Australian Government
Department of Defence
Defence Science and
Technology Organisation

An Evaluation of the FIDAP Computational Fluid Dynamics Code for the Calculation of Hydrodynamic Forces on Underwater Platforms

D.A. Jones and D.B. Clarke

Maritime Platforms Division
Platforms Sciences Laboratory

DSTO-TR-1494

ABSTRACT

Maritime Platforms Division within DSTO is currently studying the science and technology of autonomous underwater vehicles for defence applications. Part of this work involves a study of the hydrodynamics and manoeuvrability of these vehicles and the development of methods to determine the hydrodynamic coefficients of submerged bodies as a function of their shape. This report describes the application of the FIDAP Computational Fluid Dynamics package to the calculation of lift and drag forces on relatively simple underwater vehicle shapes including cylinders, spheres, flat plates and wing profiles. The degree to which FIDAP accurately reproduces known experimental data on these shapes is described and the applicability of other Computational Fluid Dynamics packages is discussed.

Published by

DSTO Platforms Sciences Laboratory
506 Lorimer St
Fishermans Bend, Victoria 3207 Australia

Telephone: (03) 9626 7000
Fax: (03) 9626 7999

AR-012-893
August 2003

APPROVED FOR PUBLIC RELEASE

An Evaluation of the FIDAP Computational Fluid Dynamics Code for the Calculation of Hydrodynamic Forces on Underwater Platforms

Executive Summary

Maritime Platforms Division (MPD) within DSTO is currently studying the science and technology of autonomous underwater vehicles for defence applications. Part of this work involves a study of the hydrodynamics and manoeuvrability of these vehicles and the development of methods to determine the hydrodynamic coefficients of submerged bodies as a function of their shape. These coefficients are specific to the vehicle and provide the description of the hydrodynamic forces and moments acting on the vehicle in its underwater environment.

The calculation of these hydrodynamic coefficients has previously been performed using methods adapted from the aeronautical literature. These methods are satisfactory when applied to axisymmetric underwater vehicle shapes but provide poor estimates of the coefficients when applied to flatfish type bodies such as the DSTO concept demonstrator vehicle Wayamba. There is a distinct need for a more widely applicable method for the calculation of hydrodynamic coefficients for non-generically shaped vehicles, in particular those with non-axisymmetric body shapes. One alternative approach, currently under investigation in MPD, is to combine experimental techniques with current Computational Fluid Dynamics (CFD) capabilities.

The purpose of this report is to document and describe the applicability of the FIDAP (Fluid Dynamics Analysis Package) finite-element CFD code to the calculation of lift and drag forces on relatively simple underwater vehicle shapes such as cylinders, spheres, flat plates, and wing profiles. The results indicate that the FIDAP code has several problems with regard to accurate simulations of turbulent flow around underwater bodies. The main problem appears to be the inability of the turbulence models incorporated in the code to fully capture the relevant physics required to simulate flow separation effects at high Reynolds numbers. The results showed that the models were unable to accurately calculate drag coefficients for two-dimensional cylinders and three-dimensional spheres. In particular, the models were unable to reproduce the strong variation of drag coefficient for each of these shapes in the critical region just after transition from laminar to turbulent flow in the boundary layers.

A further problem with the code was the difficulty in obtaining converged solutions on unstructured hybrid grids containing mixtures of quadrilateral and triangular elements (in two-dimensions), or hexahedral and tetrahedral elements (in three-dimensional flows). Because of these, and other problems, it is recommended that FIDAP should be replaced by a CFD code containing both more advanced turbulence models, as well as the ability to more easily obtain converged solutions on hybrid unstructured meshes. The finite-volume code Fluent, which is used in Air Operations Division for aeronautical applications and in Airframes and Engines Division for the simulation of combustion flows, is the recommended alternative.

Authors

David A. Jones
Maritime Platforms Division

Dr. David A. Jones obtained a B.Sc. (Hons) and Ph.D. in Theoretical Physics from Monash University in 1973 and 1976 respectively. He joined the then Materials Research Laboratories in 1983 after postdoctoral positions at the University of Strathclyde, Glasgow; Queen Mary College, London University, and the University of New South Wales, Sydney. During 1987/88 he was a visiting scientist at the Laboratory for Computational Physics and Fluid Dynamics at the Naval Research Laboratory, Washington, DC. He has authored 80 journal articles and technical reports and given more than 60 presentations at scientific meetings. His research has covered a variety of areas including polymer dynamics, the application of chaos theory to atomic and molecular physics, laser-plasma interaction theory, warhead design, air blast, detonation physics and computational fluid dynamics. He is currently using new Lagrangian vorticity methods to model hydrodynamic flow around underwater vehicles.

David B. Clarke
Maritime Platforms Division

David Clarke commenced work in the Maritime Operations Division (MOD) at AMRL in 1988 after completing a BSc at Sydney University. His work in MOD focused on magnetic sensors and instrumentation. He obtained a Graduate Diploma in computer engineering from RMIT in 1996. In 1998 he joined the Maritime Platforms Division to work on the hydrodynamics of underwater vehicles. His work on underwater vehicles has encompassed experiment, empirical and computational hydrodynamics.

Contents

1. Introduction

Maritime Platforms Division (MPD) is currently studying the use of autonomous underwater vehicles for defence applications. Part of this study involves the development of a capability to model the hydrodynamic coefficients of submerged bodies as a function of their shape. These coefficients are specific to the vehicle and provide the description of the hydrodynamic forces and moments acting on the vehicle in its underwater environment, and hence determine the stability and manoeuvrability of these vehicles.

A recent report by Jones et al. [1] presented a detailed discussion and evaluation of several methods previously used for the calculation of these hydrodynamic coefficients. Sample calculations were presented, and the accuracy and applicability of these methods to the underwater vehicles of interest to the DSTO were described. For axisymmetric bodies they provide a reasonable first estimate of the magnitude of these coefficients and a software suite, known as HYGUESS, has been written based on some of the algorithms described in reference [1]. HYGUESS calculates the linear hydrodynamic coefficients of axisymmetric underwater vehicles, as well as the total (non-linear) lift and drag coefficients for the complete vehicle. The code is written in the C++ programming language and a description of the overall structure of the code, including the table handling software and detailed class information, is contained in the report by Clarke et al. [2].

None of these methods have the necessary generality to model non-axisymmetric shapes of particular interest to MPD however, such as the flatfish type bodies of the autonomous underwater vehicle Marius [3], and the DSTO concept demonstrator vehicle Wayamba [4]. Also, as clearly demonstrated by Peterson [5], the hydrodynamic coefficients predicted by the empirical methods used in software packages such as HYGUESS, and the HYCOF subroutine in Peterson's HYSUB system [6], often display appreciable errors when applied to vehicle shapes displaying only minor physical differences from those on which the empirical methods were developed. There is a distinct need for a more widely applicable method for the calculation of hydrodynamic coefficients for non-generically shaped vehicles, and in particular those with non-axisymmetric body shapes. One alternative approach, currently under investigation in MPD, is to combine experimental techniques with current Computational Fluid Dynamics (CFD) capabilities. In the last two decades both the sophistication of CFD codes and the computing power of standard desk top workstations have increased significantly, and the possibility of using CFD to determine hydrodynamic derivatives is now just becoming feasible [7].

The current program on unmanned underwater vehicles in MPD will use a combination of both experimental and computational techniques to determine hydrodynamic coefficients for underwater vehicles of interest. The experimental facilities at the Australian Maritime Engineering College in Launceston have been used

to measure hydrodynamic coefficients on scale models of selected underwater vehicle shapes. These results will be used to benchmark simulation results for these scale models obtained from calculations using a commercial CFD software package. Once the simulation results have been tuned to the experimental results the CFD code can then be used to perform a parametric study on related underwater vehicle shapes to determine the manoeuvrability characteristics of each of these vehicles.

The purpose of this report is to document and describe the applicability of the FIDAP (Fluid Dynamics Analysis Package) finite-element CFD code to the calculation of lift and drag forces on relatively simple underwater vehicle shapes such as cylinders, spheres, flat plates, and wing profiles. FIDAP is one of several commercially available software packages with the potential capability to perform these calculations [8]. Other CFD codes currently in use within MPD include the finite-volume codes Fluent [9] and CFX [10]. FIDAP was chosen initially because it was the only code which had the capability to model free surface effects with some degree of accuracy, and this aspect of the code was of interest to MPD because of its work on surface platforms. This report discusses in detail the application of the FIDAP code to the calculation of the hydrodynamic flow around the above mentioned simple shapes. A comparison is made between the simulated results and experimental data, and the degree to which FIDAP accurately reproduces the experimental data is discussed. Some shortcomings in the code are noted, particularly with respect to the type of turbulence models available to the user, and an alternative software package is recommended for future use.

2. The FIDAP Code

FIDAP was the first commercial CFD code based on the finite-element method and has been commercially available since 1983. It was developed by Fluid Dynamics International Inc. in the USA and is now owned by Fluent Inc., Lebanon, NH, USA. One advantage of the finite-element method over either the finite-difference or finite-volume methods in their early stages of development was that it allowed any model vehicle within the flow to be arbitrarily complex in shape and connectivity. In recent years however new unstructured grid methods have allowed both finite-difference and finite-volume codes to deal with complex geometrical domains as readily as finite-element schemes, so that finite-element schemes no longer have an inherent advantage in this area. FIDAP solves the equations of mass, momentum and energy conservation for both compressible and incompressible viscous fluids for both laminar and turbulent flows and offers the user a multitude of choices with regard to both problem set-up and solution procedure. One of the reasons for briefly reviewing the code and its capabilities here is to point out which of the many model options and solution strategies were found to be appropriate for obtaining accurate solutions for flows around typical underwater shapes at realistic Reynolds number values, which for the flows considered here are of the order of 1.0×10^6.

