in each element, namely, $[u_i, v_i, p_i]_{i=1,3}$.

The shape of the approximate solution functions is also prescribed when using the FEM. In this solver, piecewise linear functions are chosen, both for the velocity components and the pressure. In finite element terminology, this is referred to as the use of P1/P1 elements. Note that, the $[u_i]_{i=1,3}$ values in an element are sufficient to describe the linear variation over the element of the approximate solution $\hat{u}(x,y)$, and that the same goes for $\hat{v}(x,y)$ and $\hat{p}(x,y)$.

The piecewise linear approximation solutions can be written as a linear combination of specific piecewise linear interpolation functions. These functions form a basis from the trial solution function space. Hence, they are sometimes called the 'basis functions'. To each point j at which unknowns are stored, a 'tentshaped' interpolation function $N_j(x,y)$ is associated. This is illustrated in figure (3-1).

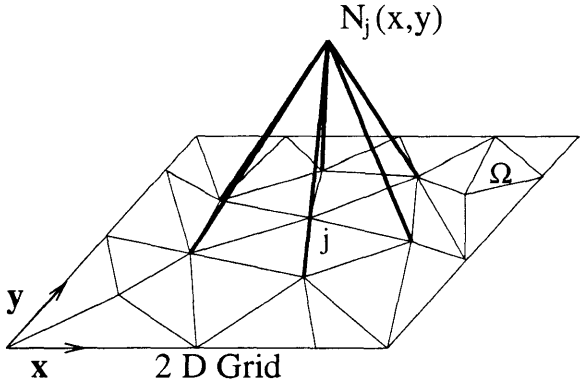

Figure 3-1: Piecewise Linear Interpolation Function $N_j(x,y)$

Hence the vertices of the elements are called interpolation nodes. Notice that in a P1/P1 element the interpolation nodes coincide with the geometrical nodes used to describe the geometry of the triangle. The piecewise linear interpolation functions

are defined only in the elements to which the corresponding node belongs. Furthermore, they have a unit value at that particular node, vary linearly in its surrounding elements and are zero at all other nodes:

$$N_j(x_i, y_i) = \delta_{ij}, \tag{3.1}$$

for any two nodes i and j. The approximate solution functions can then be written as follow :

$$\hat{u}(x, y) = \sum_j u_j N_j(x, y) \tag{3.2}$$

$$\hat{v}(x, y) = \sum_j v_j N_j(x, y) \tag{3.3}$$

$$\hat{p}(x, y) = \sum_j p_j N_j(x, y) \tag{3.4}$$

in which the coefficients u_j, v_j and p_j are the unknowns of the discretized problem.

The derivatives of the piecewise linear approximation functions are also defined. For example, the x- and y- derivative of the pressure field p can be represented as:

$$\frac{d\hat{p}(x, y)}{dx} = \sum_j p_j \beta_j \tag{3.5}$$

$$\frac{d\hat{p}(x, y)}{dy} = \sum_j p_j \gamma_j \tag{3.6}$$

where β_j and γ_j are the x- and y- derivatives of the piecewise linear interpolation function $N_j(x, y)$, which are constants for each element.

3.3 Fractional Step Algorithm

3.3.1 General Principle

The incompressible mass and momentum equations are simulated using a finite element discretization [12] of the following three-fractional-step time-marching scheme :

$$\frac{\mathbf{u}^* - \mathbf{u}^{(n)}}{\Delta t} = -\mathbf{u}^{(n)} \cdot \nabla \mathbf{u}^{(n)} + \nu \nabla^2 \mathbf{u}^{(n)} \tag{3.7}$$

$$\nabla^2 p^{(n)} = \rho \frac{\nabla \cdot \mathbf{u}^*}{\Delta t} \tag{3.8}$$

$$\frac{\mathbf{u}^{(n+1)} - \mathbf{u}^*}{\Delta t} = -\frac{\nabla p^{(n)}}{\rho} \tag{3.9}$$

Here $\mathbf{u}^{(n)}$ consists of the x- and y- components of velocity and $p^{(n)}$ represents the pressure. The superscripts denote the time level of each variables, ρ and ν are the fluid density and kinematic viscosity respectively, and Δt is the time step.

By solving equation (3.7) with the known values of $\mathbf{u}$ at time level n, we obtain the solution for $\mathbf{u}^*$. Substituting it into equation (3.8), we obtain the pressure solution at time level n. And by using the pressure solution $p^{(n)}$, equation (3.9) solves for the velocity vector components at time level $n+1$.

Notice that steps (3.7) and (3.9) together represent an explicit time-stepping solution to the Navier-Stokes equations;

$$\frac{\partial \mathbf{u}}{\partial t} = -\mathbf{u} \cdot \nabla \mathbf{u} - \frac{\nabla p}{\rho} + \nu \nabla^2 \mathbf{u} \tag{3.10}$$

while the pressure calculation in step (3.8) enforces the incompressibility condition $\nabla \cdot \mathbf{u} = 0$. Stability of the convective term is accomplished by including a higher order approximation to the time derivative in step (3.7), which appears on the right hand side in the form of an artificial viscosity.

3.3.2 Weak Formulation

In this section we describe the weighted residual method used to formulate the three equations shown in the last section.

Step One :

The Taylor series of $\mathbf{u}^{n+1}$ can be expressed as:

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t \left.\frac{\partial \mathbf{u}}{\partial t}\right|^n + \frac{\Delta t^2}{2} \left.\frac{\partial^2 \mathbf{u}}{\partial t^2}\right|^n + O(\Delta t^3) \tag{3.11}$$

However, by truncating the higher order terms and substituting $\frac{\partial \mathbf{u}}{\partial t}$ and $\frac{\partial^2 \mathbf{u}}{\partial t^2}$ which are evaluated by using equation (3.7), we obtain an intermediate updated value $\mathbf{u}^*$ as :

$$\begin{aligned} \mathbf{u}^* - \mathbf{u}^{(n)} = -\Delta t \, [\mathbf{A} \, \frac{\partial \mathbf{u}}{\partial x} + \mathbf{B} \, \frac{\partial \mathbf{u}}{\partial y} - \nu \, (\frac{\partial^2 \mathbf{u}}{\partial x^2} + \frac{\partial^2 \mathbf{u}}{\partial y^2})] + \\ \frac{\Delta t^2}{2} \, [\mathbf{A} \, \frac{\partial}{\partial x}(\mathbf{A} \, \frac{\partial \mathbf{u}}{\partial x} + \mathbf{B} \, \frac{\partial \mathbf{u}}{\partial y}) + \mathbf{B} \, \frac{\partial}{\partial y}(\mathbf{A} \, \frac{\partial \mathbf{u}}{\partial x} + \mathbf{B} \, \frac{\partial \mathbf{u}}{\partial y})] \end{aligned} \tag{3.12}$$

where

$$\mathbf{A} = \begin{pmatrix} u & 0 \\ 0 & u \end{pmatrix} \; ; \quad \mathbf{B} = \begin{pmatrix} v & 0 \\ 0 & v \end{pmatrix}$$

Now we can use the weighted residual method to solve for the unknown $\mathbf{u}^*$. The left side of equation (3.12) becomes :

$$\int_\Omega (\mathbf{u}^* - \mathbf{u}^{(n)}) \; N_l \; d\Omega \tag{3.13}$$

Notice that both for here and the rest of the formulations the weighting function will be chosen as the piecewise linear shape functions. By substituting $\mathbf{u}^*$ and $\mathbf{u}^{(n)}$ as the linear trial function described in equation (3.2), (3.3) and (3.4), the Galerkin form of the left hand side can then be expressed immediately as

$$\mathbf{K} \; \mathbf{\Phi} = \text{RHS} \tag{3.14}$$

where $\mathbf{\Phi}$ is a column vector consist of the unknown values $\mathbf{u}^* - \mathbf{u}^{(n)}$ for each node. In our finite element algorithm, instead of storing the matrix $\mathbf{K}$ itself, we simply calculate and store each element's contribution to the $\mathbf{K}$ matrix. When solving the matrix equation by the PCG method, we just need to sum up the contributions at each node from its neighboring elements by using information provided by the connectivities for each element. For the rest of this chapter, we will discuss the weak formulation for each of our triangular elements which consists of three nodes.

By using equation (3.13) the contribution to matrix $\mathbf{K}$ from each element can be expressed as:

$$K_{lm} = \begin{cases} \frac{\Delta_e}{6} & \text{if } l = m \\ \frac{\Delta_e}{12} & \text{if } l \neq m \end{cases} \tag{3.15}$$

where Δ_e is the area of a specific element being considered. The unknown velocity vector for a given triangular element is

$$\phi = (u_1 \; v_1 \;\; u_2 \; v_2 \;\; u_3 \; v_3)^T \tag{3.16}$$

in which the indices $1, 2$ and 3 are the three vertices of an element while u and v are the x- and y- components of velocity.

By using the same weighted residual method, and the linear trial function for u and v in shown equation (3.2) and (3.3), the right hand side of equation (3.12) becomes:

$$\begin{aligned} \text{RHS} = & \underbrace{-\Delta t \int_\Omega (\mathbf{A}\frac{\partial \mathbf{u}}{\partial x} + \mathbf{B}\frac{\partial \mathbf{u}}{\partial y})\, N_l \, d\Omega}_{\Psi_1} + \underbrace{\nu\, \Delta t \int_\Omega (\frac{\partial^2 \mathbf{u}}{\partial x^2} + \frac{\partial^2 \mathbf{u}}{\partial y^2})\, N_l \, d\Omega}_{\Psi_2} + \\ & \underbrace{\frac{\Delta t^2}{2} \int_\Omega [\mathbf{A}\frac{\partial}{\partial x}(\mathbf{A}\frac{\partial \mathbf{u}}{\partial x} + \mathbf{B}\frac{\partial \mathbf{u}}{\partial y}) + \mathbf{B}\frac{\partial}{\partial y}(\mathbf{A}\frac{\partial \mathbf{u}}{\partial x} + \mathbf{B}\frac{\partial \mathbf{u}}{\partial y})]\, N_l \, d\Omega}_{\Psi_3} \end{aligned} \tag{3.17}$$

First, the integral denotes as Ψ_1 in equation (3.17) can be simplified to

$$\Psi_1 = -\Delta t \int_\Omega [\mathbf{A} \, (\sum_i \mathbf{u}_i^n \, \frac{\partial N_i}{\partial x} + \mathbf{B} \, (\sum_i \mathbf{u}_i^n \, \frac{\partial N_i}{\partial y})] \, N_l \, d\Omega \tag{3.18}$$

Since the variables inside the integral is constant within the element, we can write it as :

$$\text{x-component:} \quad - \Delta t \, \frac{\Delta_e}{3} \, [\bar{u} \, (u_1^n \, \beta_1 + u_2^n \, \beta_2 + u_3^n \, \beta_3) + \bar{v} \, (u_1^n \, \gamma_1 + u_2^n \, \gamma_2 + u_3^n \, \gamma_3)] \tag{3.19}$$

$$\text{y-component:} \quad - \Delta t \, \frac{\Delta_e}{3} \, [\bar{u} \, (v_1^n \, \beta_1 + v_2^n \, \beta_2 + v_3^n \, \beta_3) + \bar{v} \, (v_1^n \, \gamma_1 + v_2^n \, \gamma_2 + v_3^n \, \gamma_3)] \tag{3.20}$$

where $\bar{u}$, $\bar{v}$ are defined as:

$$\bar{u} = u_l + u_1 + u_2 + u_3 \tag{3.21}$$

$$\bar{v} = v_l + v_1 + v_2 + v_3 \tag{3.22}$$

Note that the subscript $1, 2, 3$ denotes the three vertices for a given triangular element. The subscript $l, l = 1, 3$ denotes the difference in the equations when considering contribution from each of the three nodes in a triangle. The values for $[\beta_i]_{i=1,3}$ and $[\gamma_i]_{i=1,3}$ are the derivatives of the piecewise linear function with respect to x- and y- coordinate as described in the last section. Finally the superscript n is the time level that the variables are evaluated at.

By using Green's theorem, the integral denoted as Ψ_2 in equation (3.17) can be

written as :

$$\Psi_2 = -\ \nu\ \Delta t\ \int_\Omega (\frac{\partial \mathbf{u}}{\partial x}\ \frac{\partial N_l}{\partial x} + \frac{\partial \mathbf{u}}{\partial y}\ \frac{\partial N_l}{\partial y})\ d\Omega + \nu\ \Delta t\ \int_\Gamma (\frac{\partial \mathbf{u}}{\partial x}\ n_x + \frac{\partial \mathbf{u}}{\partial y}\ n_y)\ N_l\ d\Gamma \tag{3.23}$$

in which n_x, n_y are the x- and y- normals of the control surfaces. This can further be approximated as:

$$\Psi_2 = -\ \nu\ \Delta t\ \Delta_e\ \sum_i\ \mathbf{u}_i\ (\beta_i\ \beta_l + \gamma_i\ \gamma_l) + \nu\ \Delta t\ \sum_i\ \mathbf{u}_i\ (\beta_i\ n_x + \gamma_i\ n_y)\ \frac{S}{2} \tag{3.24}$$

where S is the boundary length of the element. Note that the second term in equation (3.24) is the natural boundary condition and we only apply it for boundary nodes which do not have zero stress.

Finally, the integral denotes as Ψ_3 in equation (3.17) can be approximated as :

$$\text{x-component:}\quad rhs_1 = -\ \frac{\Delta t^2}{2}\ \Delta_e\ (\bar{u}\ \beta_l + \bar{v}\ \gamma_l)\ [\bar{u}\ (u_1^n\ \beta_1 + u_2^n\ \beta_2 + u_3^n\ \beta_3) + \bar{v}\ (u_1^n\ \gamma_1 + u_2^n\ \gamma_2 + u_3^n\ \gamma_3)] \tag{3.25}$$

$$\text{y-component:}\quad rhs_2 = -\ \frac{\Delta t^2}{2}\ \Delta_e\ (\bar{u}\ \beta_l + \bar{v}\ \gamma_l)\ [\bar{u}\ (v_1^n\ \beta_1 + v_2^n\ \beta_2 + v_3^n\ \beta_3) + \bar{v}\ (v_1^n\ \gamma_1 + v_2^n\ \gamma_2 + v_3^n\ \gamma_3)] \tag{3.26}$$

Step Two :

The integral formulation for the second step can be written as:

$$\int_\Omega \nabla^2\ p\ N_l\ d\Omega = \frac{\rho}{\Delta t}\ \int_\Omega \nabla \cdot \mathbf{u}^*\ N_l\ d\Omega \tag{3.27}$$

By using Green's theorem, the left hand side of equation (3.27) can be written as:

$$- \int_\Omega (\frac{\partial p}{\partial x}\ \frac{\partial N_l}{\partial x} + \frac{\partial p}{\partial y}\ \frac{\partial N_l}{\partial y})\ d\Omega + \int_\Gamma (\frac{\partial p}{\partial x}\ n_x + \frac{\partial p}{\partial y}\ n_y) N_l\ d\Gamma \tag{3.28}$$

By using the linear trial function for pressure described in equation (3.4), the integral equation now becomes:

$$\sum_i\ p_i^n\ \Delta_e\ (\beta_i\ \beta_l + \gamma_i\ \gamma_l) = -\rho\ \frac{\Delta_e}{3\ \Delta t}\ (\sum_i\ u_i^*\ \beta_i + \sum_i\ v_i^*\ \gamma_i) + \int_\Gamma\ \frac{\partial p}{\partial n}\Big|_e^n\ N_l\ d\Gamma \tag{3.29}$$

As a result, the left hand side contribution to the symmetric matrix $\mathbf{K}$ from each element is :

$$\mathbf{K} = \Delta_e \begin{pmatrix} \beta_1^2 + \gamma_1^2 & \beta_1\ \beta_2 + \gamma_1\ \gamma_2 & \beta_1\ \beta_3 + \gamma_1\ \gamma_3 \\ \beta_2\ \beta_1 + \gamma_2\ \gamma_1 & \beta_2^2 + \gamma_2^2 & \beta_2\ \beta_3 + \gamma_2\ \gamma_3 \\ \beta_3\ \beta_1 + \gamma_3\ \gamma_1 & \beta_3\ \beta_2 + \gamma_3\ \gamma_2 & \beta_3^2 + \gamma_3^2 \end{pmatrix} \tag{3.30}$$

The right hand side can be written as:

$$\text{rhs} = -\ \rho\ \frac{\Delta_e}{3\ \Delta t}\ (\sum_i\ u_i^*\ \beta_i + \sum_i\ v_i^*\ \gamma_i) + \frac{S}{2}\ \left[\ \frac{\partial p}{\partial n}\right]_e^n \tag{3.31}$$

where the gradient of p can be derived by using the third step [equation(3.9)], the result is:

$$\frac{\partial p}{\partial n}\Big|_e^n = \frac{(\bar{\mathbf{u}}^* - \mathbf{u}^{n+1}) \cdot \mathbf{n}}{\Delta\ t} \tag{3.32}$$

in which $\bar{\mathbf{u}}^*$ is the average of the $\mathbf{u}^*$ values at the two boundary nodes of the element, namely,

$$\bar{\mathbf{u}}^* = \frac{\mathbf{u}_1^* + \mathbf{u}_2^*}{2} \tag{3.33}$$

Step Three :

Equation (3.9) can be expressed as:

$$\int_\Omega \sum_i \mathbf{u}_i^{n+1}\, N_i\, N_l\, d\Omega = \int_\Omega \sum_i \mathbf{u}_i^*\, N_i\, N_l\, d\Omega - \frac{\Delta t}{\rho} \int_\Omega \sum_i p_i^{n+1} \left(\frac{\partial N_i}{\partial x}\,\mathbf{i} + \frac{\partial N_i}{\partial y}\,\mathbf{j}\right) N_l\, d\Omega \tag{3.34}$$

where **i** and **j** are the unit vectors for the x- and y- Cartesian coordinates. The left hand side can be expressed as:

$$\mathbf{K} = \Delta_e \begin{pmatrix} \frac{1}{6} & \frac{1}{12} & \frac{1}{12} \\ \frac{1}{12} & \frac{1}{6} & \frac{1}{12} \\ \frac{1}{12} & \frac{1}{12} & \frac{1}{6} \end{pmatrix} \tag{3.35}$$

The x-component of the right hand side can be represented for each node of an element as:

$$\text{Node 1 :} \quad \frac{\Delta_e}{12}\,(2\,u_1^* + u_2^* + u_3^*) - \frac{\Delta t\ \Delta_e}{3}\,(\beta_1\,p_1^{n+1} + \beta_2\,p_2^{n+1} + \beta_3\,p_3^{n+1}) \tag{3.36}$$

$$\text{Node 2 :} \quad \frac{\Delta_e}{12}\,(u_1^* + 2\,u_2^* + u_3^*) - \frac{\Delta t\ \Delta_e}{3}\,(\beta_1\,p_1^{n+1} + \beta_2\,p_2^{n+1} + \beta_3\,p_3^{n+1}) \tag{3.37}$$

$$\text{Node 3 :} \quad \frac{\Delta_e}{12}\,(u_1^* + u_2^* + 2\,u_3^*) - \frac{\Delta t\ \Delta_e}{3}\,(\beta_1\,p_1^{n+1} + \beta_2\,p_2^{n+1} + \beta_3\,p_3^{n+1}) \tag{3.38}$$

Similar expression can be written for the y-component of the right hand side with v in place of u and γ in place of β.

In conclusion, by using the finite element formulation described in this chapter, our incompressible Navier-Stokes solver called 'NSV.f' was written and used for both simulating the cylinder/foil flow cases and deriving the low order model of the Navier-Stokes equations. Details of the solutions obtained by this code will be presented in the later chapter concerning examples. The accuracy of our formulation is comparable to the codes that had been reported in the literature.

Chapter 4

Mesh Generation and Movement

4.1 Grid Generation – The Felisa System

The mesh for our cylinder/airfoil problem is generated by using the Felisa system [20]. By specifying the inner and outer boundary node points, and also the grid density in specific regions, the Felisa system uses the advancing front triangulation method to discretize the computational domain into numbers of unstructured triangles. The unstructured triangular mesh is characterized by data sets including node coordinates, element connectivity, and boundary node numbering. This information is essential for both the plotting program and the incompressible Navier-Stokes solver.

4.2 Grid Movement

4.2.1 Methodology

Many Euler or Navier-Stokes codes use a computational grid with a fixed underlying geometrical structure. These algorithms just keep on iterating until a steady state solution for which the residual become smaller than a given tolerance is reached. However, many other fluid problems, as in our low Reynold's number cylinder and foil flow problem, address the relationship between the dynamical behavior of the structure and the fluid flow. The computation for this class of problems requires a

mesh that is capable of movement.

In this research, the cylinder wake was controlled by an airfoil moving behind it. Thus there was a need to be able to move the mesh at each iteration. The mesh movement algorithm described by Batina [10] was used for our 2-D mesh movement.

To move the mesh in our cylinder/airfoil test case, the boundary nodes were held fixed and the interior nodes were moved in such a way that the only change in the characteristic of our unstructured mesh is just the coordinates of each node. In other words, we keep the connectivities for each element to be the same at every iteration and the prescribed motion of the nodes on the surface of the foil simply becomes a forcing function which governs the displacement of the interior nodes.

The constraint on the movement of each node is set up by using a spring analogy. Imagine the outer boundary grid points are fixed and form a rigid frame box. All the interior edges of the mesh are represented as springs and the nodes are the connections among the springs. The moving objects (the cylinder and the foil) which are characterized by the structured interior boundary nodes act as if they are the weights which were held in equilibrium by the whole spring system. As a result, when the objects (interior boundary node points) are moved by an external force of some sort, all the springs and the connecting nodes will also displace with them in order to reach an equilibrium state. The actual displacement of a given node depends on both the stiffness of the nearby springs and also on how close the node is to the moving objects.

The whole spring system can be formulated by using Newton's Second Law of Motion in the x- and y- direction, namely,

$$F_{x_i} = \delta_{x_i} \sum_{j=1}^{m} k_m = \sum_{j=1}^{m} k_m \, \delta_{x_m} \tag{4.1}$$

$$F_{y_i} = \delta_{y_i} \sum_{j=1}^{m} k_m = \sum_{j=1}^{m} k_m \, \delta_{y_m} \tag{4.2}$$

where F_x and F_y are the x- and y- component of forces, δ_x and δ_y are the displacements of a given node, and k is the spring stiffness assigned to each edge. Since the variables we want to solve are the displacement of each node, we could rewrite the equation in

the more convenient form:

$$\delta_{x_i} = \frac{\sum_{j=1}^{m} k_m \, \bar{\delta}_{x_m}}{\sum_{j=1}^{m} k_m} \tag{4.3}$$

$$\delta_{y_i} = \frac{\sum_{j=1}^{m} k_m \, \bar{\delta}_{y_m}}{\sum_{j=1}^{m} k_m} \tag{4.4}$$

where i is the node being considered and j, ranging from 1 to m, are the neighboring nodes. In figure (4-1) below, node i is being considered so that the stiffness k_m and the displacement δ_{x_i} and δ_{y_i} of the neighboring seven points are substituted in equation (4.3) and (4.4).

