The primary research objective of this contract was to investigate and develop fluid flow control procedures by utilizing a hierarchical modeling approach. This project included research on control and identification methods for vortex wakes, with the primary example being stabilization of vortices behind a flat plate. Vortex blob and finite difference methods were constructed to provide the flow dynamics. Linear time-invariant feedback controllers have been developed, as well as identification results for a class of input/output models that can be used to design more sophisticated controllers. Additionally, Java based software components that support hierarchical modeling efforts were created.
DTIC-OCP
8725 John J Kingman Road, Suite 0944
Fort Belvoir VA 22060-6218

Ladies and Gentlemen:

Enclosed please find two copies of the final technical reports for the following AFOSR grants:

F49620-96-1-0327
F49620-97-1-0132.

These reports have been approved by the AFOSR technical monitor, Dr. Marc Q. Jacobs, at AFOSR/NA.

Sincerely,

J.S. Gibson
Professor of Engineering
and Applied Science

cc: AFOSR/PKA
UCLA - Office of Contract & Grant Admin.

JLS:cg
Hierarchical Modeling and Simulation Techniques with Application to Computational Fluid Dynamics and Fluid-Flow Control
AFOSR F-49620-96-1-0327

Final Report

C.R. Anderson
Department of Mathematics
University of California, Los Angeles
90095-1555

J.S. Gibson
Mechanical and Aerospace Engineering
University of California, Los Angeles
90095-1597

Executive Summary
The primary research objective of this contract was to investigate and develop fluid flow control procedures by utilizing a hierarchical modeling approach. Constituent tasks were

- The creation of a hierarchy of mathematical models (and their computational implementation) for the simulation of vortex shedding phenomena.
- Development of adaptive identification and control procedures appropriate for problems involving vortex shedding.
- Development and implementation of a software infrastructure suitable for hierarchical modeling efforts.

This project included research on control and identification methods for vortex wakes, with the primary example being stabilization of vortices behind a flat plate. Vortex blob and finite difference methods were constructed to provide the flow dynamics. Linear time-invariant feedback controllers have been developed, as well as identification results for a class of input/output models that can be used to design more sophisticated controllers. Additionally, Java based software components that support hierarchical modeling efforts were created. The results of this research have been presented in publications and reports [1] - [14].

Personnel Supported
Faculty : Prof. J.S. Gibson, Prof. C.R. Anderson
Post-Doc : Yen-Cheng Chen
Graduate Students : Rachel Caïden
Other : Mark Hoefer, Tyler Quoc Dinh (UCLA undergraduates).
Results Summaries

Hierarchical Modeling

The flow dynamics of our target control problem were modeled using a vortex blob method. In this method the evolution of the vorticity in the wake is approximated by evolving a collection of discrete vortices. The vortex shedding phenomenon (the principal viscous effect) is modeled by employing a dynamic Kutta condition. As discussed in a number of previous papers [15, 16, 18, 19] vortex methods provide a description of the dynamics as a model with far fewer dimensions than direct numerical simulation. This reduced dimensionality makes vortex methods particularly attractive as simulations to supply approximate dynamics for the task of developing identification and control procedures. However, in order to make the vortex model suitable for identification and control, it was necessary to modify the standard discrete vortex method. Modifications discussed in [1, 3] include an adaptive time-stepping scheme, vortex merging, and a procedure for reducing the noise in velocity measurements near the body.

Another part of the hierarchical modeling effort consisted of constructing a finite difference method that computed the two-dimensional flow about a flat plate (this method is used to validate the vortex model). A major difficulty in creating this method was the proper treatment of the infinite extent of the domain. Previously this problem had been dealt with by employing a finite difference version of the panel method — i.e. one uses velocity field basis elements that are appropriate for an infinite domain. In such a technique one must solve a linear system of equations that is a discrete version of a first-kind integral equation. This procedure has the unsatisfactory property that as the mesh is refined the solution of the linear system becomes more and more difficult to obtain — effectively limiting the resolution that can be employed. In [13] an infinite domain “projection method” was developed that overcomes this problem, thus allowing us to compute solutions of the Navier-Stokes equation in infinite domains without compromising accuracy. Sample results of this calculation are presented in Figure 1(a)-(b).

In addition, work on finite difference methods that simulate the motion of a fluid with large density ratios (such as that occurring in the combustion of fuel droplets) was carried out. In the procedure developed, the fluid was treated as a mixture of compressible and incompressible fluids with numerical methods appropriate for each type of fluid being employed. By utilizing a compressible/incompressible approach,
problems that occur when treating the fluid as a single compressible fluid were avoided (e.g. the severe timestep restriction). Aspects of the work included the development of appropriate boundary conditions at the compressible/incompressible interface and extension of level set method techniques to the problem of evolving a compressible/incompressible interface. Details of the procedure are described in [14].