2.1 Basic Fluid Equations

The basic equations governing the flow of an isothermal viscous fluid are derived from the conservation of mass and momentum and are described in many standard texts [11]. The mathematical representation of the equation for the conservation of mass is known as the continuity equation and has the form

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{1}$$

In equation (1) **u** represents the flow velocity, ρ the density, and t the time. If the fluid is incompressible (which is assumed here) then the density is constant and equation (1) has the simpler form

$$\nabla \cdot \mathbf{u} = \frac{\partial u_i}{\partial x_i} = 0 \tag{2}$$

where u_i and x_i represent the components of the velocity and position vectors, and the summation convention for indices has been used.

The equation representing the conservation of linear momentum has the following form

$$\rho\left(\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j}\right) = -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} \tag{3}$$

where τ_{ij} is the viscous stress tensor and p is the dynamic pressure. For a Newtonian fluid (which is also assumed here) the viscous stress tensor can be written in the form

$$\tau_{ij} = \mu\left(\frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j}\right) \tag{4}$$

where μ is known as the coefficient of viscosity of the fluid. Equation (3) can then be written

$$\rho\left(\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j}\right) = -\frac{\partial p}{\partial x_i} + \mu \frac{\partial}{\partial x_j}\left(\frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j}\right) \tag{5}$$

which is known as the Navier-Stokes equation.

2.2 Finite-Element Method of Solution

Equations (2) and (5) form a set of partial differential equations for the variables p and u_i over a continuous domain representing an infinite number of degrees of freedom. To solve these equations computationally they first need to be transformed into a discrete domain representing a finite number of degrees of freedom. In the Finite-Element Method this is done by first dividing the continuum region of interest into a number of simply shaped regions called elements. FIDAP allows both quadrilateral and triangular elements for two-dimensional (2D) problems, and brick (hexahedral), tetrahedron, and wedge (triangular prism) elements for three-dimensional (3D) problems. Within each of these elements the dependent variables u_i and p are approximated by interpolating functions (also known as basis functions or shape functions), so that the problem reduces to finding the unknown values of the variables at the node points of the elements.

In FIDAP the method used to transform the continuum equations into a finite set of equations for these unknown nodal values is the Galerkin form of the Method of Weighted Residuals [12]. This minimizes the errors (the residuals) in equations (2) and (5) in a weighted sense by requiring that the residuals be orthogonal to the interpolation functions within each element. The orthogonality conditions are expressed by forming integrals of the assumed solution with the interpolation functions for each element, and this results in a set of matrix equations for the unknown values at the nodal points.

FIDAP allows the user considerable choice in the combination of basis functions, element types, and pressure approximations. For example, the two-dimensional quadrilateral element can be either a 4-node linear element or an 8 or 9-node quadratic element. If the 4-node linear element is chosen then the velocity components u_i are approximated using bilinear interpolation functions, but two pressure approximations are possible; a bilinear continuous approximation with the pressure degrees of freedom located at the four nodes, or a piecewise constant discontinuous pressure approximation with the pressure degree of freedom associated with the pressure centroid. If a 9-node quadrilateral element is chosen then the velocity is approximated using biquadratic interpolation functions, and three different forms of the basis functions for pressure are available. If the domain is meshed using triangular elements then the user has a choice of either a 3-node triangle using linear basis functions for the velocity and two different types of pressure discretization, or 6 or 7-node triangle elements using biquadratic basis functions for the velocities and either linear or quadratic basis functions for the pressure.

In three dimensions there are also a large number of choices. Brick elements can be either 8-node linear elements or 27-node quadratic elements, with a choice of two possible pressure discretizations for the 8-node element, or three possible pressure discretizations for the 27-node element. Tetrahedron elements are either 4-node linear elements or 10 node quadratic elements, with different pressure approximations

similar to those for the corresponding triangles. Wedge elements are available as either 6-node linear elements or 18-node quadratic elements, and two pressure discretizations are possible with each of these elements.

The element type (eg. quadratic or triangle in 2D) and element order (eg. 4-node, 8-node or 9-node for a quadratic element) are set in the GAMBIT preprocessor software when the initial meshing of the problem is performed. The choice of element order fixes the order of the basis functions for the velocity (linear or quadratic), but the specific pressure discretisation method used with the element is specified in the PRESSURE command in the FIDAP software.

FIDAP was originally written to use quadrilateral elements for 2-D problems and hexahedral elements for 3-D problems. The use of these element shapes results in structured grids and gives the user much greater control over the quality of the entire finite element mesh. They are particularly easy and appropriate to use when the objects to be meshed have relatively simple geometries. For more complicated geometries the use of triangular and tetrahedral elements often allows the initial meshing of the problem to be considerably simplified. These elements are also particularly useful when the boundaries are required to be placed at a considerable distance from any object located in the flow. The unstructured nature of the resulting grid allows the problem to be meshed with considerably fewer elements than would be the case if a structured grid using hexahedral elements was employed.

FIDAP has previously been used by Givler et al. [13] for three-dimensional modelling of flow past a prolate spheroid and two model submarine hulls, and by Williams et al. [14] and Hajiloo et al. [15] to model flow around car-like shapes. In each of these applications the finite-element mesh was generated using hexahedral elements and the computed results, such as the value of the drag coefficient and wake velocities, compared favourably with experimental values. Equivalent benchmark calculations using triangular and tetrahedral elements do not appear in the literature however. One of the objectives of this work is to investigate the ease of use, and the resulting simulation accuracy, when FIDAP is run using these element shapes.

2.3 Turbulence Models

The dimensions and velocities of typical underwater vehicles are such that the Reynold's number will normally be greater than 1.0×10^6, indicating that there will be considerable turbulence in the flow field. In turbulent flow the fluid motion is highly random, unsteady, and three-dimensional. In principle such flows can be described by direct application of the time-dependent equations described in the preceding section, an approach which is known as Direct Numerical Simulation (DNS) of turbulence. This approach is impractical in most situations however because turbulent flows contain information on scales which are many orders of magnitude smaller than the extent of the flow domain. To resolve these scales requires very fine meshing, which in turn

requires very small time steps. Calculations of this type for realistic underwater vehicles are currently beyond the capabilities of current computers.

To overcome these problems and create a useable numerical model of a turbulent flow field FIDAP describes the turbulent motion by separating each variable into a mean component and a fluctuating component, and then averaging the flow equations over a time scale which is long compared to that of the fluctuating components. This is a standard approach in turbulence modelling [16]. Each of the field variables is therefore written in the form

$$\eta = \overline{\eta} + \eta' \tag{6}$$

where $\overline{\eta}$ and η' represent the mean and fluctuating components respectively. If this approach is applied to the variables in equations (2) and (5) and the equations are then time averaged the following equations for the mean components are obtained

$$\frac{\partial \overline{u}_i}{\partial x_i} = 0 \tag{7}$$

$$\rho\left(\frac{\partial \overline{u}_i}{\partial t} + \overline{u}_j \frac{\partial \overline{u}_i}{\partial x_j}\right) = -\frac{\partial p}{\partial x_i} + \mu \frac{\partial}{\partial x_j}\left(\frac{\partial \overline{u}_j}{\partial x_i} + \frac{\partial \overline{u}_i}{\partial x_j}\right) - \rho \frac{\partial}{\partial x_j}(\overline{u_i' u_j'}) \tag{8}$$

Equation (7) has the same form as equation (2), only now the components of the fluid velocity $\overline{u}_i$ represent the average components of the velocity. Equation (8) is also almost identical to equation (5), except for the presence of the new term

$$-\rho \frac{\partial}{\partial x_j}(\overline{u_i' u_j'}),$$

which represents the correlation between velocity fluctuations in the turbulent flow field. Physically, these correlations represent the net transport of momentum due to the fluctuations. The expression $-\rho\overline{u_i' u_j'}$ represents the i^{th} component of momentum in the j^{th} direction (or vice-versa) and is known as the Reynolds stress tensor. To solve equations (7) and (8) the Reynolds stress tensor must be expressed in terms of the mean-flow quantities. This is known as a constitutive relationship, and FIDAP allows the user to choose from one of three constitutive relations; Boussinesq, Speziale, and Launder. The simplest of these is the Boussinesq relation, which assumes that the turbulence is isotropic. The Speziale and Launder relations are extensions of the Boussinesq relation which are designed to model anisotropic turbulent flows.
The Boussinesq relation assumes that the components of the Reynolds stress tensor are proportional to the mean velocity gradients, ie.

$$-\rho\overline{u_i'u_j'} = \mu_t\left(\frac{\partial\overline{u}_i}{\partial x_j}+\frac{\partial\overline{u}_j}{\partial x_i}\right)-\frac{2}{3}\rho k\delta_{ij} \tag{9}$$

In the above equation δ_{ij} is the Kroecker Delta, and the parameter μ_t is known as the eddy-viscosity. It depends on the turbulence in the flow field and is therefore a function of position. The parameter *k* is the turbulent kinetic energy per unit mass and is also a function of position. The definition of *k* is

$$k = \frac{1}{2}\overline{u_i'u_i'} \tag{10}$$

Subtraction of the turbulent kinetic energy term on the right hand side of equation (9) is required because the trace (ie. the sum of the diagonal terms) of the first term is zero due to equation (7). Before equation (9) can be used an eddy-viscosity model needs to be defined to provide a means to estimate μ_t, and a transport equation needs to be derived to model the turbulent kinetic energy *k*.