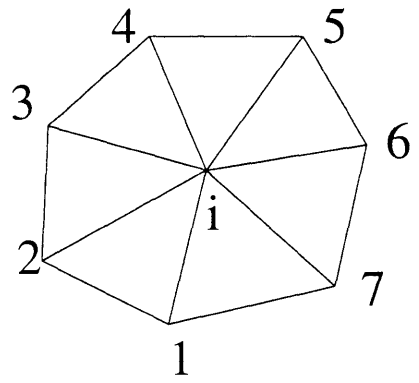

Figure 4-1: A Control Volume
The stiffness and the displacement of the neighboring edges are used to determine the movement of the center node.

The spring stiffness for a given edge i – j is taken to be inversely proportional to the length of the edge as

$$k_m = \frac{1}{\sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}} \tag{4.5}$$

By using this definition, the static equilibrium equations (4.3) and (4.4), which resulted from a summation of forces for every node, form an implicit algebraic system of equations because the displacement of a given node depends on the displacement of all the neighboring nodes. In other words, we have two matrix equations with dimension equal to the total number of nodes. By solving these two system individually, the unknown x- and y- displacement arrays can then be computed.

To solve these sparse matrix problems, a predictor-corrector procedure is used. It

first predicts the displacements of the nodes by extrapolation from grids at previous time levels according to:

$$\tilde{\delta_{x_i}} = 2\delta_{x_i}^n - \delta_{x_i}^{n-1} \tag{4.6}$$

$$\tilde{\delta_{y_i}} = 2\delta_{y_i}^n - \delta_{y_i}^{n-1} \tag{4.7}$$

These displacements were then corrected by using several Jacobi iterations described in equation (4.3) and (4.4). Then the displacements $\delta_{x_i}^{n+1}$ and $\delta_{y_i}^{n+1}$ of the nodes could be obtained and the new locations of the interior nodes could then be updated by the following equations :

$$x_i^{n+1} = x_i^n + \delta_{x_i}^{n+1} \tag{4.8}$$

$$y_i^{n+1} = y_i^n + \delta_{y_i}^{n+1} \tag{4.9}$$

This predictor-corrector procedure was found to be very efficient for our problem. Since the airfoil was moved by a very small amount for each time level, two Jacobi iterations were found to be sufficient for the grid points to come to equilibrium.

4.2.2 Examples

In this section we discuss the results of our mesh movement computation. The original cylinder and airfoil(NACA 0012) mesh is shown in figure (4-2) below. Notice that the elements around the cylinder and the airfoils have to be sufficiently small so that the unsteady and viscous effects near the objects could be captured by the incompressible Navier-Stokes solver accurately.

By zooming into the vicinity of the airfoil, we can see clearly how the elements are arranged near the surface of the foil in figure (4-3). The foil could be moved and rotated by using the techniques described in the previous section. For example, starting from the original position, by setting the moving foil in a sinusoidal translational and rotational motion, after a quarter of the sinusoidal cycle, the airfoil has moved upward 0.667 unit length with respect to the center line of the geometry and has an angle of attack of 30 degrees with respect to the one-third chord point. The resulting mesh is displayed in figure (4-4).

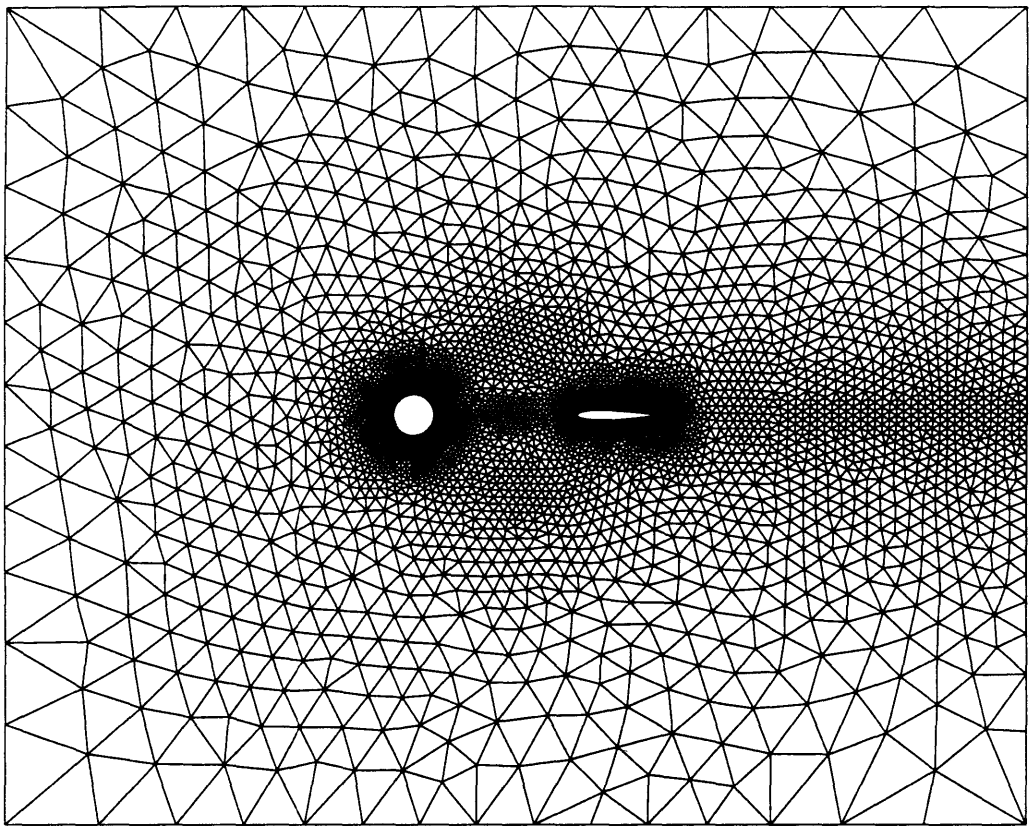

Figure 4-2: Original Mesh
Mesh used in the moving mesh algorithm.

In the figure, we can clearly see that the resulting mesh is a failure because the elements at the leading edge overlap with each other. This causes a negative area for some of the control volume and so this mesh is completely useless for our incompressible Navier-Stokes solver. Notice that even if the elements did not overlap with each other but were sheared so much that they had very small areas, it would still not be useful for our solver because the time step limit for our scheme depends on the size of the smallest element. In other words, if the elements near the airfoil become very small, the time step has to be reduced significantly in order to make the scheme stable. As a result, we need a better way to move the nodes around the airfoil.

There were two different methods that we have tried in order to obtain an ap-

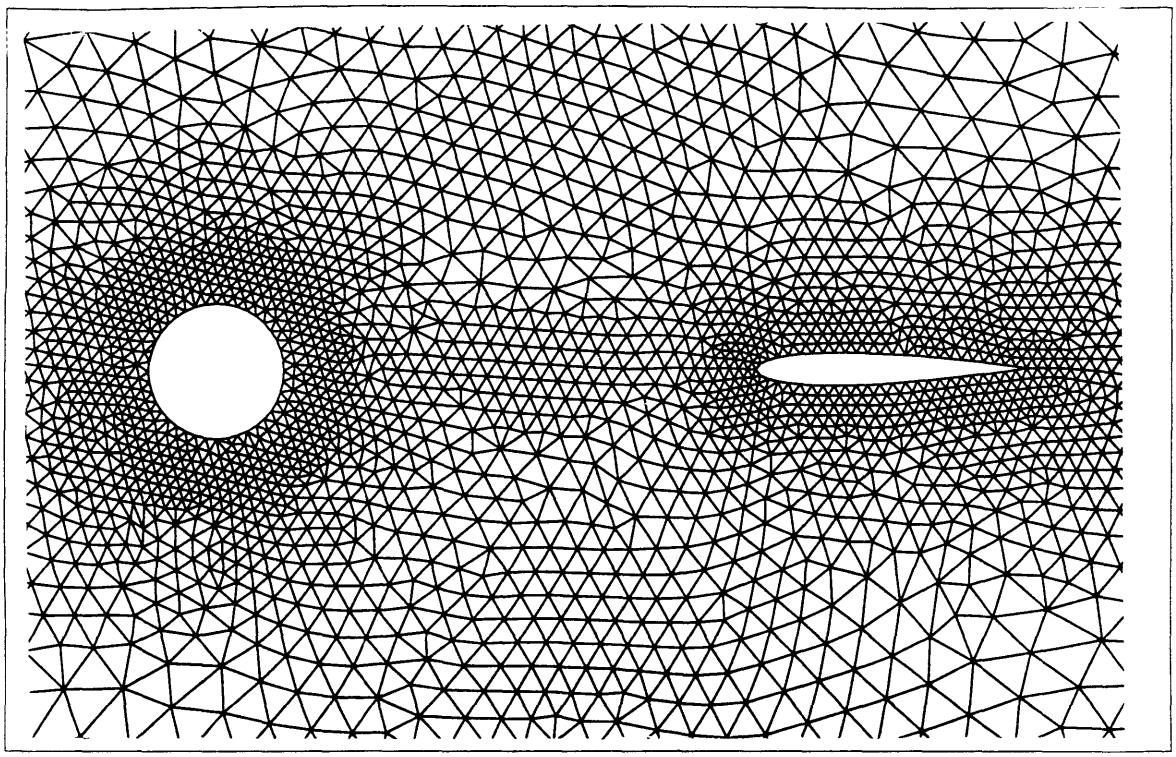

Figure 4-3: Enlargement at the vicinity of the cylinder and foil

propriate mesh. The first one was to move the layers of elements around the airfoil together with the prescribed motion on the foil. This eliminated the problem shown in figure (4-4). However, we found that the sheared small element problem still arises for the elements near the layers of elements which have the prescribed motion. The alternative is to re-assign the stiffness of the edges near the airfoil. By setting the first layer of edges to have the highest stiffness and the stiffness for the next five layers of edges to be gradually smaller, we obtain a satisfactory mesh. This mesh is shown in figure (4-5).

In conclusion, by using a modified spring analogy technique, we are capable of moving the mesh very efficiently to a very high angle of attack and a long translational distance. Most of the applications of the moving airfoil require only a small change in the a.o.a and the translational distance; therefore this mesh movement technique is quite sufficient.

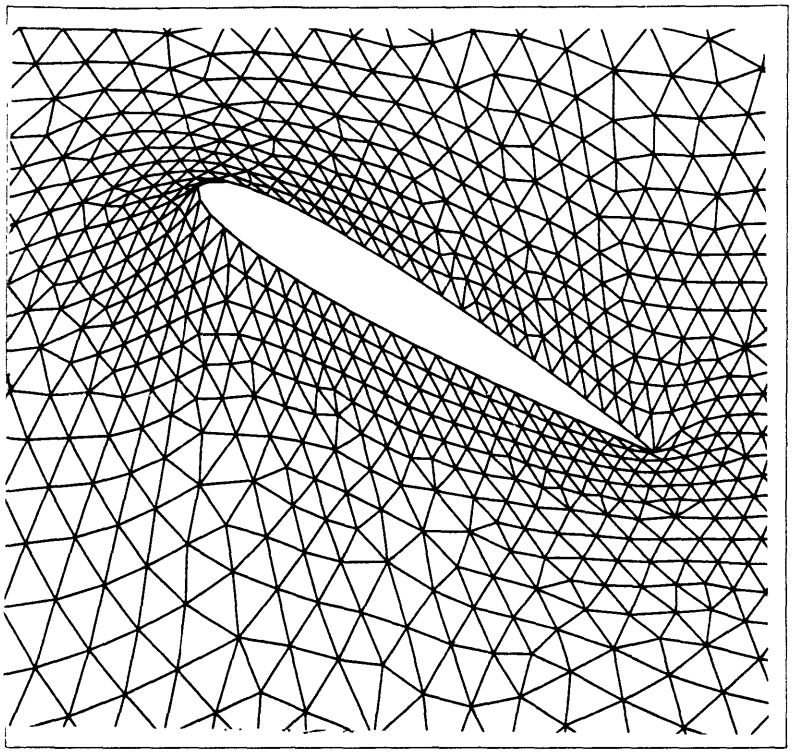

Figure 4-4: Mesh Movement
Airfoil at 0.667 Vertical Distance and 30 degrees A.O.A

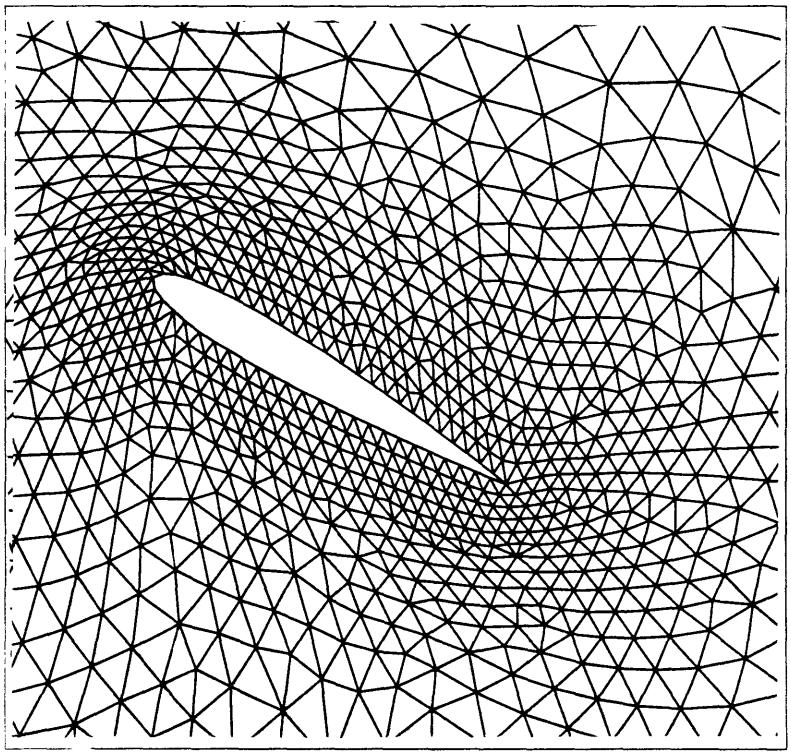

Figure 4-5: Modified Mesh Movement
Airfoil at 0.667 Vertical Distance and 30 degrees A.O.A

Chapter 5

Techniques for Solving Systems of Equations

5.1 Conjugate Gradient Method

To solve the three matrix equations which resulted from our three-step time-marching scheme, we have to choose a numerical method which can solve the equations in the most efficient way. Since the matrices derived by using the FEM are very sparse, an iterative method is chosen over a direct method. In this section we will discuss briefly the preconditioned conjugate gradient (PCG) method used in our solver. The purpose of using preconditioning [1] is to accelerate the convergence of the CG method [14]. However, finding a good preconditioner for solving a given sparse linear systems sometimes seems to be more of an art than a science and we will discuss it in the next section. The "Sparskit", which is a collection of computer programs for sparse matrix computations, developed by professor Youcef Saad at the University of Minnesota was found very useful for this part of our research.

Conjugate gradient is the most prominent iterative method of solving large sparse systems of linear equations. The matrix equations have the form

$$Ax = b \tag{5.1}$$

where x is an unknown vector, b is a known vector, and A is a known, square, symmetric, positive-definite, n by n matrix. The conjugate gradient method is in the category of the Krylov subspace methods. These techniques are based on projections processes, both orthogonal and oblique, onto Krylov subspaces. The mathematical background of these class of iterative techniques can be found in many linear algebra text books [7].

The conjugate gradient algorithm can be summarized as:

- **Start:** Choose an initial guess x_0 then compute the residual vector $r_0 = b - Ax_0$. Set $p_0 = r_0$.

- **Iterate:** For $j = 0, 1, 2, ..,$ until convergence(in which the residual is less than a given tolerance), do:

  1. $\alpha_j = \frac{(r_j, r_j)}{(Ap_j, p_j)}$
  2. $x_{j+1} = x_j + \alpha_j p_j$
  3. $r_{j+1} = r_j - \alpha_j A p_j$
  4. $\beta_j = \frac{(r_{j+1}, r_{j+1})}{(r_j, r_j)}$
  5. $p_{j+1} = r_{j+1} + \beta_j p_j$

Note that the notation of (a, b) denotes the inner product between two vectors. The first and the last step of our three-step time-marching Navier-Stokes solver can be calculated very effectively by using just 8 conjugate gradient iterations while the second part solving for the pressure field takes more efforts (an average of 60 iterations). In the next section, we will see that by using preconditioning, the number of iterations required for convergence can be decreased substantially.

5.2 LU Preconditioner

Preconditioning is a technique for improving the condition number of a matrix. Suppose that M is a symmetric, positive-definite matrix that approximates A, but is

easier to invert. We can solve $Ax = b$ indirectly by solving

$$M^{-1}Ax = M^{-1}b. \tag{5.2}$$

If the condition number of $M^{-1}A$ is much less than the condition number of A, or if the eigenvalues of $M^{-1}A$ are better clustered than those of A, we can iteratively solve equation(5.2) more quickly than the original problem. However, the catch is that $M^{-1}A$ is not generally symmetric nor definite, even if M and A are.

The best choice of the preconditioner is the one which can be easily inverted and also make the condition number to be small. The simplest preconditioner is a diagonal matrix whose diagonal entries are identical to those of A. The process of applying this preconditioner, known as *diagonal preconditioning* or *Jacobi preconditioning*, is equivalent to scaling the quadratic form along the coordinate axes. A diagonal matrix is trivial to invert, but is often only a mediocre preconditioner. The number of iterations needed for convergence in this case is about 20% lower than the one for conjugate gradient method without preconditioning.

A more effective and common way to define a preconditioner is to perform an incomplete LU factorization of the original matrix A. This entails a decomposition of the form $A = LU$ where L and U have the same nonzero structure as the lower and upper parts of A respectively. The inverse of the preconditioner LU is simply $U^{-1}L^{-1}$ which does not take too much effort to compute.

By using the LU preconditioner technique together with the conjugate gradient technique, the required number of iterations for convergence is much less. In our solver, the first and the third step take only 3 iterations to converge while the second part takes about 10 iterations.

5.3 Reverse Cuthill-McKee Graph Reordering

The node numbering of the Felisa mesh generation system is specific, it starts from numbering all the boundary nodes before the interior nodes are being numbered.

However, this node numbering technique might not be very desirable because node ordering determines the position of non-zero elements in a sparse matrix. In other words, by using the node numbering generated by the Felisa system, the bandwidth of the sparse matrix is not at all optimized and can be very large. This would increase the cost of performing LU decomposition of a sparse matrix.

Consider grid shown in figure (5-1) as an example. There are 1,182 nodes in the mesh and the number of non-zero elements in the sparse matrix created by the finite element formulation for the second part of our three-step method is 8,120. The sparse matrix resulting directly from the Felisa program is shown in figure (5-2). Notice that there are many off-diagonal terms around the upper right and lower left corner of the sparse matrix.

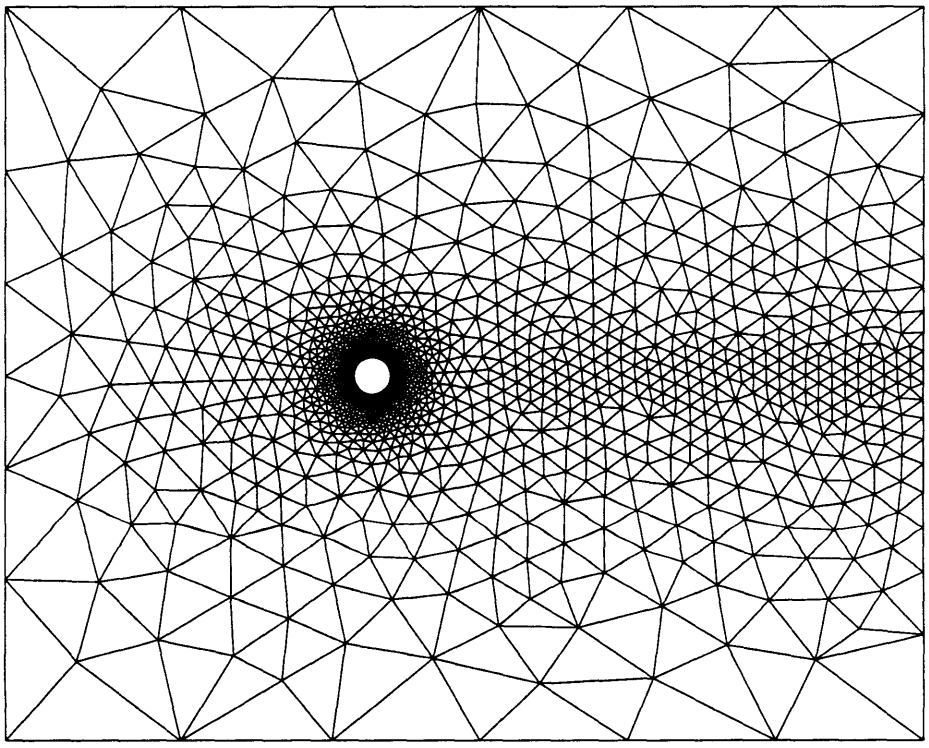

Figure 5-1: Circular Cylinder Mesh

The maximum bandwidth is 1,177 which is almost the same as the dimension of the sparse matrix. This will significantly slow down and decrease the accuracy of the LU decomposition on the sparse matrix.

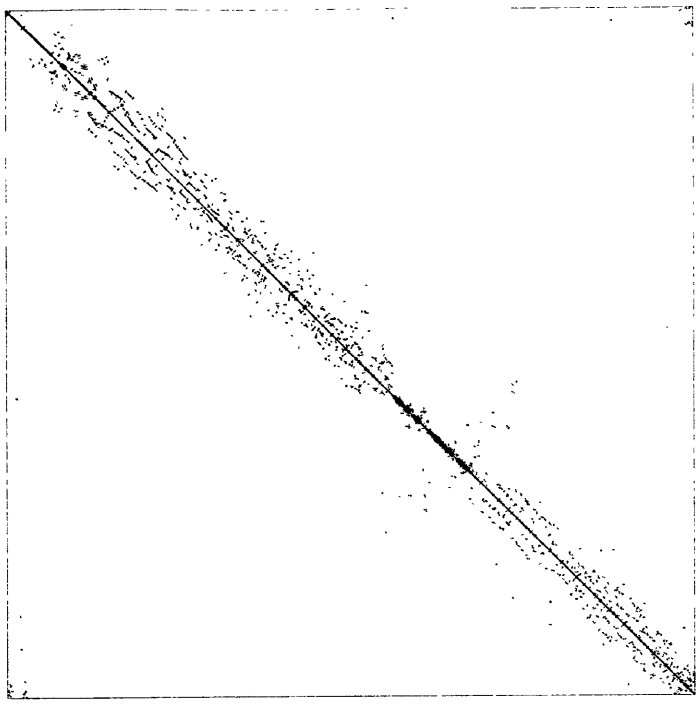

Figure 5-2: Original Sparse Matrix
Sparse matrix resulted from the Felisa system.