Research on Flow Control

Our research on control and identification methods for vortex wakes are presented in [1, 2, 3]. This work concerns the stabilization of vortices behind a flat plate, using backside suction as an actuator. The flow dynamics are modeled with a discrete vortex method. Feedback control results for a linear PI controller are presented as well as identification results for a class of input/output models that can be used to design more sophisticated controllers.

In the case of constant free-stream velocity, vortex shedding occurs and a vortex wake forms behind the plate. Our results demonstrate the feasibility of using feedback control in this problem by applying a constant-gain linear feedback to trap the vortices and inhibit shedding when there is no disturbance to the flow field. Most real flow-control applications involve time-varying free-stream velocities and unmodeled disturbances, and these will require more sophisticated controllers. Hence, our research also has investigated the identification of input/output models that can be used to design such controllers.

Our results illustrate that with appropriate modifications, discrete-vortex models can be quite useful for control design and simulation. In particular, the agreement found between the discrete-vortex model and the identified input/output model with respect to the minimum phase and nonminimum phase characteristics of the flow field indicates that the identified input/output models obtained from the discrete-vortex model capture the characteristics of the flow that are important in control system design.

The uncontrolled flow past the plate is shown in Figure 2(a) and is characterized by the formation of vortex street. The vortex shedding causes unsteady drag forces to be exerted upon the plate. In an effort to reduce these unsteady forces, the control problem becomes one of trying to trap the vortices behind the plate and thus inhibit the vortex shedding process. Suction on the centerline of the downstream side of the plate is used as an actuator.

The possibility that the vortices can be trapped and vortex shedding inhibited is demonstrated in Figure 2(b), where we show the results of applying an open-loop constant control. In general, the amount of suction necessary to trap the vortices when there is non-constant free stream velocity and/or unmodeled disturbances is not known, therefore closed-loop control strategies capable of determining the appropriate suction are desired.

For design of adaptive and other sophisticated controllers, an input/output model must be identified. The full dynamics of the flow are too nonlinear to be represented by a single linear input/output model, so we
chose to identify a linear input/output model of the vortex dynamics for perturbations from a nominal steady-state flow. The identified discrete-time input/output model has the form of the ARX (auto-regressive with exogenous input) model. The measurements used for system identification are point velocity measurements on the downstream side of the plate.

Typical results of the system identification are shown in Figure 3. Amplitude and phase responses corresponding to single-frequency inputs are shown by asterisks. The amplitude matches very well, and the phase matches very well at low frequency. At frequencies around half the Nyquist frequency, the phase shows significant errors. However, because the amplitude is very small in this frequency range, these phase errors do not represent a significant difference between the frequency response of the vortex simulation and the frequency response of the ARX model.

Software Infrastructure

A primary research objective of this contract was to investigate and develop fluid flow control procedures by utilizing a hierarchical modeling approach. To achieve this goal required that we construct simulations that were combinations of control codes, fluid simulation codes, and visualization/input-output codes. Many of the codes were written in different languages and were functional on different types of machines. (e.g. Unix workstations and PC's). The problem was to figure out a way that simulations could be created without rewriting component codes and without having to port all the codes to a single computational platform.

Our solution to this problem consisted of developing a software infrastructure that supported the creation of distributed applications; e.g. applications in which individual components exist on separate machines and communicate over a network. This solution avoids the cost of rewriting codes, since the construction of the distributed application only requires that the component codes be "wrapped" with code that enables them to be managed by a distributed application server. In addition, the component codes can remain on the machine that they were developed thus avoiding the cost of porting codes to a single platform.

Our software infrastructure was constructed using the Java programming language. To create distrib-
uted applications requires software that allows components residing on remote machines to be managed — dynamically loading components, establishing communication links, and controlling execution. The JAVA package cam.netapp was created to provide this capability. A report [4] as well as documentation [5] and source [6] are available on the Web. Additionally, to use this Java infrastructure with C, C++ or Fortran codes requires that one interface Java to these codes. While not particularly difficult, there are technical details that are useful to know, and these are discussed in [7]; also available on the Web.

After some experimentation with this infrastructure, it became apparent that getting component codes into a form that allowed them to be integrated into a distributed application requires a considerable amount of tedious programming. As described in [8], creating distributed applications should be as easy as creating web pages that have links to remote pages. To reduce the time it takes to create distributed applications, we automated the process of programming the “wrappers” for the component codes. The description of this process is described in [9], with source and documentation available in [10] and [11].

During the contract, C. Anderson was co-organizer of a workshop concentrating on software issues in scientific technical computing. Proceedings of this workshop have been published [12].

Publications and Reports


**Additional References**