FIDAP provides seven eddy-viscosity models. Two of these are zero equation models, in which algebraic equations are used to describe the relationship between μ_t and the flow variables. The particular algebraic model used by FIDAP is a mixing length model, and the user has the choice of an automatic mixing length computation or one calculated from a user supplied subroutine. These models will not be discussed further however as it is well known that algebraic models are quite unreliable for predicting flow separation effects [17].

The remaining turbulence models are known as two-equation models, and these include two extra partial differential equations to describe the relationship between μ_t and the flow variables. The default model employed in FIDAP is a two-equation eddy-viscosity model known as the standard *k*-ε model. In this model the turbulence field is characterised in terms of two variables, the turbulent specific kinetic energy *k*, and the viscous dissipation rate per unit mass of turbulent kinetic energy ε, which is defined as

$$\varepsilon = \left(\frac{\mu}{\rho}\right)\overline{\frac{\partial u_i'}{\partial x_j}\frac{\partial u_i'}{\partial x_j}} \tag{11}$$

In the standard *k*-ε model *k* and ε are obtained from the following semi-empirical coupled transport equation:

$$\rho\frac{\partial k}{\partial t}+\rho\overline{u}_j\frac{\partial k}{\partial x_j} = -\rho\overline{u_i'u_j'}\frac{\partial\overline{u}_i}{\partial x_j}-\rho\varepsilon+\frac{\partial}{\partial x_j}\left[\mu+\frac{\mu_t}{\sigma_k}\frac{\partial k}{\partial x_j}\right] \tag{12}$$

$$\rho\frac{\partial\varepsilon}{\partial t}+\rho\bar{u}_j\frac{\partial\varepsilon}{\partial x_j}=-c_1\frac{\varepsilon}{k}\rho\overline{u_i'u_j'}\frac{\partial\bar{u}_i}{\partial x_j}-c_2\rho\frac{\varepsilon^2}{k}+\frac{\partial}{\partial x_j}\left[\mu+\frac{\mu_t}{\sigma_\varepsilon}\frac{\partial k}{\partial x_j}\right] \quad (13)$$

Here c_1 and c_2 are empirical constants and σ_k and σ_ε are the turbulent Prandtl/Schmidt numbers for k and ε respectively. The relationship between k and ε has the form

$$\mu_t=c_\mu\rho\frac{k^2}{\varepsilon} \quad (14)$$

where c_μ is an empirical constant with a value of 0.09.

The k-ε model was developed by Jones and Launder [18], and is described in some detail by Wilcox [17]. The model has been widely tested for a variety of flows. For isothermal flows with no mass transfer the recommended values of the various constants are as follows:

$$\sigma_k = 1.00,\ \sigma_\varepsilon = 1.30,\ c_1 = 1.44,\ c_2 = 1.92,\ c_\mu = 0.09$$

These are the default values used in FIDAP, and are also the values which have been used in the applications of the k-ε model described in this report. The advantages of the standard k-ε model are that it is robust, economical, and reasonably accurate for moderately difficult problems when used in conjunction with the Boussinesq eddy-viscosity constitutive relation. The disadvantage, however, is that it performs badly for complex flows involving severe pressure gradients or strong streamline curvature.

FIDAP also contains several improved versions of the standard k-ε model, these being the RNG k-ε model, the Extended k-ε model, and the Anisotropic k-ε model. Each of these models offers some degree of improvement for modelling flows involving pressure gradients and strong streamline curvature. The RNG model was developed by Yakhot et al. [19] and is based on renormalised group theory arguments. It contains an additional source/sink term in the ε equation and takes better account of the physics. McKenzie has used the RNG k-ε model in the Fluent CFD code to model flow around submarine shapes and has noted that whilst the standard k-ε model performs badly in this application, in that it is unable to calculate the position of flow separation points accurately, the RNG k-ε model gave quite acceptable results [20].

The Extended k-ε model used in FIDAP was developed by Chen and Kim [21]. It is based on the assumption of a second time scale which beomes important when the turbulence deviates from local equilibrium. This results in a modified equation for ε and an additional empirical constant which must be fitted to experimental results. The

extended k-ε model yields results similar to those of the standard model for simple flows, but is claimed to yield better predictions for complex flows involving recirculation and streamline curvature.

The Anisotropic k-ε model is a simple extension of the standard model which allows terms in the ε equation to relate to the anisotropic nature of the turbulence field. The form of the k and ε equations for the anisotropic model are identical to those of the standard k-ε model, but the coefficient c_2 is now a function of the second and third invariants of the anisotropy tensor. This approach allows improved predictions of anisotropy effects which are not strictly modelled by two-equation models. These effects can be significant for difficult flows involving strong streamline curvature.

Both the standard k-ε model and the three variants of the model used in FIDAP are only appropriate for high Reynolds number flows and hence cannot be used in the near-wall boundary layer regions where viscous effects dominate. To overcome this problem FIDAP uses specialised WALL elements, which use special velocity basis functions based on universal experimental velocity profiles for fully turbulent boundary layers. The WALL elements extend from the surface of the solid wall into the buffer region of the boundary layer, and in this manner account for the rapid variation in flow properties which occur in the lower most part of the boundary layer. Use of the WALL elements means that a very fine mesh is not required close to any solid surfaces in the flow and the high Reynolds number k-ε models are not solved in inappropriate regions of the flow. A disadvantage of this approach is that, because the velocity is forced to have a particular profile close to the solid surface, boundary layer separation is less likely to be modelled accurately because the velocity profile near the separation point will be quite different to that used for the velocity basis functions.

Better predictions of boundary layer separation would be obtained by avoiding the use of the WALL element approach and k-ε models and instead using a more appropriate low Reynolds number turbulence model in conjunction with a very fine mesh through the boundary layer. In this way, more accurate simulation of the velocity profile in the boundary layer would be obtained. One turbulence model available in FIDAP which has these properties is the Wilcox Low Reynolds Number k-ω model. In this model separate transport equations are solved for k and ω (where ω is defined by $\varepsilon = k\omega$) over a fine near-wall mesh to model both the mean flow variables and the turbulence variables near the wall, and to resolve any geometric features which may be present in the viscous sub-layers. While the Wilcox Low Reynolds Number k-ω model would seem to be the best model to use if interest is centred on accurate evaluation of boundary layer separation points, it is unfortunately not appropriate for the applications considered here. It is primarily intended for simulating globally low Reynolds number internal flows where Re < 10,000, and is particularly unsuited for predicting external flows because the results are very sensitive to the free-stream value of ω [22].

The above discussion would seem to indicate that the standard k-ε turbulence model might not be particularly appropriate for the applications considered here. Nevertheless, it is important to note that the model, as implemented in FIDAP, has been used by Givler et al. [13] to model flow past three-dimensional submarine models, and by Williams et al. [14] and Hajiloo et al. [15] to model flow around car-like shapes. Givler et al. [13] used FIDAP with the standard k-ε model to simulate flow around a 6:1 prolate spheroid and two model submarine hulls with an attached fairwater and sternfins. For the prolate spheroid comparisons were made between computed and experimental results and good agreement was obtained for some of the data. For example, computed axial velocity profiles at four locations within the wake were in excellent agreement with the experimental measurements. The computed separation streamline behind the body was poorly predicted however, with the numerical model underpredicting the length of the recirculation eddy by a factor of four. This lead to overprediction of the pressure in the near wake, but excellent agreement was obtained between simulated and experimental pressure values further downstream. Although the separation streamline was poorly predicted, surprisingly, the computed drag coefficient of 0.064 was in excellent agreement with the experimental value of 0.06.
Williams et al. [14] used FIDAP with the standard k-ε turbulence model to simulate the flow around two car-like shapes without performing any grid convergence studies and found that they were able to qualitatively simulate all significant flow structures, as well as obtaining excellent agreement between calculated and measured centreline pressure distributions. The calculated drag coefficients agreed with the measured vales to within 5% and 15% respectively. Hajiloo et al. [15] performed a limited mesh refinement study for one of these car-like shapes, again using FIDAP and the standard k-ε turbulence model, and found that their finest mesh gave a simulated drag coefficient which agreed with the measured value to within 2%.

The success of the standard k-ε turbulence model in calculating reasonably accurate drag coefficients for car-like shapes is not surprising as these bodies have comparatively blunt rear ends, and the location of any separation point is relatively easy to predict. For more streamlined shapes such as 6:1 spheroids, submarines, or flatfish UUVs, it would be expected that the standard k-ε model would have less success, and this was indeed experienced by Givler et al. [13] for the 6:1 spheroid and McKenzie for submarine shapes [20] where, in both cases, the location of the separation points were poorly predicted. For these more streamlined shapes the RNG k-ε model would be expected to provide improved simulation results.

2.4 Boundary and Initial Conditions

The solution of any fluid flow problem in a finite domain requires specification of appropriate boundary and initial conditions. All the problems considered here are external, unconfined flows of turbulent fluid, so the variables of interest are the pressure, the three components of velocity (for a 3D problem), as well as the turbulent kinetic energy k and the dissipation function ε. As well as boundary conditions for each of these variables, initial conditions must also be stipulated. In a transient simulation

these values prescribe the state of the system at the initial time, whereas in steady-state simulations, such as those considered here, these values serve as initial guesses for the iterative solution procedure.

All the simulations described in this report are effectively numerical wind tunnel or cavitation tunnel experiments. There is an inlet boundary, typically on the left side of the grid, through which the undisturbed flow enters the computational domain. The centre of the grid contains the solid body of interest, and there is an outlet boundary on the right side of the grid. The remaining boundaries can either be solid, representing the real walls of either the wind tunnel or the cavitation tunnel, or they can be entrainment boundaries, which allow fluid to enter or leave the computational domain and hence simulate an unbounded region of flow. For computational convenience, and also to minimise memory requirements, there may also be one or more symmetry boundaries along the computational domain.