In order to minimize the bandwidth of the sparse matrix, the reverse Cuthill-McKee graph reordering is used. This is a method based on mathematical graph theory [28]. The triangular grid is considered as a planar graph. Reordering is done by the constraint that the neighboring vertices must have numbering which are near by. The steps of performing the Cuthill-McKee reordering is summarized below.

- Find the vertex with the lowest degree of freedom. (A corner outer boundary nodes is a good choice.)
- Find all the neighboring vertices connecting to the root by incident edges. Then order them by increasing vertex degree. This will result the first level.
- Form the next other level by finding all neighboring vertices of the previous level which have not been previously ordered. Order these new vertices by increasing vertex degree.

- Repeat the previous step if vertices remain.
- After all the vertices had been numbered, just simply reverse the node numbers. For example, if we have N nodes in our graph which are numbered as $1, 2, 3, 4...N-1, N$; we just simply reverse the order by counting the node in a backward fashion, namely, $N, N-1, ..., 3, 2, 1$. In other words, node i is now numbered $n-i+1$. By doing this reverse process, the bandwidth for the sparse matrix is minimized.

Figure 5-3 shows the sparse matrix after the reversed Cuthill- McKee graph reordering has been performed. Notice that the maximum bandwidth of the reordered sparse matrix is just 108 which is far less than the bandwidth of the original sparse matrix. In conclusion, by using the Cuthill-McKee node reordering technique, the LU decomposition described in the last section could be performed in a very efficient way.

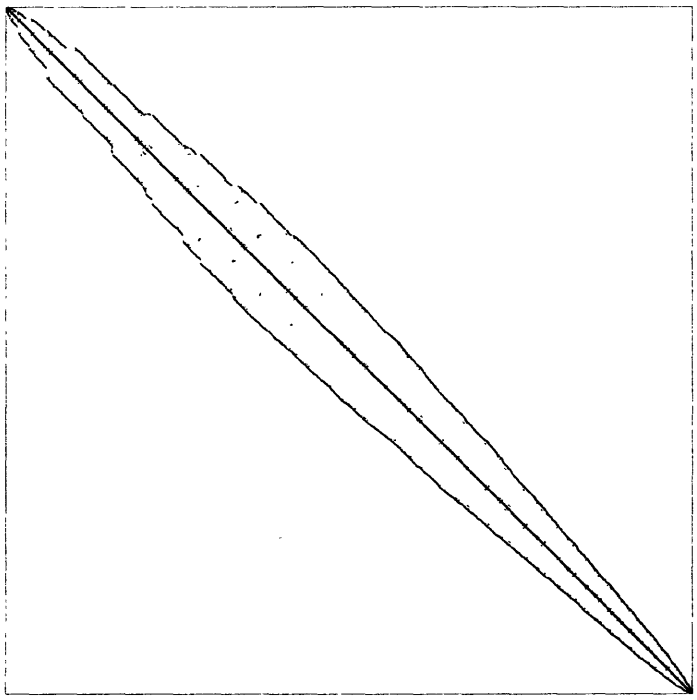

Figure 5-3: Sparse Matrix after Reversed Cuthill-McKee Reordering

Chapter 6

Results

In this chapter the performance of our algorithm is being investigated by applying it to four different test cases including the inviscid half circular cylinder case, the viscous full circular cylinder case, the viscous rotating cylinder case and the moving cylinder and airfoil case.

6.1 Half Circular Cylinder (Inviscid)

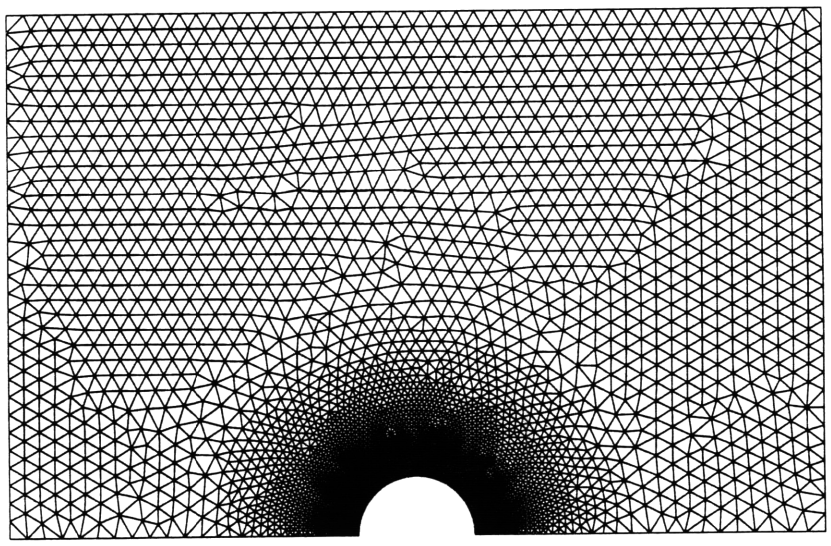

Figure 6-1: Half Circular Cylinder Mesh

The first example investigated was an inviscid flow over a half circular cylinder. This inviscid flow problem was chosen because the numerical solution obtained from our Navier-Stokes solver, with viscous term setting to zero, could be compared with the analytical solution; and thus the accuracy of our solver could be estimated. Figure (6-1) was used for this inviscid flow problem.

The radius of the cylinder is set to unity. The Dirichlet boundary conditions are illustrated in figure (6-2) below. The x-component velocity of the incoming flow has a magnitude of one and the pressure at the outflow boundary is set to zero. All the boundaries except the one on the cylinder have their y-component velocity setting to zero.

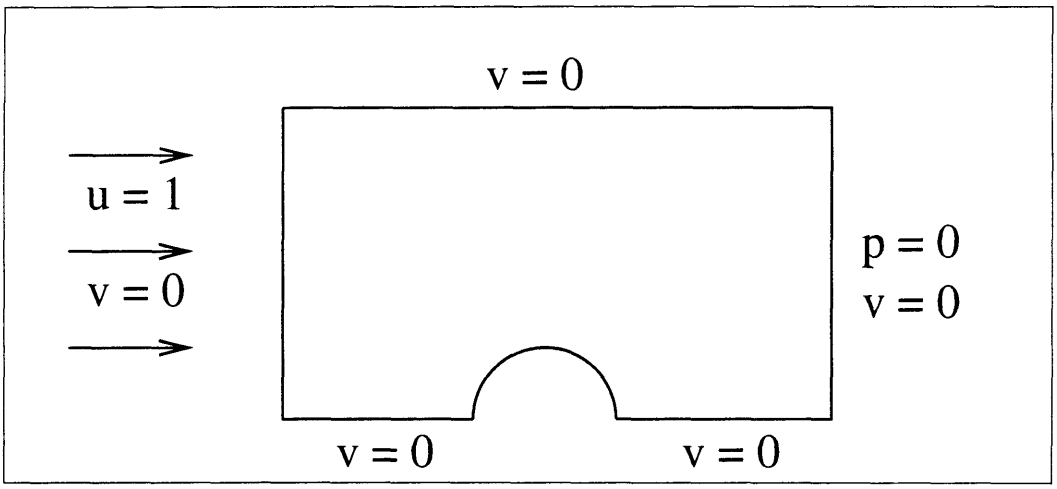

Figure 6-2: Boundary Conditions

After running the solver for about two hundred time steps, the flow field converges to a steady-state solution. The velocity and pressure contour are plotted in figure (6-3) and (6-4).

Notice that the numerical solution of the pressure contour shows clearly that the flow stagnates at the leading edge of the cylinder. In contrast , the static pressure comes to the lowest value on top of the cylinder while the velocity is maximum at that point. The analytical solution of an inviscid circular cylinder flow is simply the potential flow. For a circular cylinder centering at the origin, the potential flow solution can be easily described in the polar coordinates. The radial and tangential

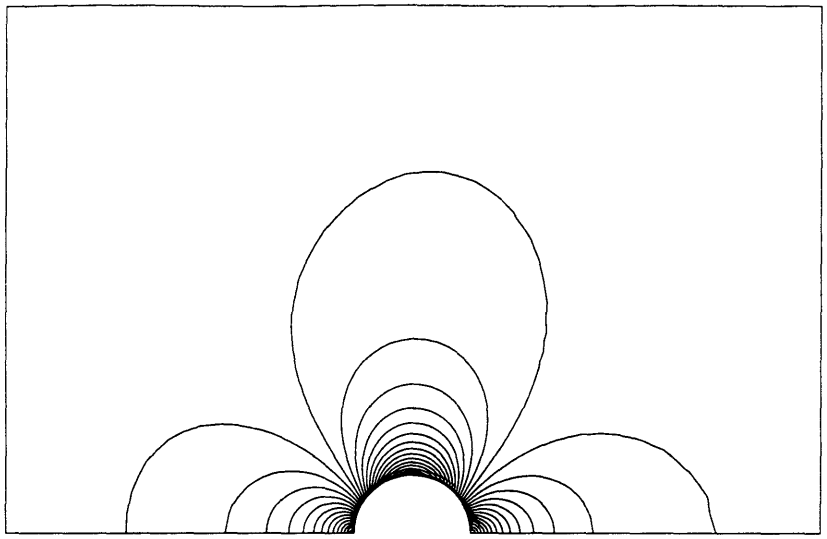

Figure 6-3: Velocity Contour of the Inviscid Flow Field

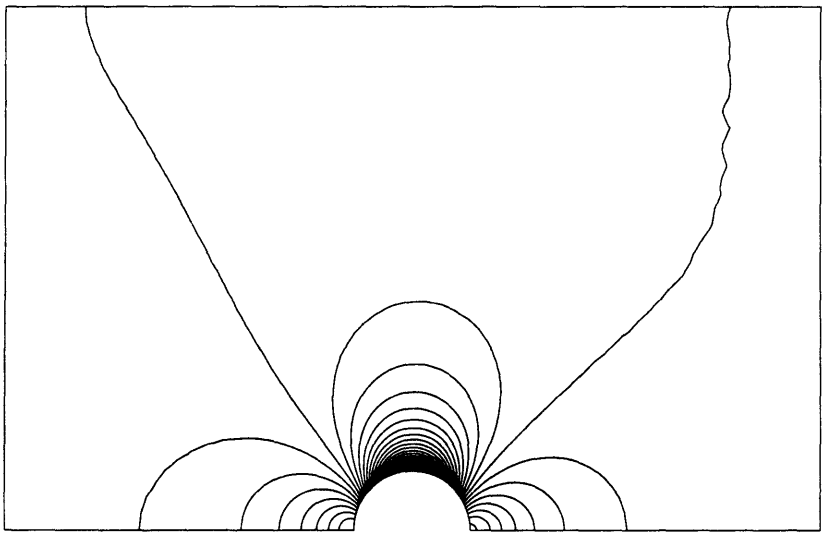

Figure 6-4: Pressure Contour of the Inviscid Flow Field

velocity of the flow field outside the cylinder are:

$$u_r = (1 - \frac{1}{r}) \cos\theta \tag{6.1}$$

$$u_t = -(1 + \frac{1}{r}) \sin\theta \tag{6.2}$$

where r is the distance between a specific point in the flow field and the origin; and θ is the angle between the x-axis and the radius arm. This is illustrated in figure (6-5) below.

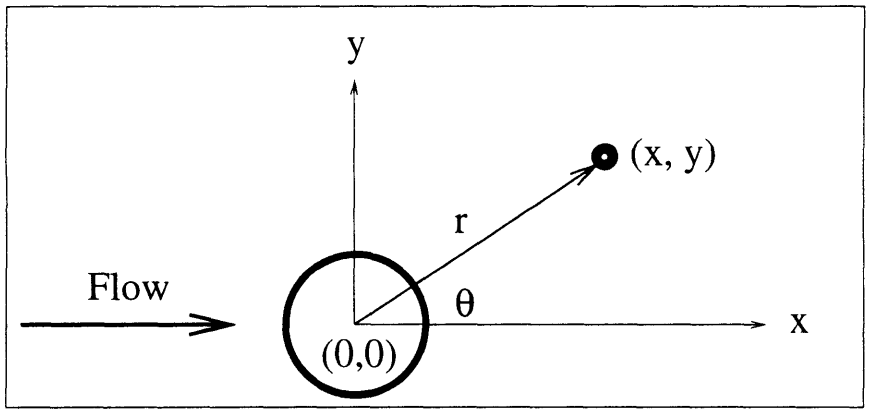

Figure 6-5: Polar Coordinate of the Flow Problem

By using equation (6.1) and (6.2), the velocity and pressure of the flow field at any specific field point can be expressed, in Cartesian coordinates as :

$$u = u_r \cos\theta - u_t \sin\theta \tag{6.3}$$

$$v = u_r \sin\theta + u_t \cos\theta \tag{6.4}$$

$$p = -\frac{1}{2r^2} - \frac{\sin^2\theta - \cos^2\theta}{r} \tag{6.5}$$

By using these equations, the velocity and pressure contour of the flow field are plotted in figure (6-6) and (6-7).

We could see that the analytical solution is very similar to the numerical solution obtained from our solver. However, in order to compare the two results qualitatively, we calculate the coefficient of pressure for both the numerical and analytical solution

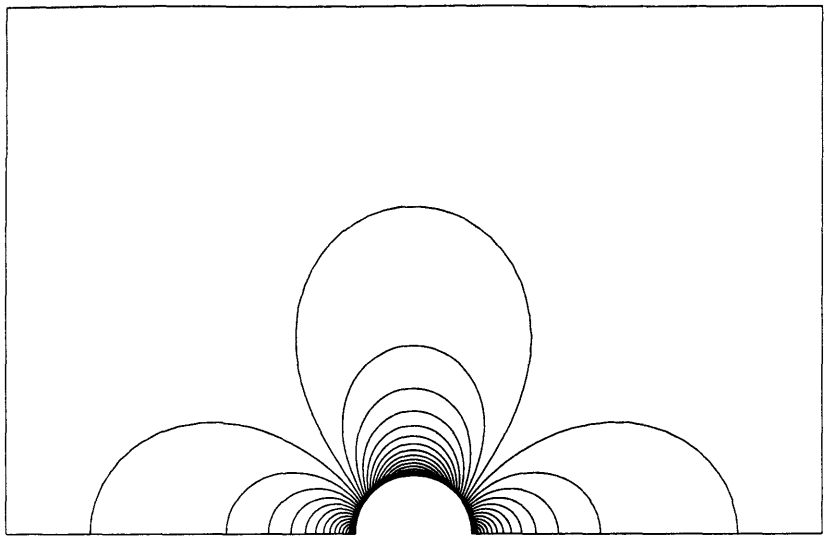

Figure 6-6: Velocity Contour (Potential Flow)

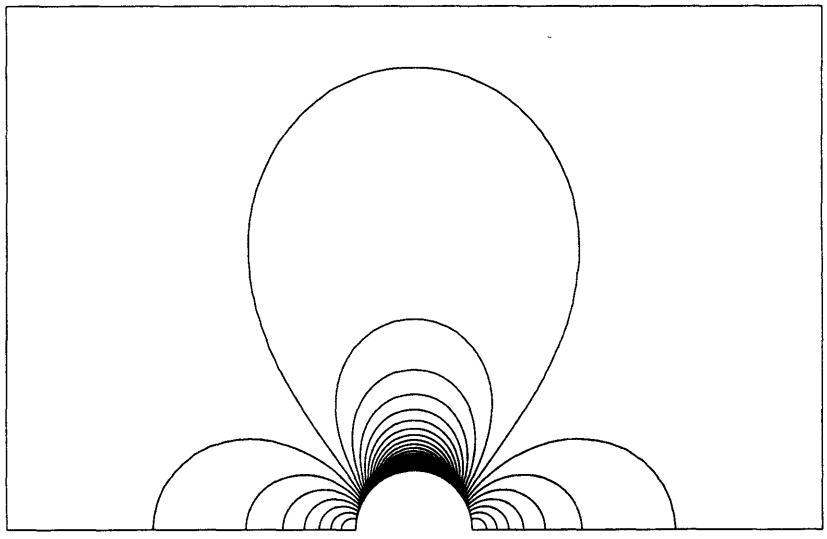

Figure 6-7: Pressure Contour (Potential Flow)

on the lower boundary of the domain. These values are plotted in figure (6-8) below.

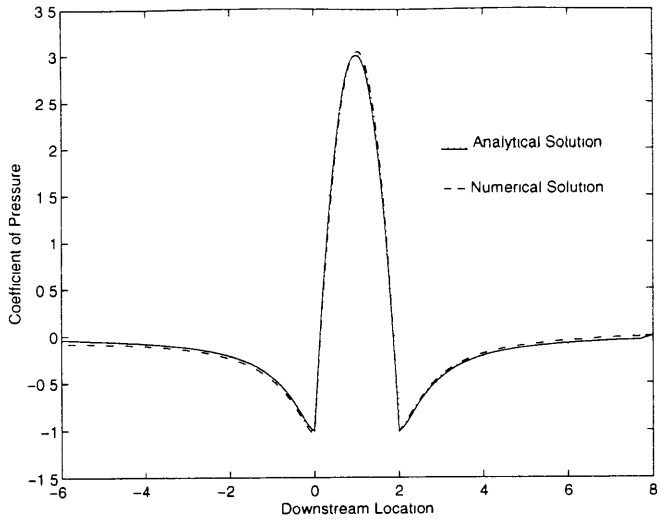

Figure 6-8: comparision between the Numerical and Analytical Solution

From these data, we can conclude that our Navier-Stokes solver over-predict the maximum static pressure coefficient occurs at the top of the cylinder. Also, in the numerical simulation, the static pressure coefficient in front of the cylinder is under-predicted; but it is over-predicted behind the cylinder. However, the difference is just in the order of 5 %.

In conclusion, the numerical solution obtained from our solver is very close to the analytical solution and thus the solver is ready to be extended to the use of the viscous full circular cylinder test case.

6.2 Full Circular Cylinder (Viscous)

6.2.1 Introduction

In this section, we consider the two-dimensional, incompressible, viscous flow past a circular cylinder at Reynold's numbers where vortex shedding occurs. This is a classical example problem for fluid dynamicists and thus the solution obtained from our Navier-Stokes solver could be compared with many references in the literature.

The characteristics of this fluid flow can be summarized as follows :

- The freestream velocity is relatively low and the corresponding Reynold's number is in the range of 100 to 500.
- The flow can be treated as an incompressible fluid.
- The large scale separation region and the global unsteadiness of the flow is due to the viscous effect (no-slip condition) on the surface of the cylinder.

6.2.2 Initial Condition

For the x- and y- component velocities, only the Dirichlet type boundary conditions are imposed; however, for the pressure term, both the Dirichlet type and the Neumann type boundary conditions are needed. Figure (6-9) summarizes the boundary conditions.

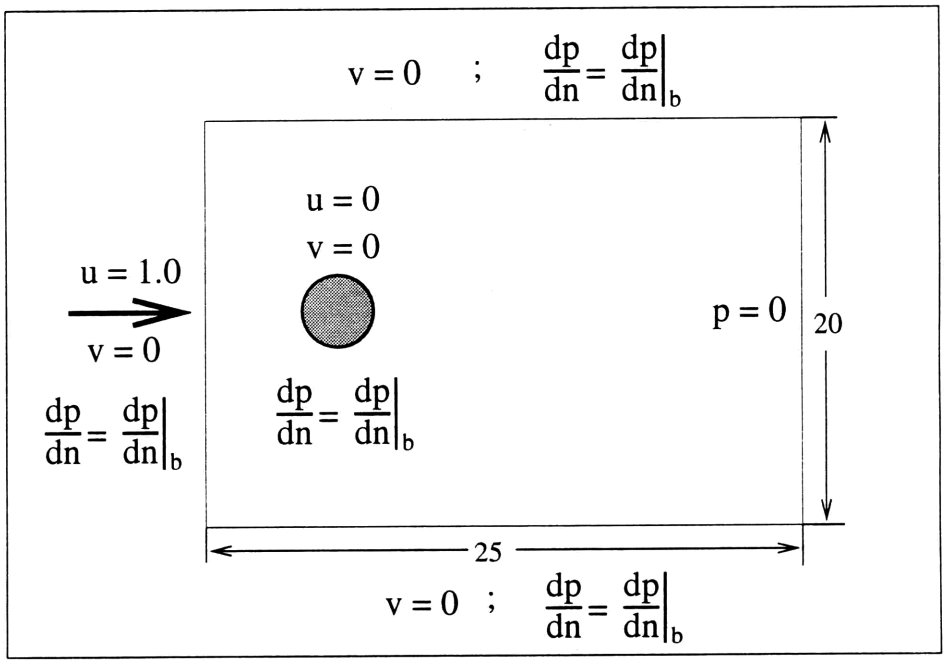

Figure 6-9: Boundary Conditions
The outer and inner boundary conditions for u,v and p are listed.

The boundary conditions are described in the following figure :

- For the inner boundaries, in order to impose the no-slip boundary conditions, the x- and y- component velocities are taken to be zero.

- The incoming flow boundary (left side of the rectangle) is taken far enough so that far-field boundary condition can be applied. In other words, the x-component velocities are normalized to unity and the y-component velocities are set to zero at this boundary.

- The lower and upper far-field boundaries have the same boundary conditions. The x-component velocities are not set. Since these far-field boundaries are taken to be far enough so that the blockage effect arise from the presence of the bodies in the flow could be neglected, y-component velocities could be assumed to be zero.

- The right side of the rectangle is the out-flow boundary of the problem. Both the x- and y- component velocities are not set in order to eliminate the numerical problem arises from a reflecting boundary.

- Only the pressure at the outflow boundary is set to be zero. Pressure at all other boundaries are not known a-priori. As a result, we have to impose the pressure gradient BC for these boundaries instead.

The cases for the rotating cylinder and the oscillating foil have similar boundary conditions as described above except that the velocities at the surfaces of the cylinder and the foil are set to be a prescribed velocity of its rotation and its oscillation respectively instead of just zero.

6.2.3 Location of the Lateral Boundaries

In many numerical simulations, the computational domain used is often only an approximation of the actual domain of the physical problem. Many types of boundary conditions used in practical applications are applicable only if they are sufficiently removed from the region where accuracy of the solution is important. The desire to limit computational cost, on the other hand, provides motivation to reduce the domain size. The best choice of the size of the computational domain can be expected when these contradictory tendencies are optimally balanced [2].

In our research, many different choices of the distance between the cylinder and lateral boundaries was experimented and the significant effect on the Strouhal number, lift coefficient, and mean drag coefficient were found and analyzed. Firstly, we found that the lateral boundaries should be removed from the cylinder at least by a distance of 8 cylinder diameters. If this is not the case, the computed Strouhal number ended up to have an artificially high value. Secondly, the lateral boundaries farther than 12 diameter distance from the center of the cylinder does not improve the solution significantly. Thus, the final choice of the lateral boundaries were set to be 15 diameter length.

6.2.4 Solution

Different meshes were experimented with for different Reynold's number with our Navier-Stokes solver. It was found that as long as the lateral boundaries are far enough and the elements near the cylinder are fine enough, the flow solution does not vary too much. An incoming flow with Reynold's number of 200 was simulated using the mesh shown in figure (6-10) was chosen to be used for illustration purposes in this thesis.

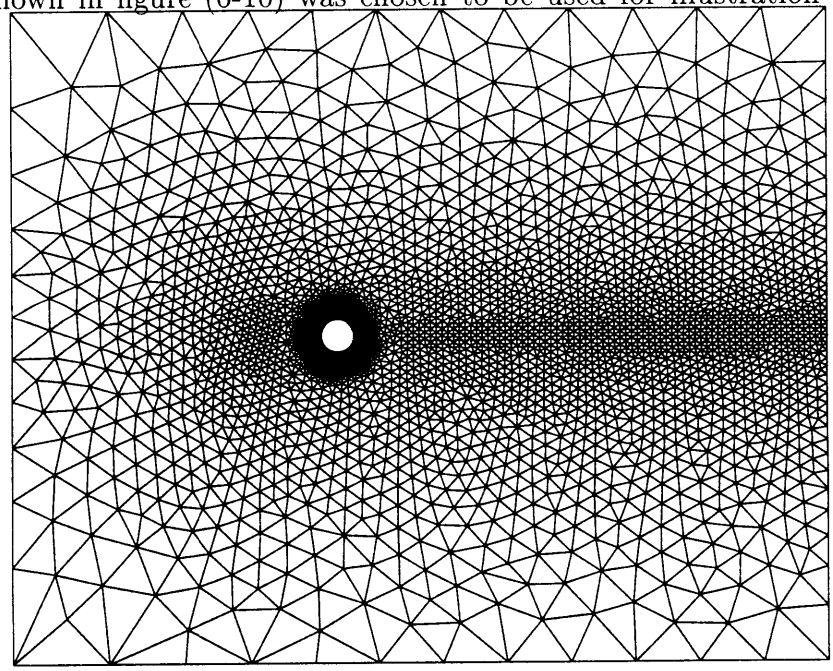

Figure 6-10: Full Circular Cylinder Mesh

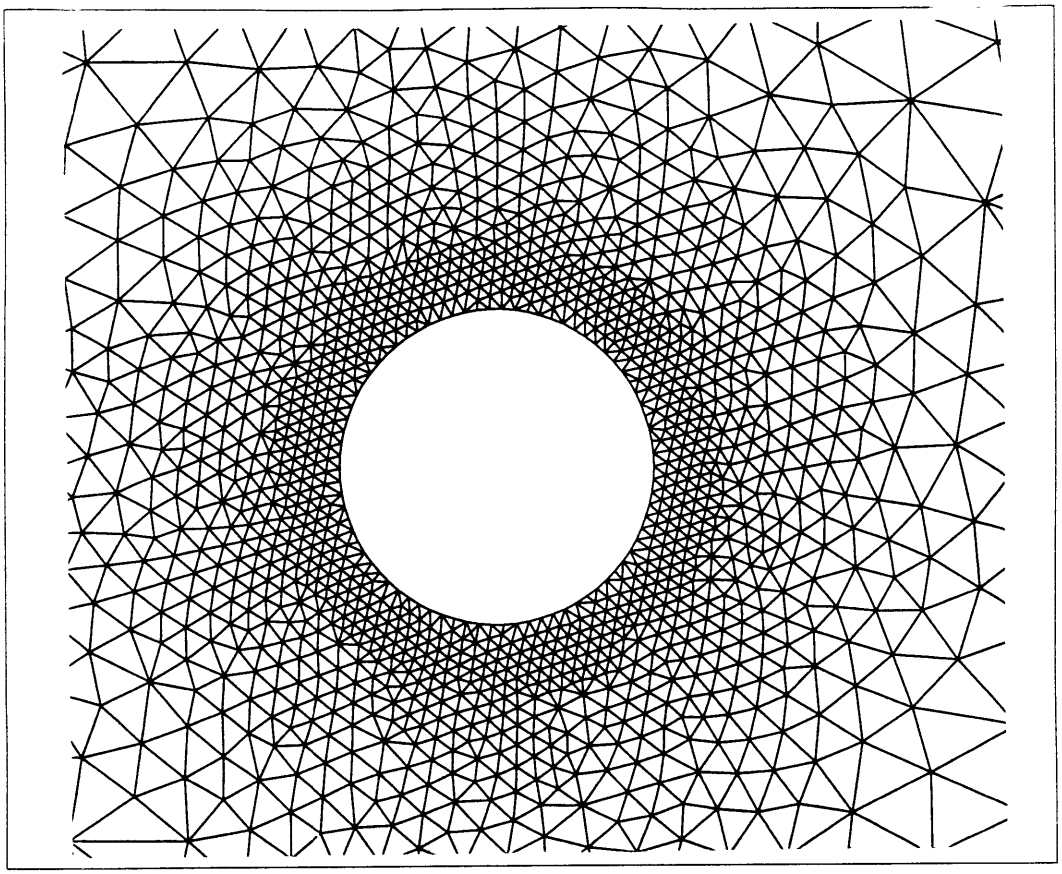

Figure 6-11: Enlargement around the Circular Cylinder

Notice that because it is the cylinder surface where the viscous effect dominated, in order to sufficiently capture the vortex generated by the cylinder, the elements in the vicinity of the cylinder have to be very fine. The details on how the elements are arranged near the cylinder could be found in figure (6-11).

At the beginning, the x-component velocity of the whole flow field was set to unity except the no-slip boundary condition applied on the cylinder. The flow begins to develop in the next 40 non-dimensional time until it reaches a state of fully developed flow. The following are a set of velocity vector figures, figure (6-12) to figure (6-16) which describe the development of the Von Karman vortex street behind the circular cylinder. Notice that the symmetric upper and lower vortex get larger and move away from the cylinder as time goes on. At around 30 non-dimensional time, one of the vortex starts becoming larger than the other one and the unsteadiness of the vortex street starts to appear.

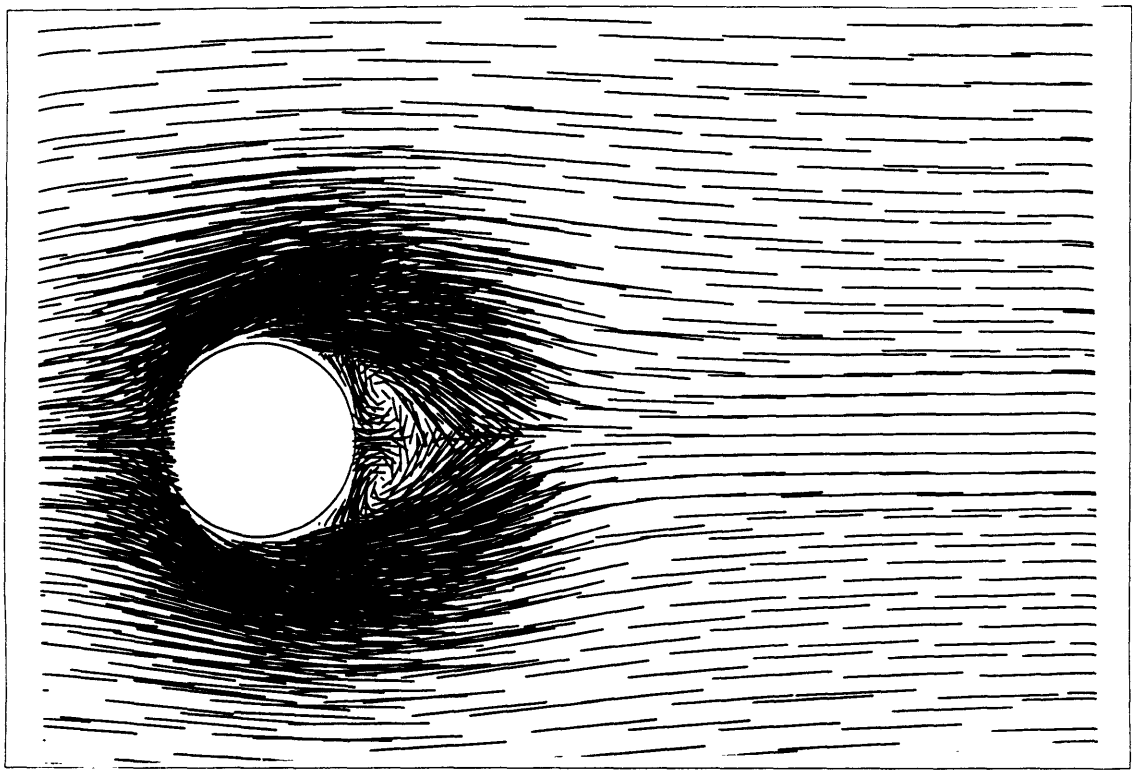

Figure 6-12: Non-Dimensional Time = 1

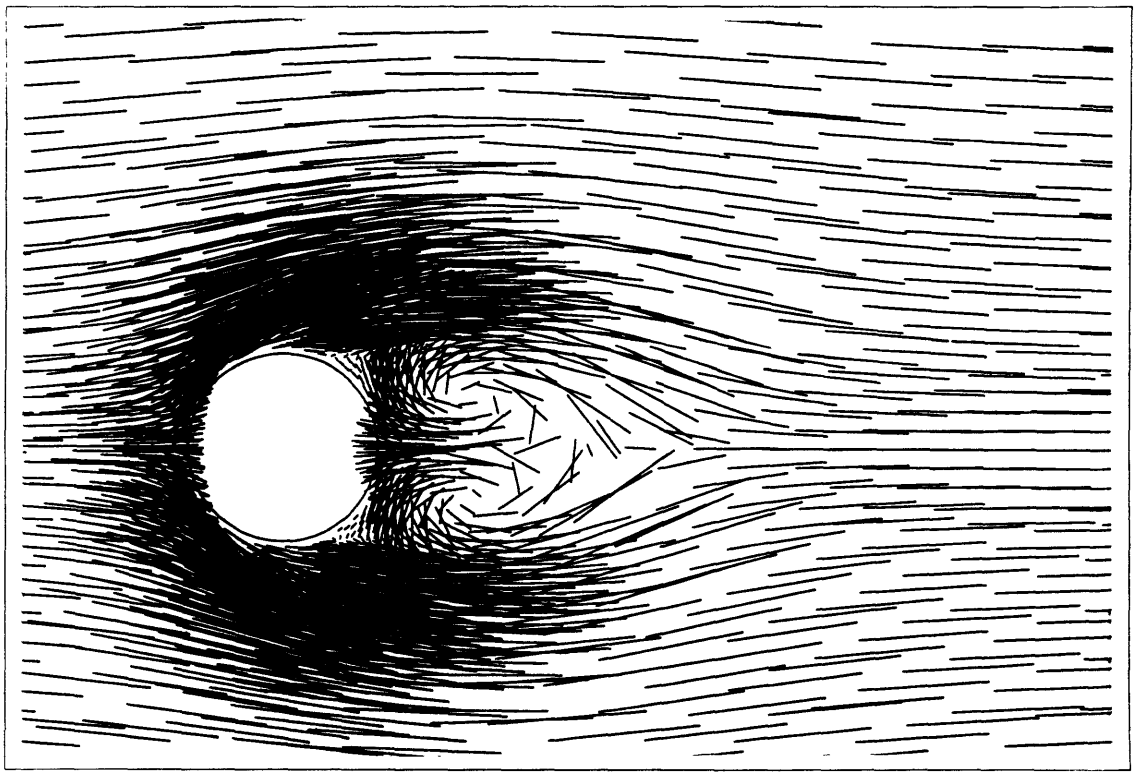

Figure 6-13: Non-Dimensional Time = 3

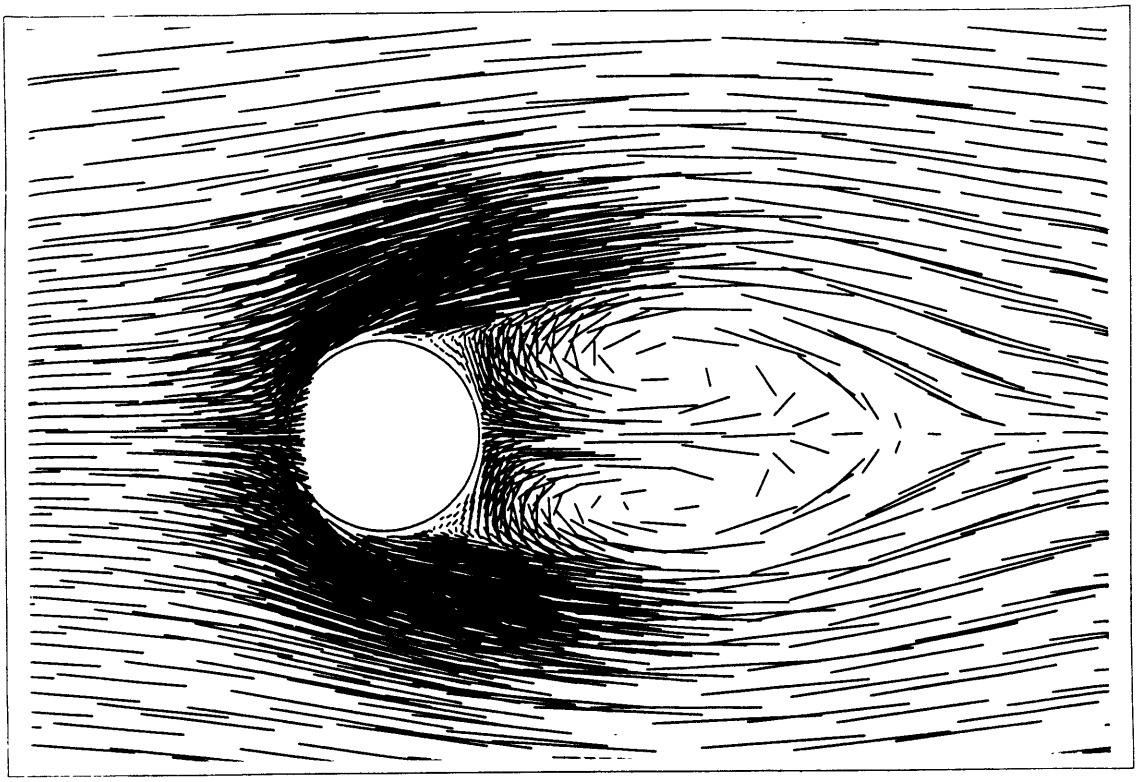

Figure 6-14: Non-Dimensional Time = 7

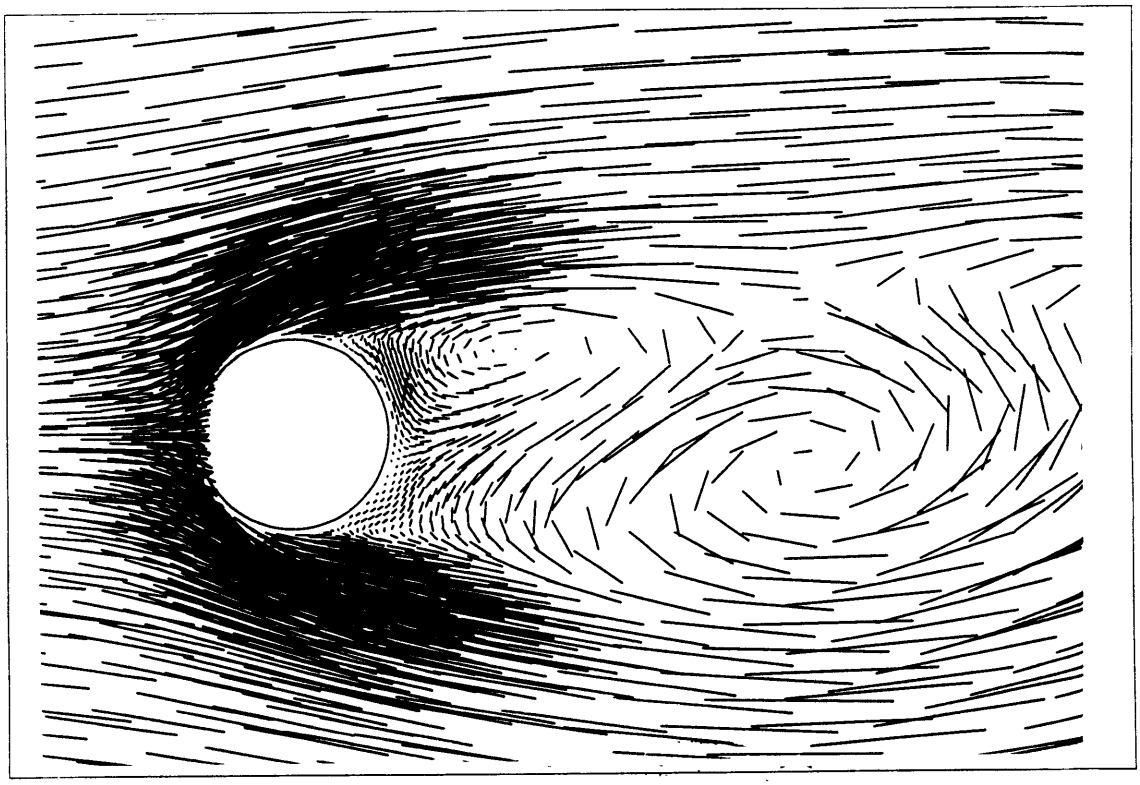

Figure 6-15: Non-Dimensional Time = 20

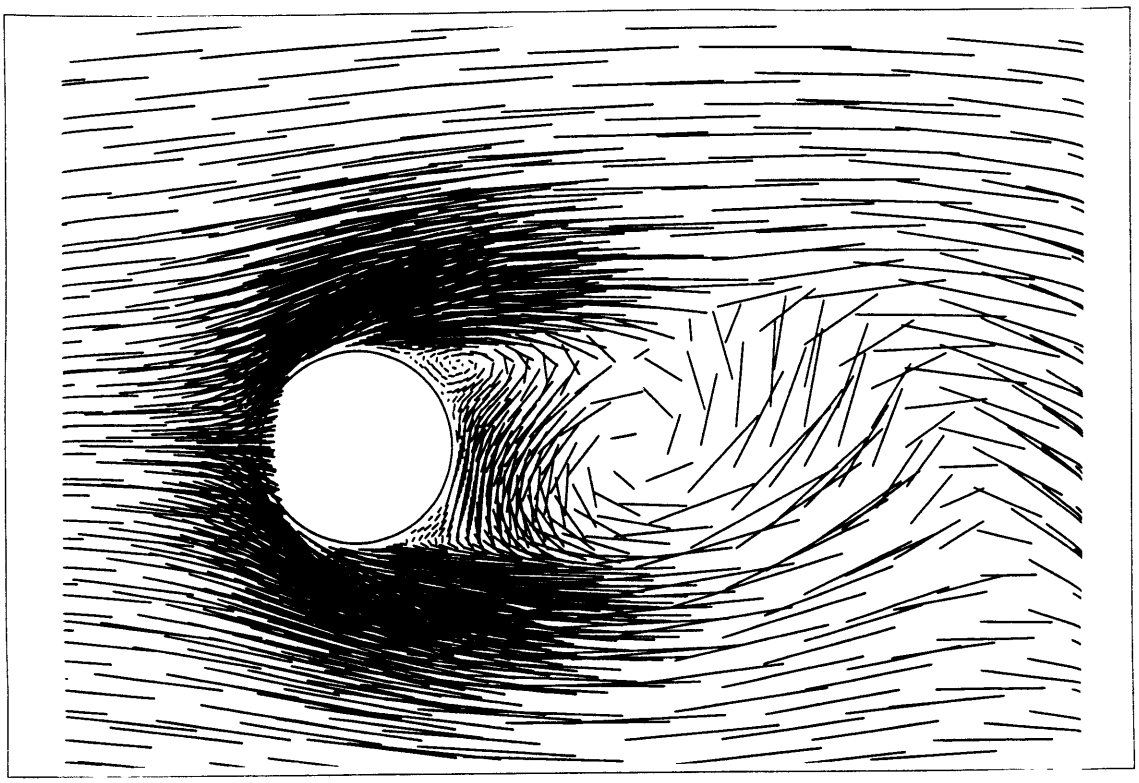

Figure 6-16: Non-Dimensional Time = 30

The lift and drag coefficient history of the developing flow is shown in figure (6-17). After the flow is developed, the lift and drag coefficients go into a limit cycle having a period of 5.140 which corresponds to a Strouhal number of 0.1946. The lift coefficient is centered at the zero lift line and the maximum and minimum lift are ± 0.64. The mean drag for the flow is 1.30 and the maximum and minimum drag coefficient are 1.345 and 1.260 respectively. The lift and drag coefficients for the developed unsteady flow is plotted in figure (6-18).

Figure (6-19) is the delay plot of the lift coefficient $C_L(t) - C_L(t+\tau) - C_L(t+2\,\tau)$ in which τ is one tenth of the period of the vortex shedding. This plot shows how the trajectories are attracted by a stable limit cycle. It is remarkable that the limit cycles obtained from different initial conditions literally overlap. This property is a manifestation of the attractive character of the limit cycle for the continuum problem, and illustrates the excellent long-term behavior of our fractional step Navier-Stokes algorithm which accurately replicates this attractor.