Appropriate boundary conditions for u_i, k and ε on the inlet boundary are Dirichlet, or prescribed, boundary conditions. Values for u_i are easily defined; the flow is typically in the x direction and so u_x has a finite value, while u_y and u_z are set to zero. The inlet boundary value for k is calculated by first assuming a particular value for the intensity of the turbulence in the free stream. The intensity I is defined as

$$I = \frac{\left(\overline{u_i'^2}\right)^{1/2}}{u_\infty} \tag{15}$$

where u_∞ is the undisturbed free stream velocity. Once a value for I has been decided upon, k is then calculated from

$$k = 1.5\left(Iu_\infty\right)^2 \tag{16}$$

The value for the dissipation function ε is then calculated from equation (14), ie.

$$\varepsilon = \frac{\rho c_\mu k^2}{R_\mu \mu} \tag{17}$$

where R_μ is the ratio of turbulent to laminar viscosity. The value of R_μ depends to a great extent on the nature of the flow and is difficult to specify exactly. For flows of the type considered in this report R_μ typically lies somewhere in a range between 100 and 500 [23]. The exact value of both I and R_μ should not be critical to the calculation. Although the location of the separation point, and hence the drag coefficient, has been found to depend on the level of free stream turbulence this is a small effect, and the nature of the turbulence models employed within FIDAP are not expected to be able to accurately reproduce this dependence. In practice, solutions obtained from FIDAP are

expected to be independent of the assumed values of I and R_μ, provided that these lie within appropriate limits.

The appropriate boundary condition for all variables on a symmetry boundary is that the gradient of the variable in the direction perpendicular to the boundary should be zero. This is also the appropriate condition on an outflow boundary, provided that the boundary is sufficiently removed from any object within the flow. This is always true for all of the simulations reported here. The zero gradient boundary condition is known as a Neumann boundary condition and FIDAP automatically assigns Neumann boundary conditions to all flow variables on computational boundaries which have not been explicitly given Dirichlet boundary conditions. Neumann boundary conditions are also appropriate on entrainment boundaries, provided that the velocity component normal to the boundary plane is small relative to the tangential velocity component. This is typically the case in all the simulations considered here.

The appropriate boundary condition for the velocity at a solid boundary is the no-slip condition, ie. the normal and tangential components of the velocity should be zero on any solid (non-moving) boundary. Implementation of this boundary condition raises several problems however. All viscous fluids form a boundary layer near the surface of a solid wall and the flow variables change very rapidly within this boundary layer. To accurately model this boundary layer would require a very fine mesh with a disproportionately large number of grid points in the immediate vicinity of the solid boundary, and this would significantly reduce the size of the mesh which could be used for the rest of the flowfield. A further complication, as mentioned in the previous section, is that the standard k-ε models used in FIDAP are only valid for high Reynolds number flows and cannot be used in near-wall regions.

FIDAP overcomes these problems by using specialised WALL elements. In this approach, as described in section 2.3, the computational domain is extended to the physical boundary and the problem of resolving the boundary layer is removed by using special WALL elements in the near-wall region between the fully turbulent outer flow field and the physical boundary. These elements use special shape functions to accurately capture the sharp variations of the mean flow variables in the boundary layer. Because the k-ε model is invalid in the boundary layer the k and ε equations are only solved in the region outside the WALL elements, and the spatial variation of the turbulent momentum diffusivity within the boundary layer is modelled using van Driest's mixing length approach [12].

The shape functions in the special elements are constructed using products of one-dimensional basis functions. In the local coordinate direction along the wall these basis functions are identical to those used in the construction of shape functions for the regular elements. In the local coordinate direction normal to the wall however special one-dimensional basis functions are used which are based on universal flow profiles in the near-wall region. These functions depend on the number of nodes in the element along the local coordinate direction which points away from the wall, but each is just a

slight variation of the Reichardt law, which describes the dimensionless flow velocity normal to the wall, u^+, as a function of the dimensionless normal distance from the wall, y^+. Reichardt's law has the form

$$u^+ = \frac{1}{\kappa}\ln\left(1+\kappa y^+\right) + 7.8\left[1-\exp\left\{-\frac{y^+}{11}\right\} - \frac{y^+}{11}\exp\left(-0.33y^+\right)\right] \qquad (18)$$

where

$$u^+ = \frac{\left(u-u_w\right)}{u_*}; \quad y^+ = \frac{\rho u_* \delta}{\mu}; \quad u_* = \left(\frac{\tau_w}{\rho}\right)^{1/2} \qquad (19)$$

τ_w is the shear stress on the wall and κ is an experimentally determined constant having a value of approximately 0.40. In equation (19) the subscript w denotes a wall value, u^* is the friction velocity, and δ is the normal distance from the wall. Equation (18) accurately reproduces known experimental data for the velocity profile within the turbulent boundary layer along a flat plate. For values of y^+ less than 5 the flow is predominantly laminar and has a linear velocity profile, while for values of y^+ greater than 30 the flow is fully turbulent with a logarithmic velocity profile. For y^+ values between 5 and 30 the flow is transitional and intermittent, with a log-linear velocity profile, and equation (18) provides a smooth transition between the three regions.

As explained above, the computational domain for the k and ε equations extends only to the "top" of the special near-wall elements. Hence appropriate boundary conditions are needed at these locations for the k and ε equations. FIDAP automatically applies the following boundary conditions for k and ε

$$\frac{\partial k}{\partial n} = 0, \qquad \varepsilon = \frac{\left(c_\mu^{1/2} k\right)^{1.5}}{\kappa\delta} \qquad (20)$$

The boundary conditions defined in equation (20) are representative of the equilibrium conditions in the near-wall regions.

When using the above scheme to model turbulent flow around objects in the flow field it is important to ensure that the first layer of elements is thick enough to completely contain the viscous sub-layer and transition region. This is done by plotting the y^+ values at the position of the WALL boundaries after the simulation has converged. Provided the y^+ values are greater than 30 for all elements then the elements have been chosen thick enough so that they completely contain the boundary layer. If the y^+ values are significantly lower than 30 then the solution of the k and ε equations will extend into the boundary layer and could result in inaccurate predictions. If this occurs the simulation needs to be re-run with a coarser grid in the direction normal to the wall.

2.5 Solution Algorithms

FIDAP offers a choice of two quite different solution strategies for the full set of coupled nonlinear equations describing the flow. For small 2D problems the *fully coupled* approach is the most appropriate. In this method the discretised equations are assembled into one large global matrix system and the equations are then solved simultaneously using direct Gaussian elimination. This approach is the most efficient, but can only be implemented when the global matrix system fits within available computer memory. For large 2D problems, and most 3D problems, particularly those involving turbulent flow where an additional two transport equations need to be solved (for k and ε), the most appropriate method is the *segregated* approach. This method substantially reduces disk storage requirements compared to the fully coupled solver because the global matrix system is never directly constructed. In the segregated approach each conservation equation is solved separately in a sequential manner. Each of the nonlinear flow equations is linearised using the fixed-point Picard method (also known as the successive substitution method) and the resulting linear equations are then solved using either direct Gaussian elimination or iterative solvers.

Linearisation using the fixed-point Picard method can be illustrated as follows; application of the Galerkin form of the Method of Weighted Residuals to the stationary Navier-Stokes equations results in a set of non-linear algebraic equations which have the following form

$$\mathbf{K}(\mathbf{u})\,\mathbf{u} = \mathbf{F} \tag{21}$$

In equation (21) $\mathbf{K}$ is the global system matrix, $\mathbf{u}$ is the global vector of unknowns (velocities and pressures), and $\mathbf{F}$ is a vector which includes the effects of body forces and boundary conditions. Picard iteration solves equation (21) using the following method

$$\mathbf{K}(\mathbf{u}_i)\,\mathbf{u}_{i+1} = \mathbf{F} \tag{22}$$

This algorithm evaluates the nonlinear terms at the currently known values of the variables $\mathbf{u}_i$ and then solves the simple linear equation (22) to calculate new values of the variables, $\mathbf{u}_{i+1}$, at the next iteration level. The convergence rate of this scheme is slow (asymptotically linear), but the method converges for a fair range of Reynolds number. The rate of convergence can often be improved, or instability in the convergence process can be avoided, by the use of a relaxation factor α. This is done by writing the solution of equation (21) at the *(i+1)th* iterate as

$$\mathbf{u}_{i+1} = \alpha\,\mathbf{u}_i + (1-\alpha)\,\mathbf{u}^* \tag{23}$$

where $\mathbf{u}^*$ satisfies the equation

$$\mathbf{K}(\mathbf{u}_i)\,\mathbf{u}^* = \mathbf{F} \tag{24}$$

The optimum values of relaxation factors are flow dependent and are usually chosen by trial and error. In FIDAP, relaxation factors can be entered selectively for each primary flow variable, and they can either be fixed in value or expressed as a function of the level of convergence of the solution.

When using the segregated solution approach the default relaxation scheme is Hybrid Relaxation. This scheme employs a combination of four different relaxation factors for each degree of freedom, and these are denoted as follows:

α - explicit relaxation factor
β_a implicit advection relaxation factor
β_d –implicit diffusion relaxation parameter
γ - implicit source-based relaxation factor

Apart from γ, which is computed dynamically by FIDAP for every nodal unknown, the three factors α, β_a , and β_d are subject to user control and can be changed by using the FIDAP command RELAXATION (HYBRID). The default values for turbulent flows of the type considered here are shown in Table 1:

Table 1: Default values for the relaxation parameters for the relevant flow variables.

Factor Type	u_x	u_y	u_z	p	k	ε
Explicit	0.3	0.3	0.3	0.6	0.3	0.3
Implicit: Advection	0.1	0.1	0.1	0.0	0.05	0.05
Implicit: Diffusion	0.01	0.01	0.01	0.0	0.01	0.01

In a three-dimensional, turbulent, iso-thermal Navier-Stokes calculation the variables to be solved for are the three velocity components, the pressure, the turbulent kinetic energy, and the energy dissipation. The equations to be solved are the three components of the momentum equation, the continuity equation, and equations for the turbulent kinetic energy and energy dissipation. Before this set of equations can be solved using a segregated solution approach it is necessary to combine the three components of the momentum equation with the continuity equation to obtain an explicit equation for the pressure. This is easily done, but the resulting explicit matrix equation for the pressure is prohibitively expensive (in terms of computer memory) to construct and solve, and so simplified forms of the pressure equation need to be found which require less memory to solve.

FIDAP provides a choice of three different approximations for the solution of the pressure equation; the pressure correction (PC) algorithm, the pressure projection (PP) algorithm, and the pressure update (PU) algorithm. Each of these algorithms solves a slightly simplified form of the pressure equation either directly for the pressure p, or indirectly for a pressure increment Δp. The PC algorithm is a direct finite element counterpart of the SIMPLE algorithm, which is a well established and successful algorithm for solving the explicit pressure equation in finite-volume CFD codes [24].

The PP algorithm is a consistent finite-element counterpart of the SIMPLER algorithm, which is an improvement of the SIMPLE algorithm, while the PU algorithm is similar to a penalty function method [25]. Haroutunian et al. [26] have presented results from a number of numerical tests which show conclusively that the PP algorithm is the most efficient solution method.

The segregated algorithm requires the solution of multiple linear equation systems at each iteration for the update of each of the variables u_i, p, k and ε (this is known as the "outer" loop). Each of these equations involves non-symmetric coefficient matrices, except for the pressure equation, which involves solution of a symmetric coefficient system. The default method in FIDAP for solving these equations is direct Gaussian elimination. The only problem with this method is that it uses excessive amounts of computer memory and disk storage space when large 3D problems are solved. To overcome this problem FIDAP has a number of iterative methods available for the solution of linear equation systems. Conjugate gradient (CG) and conjugate residual (CR) algorithms are available for the solution of linear equations with symmetric coefficient matrices, while conjugate gradient squared (CGS) and generalised minimal residual (GMRES) algorithms are available for the solution of equations with non-symmetric coefficient matrices. Each of these iterative methods uses considerably less computer memory than direct Gaussian elimination.

To be effective however each of these iterative solvers requires preconditioning of the coefficient matrices before the solution algorithms are applied. In FIDAP this preconditioning is done in two different ways; implicit preconditioning and conventional preconditioning. The implicit preconditioning is achieved by using implicit relaxation of the non-symmetric advection-diffusion type linear equation systems and explicit relaxation of the pressure in each of the solution algorithms for the pressure equation. This relaxation provides a form of preconditioning which is applied to the original linear equation systems before conventional preconditioning is applied. Conventional preconditioning of a linear equation system such as $Ax=b$ is achieved by replacing the original system with the following equivalent system

$$[P_l^{-1} A P_r^{-1}][P_r x] = P_l^{-1} b \tag{25}$$

and then attempting to make the transformed matrix $P_l^{-1} A P_r^{-1}$ as close as possible to the identity matrix. In FIDAP two types of conventional preconditioning are available; diagonal and SSOR. In diagonal preconditioning the matrix P is chosen to be diagonal, and in FIDAP applications it has the form P=diag(A). SSOR preconditioning is more involved. If the matrix A has the following decomposition

$$A = L + D + U \tag{26}$$

where L and U are lower and upper matrices, and D = diag(A), then P is given by

$$P = (U + D)^{-1} D (L + D)^{-1} \quad (27)$$

In FIDAP, both diagonal preconditioning and SSOR preconditioning can be used in the iterative solution of both symmetric and non-symmetric equation systems.

Haroutunian et al. [26] have presented results from a number of numerical tests which studied the relative effectiveness of each of the four iterative solution methods (CG and CR for symmetric coefficient matrices, CGS and GMRES for non-symmetric coefficient matrices), as well as the two conventional preconditioning methods (diagonal and SSOR, within the context of the FIDAP code. Through a number of extensive numerical tests it was shown that the CR solver with SSOR preconditioning was the optimal choice for solving symmetric equation systems, and the CGS solver with diagonal preconditioning was the optimal choice for non-symmetric equation systems.

The segregated approach requires the iterative solution of linear systems of equations at two different levels (if iterative solvers are used to replace the direct Gaussian solver in the "inner loop"), and appropriate convergence criteria are required for each of these levels. For the "outer loop", which provides an estimate of the solution variables $\mathbf{U}_i$ (at iteration i) from the variables $\mathbf{U}_{i-1}$ (at iteration $i-1$) this has the form

$$\left\| \frac{U_i - U_{i-1}}{U_i} \right\| \leq DTOL \quad (28)$$

where the norm $\| \ x \ \|$ is a root mean square norm summed over all the equations for the model. The norm is computed separately for each degree of freedom in the problem. For example, in a three-dimensional, isothermal, turbulent problem this would mean the three components of velocity, pressure, turbulent kinetic energy and dissipation. Convergence is considered to be obtained when all of these norms are simultaneously less than the specified DTOL tolerance. For the segregated solver the default value of DTOL is 0.001, although this can be changed by specifying a different value for the VELCONV keyword in the SOLUTION command.

The iterative solvers which replace the default Gaussian solution method in the "inner loop" also require convergence criterion to terminate the iterative procedure. For the CGS and GMRES methods the tolerance is specified using the NCGCONV keyword, while for the CR and CG methods it is specified using the SCGCONV keyword. The default value for each of these is 0.0001, indicating that good convergence of these "inner loop" iterative solvers is required before attempting convergence of the non-linear iterations in the "outer loop".

All CFD codes, whether finite-difference, finite-volume, or finite-element, suffer from the problem of numerical undershoots and overshoots in the flow variables caused by discretization of the convection terms in the flow conservation equations. These problems typically occur when sharp gradients in the flow variables are encountered

on the computational grid. The common solution to this problem is to add varying amounts of artificial numerical diffusion to the solution algorithm to stabilise the overall convection scheme. The diffusion has the effect of weighting the convection towards the upwind regions of the flow, hence the algorithms are referred to generically as "upwinding schemes". FIDAP allows the user the choice of three different upwinding schemes – Streamline, First-Order, or Hybrid. Detailed descriptions of each of these schemes is provided in the FIDAP Theory Manual [12]. For turbulent problems the default scheme is Streamline upwinding. This explicitly adds numerical diffusion only along the flow direction. Hence it reduces accuracy to first order in the streamwise direction, but preserves the higher order accuracy of the Galerkin scheme in the cross-wise stream direction where no numerical diffusion is added. FIDAP allows the user to set different streamline upwinding factors for each of the flow variables, and this is done using the OPTIONS (UPWINDING) command. Preliminary sample calculations showed that computed results, such as lift and drag coefficients, were insensitive to changes in the upwinding factor however and so the default value, 1.0, has been used in all the calculations reported here.

As most of the FIDAP simulations which will be undertaken in the future to calculate hydrodynamic coefficients for underwater vehicles will involve large scale three-dimensional turbulent simulations, all the calculations reported here have used the segregated solution approach with the PP algorithm for the pressure equation, and the recommended iterative solvers for the linear equation systems.

3. Drag on a Cylinder - Introduction

The flow around a smooth, infinitely long circular cylinder with its axis perpendicular to the flow is well described by Massey [27]. The flow is approximately two-dimensional, and the nature of the flow is governed by the value of the Reynolds number. For very low values (eg. $Re < 0.5$) the inertia forces are negligible and the flow is laminar. As the Reynolds number increases ($2 < Re < 30$) the laminar boundary layer separates symmetrically from the downstream side of the cylinder and two eddies are formed which rotate in opposite directions. At a Reynolds number of about 90 these eddies break off from the cylinder and then eddies are continuously shed from each side of the cylinder, forming what is known as a Von Kármán vortex street.

At higher Reynolds numbers the individual vortices disintegrate into random turbulence close to the cylinder and a regular vortex street can no longer be observed. For Reynolds numbers between approximately $Re = 2{,}000$ and $Re = 2.0 \times 10^5$ the wake has a width approximately equal to the diameter of the cylinder, the boundary layer is laminar, and the drag coefficient C_D is approximately constant and has a value between 0.9 and 1.2. At a Reynolds number slightly greater than 2×10^5 the boundary layer becomes turbulent. As turbulent boundary layers are better able to withstand an adverse pressure gradient the separation point now moves further downstream and the wake narrows. This leads to better pressure recovery on the downstream side of the

cylinder and therefore a reduction in drag, which is now determined almost totally by the pressure imbalance on the front and rear of the cylinder.

Between Re = 2.0 $\times 10^5$ and Re = 5.0 $\times 10^5$ the drag coefficient drops from a value of 1.2 to 0.3. According to Massey [27], over the range 5.0 $\times 10^5$ < Re < 3.0 $\times 10^6$ C_D then rises again from 0.3 to approximately 0.7, and has a value of 0.40 at Re = 2.0×10^6. Delany and Soresen [28] present much more detailed data for C_D as a function of Re for circular cylinders but their data is multivalued, with C_D lying in the range 0.2 to 0.3 at Re = 5.0×10^5, and having a value of 0.5 at Re = 2.0×10^6. There is very little experimental data on values of the drag coefficient for smooth cylinders in the Reynolds number range 10^6< Re <10^7. The most detailed set of data is given by Roshko [29]. His experiments show that C_D increases in the range 1.0 $\times 10^6$< Re < 3.5 $\times 10^6$ from a value of about 0.3 to about 0.7, and then levels off at the latter value.