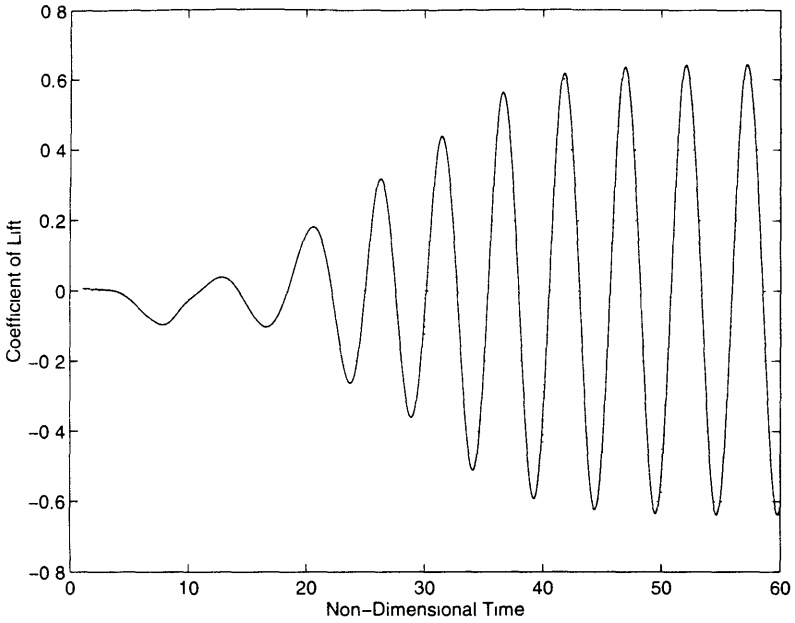

Figure 6-17: Lift Coefficient History of the Developing Flow

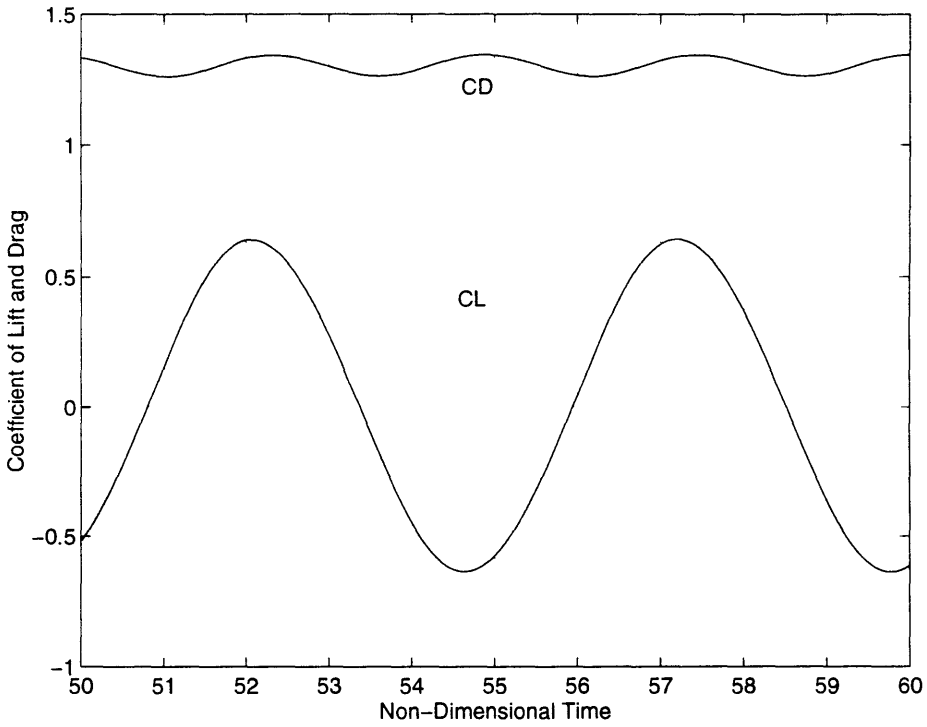

Figure 6-18: Lift and Drag Coefficient History of the Developed Flow

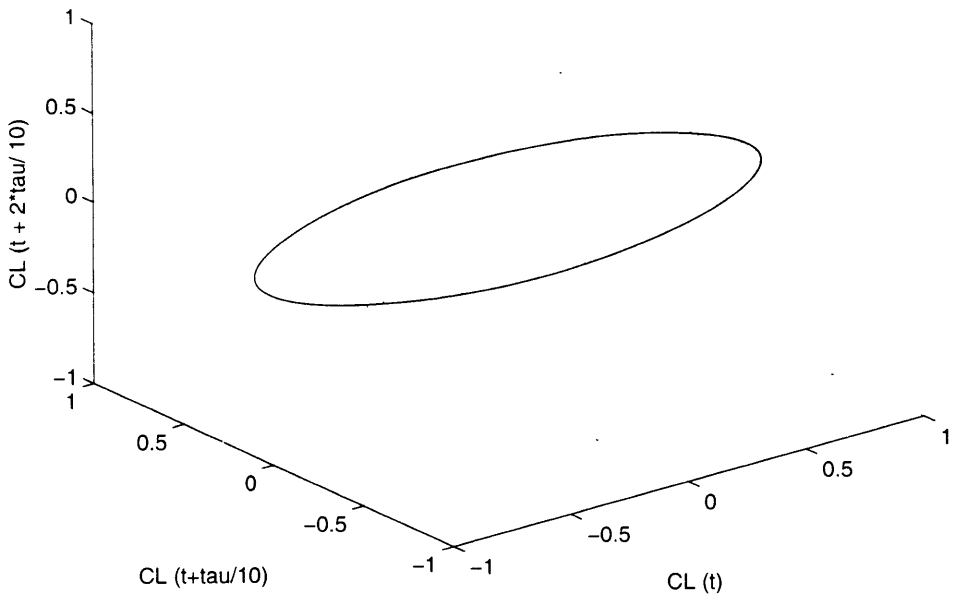

Figure 6-19: Delay Plot of the Lift Coefficient

The velocity contour, pressure contour and the streamline at time 60 are plotted in figures (6-20),(6-21) and (6-22). From the pressure contour, we can see that the large eddies which were generated by the flow separation on the cylinder, travel downstream. The eddies with positive (counter-Clockwise sense) vorticity which are those below the center line, travel in a staggered manner with the negative vortices. This is due to the fact that while the vortex on one side of the cylinder is being shed, the one on the other side is reforming.

6.2.5 Comparision

There are many calculations of two-dimensional flow over a circular cylinder which can be found in the literature. By no means would we be able to list them all here, a few of them are chosen and used as a comparison of results obtained from the research on which this thesis is based.

In the following table, the coefficient of lift, drag and Strouhal number obtained from different methods are listed. The first one listed is the result from our fractional step Navier-Stokes scheme and the following six are results obtained from other numerical schemes [3]. Then the results in [27] suggest that the continuation of the

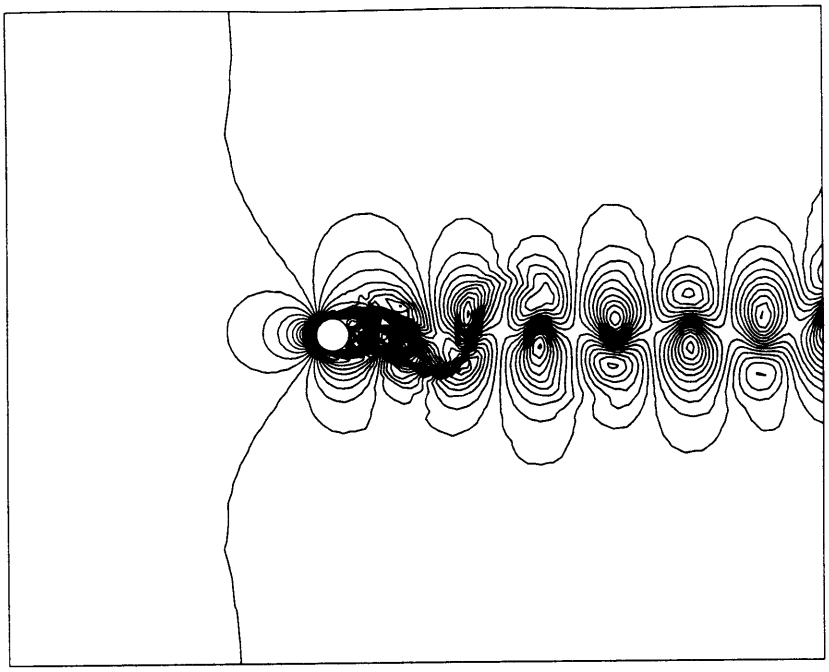

Figure 6-20: Velocity Contour at Time = 60

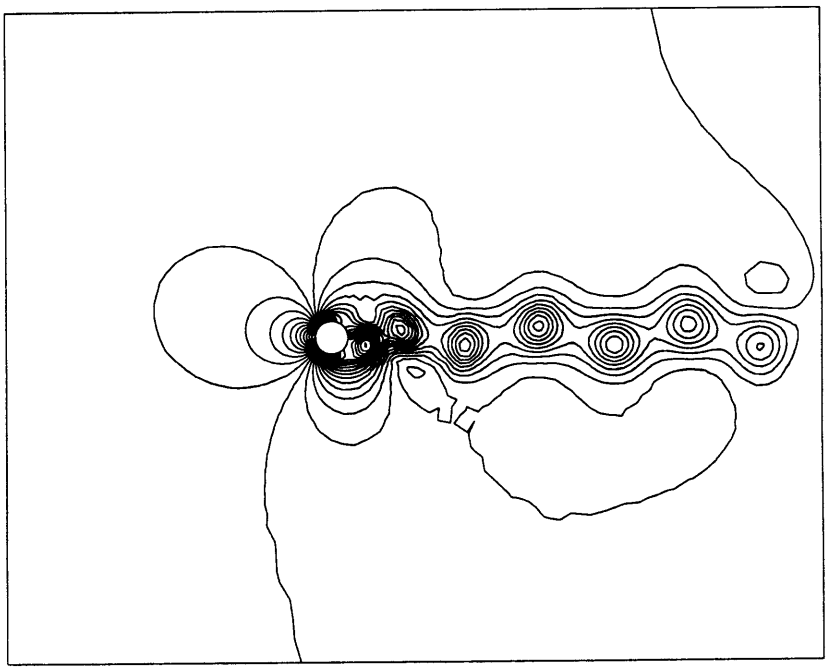

Figure 6-21: Pressure Contour at Time = 60

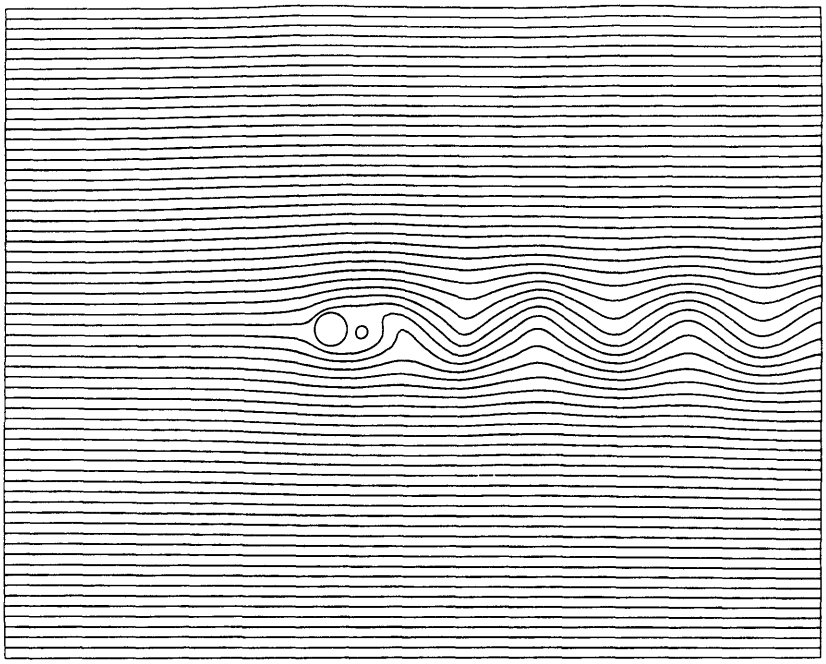

Figure 6-22: Stationary Streamline at Time = 60

universal curve corresponding to the parallel shedding provide us the results from the empirical method. Finally, actual experimental studies of the wake of circular cylinders [5],[18] results are also listed. These results provide a good basis for validation of our two dimensional Navier-Stokes algorithm.

Reference	Type	Lift Coefficient	Drag Coefficient	Strouhal Number
Present	Computation	$\pm$ 0.64	1.30 $\pm$ 0.042	0.195
Belov	Computation	$\pm$ 0.64	1.19 $\pm$ 0.042	0.193
Rogers, Kwak	Computation	$\pm$ 0.65	1.23 $\pm$ 0.05	0.185
Miyake et al	Computation	$\pm$ 0.67	1.34 $\pm$ 0.043	0.196
Lecointe et al	Computation	$\pm$ 0.5	1.58 $\pm$ 0.0035	0.194
Lin et al	Computation		1.17	
Henderson	Computation			0.197
Roshko	Curve Fit			0.190
Williamson	Curve Fit			0.197
Kovaznay	Experimental			0.19
Roshko	Experimental			0.19

Table 6.1: Comparision Among Different Results Presented in the Literature

6.3 Rotating Cylinder (Viscous)

In this section we extend our Navier-Stokes model to simulate flow around a rotational circular cylinder. There are two examples given in this section including the one with steady rotation and rotation with varying phase. Both of these examples are done with a Reynold's number of 100.

6.3.1 Steady Rotation

The flow associated with a circular cylinder which begins its rotational motion impulsively in a moving fluid is a rather complex one. It includes the unsteady boundary layer separation flow which interacts with the thin shear layers and wake flow, and generates complex unsteady lift and drag forces.

Before rotation begins, the simulation is brought to a steady-state non-rotating cylinder flow. Then by imposing the velocity boundary conditions on the cylinder having an angular velocity with a magnitude of two in the clockwise direction, the vortex street then became affected and started to turn below the center line while a net lift in the upward direction is created on the cylinder. Figure (6-23) shows the history of the coefficient of lift and drag before and after the cylinder begins to rotate. Notice that it takes very short time (about 10 non-dimensional time) for the transition between when the rotation starts and when the flow comes to a new steady-state. The velocity, pressure and streamline contour for a rotating cylinder with angular velocity of two are plotted in figure (6-24), (6-25) and (6-26).

Similar simulations were performed for rotating cylinder having angular velocity equals to $0.25, 0.5$ and 1.0. The following table summarizes the coefficient of lift, drag and Strouhal number for different values of steady angular rotation. From the table, we can conclude that the mean coefficient of lift increases with increasing angular velocity while the mean coefficient of drag decreases. However, the Stouhal number decreases for small angular velocity but it increases as the angular velocity gets larger.

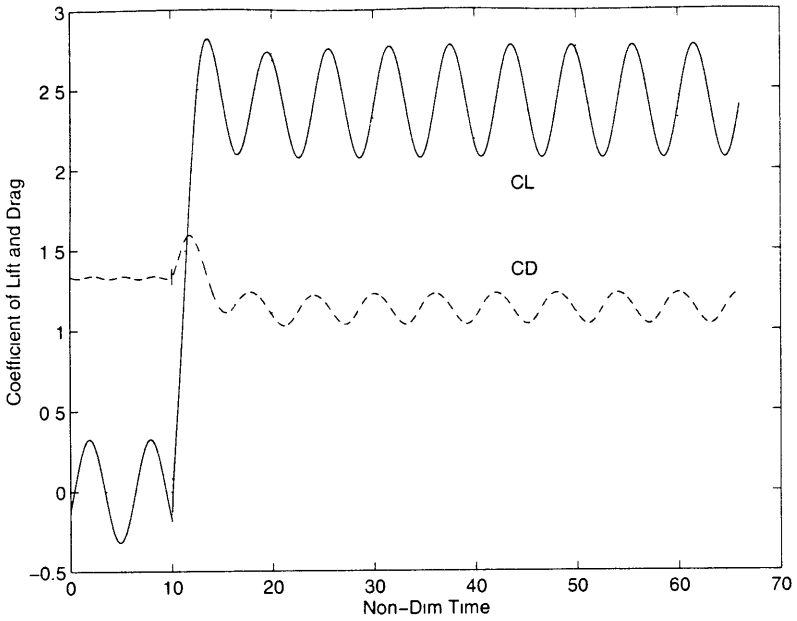

Figure 6-23: CL and CD Plot for the Rotating Cylinder ($\omega = 2$)

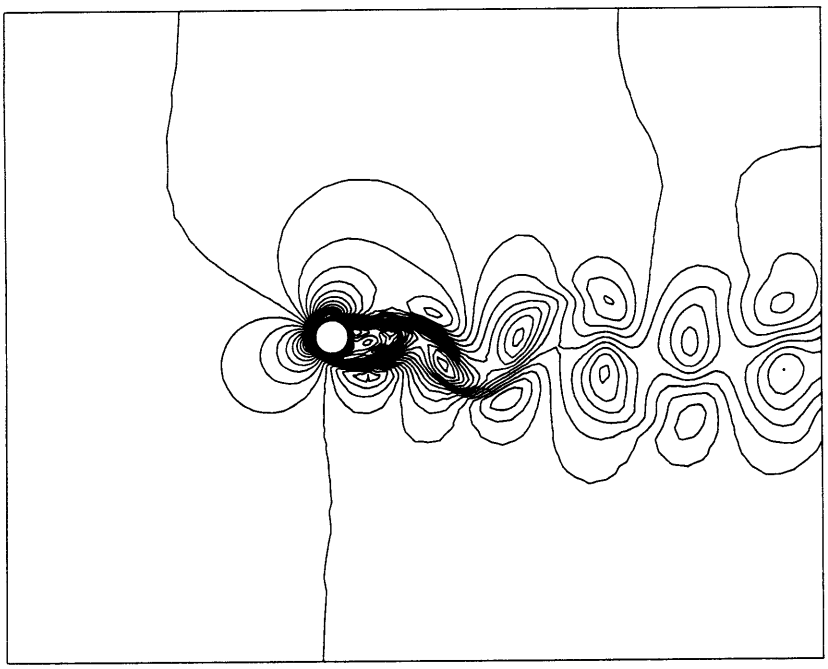

Figure 6-24: Velocity Contour for Rotating Cylinder ($\omega = 2$)

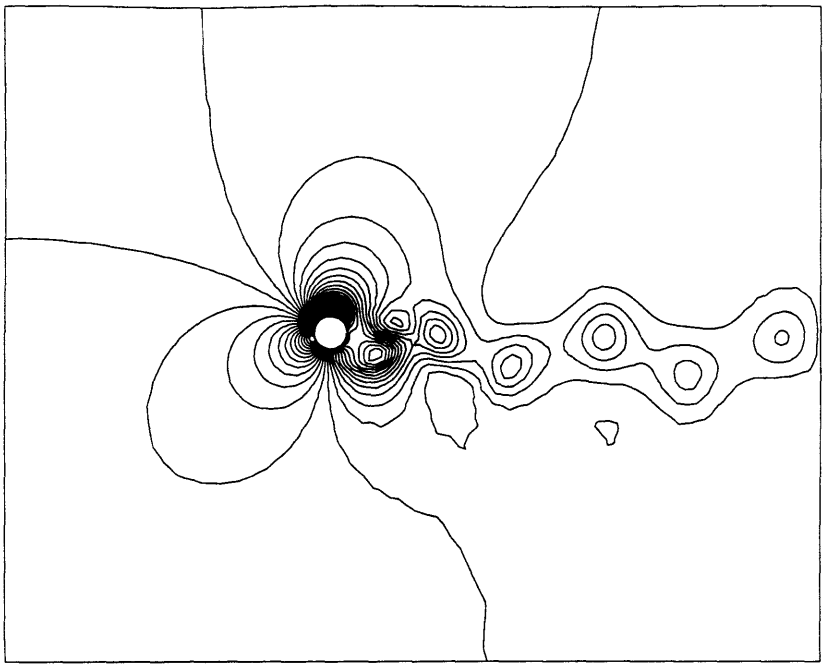

Figure 6-25: Pressure Contour for Rotating Cylinder ($\omega = 2$)

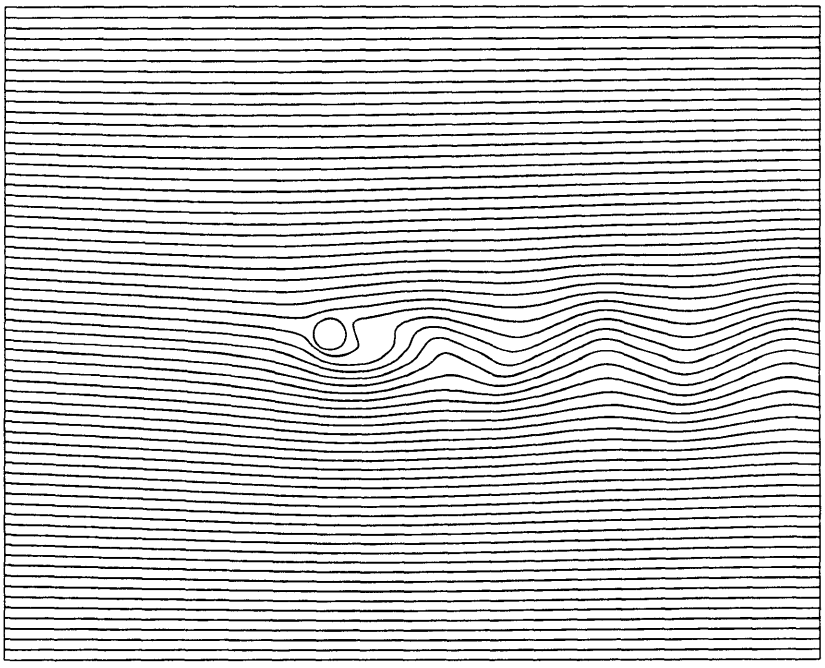

Figure 6-26: Stationary Streamline for Rotating Cylinder ($\omega = 2$)

Angular Velocity	Mean Coeff. of Lift	Mean Coeff. of Drag	Strouhal Number
0	0.001	1.33	0.1656
0.25	1.99	1.24	0.1603
0.5	2.16	1.22	0.1594
1.0	2.27	1.16	0.1680
2.0	2.41	1.14	0.1766

Table 6.2: Comparison of Characteristic Numbers for Different Angular Velocity

6.3.2 Rotation with Varying Phase

When the cylinder is set into an oscillatory rotation at frequencies close to the natural shedding frequency, the flow "locks on" (synchronizes with the cylinder rotation) and the vortex shedding becomes much stronger.

For illustration purpose, a rotating cylinder starts its rotation from the non-rotating vortex shedding flow initial condition with an angular velocity of :

$$\gamma = 2 \sin(\frac{2 \pi t}{6}) \tag{6.6}$$

in which its amplitude and period of rotation are two and six respectively. The history of the coefficient of lift and drag is shown in figure (6-27). The velocity and pressure contour are plotted in figure (6-28) and (6-29). Notice that the mean lift is still zero because the magnitude of the angular velocity has a sinusoidal shape. However, the maximum and minimum lift are now at ± 2.21 which is four times more than the one for a non-rotating cylinder. Also the coefficient of drag increases to 2.245 ± 0.22 while the Strouhal number remains the same value as the non-rotating cylinder flow.