All of the *k*-ε turbulence models in FIDAP assume that the flow in the boundary layer has already undergone a transition from laminar to turbulent flow. Hence our simulation of turbulent flow around a cylinder using FIDAP is restricted to Reynolds numbers greater than 2.0×10^5 . In the next section we present drag coefficients calculated at flow speeds of 0.25 m/s and 1.0 m/s for several different turbulence models. These speeds correspond to Reynolds numbers of 5.0×10^5 and 2.0×10^6 respectively. Hence we expect to obtain drag coefficients in a range between 0.20 and 0.30 at 0.25 m/s, and in the range 0.40 to 0.50 at 1.0 m/s.

3.1 Calculations using Quadrilateral Elements

The cylinder is located at the origin of a rectangular mesh and has a radius of 1.0 m. The working fluid is water and is assumed to be incompressible. The far-field boundaries are located equidistant from the cylinder in the ± x and ± y directions. It is important that these are placed far enough away from the cylinder so that their location has a negligible effect on the calculated value of C_D. Hence a number of runs were performed in which the far-field boundaries were progressively moved further away from the cylinder until further changes in their location produced less than a 1.0% change in C_D. The results from these runs are shown in Table 2.

The automatic Paving algorithm within GAMBIT was used to mesh the grid with first-order quadrilateral elements and the size of the elements in the direction normal to the surface of the cylinder was controlled using the Boundary Layer function within GAMBIT. The thickness of the first row of elements was 1.5 mm and the growth rate in the normal direction was typically 1.15 for the first 20 to 30 rows. The circumference of the cylinder was divided into 240 intervals of equal length and the angular resolution around the cylinder was kept fixed during the first series of runs. The fluid had a velocity of 1.0 m/s, or a Reynolds number of Re = 2.0×10^6 based on diameter. The initial assumed free stream turbulent intensity was 5.0% and the standard k-ε turbulence model was used with the Boussinesq constitutive relationship. The initial

values for k and ε were 0.004 and 0.006 respectively. The segregated solution algorithm with the recommended combination of iterative solvers was used for each of the runs.

The results in Table 2 indicate that the location of the far-field boundaries does have a significant effect on the calculated value of the drag coefficient. These need to be at least 7 to 8 cylinder diameters from the cylinder before their effect on the numerical value falls below the 1% level. This is consistent with the experience found by other code users [30], who routinely ensure that boundaries are at least 10 to 20 chord lengths distant from objects of interest in the flow. At these distances the assumed boundary condition on each of the variables also had negligible effect on the calculated drag coefficient. The far field boundary conditions for all the runs listed in Table 2 for example were performed without imposing prescribed boundary conditions on either the pressure, or the two velocity components u_x and u_y. Hence, as described in Section 2.4, Neumann boundary conditions are automatically imposed (i.e. the gradient of the variable in the direction perpendicular to the boundary is set to zero). It was found that imposing the condition $u_x = 1.0$ on the top and bottom of the grid had no effect on the computed value of C_D when the far field boundaries where located at ±16 cylinder radii, while imposing the condition $u_y = 0.0$ on the top and bottom of the grid changed C_D by no more than 0.5% at these distances.

Table 2: Drag coefficient C_D as a function of the location of the outer boundaries.

Grid Size	Number of elements	VELCON	C_D	y^+_{min}	y^+_{max}
8m × 8m	17188	1.0×10^{-3}	0.4100	33.89	111.18
12m × 12m	20944	1.0×10^{-3}	0.3150	31.31	104.82
16m × 16m	22691	1.0×10^{-3}	0.2984	30.38	102.34
20m × 20m	23545	1.0×10^{-3}	0.2862	30.22	101.18
24m × 24m	23923	1.0×10^{-3}	0.2814	29.94	100.39
28m × 28m	24159	1.0×10^{-3}	0.2781	29.66	99.84
32m × 32m	24391	1.0×10^{-3}	0.2760	29.99	99.78
32m × 32m	24391	1.0×10^{-4}	0.2743	30.13	99.90

Flow speed = 1.0 m/s

Table 2 also lists the maximum and minimum y^+ values around the circumference of the cylinder. These values depend on the height of the first element above the cylinder surface, as described in Section 2.4, and these are estimated and re-adjusted as required to ensure that the elements have been chosen thick enough to completely contain the viscous sub-layer and transition region. According to the FIDAP Tutorial Manual [31], "In isothermal flows, the predicted velocity field is generally insensitive to y^+ values in

the range $10 < y^+ < 1000$ for most wall boundaries where the flow remains attached to the wall". However, "In flow problems involving subtle separation phenomena, such as separation occurring on gently sloping surfaces or on curved surfaces, the predicted flow field will be sensitive to the y^+ values upstream of the separation point. In these situations the most accurate predictions will be obtained if the y^+ values upstream of potential flow separation are kept in the range $30 < y^+ < 100$."

The height of the first element (denoted by Δh) for the runs in Table 2 and Table 3 was 1.5 mm, which resulted in acceptable y^+ values, as shown. This value was determined from several trial runs on a 4.0 m × 4.0 m sized grid using Δh values of 1.0 mm, 1.5 mm and 2.0 mm. The minimum y^+ values around the circumference of the cylinder for each of these runs was 20.69, 32.73 and 46.25 respectively, with the corresponding maximum y^+ values being 73.77, 107.21, and 142.04, indicating that a first element height of 1.5 mm was optimal. It is interesting to note that the calculated drag coefficient for each of these runs was 0.416, 0.414, and 0.397 respectively, indicating that the height of the first element, at least in this range, has little effect on the computed value of C_D. This was found to be true in general for the standard k-ε turbulence model when used with the Boussinesq constitutive relationship. However, when the standard k-ε model was used with the non-linear constitutive relationships there was found to be quite a sensitive dependence of C_D on Δh.

Table 3: Drag coefficient C_D as a function of angular resolution.

Number of intervals	Number of elements	C_D	Y^+_{min}	Y^+_{max}
240	24,391	0.2743	30.14	99.90
280	26,920	0.2668	30.08	99.99
320	31,507	0.2611	29.93	100.07
360	36,383	0.2569	29.90	100.22
400	42,070	0.2534	29.85	100.30
440	48,729	0.2505	29.83	100.42
480	64,298	0.2481	29.81	100.51

Flow speed = 1.0 m/s

Having established a suitable distance for the location of the far-field boundaries and a reasonable thickness for the first layer of elements on the cylinder, the angular resolution around the cylinder was then increased by performing a number of runs with increasing numbers of intervals around the circumference of the cylinder. The results are shown in Table 3. The increased angular resolution also had a significant effect on the drag coefficient, doubling the resolution from 240 to 480 intervals for example produced a 10% drop in the value of C_D. This is perhaps not surprising as the

drag coefficient is very much dependent on the location of the separation points on the downstream side of the cylinder, and the increased angular resolution around the cylinder allows these to be more accurately determined.

3.2 Effect of Turbulence Model

From the results given in Tables 2 and 3 we can conclude that the standard k-ε turbulence model in FIDAP predicts a value for the drag coefficient of approximately 0.25 at a Reynolds number of 2.0×10^6. The experimental results in section 3.0 indicate that the true value lies somewhere in the range 0.4 – 0.5. The disagreement is not surprising however as it is well known that the standard k-ε turbulence model is unable to accurately calculate the position of separation points in the presence of strong adverse pressure gradients. As discussed in section 2.3, the relevant turbulence models within FIDAP which can be reasonably applied to the flows considered here are the Standard k-ε model, the Extended k-ε model, the RNG k-ε model, and the Anisotropic k-ε model. Each of these two-equation models can be utilised with either the Boussinesq, Speziale or Launder constitutive relationships (although the Boussinesq constitutive relationship is obviously not appropriate for the Anisotropic k-ε model). Table 4 shows the results of applying each of these models and constitutive relationships to the flow around a smooth cylinder at a flow velocity of 1.0 m/s. The far field boundaries were located at ±16 cylinder radii, the angular resolution around the circumference of the cylinder was 360 elements, VELCON was set to 10^{-3}, and Neumann boundary conditions were applied at the far-field entrainment boundaries.

Table 4: Drag coefficient C_D for each of the applicable turbulence models.

	Standard k-ε	Extended k-ε	RNG k-ε	Anisotropic k-ε
Boussinesq	0.2569	0.2625	0.2683	-
Speziale	0.2419	0.2320	0.2321	0.2457
Launder	0.2978	0.2994	0.3010	-

Flow speed = 1.0 m/s

As Table 4 shows, neither the Extended k-ε model nor the RNG k-ε model offers much improvement when used with the Boussinesq constitutive relationship. The Extended k-ε model increases the computed value by 2.2%, while the RNG model shows a 4.4% increase. When used with the Launder constitutive model each of the k-ε models does provide a slightly more realistic value for $C_{D,}$ with the average value being approximately 0.30, which is a 16% increase on the value calculated using the standard k-ε model and the Boussinesq constitutive relationship. The height of the first element had to be adjusted slightly to 1.25 mm when using the Launder constitutive relationship to ensure that the y^+ remained within the appropriate range, and it was more difficult to obtain convergence using this constitutive relationship.