6.4 Moving Cylinder and Airfoil (Viscous)

In order to demonstrate that vortices can be controlled directly, in the sense of altering their position and strength; and also that energy can be extracted from large scale patterns, in this section we use our Navier-Stokes solver to simulate a flow situation in which both a circular cylinder and a moving airfoil is present in the flow field.

A NACA 0012 airfoil, with a chord length of two, is placed in the wake of a cylinder

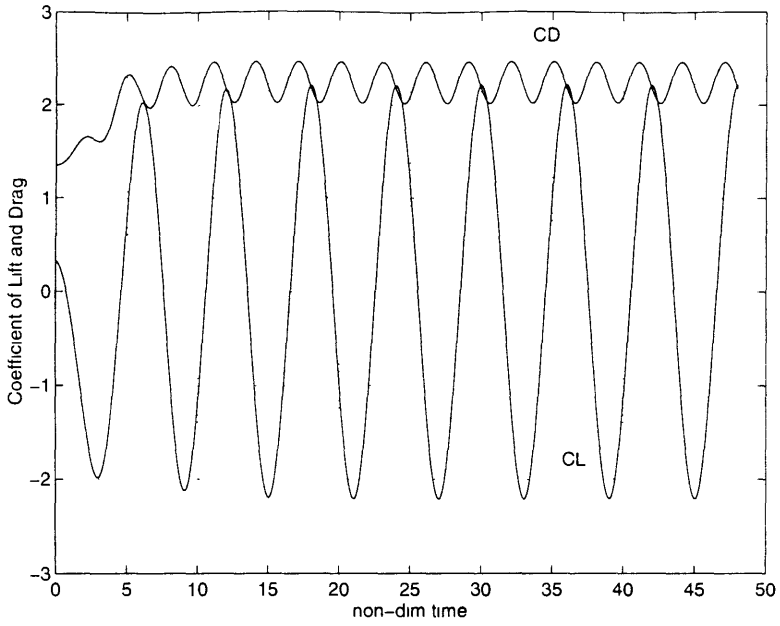

Figure 6-27: CL and CD Plot for the Rotating Cylinder ($\omega = 2\ \sin\ (\frac{2\ \pi\ t}{6})$)

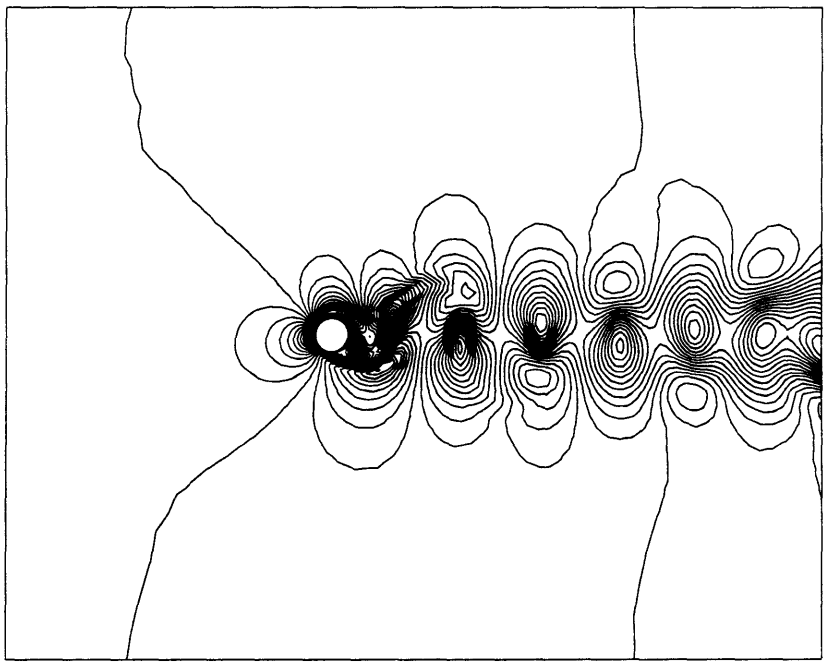

Figure 6-28: Velocity Contour for Rotating Cylinder ($\omega = 2\ \sin\ (\frac{2\ \pi\ t}{6})$)

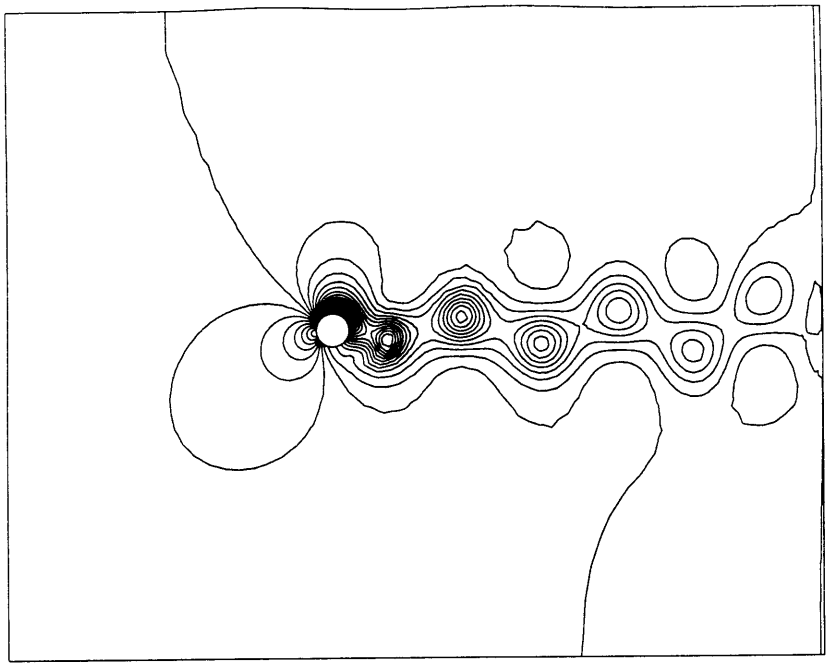

Figure 6-29: Pressure Contour for Rotating Cylinder ($\omega = 2\ \sin\ (\frac{2\ \pi\ t}{6})$)

(diameter = 1.0), sufficiently far behind the cylinder so that it does not interfere with the vortex formation process. In our case, we chose the distance to be 4 $\frac{2}{3}$ so that our numerical simulation is consistent with the experiment done by Triantafyllou et al [9]. The grid displayed in figure (6-30) is used for this simulation.

By using the grid movement technique discussed in chapter four, we can simulate the flow in which the foil performs a heaving and pitching oscillation at a frequency close to the Strouhal frequency of the cylinder while the cylinder and foil move forward at Reynold's number of 100. We could then compare our simulation result with the flow visualization result obtained by Triantafyllou et al.

Before solving the flow solution for a moving airfoil, we first simulate a flow field with a fixed cylinder and a fixed airfoil with an angle of attack of 10 degrees. The velocity and pressure contour are shown in figure (6-31) and (6-32) for time equals to 80 non-dimensional time units. Notice that this simulation starts with initial condition of unity y-component velocity and zero x-component velocity everywhere, except for the boundary points on the cylinder and airfoil for which the velocity components are set of be zero at all time.

From these figures, we could see that the vortices generated by the cylinder were

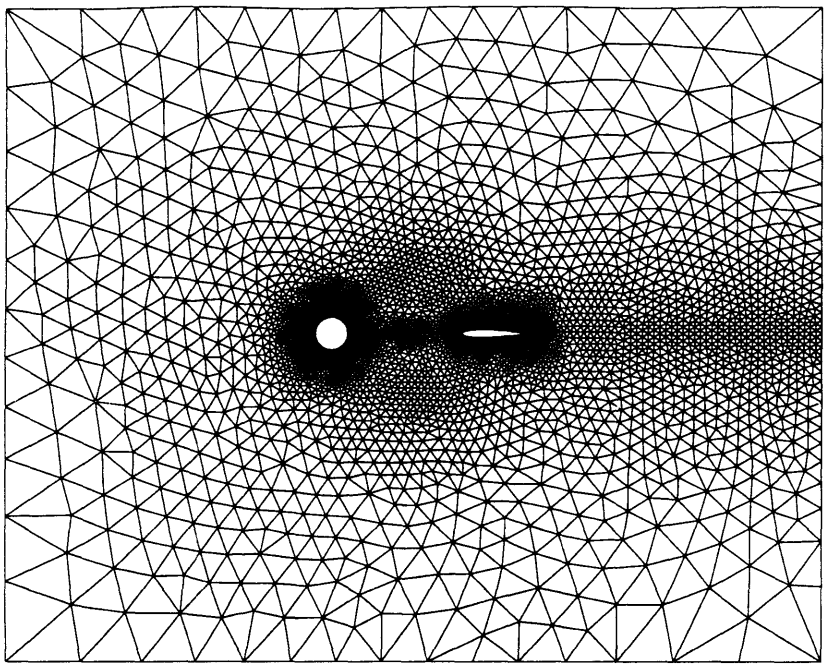

Figure 6-30: Grid Used for Cylinder/Airfoil Simulation

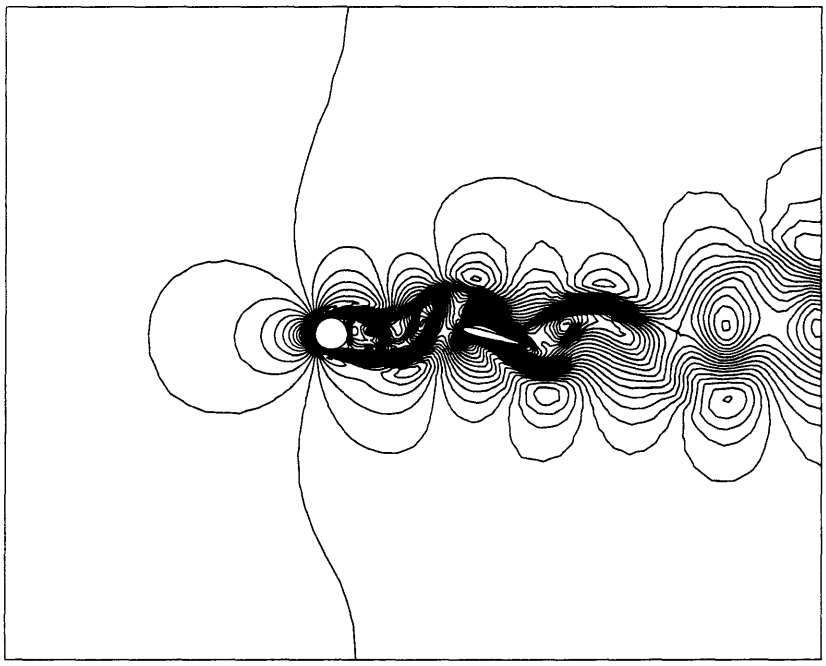

Figure 6-31: Velocity Contour (A.O.A = 10 degrees)

destroyed by the presence of the airfoil and then the vortices are regenerated by the airfoil in the downstream. Little information exists in the literature on the behavior of vortex regeneration behind a cylinder and an airfoil. However, the result from our Navier-Stokes solver seems to be reasonable and so we are ready to simulate the moving airfoil flow field.

As in the Triantafyllou's experiment, the airfoil motion consists of a combined transverse translation (heave) and rotation (pitch). The heave displacement is given as follows:

$$y(t) = A\ sin(2\ \pi\ f\ t + \psi) \tag{6.7}$$

The pitching motion has the same frequency as the heave motion; it has the form of:

$$\theta\ (t) = \theta_0\ sin(2\pi f t + \psi + \phi) \tag{6.8}$$

The experimental visualization result obtained by Triantafyllou et al were done with a Reynold's number of 550. They tested with many settings for different choices of heave/pitch frequency and amplitude. The main result they found was that the oscillating foil altered substantially the position and strength of the incoming cylinder vortices. The effect of the foil on the vortices was felt only near and downstream of the foil, while the near wake of the cylinder was not changed noticeably. Also, they identified the three distinct stable modes:

- Expanding Wake : When oncoming cylinder vortices pair up with vortices shed by the foil to form mushroom like structures, drifting away transversely to the flow.

- Destructive interaction: When oncoming cylinder vortices are repositioned by the foil and they interact with vorticity of the opposite sign shed by the trailing edge of the foil to form ultimately weakened vortices, or even vortices of the opposite circulation.

- Constructive interaction : When oncoming cylinder vortices meet with vortices

of the same sign shed by the foil to form much stronger vortices compared to their original strength.

In our simulation, we chose the amplitude and frequency of the heaving and pitching motion to be 0.667 and 0.2 respectively; and ψ, ϕ to be zero and $\pi/2$ respectively. The velocity and pressure contour for the flow at time unit equals to 12 starting from the non-moving condition are shown in figure (6-33) and(6-34). These show that the simulation gives a sensible solution. However, because of the relatively long running time the Navier-Stokes code requires; it was not too easy to perform the same kind of visualization procedures done in their experiment. However, the expanding wake could be found quite distinctively from our simulation. We could see that the vortex generated by the cylinder joins up with a newly formed foil vortex of the opposite sign to form a pair, slowly drifting away from the center line of the wake, owing to the mutual velocity induction. Thus a street of vortex pairs is formed, resulting in an expanding wake. Unfortunately, the other modes they described in the experiment result could not be found very clearly from our simulation and this probably is due to the fact that the Reynold's number that we are running is 100 instead of 550. Also, a different suitable setting has to be found for our case in which Reynold's number equals to 100. Due to the limitation of our time, research on this will be done and reported more extensively in the near future.

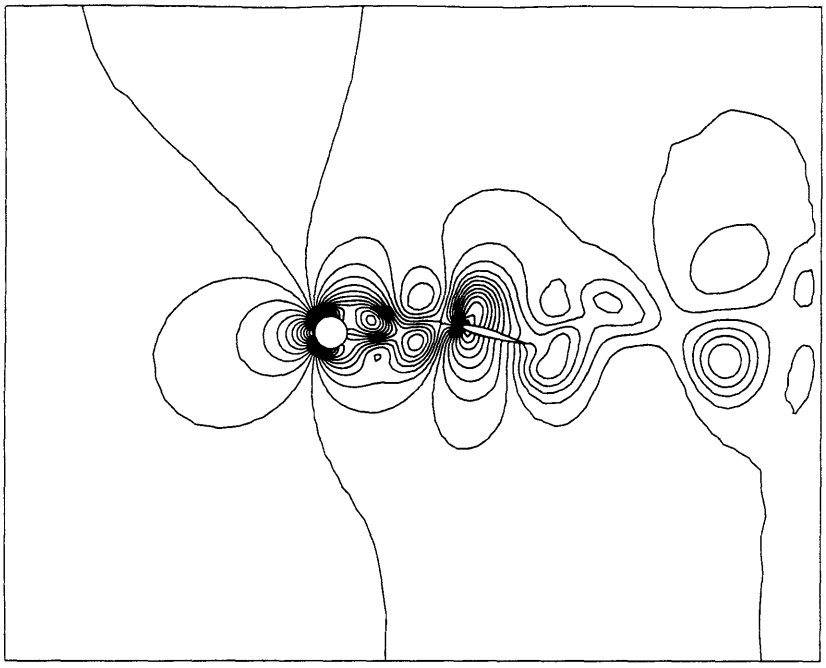

Figure 6-32: Pressure Contour (A.O.A = 10 degrees)

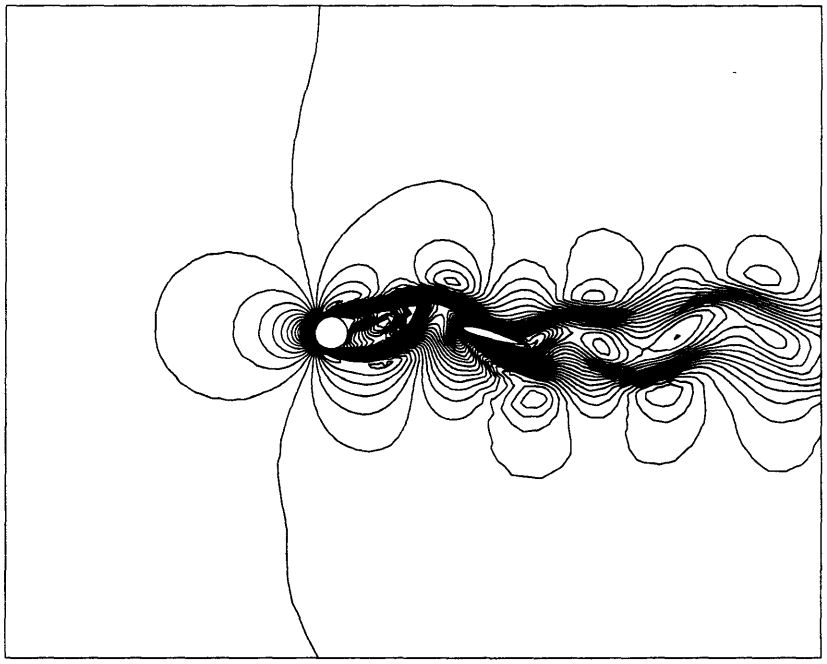

Figure 6-33: Velocity Contour for the Moving Airfoil case

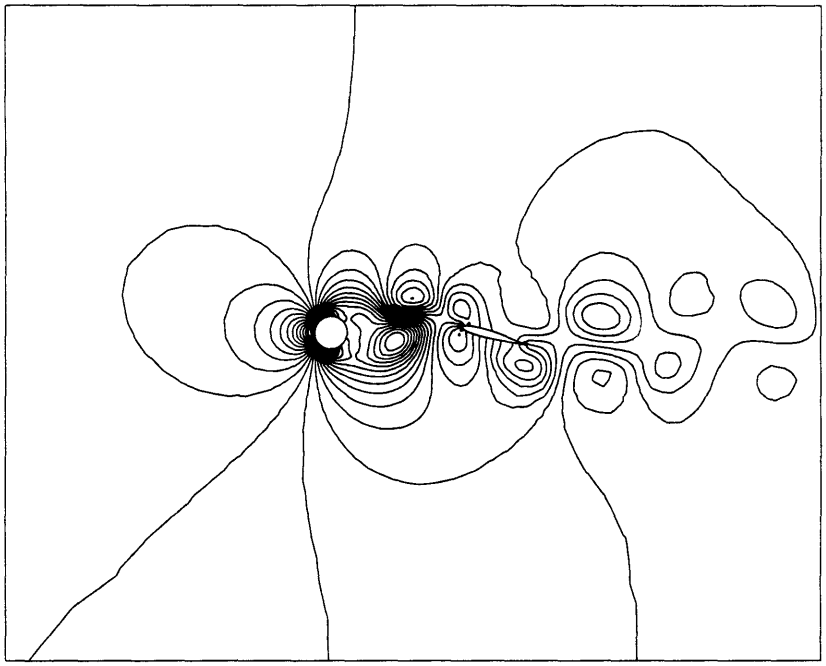

Figure 6-34: Pressure Contour for the Moving Airfoil case

Chapter 7

Active Flow Control

7.1 Introduction

With the advent of high speed micro-electronics, the concept of active control of the properties of a fluid flow has become accepted as a technical possibility, with many potential benefits and significant advantages over conventional, passive control methods. To date, much of the work in this area has been directed towards the suppression of instabilities using a simple heuristic model [23] or neural network techniques [29]. However, many fluid problems are highly non-linear and they are too complex for the heuristic approach to model.

In these cases, a well established technique for extracting the dominant features or degrees of freedom from a sample data set is the proper orthogonal decomposition(POD) which yields a set of orthogonal basis functions which are optimal in the sense of energy captured from the data set [15, 16, 17]. These functions may then be used as a basis for series expansions of the flow variables, and a Galerkin projection of the governing equations subsequently yields a set of ordinary differential equations representing the system.

This approach has been employed extensively in turbulence studies [4], and also in examples of larger scale unsteadiness [6]. However, with some exceptions, previous work has been largely descriptive, in that the low order model has generally been used to simulate the flow which gave rise to it. The technique will thus require development

for our purposes.

The test case we choose is the two-dimensional flow past a circular cylinder, with control action achieved via cylinder rotation. The flow data is obtained by the Navier-Stokes code we have discussed in the previous chapters.

In this chapter, we will first discuss the POD process and the results of the decomposition obtained for our test cases. Then we will discuss how we use the POD results to generate a set of low order models to simulate our flow solution. Finally, we will illustrate the application of the resulting low order models to the optimal control problem.

7.2 Proper Orthogonal Decomposition of the Cylinder Flow

7.2.1 Introduction

In recent years, the Proper Orthogonal Decomposition (POD) has become popular as means of extracting dominant energy-containing structures from flow field data and, by using these structures as basis functions, generating low order dynamical models for the associated systems. Many of the studies in this area have been in the field which originally motivated the POD, for instance, turbulence flow; however, the technique has also found application on a number of larger scale flows [4, 6]. A very good reference for the subject can be found in the paper by Sirovoch [15, 16, 17]. In this section, we briefly describe the general POD approach and then in the next section we will discuss the result for our cylinder test case.

Given a set of flow realizations, $\mathbf{u}(\mathbf{x},t)_j, j = 1...N$, the 'snapshot' form of the POD generates N basis functions ('modes'), $\psi_i(\mathbf{x})$ which are mutually orthogonal. These basis function together with the mean flow form the following approximate representation of the flow field.

$$\mathbf{u}(\mathbf{x},t) = \mathbf{u}_m(\mathbf{x}) + \sum_{i=1}^{N} y_i(t)\psi_i(\mathbf{x}) \tag{7.1}$$

This representation is optimal, in the sense that it maximizes the captured unsteady energy, $|\mathbf{u}(\mathbf{x},t) - \mathbf{u}_m(\mathbf{x})|^2$. In other words, given a data set, the POD extracts the main energy-containing structures in that data set.

In this sense, the POD has a purely descriptive nature. However, the basis functions it produces may be used to generate a predictive model via a Galerkin projection of the Navier-Stokes equations described as follows :

$$(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} + \nabla p - \nu \nabla^2 \mathbf{u},\ \psi_i) = 0 \tag{7.2}$$

Note that the inner product is defined as:

$$(\mathbf{a}, \mathbf{b}) = \int_\Omega \mathbf{a} \cdot \mathbf{b}\ d\Omega \tag{7.3}$$

Since the basis functions ψ_i are divergence-free by construction, equation (7.4) equals to zero for typical boundary conditions.

$$\int_\Gamma \psi_i \cdot \nabla p d\Omega = \int_\Gamma p \psi_i \cdot \mathbf{n}\ d\Gamma \tag{7.4}$$

Notice that p and $\mathbf{n}$ denote the pressure and the boundary normal respectively. Substitution of equation (7.1) into equation (7.2) then yields a set of evolution equations for the mode amplitudes $y_i(t)$, which may be used to predict the time history of the flow from a given initial condition. Often the number of degrees of freedom required for this low order model to be accurate is small and a significant decrease in complexity is achieved in comparison of the original system.