When used with the Speziale constitutive relationship each of the variants of the *k-ε* model resulted in slightly lower values for C_D, varying between 5% and 10% lower. In each of these cases the height of the first element also had to be altered considerably, from 1.5 mm to 5.5 mm, and convergence was again more difficult to obtain than with the Boussinesq constitutive relationship. It was also noted that C_D varied more markedly with Δh than with either of the other constitutive relationships, and that these variations occurred in a non-uniform manner.

3.3 Variation of Drag Coefficient with Flow Speed

As noted in section 3.0, the value of C_D varies significantly in the Reynolds number range between 5.0×10^5 and 2.0×10^6 (ie. at flow speeds between 0.25 m/s and 1.0 m/s respectively). To test the ability of the turbulence models considered here to capture this variation with flow speed several runs were performed using the Boussinesq constitutive relationship with the Standard *k-ε* model, the Extended *k-ε* model, and the RNG *k-ε* model at a flow speed of 0.25 m/s. The results are shown in Table 5. As is clearly evident, each of these models is completely incapable of capturing the experimental variation with flow speed. Over the considered range the experimental value of C_D changes by approximately 50%, while the maximum change for any of the *k-ε* models is less than 2%.

Table 5: Drag coefficient C_D for each of the applicable turbulence models for the Boussinesq constitutive relationship at different flow speeds.

	Standard *k-ε*	Extended *k-ε*	RNG *k-ε*	Experiment
u_x = 1.0 m/s	0.2569	0.2625	0.2683	0.4 –0.5
u_x = 0.25 m/s	0.2567	0.2670	0.2684	0.2 – 0.3

The effect of the initial assumed free stream turbulence intensity on these calculations was checked for many of these runs and was found to have negligible effect on the calculated value of C_D (although it did sometimes have a considerable effect on the convergence rate of the calculation). For example, Table 6 presents the results from a model calculation performed at initial intensity levels of 0.5%, 1.0% and 5.0%. As is clearly evident, the initial free stream intensity level has almost no effect on the calculated drag coefficient, a 10% change in free stream turbulence level making less than a 1% change in the drag coefficient.

Table 6. Drag coefficient C_D as a function of assumed free stream turbulence intensity.

Turbulent Intensity Level	0.5%	1.0%	5.0%
k value	0.00004	0.00015	0.004
ε value	0.000001	0.00001	0.006
Drag coefficient	0.339	0.340	0.342

3.4 Calculation using Triangular Elements

As discussed in section 2.2, FIDAP users can chose between quadrilateral or triangular elements to mesh two-dimensional problems, and can also combine both types of elements to create a hybrid unstructured mesh. This offers some advantages for the types of problems considered here as it has already been noted that the boundaries must be placed at considerable distances from the objects of interest in the flow. This constraint causes problems when the grid is meshed purely with quadrilateral elements as it was found that the automatic paving algorithm within GAMBIT gave the user very little control over the size of the elements as they covered the grid. The elements are required to have small dimensions near the surface of the object to capture boundary layer details, but ideally they should then increase in size, in a uniform manner, as the far-field boundaries are approached. The automatic paving algorithm usually produced a uniformly fine mesh near the surface of the body and a uniformly coarse mesh near the far-field boundaries, but was not particularly successful at grading the element sizes in the regions where the two meshes combined. This often resulted in both very irregularly shaped elements in certain regions of the grid, which led to convergence problems when trying to solve the underlying equations, as well as to an unacceptably large number of elements on the grid. These problems can be overcome by dividing the mesh into a number of sub-spaces and using appropriately spaced nodes along the boundaries of these sub-spaces to control the size of the quadrilateral elements, but the process can become exceedingly tedious and time consuming when complicated geometries are involved. Triangular elements however can grow uniformly from millimetre dimensions near surfaces to metre dimensions near boundaries. When combined with the GAMBIT paving algorithm it was found that they gave much greater flexibility in meshing more complex shapes on grids covering several orders of magnitude of spatial dimensions.

It should be noted however that the triangular elements employed in commercial codes such as FIDAP have proven to be particularly inappropriate at modelling flows near surfaces [32]. This is due to the very rapid variation in flow speed through the boundary layer normal to the surface compared with the relatively slower variation in speed in the streamwise direction. This problem can be overcome in specialised non-commercial codes by using local stretching. When this option is unavailable then the most appropriate approach is to use are quadrilateral elements with high aspect ratios. Hence the most appropriate mesh is a hybrid grid consisting of quadrilateral elements near the surfaces and triangular elements elsewhere. FIDAP allows the use of such unstructured hybrid meshes, but it was found that extra effort was required to obtain solutions on these types of grids. Initially, convergence was found to be much more difficult to obtain than on structured meshes using just quadrilateral elements. After much experimentation it was found that convergence could usually be obtained on hybrid meshes if the continuous pressure approximation was implemented and the relaxation parameters were considerably increased.

Table 7 shows results from a number of different runs calculated on a hybrid mesh for a smooth 1.0 m diameter cylinder immersed in a flow of 1.0 m/s. The Standard k-ε model with the Boussinesq constitutive relationship was used for each of these calculations. The first run on a 20m×20m grid using 33,702 first-order triangular elements gave C_D = 0.3122. The corresponding result using 23,545 first-order quadrilateral elements was (from Table 2) C_D = 0.2862. Increasing the convergence limit from 10^{-3} to 10^{-4} dropped the value to 0.3102, but there is still an 8% difference from the result calculated using quadrilateral elements. This may be due to the difference in grid resolution, ie. the number of elements in each calculation. On a 32m×32m grid using 34,890 elements the calculated value is 0.3001, while the comparable value calculated using 24,391 quadrilateral elements (from Table 2) is 0.2760, which again shows an 8% difference. Repeating this run using 11,754 second order elements gave a calculated value of 0.3458, although the y^+_{min} value was a little too low. Increasing Δh from 1.5 mm to 2.0 mm increased y^+_{min} to 34.37, but this had almost negligible effect on the value of C_D, which changed from 0.3458 to 0.3459.

Table 7: Drag coefficient calculated using triangular elements.

Order	Grid Size (m^2)	Angular resolution	y^+_{min}	y^+_{max}	Velcon	Δh (mm)	C_D
1st	20×20	240	28.73	100.68	10^{-3}	1.5	0.3122
1st	20×20	240	28.98	100.82	10^{-4}	1.5	0.3102
1st	32×32	240	28.05	98.81	10^{-3}	1.5	0.3001
2nd	32×32	240	24.42	92.62	10^{-3}	1.5	0.3458
2nd	32×32	240	34.37	121.75	10^{-3}	2.0	0.3459

4. Lift on a Flat Plate at 5° angle of incidence

The subject of classical aerofoil theory is treated in many texts on elementary fluid dynamics. Acheson [33] for example derives the standard Kutta-Joukowski Lift Theorem. If a two-dimensional body with a cross-section described by a simple closed curve C is located in a uniform flow with speed u_x in the x direction, then the forces on the body in the directions parallel and perpendicular to the flow (F_x and F_y respectively) are given by:

$$F_x = 0.0, \qquad F_y = -\rho\, u_x\, \Gamma \tag{29}$$

where Γ is the circulation around the body. The first expression is simply D'Alembert's paradox, which states that the steady uniform flow of an ideal fluid past a fixed body gives no drag on the body. The second expression gives a formula for the lift on the

body. To use this expression we first have to evaluate the circulation. Acheson further proves that for uniform irrotational flow past an aerofoil with a sharp trailing edge there is just one value of the circulation Γ for which the velocity is finite everywhere. This is the standard Kutta-Joukowski condition. In the case of a thin symmetrical aerofoil of length l making an angle of attack α with the oncoming stream the value of Γ is given by

$$\Gamma = - \pi u_x l \sin \alpha \tag{30}$$

Combining equation (30) with equation (29) results in the following expression for the lift force on the wing

$$L = F_y = \pi \rho u_x^2 l \sin \alpha \tag{31}$$

In terms of the lift coefficient C_L this is simply

$$C_L = 2 \pi \sin \alpha \tag{32}$$

Equation (32) is applicable to either a thin symmetric wing or a flat plate. At a 5^0 angle of incidence equation (32) results in a value for the lift coefficient of 0.5476. Equation (32) is only valid of course for small angles of attack where the boundary layer flow around the wing remains attached and the flow can be described by ideal potential flow theory. At angles of attack near 10^0 the boundary layer begins to separate and the flow must be described by the Navier-Stokes equation.

4.1 Calculation using quadrilateral elements

To check the ability of FIDAP to accurately calculate the lift coefficient for simple aerodynamic shapes a number of runs were performed to calculate the lift coefficient for a flat plate at a 5^0 angle of incidence. The plate was 1.0 m long and 50 mm thick and the ends of the plate consisted of semi-circular segments. The plate was placed at the centre of a rectangular computational grid initially having a length of 4.0 m and a height of 2.0 m. The flow velocity was 1.0 m/s and the Reynolds number was 1.0×10^6.

The mesh was constructed by dividing the semi-circular end segments into 60 equally spaced intervals and the 1.0 m long straight sections of the plate into 300 intervals. A spacing ratio of 1.05 was used so the interval spacing near the ends of the plate were similar to those on the curved ends. The GAMBIT boundary layer function was used to ensure a uniform mesh of quadrilateral elements near the surface of the plate. The height of the first row of elements was 1.5 mm, the growth factor was 1.05, and the boundary layer extended for 20 rows. To ensure the continuation of a uniform mesh outside this limited boundary layer region the computational domain was divided into separate areas by constructing boundaries on the grid corresponding to plates having slightly longer lengths and greater thicknesses. These areas were then meshed using the GAMBIT MAPPING function to ensure that a high quality grid (low EquiAngle

Skew values) was obtained in close proximity to the plate. The remaining area was meshed using the GAMBIT automatic PAVING algorithm. This resulted in a grid having 54,019 first order quadrilateral elements. Less than 0.11% of these elements had an EquiAngle Skew value greater than 0.5. An illustration of the way in which the grid was divided into a number of separate areas to control the quality of the meshing is shown in Figure 1 below.The results from calculations on this grid (or an extended version of the grid) are shown in Table 8. The initial assumed free stream turbulent intensity was 5.0% and the standard k-ε turbulence model was used.