7.2.2 POD for Stationary Cylinder Flow

Derivation of POD Basis Functions

In this case, by using technique described in the previous section, we derive the POD basis functions using the "method of snapshots" [6]. Here we take a number of flow realizations (snapshots) and, having subtracted their average value (the mean flow),

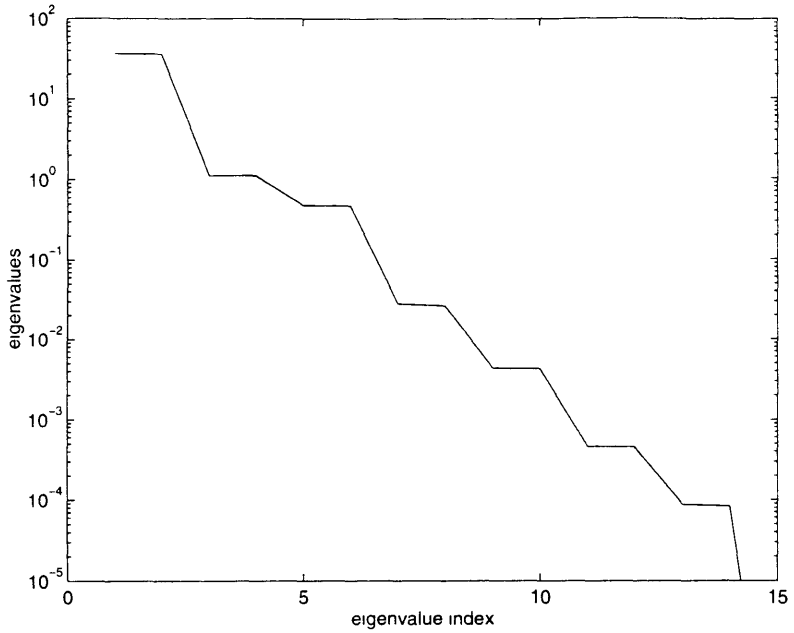

Figure 7-1: Eigenvalue Spectrum

form a correlation matrix whose entries consist of the inner products of snapshot pairs. The eigenvalues of this matrix then represent the energies of the POD modes present in the snapshots, and the elements of the associated eigenvectors give the relative weightings to be used in constructing the modes form the snapshots.

Figure (7-1) shows the eigenvalue spectrum from the correlation matrix formed from a set of 15 snapshots of the cylinder flow, taken over two vortex shedding periods. Clearly, most of the fluctuation energy is captured by the first two modes alone, and if six modes are taken the remaining eigenvalues account for less than 0.1% of the unsteadiness energy in the snapshots. The decomposition of the flow achieved here is thus highly efficient. These results are in agreement with those of Deane el al [6], and the associated basis functions also correspond satisfactorily. Velocity contours for the first four are plotted in Figure (7-2) to (7-5).

Note the pairing of similar patterns, shifted spatially, a result of the convective nature of the flow. The amplitudes of these basis functions present in the snapshots are plotted in figure (7-6), from which it is clear that the basis functions correspond very closely to the coefficients of a temporal Fourier series expansion of the periodic

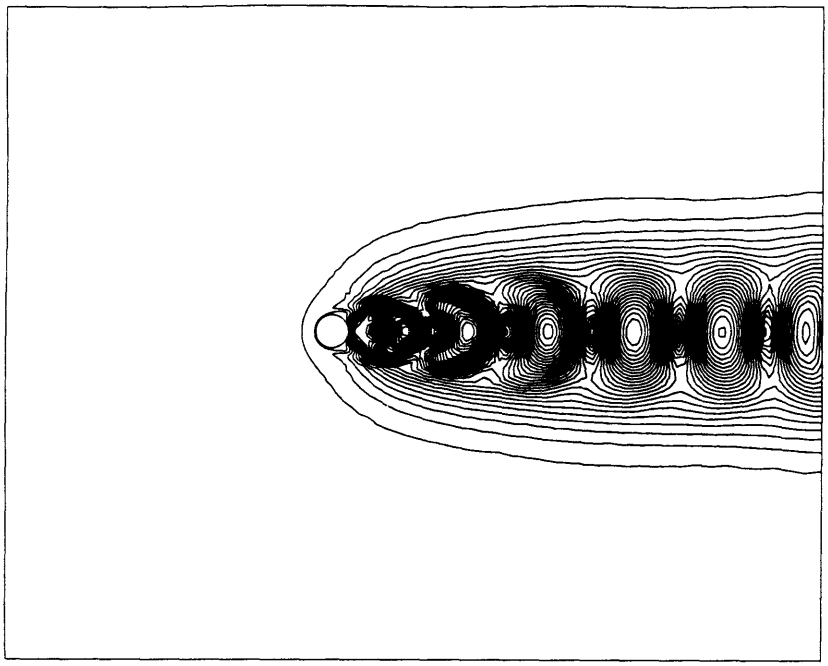

Figure 7-2: Velocity Contour of the First Mode

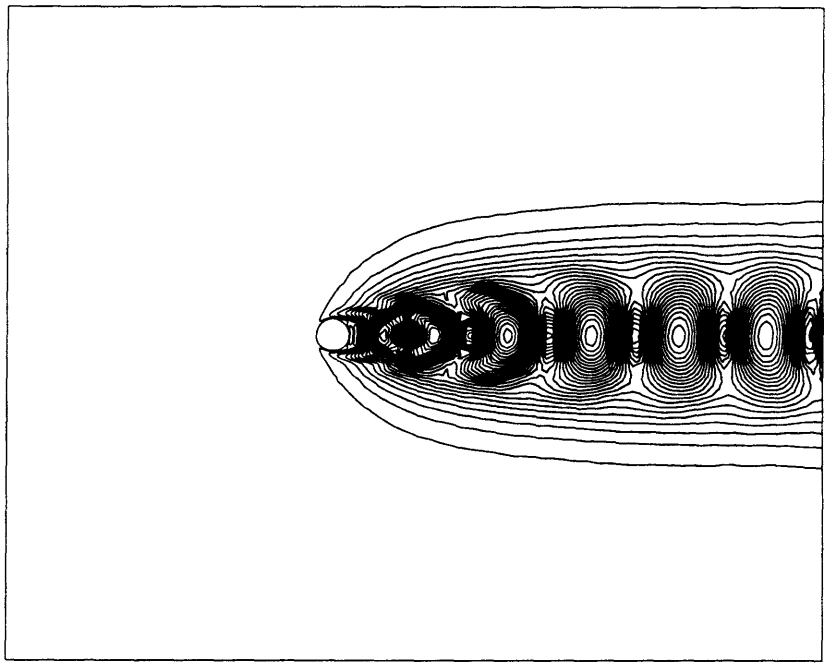

Figure 7-3: Velocity Contour of the Second Mode

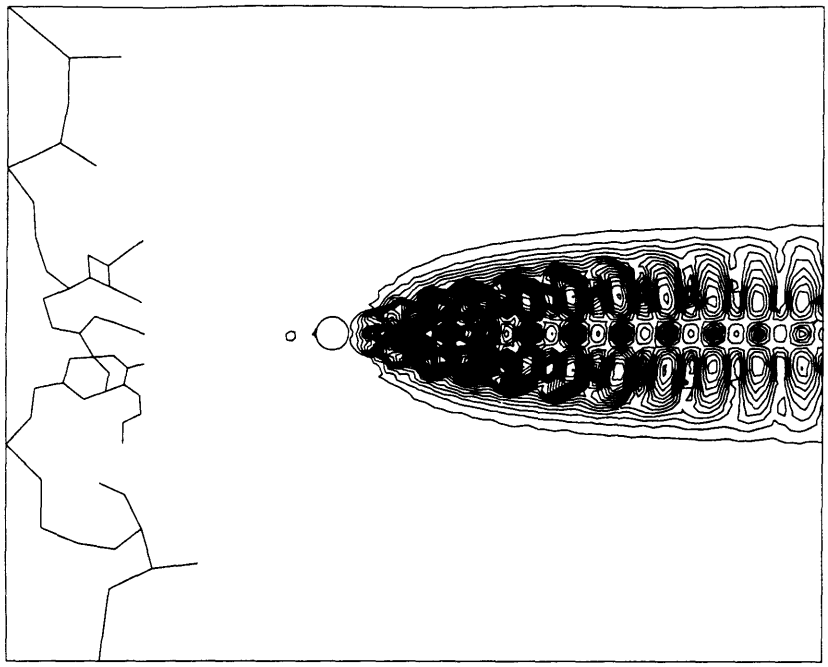

Figure 7-4: Velocity Contour of the Third Mode

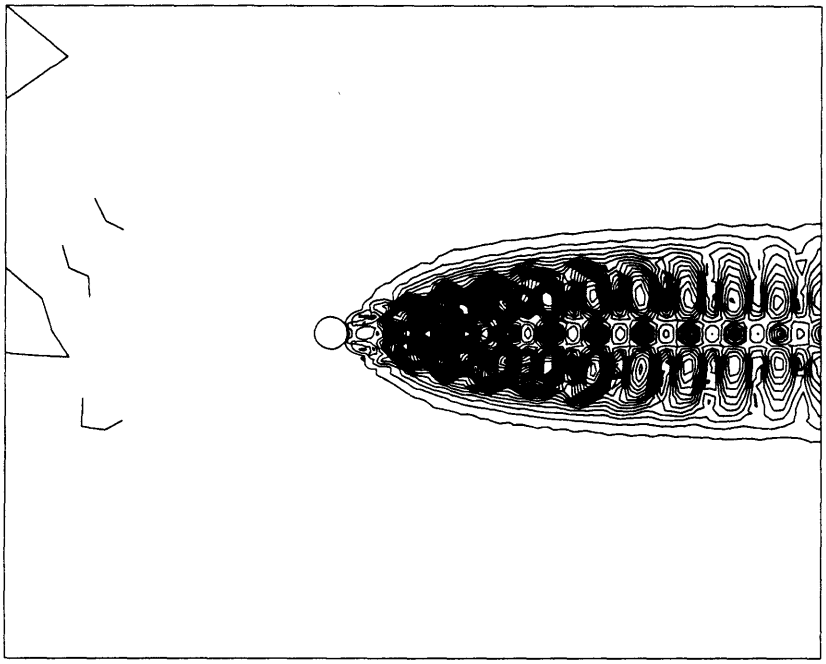

Figure 7-5: Velocity Contour of the Fourth Mode

flow. This correspondence would be exact for a perfectly convective flow; here it is remarkably close to being so, even for the higher basis functions/Fourier coefficients. An interesting corollary is that the mean flow is very nearly orthogonal to the basis functions and may be found as a mode if the correlation matrix is formed without first subtracting it from the snapshots.

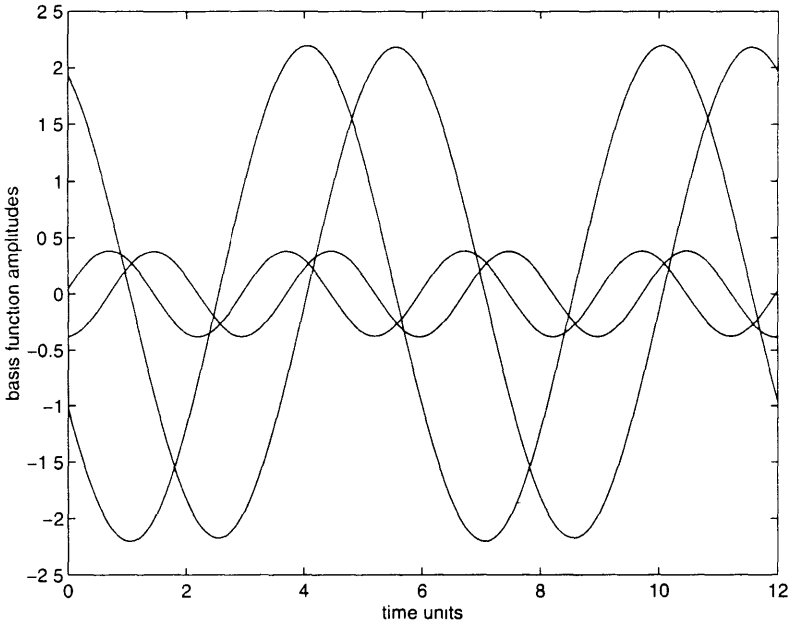

Figure 7-6: Basis Function Amplitudes for the First Six Modes

Low Order Model for the Flow

Having determined the basic functions for this flow, normalized so that $(\psi_i, \psi_i) = 1$, we can now obtain a low order model of the circular cylinder flow from the Galerkin projection give by equation (7.2). Note first that $\psi_i \cdot \mathbf{n} = 0$ on the inflow, upper, lower and cylinder boundary points; and $p = 0$ on the outflow boundary surface; so that by equation (7.4) the pressure term drops out. The resulting N coupled ordinary differential equations can then be written as follows:

$$\frac{dy_i}{dt} = a_i + b_{ij}y_j + c_{ijk}y_j y_k \tag{7.5}$$

Note that, summation over repeated indices ranging from 1 to N is assumed. The coefficients are given by the following expressions:

$$a_i = -(\psi_i, (\mathbf{u}_m \cdot \nabla)\mathbf{u}_m) + \frac{1}{Re}\,(\psi_i, \nabla^2 \mathbf{u}_m) \tag{7.6}$$

$$b_{ij} = -(\psi_i, (\psi_j \cdot \nabla)\mathbf{u}_m) - (\psi_i, (\mathbf{u}_m \cdot \nabla)\psi_i) + \frac{1}{Re}\,(\psi_i, \nabla^2 \psi_j) \tag{7.7}$$

$$c_{ijk} = -(\psi_i,\ (\psi_j \cdot \nabla)\psi_k) \tag{7.8}$$

For numerical implementation, these expressions are further simplified by using Gauss' theorem to reduce the order of the Laplacian operator in the viscous terms. After forming the coefficients from the first six basis functions, equation (7.5) is readily integrated from a given set of initial conditions. By employing a fourth order Runge-Kutta scheme, we can obtain a set of predicted time histories for the mode amplitudes which may be compared with the projection of the numerical simulation onto these functions. As shown in figure (7-7), the integrated time histories for the first two modes match the observed results extremely well, while the amplitudes of the higher modes are less accurately predicted. This again agrees with the results of Deane et al [6]. As a result, we conclude that a six mode model provides an entirely satisfactory description of the long term dynamics for this case.

7.2.3 POD for the Locked-On Flow Past a Rotating Cylinder

As illustrated in the last Chapter, when the cylinder is set into an oscillatory rotation at frequencies close to the natural shedding frequency, the flow "locks on" and the vortex shedding becomes stronger. On performing the POD as described previously, we find that the eigenvalue spectrum is still concentrated around the first few modes. The eigenvalues distribution can be found in figure (7-8). Also the modes have similar

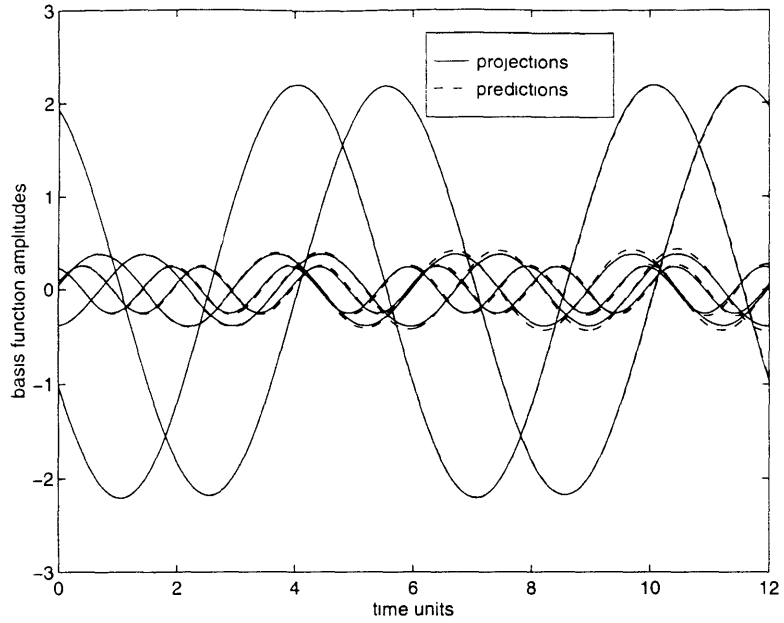

Figure 7-7: Mode Amplitude Comparison

structure to those derived for the stationary cylinder. One might thus expect to be able to form a low-dimensional model exactly as before. However our cylinder is now rotating, so that the velocity expansion in equation (7.1) is not a valid candidate for $\mathbf{u}$ in the Galerkin projection. In other words, equation (7.2) no longer automatically satisfies the boundary conditions for any arbitrary set of mode amplitudes y_i. This problem is the first indication of the difficulties involved in applying t he POD to active control of fluid flows. Since it arises from the presence of non-homogeneous boundary conditions, it will inevitably crop up in any situation where control is applied at the boundaries. Thus in the following section ,we consider how this difficulty may be addressed.

A further issue which should be raised initially at this point is the extent to which the extreme optimality of the POD basis functions derived here may be a disadvantage in control applications. Once control action is applied to the flow it may be expected to diverge from its steady state, and the ability of the low order model to capture that divergence will be crucial to successful control. Thus a set of basis functions which is perfectly tuned to the steady state is likely to be unsatisfactory, as we shall

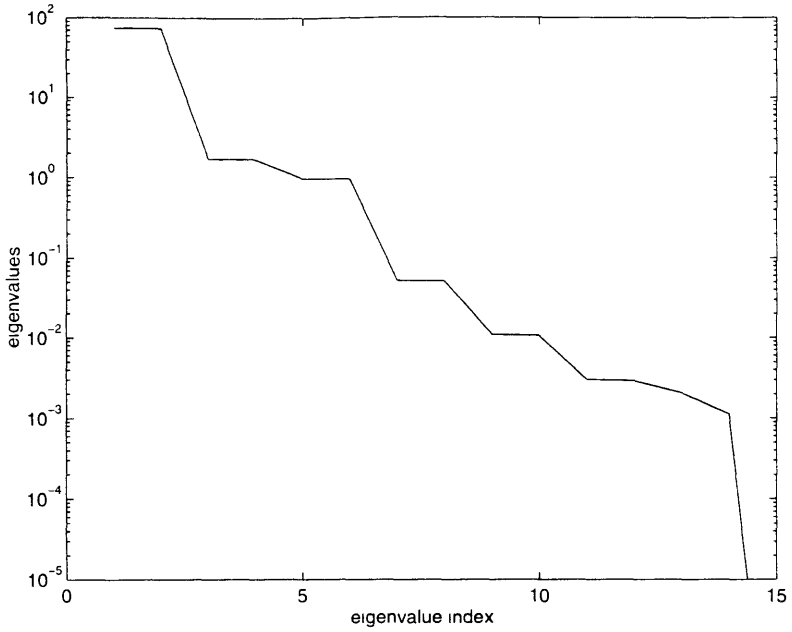

Figure 7-8: Eigenvalues for the Locked-On Flow

find subsequently.

7.3 Incorporation of Control Surface Motion into the Low Order Model

To introduce control into the problem, we consider cylinder rotation as a means of actuation. A similar control idea has been suggested by Burns [32]. In this approach we remove the inhomogeneous boundary condition on the cylinder by subtracting a suitable "control function" form each snapshot before generating the POD basis functions. To retain the divergence-free property of the basis functions, our control function must also be divergence-free; we additionally require that it satisfy the appropriate homogeneous boundary conditions on all the other boundaries. Beyond this, the function is arbitrary; however it is convenient to take it to be the flow solution generated by the simulation for the steady cylinder rotation flow with unit angular velocity; denoting this solution by $\mathbf{u}_c(\mathbf{x})$, the modified snapshot set can now

be represented as :

$$\mathbf{u}_c(\mathbf{x}, t_i) - \gamma(t_i)\ \mathbf{u}_c(\mathbf{x}) \tag{7.9}$$

in which t_i is the time of the i th snapshot. The mean flow and the basis functions are obtained from this modified set in the usual way. The Ψ_i thus generated have zero velocity on the cylinder surface, and the velocity expansion is then:

$$\mathbf{u}(\mathbf{x}, t) = \mathbf{u}_m(\mathbf{x}) + \gamma(t)\mathbf{u}_c(\mathbf{x}) + y_i(t)\mathbf{\Psi}_i(\mathbf{x}) \tag{7.10}$$

On substituting this expansion into the Galerkin projection in equation (7.2), we obtain the following system of equations :

$$\frac{dy_i}{dt} = a_i + b_{ij}y_j + c_{ijk}y_jy_k + d_i\frac{d\gamma}{dt} + (e_i + f_{ij}y_j)\ \gamma + g_i\gamma^2 \tag{7.11}$$

where a_i, b_{ij}, and c_{ijk} are given by equation (7.6) to (7.8), and

$$d_i = -(\Psi_i, \mathbf{u}_c) \tag{7.12}$$

$$e_i = -(\Psi_i, (\mathbf{u}_m \cdot \nabla)\mathbf{u}_c) - (\Psi_i, (\mathbf{u}_c \cdot \nabla)\mathbf{u}_m) + \frac{1}{Re}(\Psi_i, \nabla^2\mathbf{u}_c) \tag{7.13}$$

$$f_{ij} = -(\Psi_i, (\Psi_j \cdot \nabla)\mathbf{u}_c) - (\Psi_i, (\mathbf{u}_c \cdot \nabla)\Psi_j) \tag{7.14}$$

$$g_i = -(\Psi_i, (\mathbf{u}_c \cdot \nabla)\mathbf{u}_c) \tag{7.15}$$

Given initial conditions on the y_i's and a prescribed time history for γ, these equations may be integrated forwards to yield predicted y_i's, and hence predicted flow fields can be obtained by using equation (7.10).

7.4 The Optimum Control Problem

In this section we consider the problem of determining a control $\gamma(t)$ satisfying an initial condition $\gamma(0) = \gamma_0$ that will minimize a given cost function. Of interest here, are cost functions which can be written in the following general form of :

$$E(\mathbf{y}, \gamma) = \int_0^T f(\mathbf{y}, \gamma) dt \tag{7.16}$$

where $\mathbf{y}$ is a vector whose i-th component is the mode amplitude y_i. An iterative algorithm can be devised to evaluate $\gamma(t)$ from a sequence of updates starting from an initial guess γ^0 and corresponding solution $\mathbf{y}^0$. For each iteration a gradient function $G(\mathbf{y}^k, \gamma^k)$ satisfying the condition :

$$\delta E = G(\mathbf{y}^k, \gamma^k)\delta\gamma \tag{7.17}$$

is evaluated. Then, a straightforward steepest descent update :

$$\gamma^{k+1} = \gamma^k - \alpha_k G^k \tag{7.18}$$

guarantees that $E(\gamma^{k+1}) \leq E(\gamma^k)$ for a sufficiently small scalar α_k. The iteration process is stopped when the magnitude of the gradient function becomes smaller than a prescribed tolerance. Clearly, more sophisticated update algorithms can be used. In particular if $E(\gamma)$ is a quadratic positive definite form conjugate gradient directions could be employed.

7.4.1 Continuum Formulation

An efficient calculation of the gradient function in equation (7.17) can be accomplished using the ideas of Optimum Control Theory [13, 21]. In this approach the equation set (7.11) is regarded as a constraint on the problem variables y_i and γ. Introducing co-state variables $\{\phi^i, i = 1 \ldots N\}$ the cost function in equation (7.16) is replaced by :

$$E(\mathbf{y},\gamma) = \int_0^T f(\mathbf{y},\gamma)dt -$$
$$\int_0^T \phi^i(\frac{dy_i}{dt} - a_{ij}y_j - b_{ijk}y_jy_k - \gamma c_{ij}y_j - \gamma d_i - \gamma^2 f_i - g_i\frac{d\gamma}{dt} - h_i)dt \qquad (7.19)$$

Clearly, if we restrict ourselves to functions y_i satisfying equations (7.11), the minimization of equation (7.19) is equivalent to the minimization of (7.16) since the additional term is identically zero for arbitrary ϕ^i functions.

Taking the first variation of the above equation we can write :

$$\delta E(\mathbf{y},\gamma) = \int_0^T (\frac{\partial f}{\partial y_i}\delta y_i + \frac{\partial f}{\partial \gamma}\delta\gamma)dt -$$
$$\int_0^T \phi^i(\frac{d\delta y_i}{dt} - a_{ij}\delta y_j - (b_{ijk}+b_{ij})y_j\delta y_k - \gamma c_{ij}\delta y_j -$$
$$c_{ij}y_j\delta\gamma - d_i\delta\gamma - 2\gamma f_i\delta\gamma - g_i\frac{d\delta\gamma}{dt})dt \qquad (7.20)$$

which after integration by parts and rearranging leads to the following equation.

$$\delta E(\mathbf{y},\gamma) = \int_0^T (\frac{d\phi^i}{dt} + [a_{ij} + \gamma c_{ij} + (b_{ijk}+b_{ij})y_k]\phi^j + \frac{\partial f}{\partial y_i})\delta y_i dt +$$
$$\int_0^T (g_i\frac{d\phi^i}{dt} + [c_{ij}y_j + d_i + 2\gamma f_i]\phi^i + \frac{\partial f}{\partial \gamma})\delta\gamma dt - [\phi^i\delta y_i]_0^T - [g_i\phi^i\delta\gamma]_0^T \qquad (7.21)$$

Since initial conditions are given on y^i and γ we consider only variations such that $\delta y^i(0) = 0$ and $\delta\gamma(0) = 0$. Therefore, if the functions ϕ^i solve the following problem :

$$\frac{d\phi^i}{dt} = A_{ij}\phi^j - \frac{\partial f}{\partial y_i}, \qquad \phi^i(T) = 0 \qquad i = 1\ldots N \qquad (7.22)$$

where $A_{ij} = -[a_{ij} + \gamma c_{ij} + (b_{ijk}+b_{ij})y_k]$, then equation (7.20) reduces to :

$$\delta E(\mathbf{y},\gamma) = \int_0^T ([g_i A_{ij} + c_{ij} y_j + d_i + 2\gamma f_i]\phi^i + \frac{\partial f}{\partial \gamma} - g_i \frac{\partial f}{\partial y_i})\delta\gamma dt \tag{7.23}$$

Note that the problem in equation (7.22) is a linear initial value problem that needs to be solved in reverse time for the N co-state functions $\phi^i(t)$. Once these are known, it is straightforward to evaluate δE for an arbitrary variation in the control $\delta\gamma$ using expression (7.23).

7.4.2 The Discrete Problem

In the discrete case, integrals are replaced by summations and the unknown control function $\gamma(t)$ is parameterized using an expansion of the form :

$$\gamma(t) \approx \gamma(0)(1 - \frac{t}{T}) + \sum_{m=1}^{N_d} \beta_m \sin(\frac{m\pi t}{2T}) \tag{7.24}$$

The vector $\boldsymbol{\beta}^k$ containing the N_d *design* variables β_m^k can therefore be updated using the discrete analogue of equation (7.18)

$$\boldsymbol{\beta}^{k+1} = \boldsymbol{\beta}^k - \alpha_k \boldsymbol{G}^k \tag{7.25}$$

and the components G_m^k of the discrete gradient vector $\boldsymbol{G}^k$ are computed as :

$$G_m^k = \Delta t \sum_{n=0}^{N_t} \{[g_i A_{ji}^{kn} + c_{ji} y_j^{kn} + d_i + 2\gamma^{kn} f_i]\phi_i^{kn} + (\frac{\partial f}{\partial \gamma})^{kn} - g_i(\frac{\partial f}{\partial y_i})^{kn}) \sin(\frac{m\pi n \Delta t}{2T})\} \tag{7.26}$$

where the superscript n denotes time level and $N_t = T/\Delta t$.

7.5 Examples on Control of the ODE System

To illustrate the above procedure some examples are presented. The cost functions are defined over a time interval T of 30 time units. This corresponds roughly to five

vortex shedding periods. Equations (7.11) and (7.22) are integrated using a four-step Runge-Kutta timestepping procedure, using 300 timesteps. Note that, this timestep is ten times larger than the one used for the Navier-Stokes simulation.

We consider a cost function made up of the squared sum of the amplitudes of the first two most energetic modes:

$$E(\mathbf{y}, \gamma) = \int_0^T (y_1^2 + y_2^2) dt \tag{7.27}$$

Since the the mode amplitude represents a departure from the mean flow one would expect that the control that minimizes the above cost function would tend to produce a flow which resembles the mean flow. Three different cases with $N_d = 20, 40$ and 60 design variables are considered and for each case 20 steepest descent steps were performed in the minimization process. In all cases the iteration process was started assuming a zero initial control. For each descent iteration the value of α_k was chosen using a line search algorithm. Figure (7-9) shows the the normalized cost function versus the iteration number for the two cases.

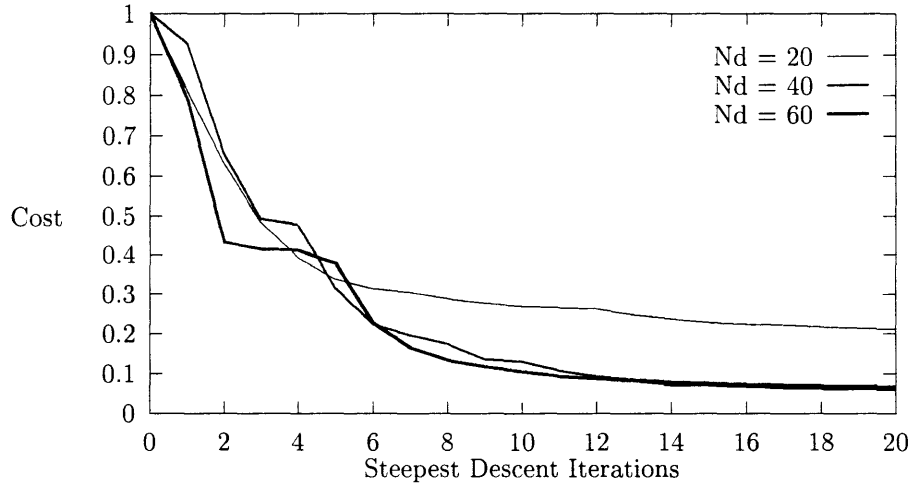

Figure 7-9: Convergence History of Iterative Process

As expected the final value of the cost function decreases for higher values of N_d, although only minor differences are observed between $N_d = 40$ and $N_d = 60$. All

cases appear to be practically converged. Figure (7-10) shows the control functions obtained in each case. There are still considerable differences between the control functions obtained for $N_d = 40$ and $N_d = 60$. Finally figure (7-11) show the first two mode amplitudes for the controlled case with $N_d = 60$. The effect of the control on the first two modes is readily apparent.

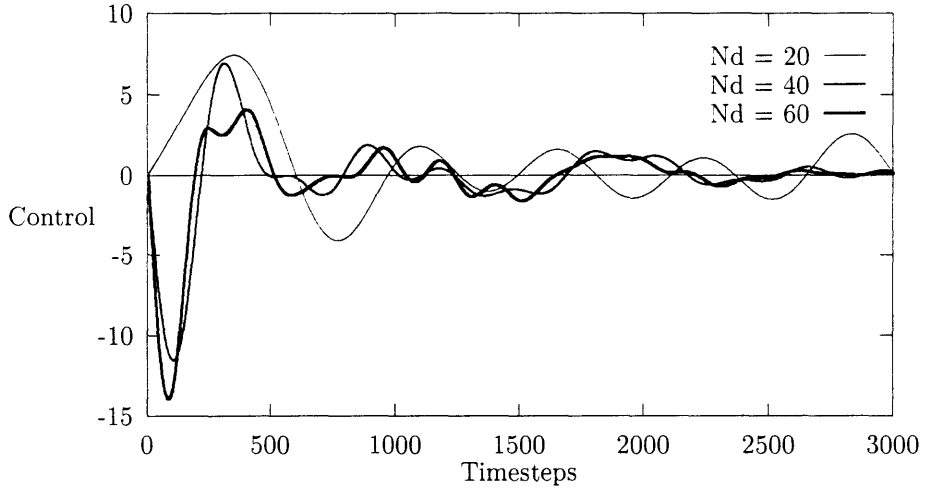

Figure 7-10: Optimum Control for $N_d = 20$. 40 and 60

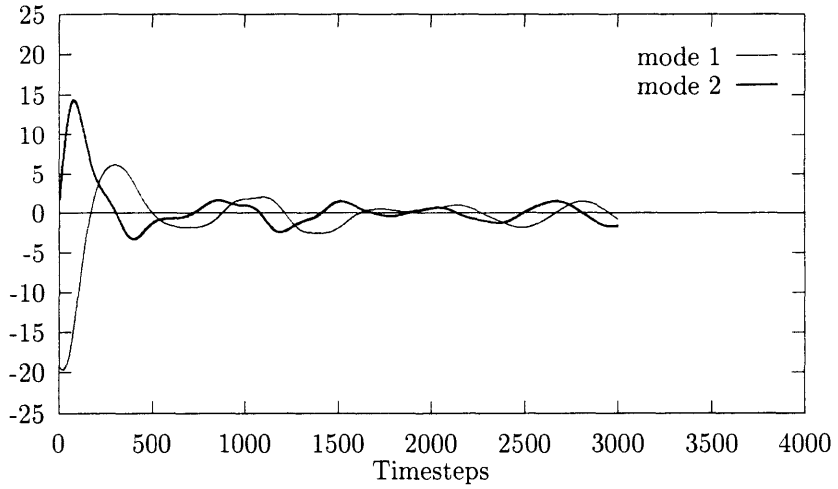

Figure 7-11: Time Evolution for Modes 1 and 2 for $N_d = 60$

7.6 Robustness of the Control Strategies

In order to demonstrate the proposed approach, the control derived from the reduced order model has been applied to a full Navier-Stokes simulation. In the example presented here the control function is re-evaluated every 12 timesteps of the Navier-Stokes computation. For practical purposes the maximum angular speed of the cylinder predicted by the controller has been truncated to one. This control $\gamma(t)$ is shown in Figure (7-12). Figure (7-13) shows the amplitude of the six modes of the controlled Navier-Stokes flow compared to the uncontrolled case. Notice that by using our proposed optimal control strategies, the amplitude of the first two modes which carry most of the energy of the flow, decrease to less than half of theirs original amplitudes and hence the vortex shedding is being suppressed by the unsteady rotations of the cylinder.

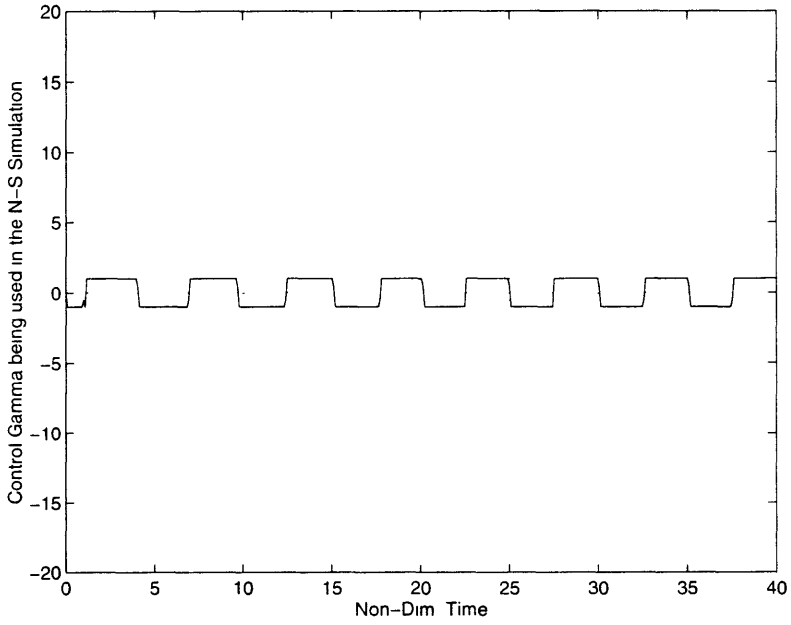

Figure 7-12: Control on the Full Navier-Stokes Simulation

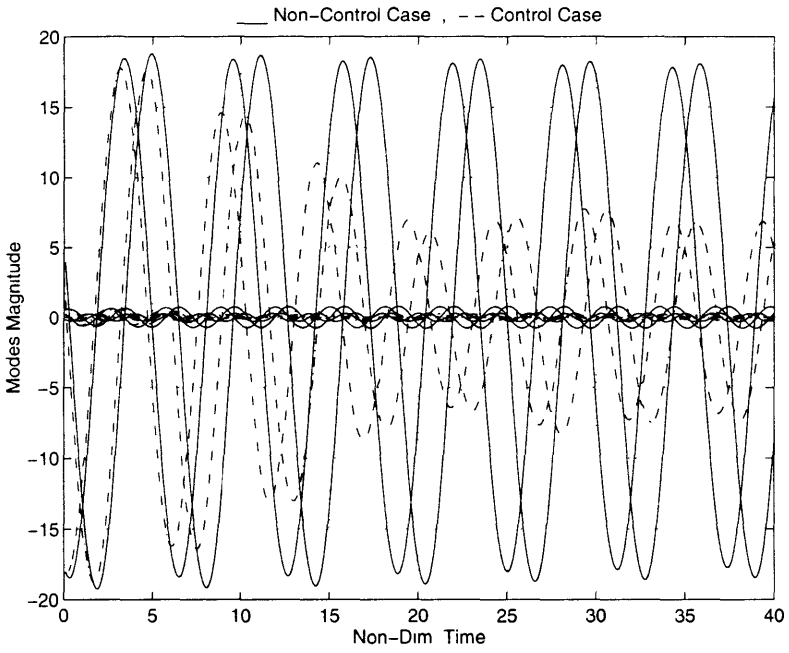

Figure 7-13: Comparision of the Controlled and Uncontrolled N-S Simulation

Chapter 8

Conclusions and Future Development

A two-dimensional, incompressible, viscous flow solver has been developed. The governing equations are written in a Cartesian coordinate system, using primitive variables. The arbitrary Langangian-Eulerian formulation of the Navier-Stokes equations is also considered for the case where mesh movement is applied. The continuous Navier-Stokes equations were discretized by using a fractional step finite element method, employing piecewise-linear interpolation functions for both the velocity and pressure components.

The fully coupled and implicit FEM discretization of the equations leads to a non-linear system of algebraic equations in the nodal velocities and pressures. Since the matrices are very sparse, an iterative method (Preconditioned Conjugate Gradient) is used. In order to minimize the time for solving the matrix equations, a reversed Cuthill-McKee graph reordering technique is performed on the matrices so that the LU preconditioner could be found more accurately and more efficiently.

Unstructured triangular grids which are generated by an advancing front method are used. The main advantage of using unstructured grids is that complex geometries can be generated quite easily. Also, the dynamics movement of the grids can be done very cheaply.

The results for the test cases were shown and their accuracy are very comparable

to the results obtained by other authors in the literatures. As a result, we conclude that our discretization technique is reliable and thus the solutions obtained from our solver can be confidently used as the basis of the development of the proper orthogonal decomposition techniques required for the control of the flow fields.

The second main thrust of this thesis research is concerned with controlling vortex production and stabilizing wake development. The theoretical methods reported in this thesis are also appropriate for many other flow control problems. Modifications of existing numerical algorithm needed for other types of control problems depend on performance and design constraints. More investigations on the control strategies and methods will be performed in the future. For example, by actively controlling the pitch and heave motion of an airfoil in the wake generated by a cylinder, more thrust could be produced or the wake could be stabilized by using active control.

It is our hope that this thesis research will represent a forward step toward the development of computational algorithms for the control of unsteady flow and it will be useful in some real flow problems in the near future.

Bibliography

[1] Ploeg A.V.D. Preconditioning techniques for large sparse, non-symmetric matrices with arbitrary sparsity patterns. In *Iterative methods in linear algebra.* Elsevier Science Publishers B.V., 1992.

[2] Mittal S. Behr M., Hastreiter D. and Tezduyar T.E. Incompressible flow past a circular cylinder: dependence of the computed flow field on the location of the lateral boundaries. *Computer Meth. in Appl. Mech. and Eng.*, 123:309–316, 1995.

[3] Martinelli L. Belov A. and Jameson A. A new implicit algorithm with multigrid for unsteady incompressible flow calculations. AIAA-95-0049, 1995.

[4] Holmes P. Berkooz G. and Lumley J.L. The proper orthogonal decomposition in the analysis of turbulent flows. *Annu. Rec. Fluid Mech.*, 25:539–75, 1993.

[5] Williamson C.H.K. and Roshko A. Measurements of base pressure in the wake of a cylinder at low Reynolds numbers. *Weltraumforsch*, 14:38–46, 1990.

[6] Karniadakis G.E. Deane A.E., Kevrekidis I.G. and Orszag S.A. Low-dimensional models for complex geometry flows : Application to grooved channels and circular cylinders. *Phys. Fluids A*, 3(10):2337–54, October 1991.

[7] Strang G. *Linear Algebra.* Wellesley-Cambridge Press, Wellesley, Ma, 1990.

[8] Kadioglu Z. Glezer A. and Pearlstein A.J. Development of an extended proper orthogonal decomposition and its application to a time periodically forced plane mixing layer. *Phys. Fluids A*, 1(8):1363–73, August 1989.

[9] Triantafyllou G.S. Gopalkrishnan R., Triantafyllou M.S. and Barrett D. Active vorticity control in a shear flow using a flapping foil. *J. Fluid Mech.*, February 1994.

[10] Batina J. Unsteady Euler algorithm with unstructured dynamic mesh for complex-aircraft aeroelastic analysis . AIAA-89-1189, 1989.

[11] Donea J. Arbitrary Lagrangian-Eulerian finite element methods. In *Computational Methods for Transient Analysis.* Elsevier Science Publishers, 1983.

[12] Donea J. A Taylor-Galerkin method for convective transport problems. *Inter. Jour. for Num. Methods in ENG*, 20:101–119, 1984.

[13] Lions J.L. *Optimal control of systems governed by partial differential equations.* Springer-Verlag, New York, NY, 1971.

[14] Shewchuk J.R. An introduction to the conjugate gradient method without the agonizing pain. School of computer science, Carnegie Mellon University, August 1994.

[15] Sirovich L. Turbulence and the dynamics of coherent structures; Part I: Coherent Structures. *Quarterly of Appl. Math.*, XLV(3):561–571, October 1987.

[16] Sirovich L. Turbulence and the dynamics of coherent structures; Part II: Symmetries and transformations. *Quarterly of Appl. Math.*, XLV(3):573–582, October 1987.

[17] Sirovich L. Turbulence and the dynamics of coherent structures; Part III: Dynamics and scaling . *Quarterly of Appl. Math.*, XLV(3):583–590, October 1987.

[18] Hammache M. and Gharib M. An experimental study of the parallel and oblique vortex shedding from circular cylinders. *J. Fluid Mech.*, 232:567–590, 1991.

[19] Lesoinne M. and Farhat C. Geometric conservation laws for aeroelastic computations using unstructured dynamic meshes. AIAA-95-1709-CP, 1995.

[20] Peraire J. Morgan K. and Peiro J. Unstructured grid methods for compressible flows. *Agard*, 787:5.1–5.39, 1991.

[21] Pironneau O. *Optimal shape design for elliptic systems.* Springer-Verlag, NewYork, NY, 1984.

[22] Zienkiewicz O.C. and Morgan K. *Finite Elements and Approximation.* Wiley-Interscience Publication, New York, NY, 1982.

[23] Moin P. and Bewley T. Feedback control of turbulence. *Appl. Mech. Rev.*, 47(6):S3–S13, June 1994.

[24] Volavani L. Paduano J.D., Epstein A.H. Active control of rotating stall in a low-speed axial compressor. *Journal of Turbomachinery*, 115(1):48–56, January 1993.

[25] Thomas P.D. and Lombard C.K. Geometric conservation law and its application to flow compuatations on moving grids . *AIAA Journal*, 17:1030–37, 1979.

[26] Karlsson S.K.F. Rajaee M. and Sirovich L. Low-Dimensional description of free-shear-flow coherent structures and their dynamical behaviour. *J. Fluid Mech.*, 258:1–29, 1994.

[27] Henderson R.D. *Unstructured spectral element methods: Parallel algorithms and simulations.* PhD thesis, MAE Dept., Princeton, June 1994.

[28] Pissanetzky S. *Sparse matrix technology.* Academic press, New York, NY, 1984.

[29] Jacobson S.A. and Reynolds W.C. Active control of boundary layer wall shear shear stress using self-learning neural networks. AIAA-93-3272, 1993.

[30] Triantafyllou G.S. Triantafyllou M.S. and Gopalkrishnan R. Wake mechanics for thrust generation in oscillating foils. *Phys. Fluids A*, 3(12):2835–37, December 1991.

[31] Triantafyllou G.S. Triantafyllou M.S. and Gopalkrishnan R. Optimal thrust development in oscillating foils with application t o fish propulsion. *Journal of Fluids and Structures*, 7:205–224, 1993.

[32] Burns J.A.and Ou Y.R. Active control of vortex shedding. AIAA-94-0182, 1994.