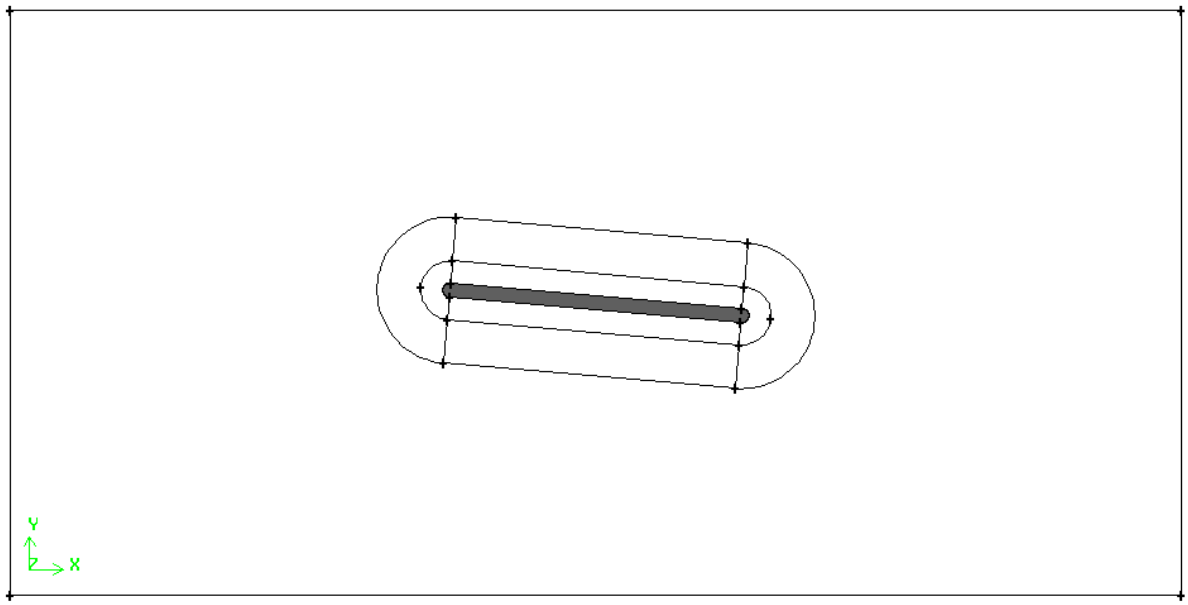

Figure 1: Schematic of the way the grid for the flat plate is divided into separate areas to control the quality of the meshing.

Table 8: Lift coefficient for a flat plate. Dirichlet boundary conditions with u_x=1.0 and u_y=0.0 on the top & bottom far field boundaries.

Grid Size (m^2)	Number of elements	VELCON	C_L	Y^+_{min}	Y^+_{max}	Boundary conditions
4×2	55,099	1.0×10^{-3}	0.4670	26.35	155.6	Dirichlet
6×4	61,761	1.0×10^{-3}	0.4369	26.39	155.96	Dirichlet
8×6	65,434	1.0×10^{-3}	0.4307	26.83	157.13	Dirichlet
8×6	65,434	1.0×10^{-5}	0.5059	28.14	160.03	Dirichlet
8×6	65,434	1.0×10^{-6}	0.5064	28.14	160.05	Dirichlet

The first three runs in Table 8 show the effect of increasing the size of the grid by moving the far-field boundaries further away from the plate. As can be seen, the calculated lift coefficient is in error by 15%, and the disagreement increases as the boundaries are moved further away. Improved agreement with the theoretical value is obtained by increasing the convergence criterion from VELCON = 1.0×10^{-3} to VELCON = 1.0×10^{-6}, but there is still an 8% error. From the discussion in section 3.1 it is obvious that the boundaries of the grid are located too close to the plate, but Table 8 shows a trend away from the correct result as the boundaries of the grid are moved further away from the plate. Believing that this may have been caused by a poor quality grid, a number of runs were also made on a slightly different grid. This had a greater resolution near the curved ends of the plate (80 node points) and lower resolution along the length of the plate (200 node points). In addition, the area of the grid near the external boundaries was further sub-divided into separate areas and node spacing along the edges of these areas was used to ensure a more uniform meshing of the grid. The resulting mesh contained 61,620 first order quadrilateral elements. The results from calculations on this mesh are shown in Table 9.

Although the new grid had much better resolution near the leading edge of the plate the results were virtually identical. On an 8m × 6m grid with VELCON = 1.0×10^{-5} the calculation using the first grid gave a value for the lift coefficient of 0.5059, while the calculation on the second grid gave a value of 0.5061. Increasing the location of the far field boundaries to distances of 10 to 20 chord lengths from the plate, which are more realistic distances for the outer edges of the grid, gave conflicting results. On a 10m × 10m grid the calculated value was lower, ie. 0.4963, while on a 20m × 20m grid the value was slightly higher, ie. 0.4978. These values are still approximately 9% too low however. It is also interesting to note that even on a 20m × 20m grid the boundary conditions are having a non-negligible effect on the results. Imposing the conditions $u_x = 1.0$ and $u_y = 0.0$ on the top and bottom of the grid results in $C_L = 0.4978$, while imposing Neumann boundary conditions on the top and bottom of the grid results in the value $C_L = 0.4701$, ie. a 6% difference in the calculated value. Increasing the size of the grid to 40m × 40m and employing Neumann boundary conditions results in $C_L = 0.4832$, but this is still in error by 12%. In an attempt to resolve this problem calculations were then made on a standard NACA wing section having a well defined profile and a sharp trailing edge. These calculations are described in the next section.

Table 9: Lift coefficient for a flat plate. Dirichlet boundary conditions imply u_x=1.0 and u_y=0.0 on the top & bottom far field boundaries.

Grid Size (m^2)	Number of elements	VELCON	C_L	Y^+_{min}	Y^+_{max}	Boundary conditions
4×2	62,618	1.0×10^{-5}	0.5640	27.92	160.45	Dirichlet
6×4	72,687	1.0×10^{-5}	0.5162	27.86	160.37	Dirichlet
8×6	85,248	1.0×10^{-5}	0.5061	27.82	161.74	Dirichlet
10×10	86,752	1.0×10^{-3}	0.4105	26.21	158.06	Dirichlet
10×10	86,752	1.0×10^{-5}	0.4963	27.80	162.21	Dirichlet
20×20	94,271	1.0×10^{-5}	0.4978	27.79	165.79	Dirichlet
20×20	94,271	1.0×10^{-5}	0.4701	27.83	163.80	Neumann
40×40	97,321	1.0×10^{-5}	0.4832	27.84	169.02	Neumann

5. Lift on a NACA Wing Profile at 5^o angle of incidence

The wing chosen was the NACA 0006 wing section. The numbering system for the NACA wing sections in the four digit series is based on section geometry. The first integer indicates the maximum value of the mean-line ordinate as a percentage of the chord length. The second integer indicates the distance from the leading edge to the location of the maximum camber in tenths of the chord. The last two digits indicate the section thickness as a percentage of the chord length. Thus the 0006 wing section has zero camber, ie. is symmetrical, and has a maximum thickness of 6% of the chord length. Geometry data for the 0006 wing section was obtained from Abbott and Von Doenhoff [34]. This reference also contains a large compilation of lift, drag and moment coefficient data for many different wing types, and the experimental data for the lift coefficient for a NACA 0006 wing section at a 5^o angle of incidence agrees within experimental error with the theoretical value for a thin wing or flat plate of $C_L = 2\,\pi \sin\alpha$, or $C_L = 0.5476$ at $\alpha = 5^0$.

5.1 Calculations using quadrilateral elements

Table 10 shows the calculated lift coefficient C_L for a number of runs on different sized grids. For the smaller sized grids the mesh was constructed in a similar manner to those for the flat plate in the previous section, ie. close to the plate the mesh consisted of mapped first order quadrilateral elements, while the far-field region was meshed using the automatic PAVING algorithm. For the larger sized grids, ie.12.5m ×10m and above, the entire grid was meshed using the MAPPING algorithm. This took

considerably more time and effort, but resulted in a better quality mesh. Neumann boundary conditions were employed at the top and bottom of the grid.

The value for C_L calculated on the smaller sized grids is considerably below the theoretical value of 0.5476. As the boundaries of the grid are moved further away from the wing section the calculated value moves closer to the theoretical value (when Neumann boundary conditions are used), and further improvement is obtained by increasing the convergence criterion from VELCON = 1.0×10^{-3} to VELCON = 1.0×10^{-4}. Even on the 24.5m ×40.5m grid however the calculated value of 0.4882 is still 11% lower than the theoretical value. It is also noticeable that even on this relatively large grid the boundary conditions are still having a significant effect on the calculated value.

Table 10: Lift coefficient for NACA 0006 – Neumann boundary conditions

Grid Size (m^2)	Number of elements	VELCON	C_L	Y^+_{min}	Y^+_{max}	Boundary conditions
4×2	24,305	1.0×10^{-3}	0.2625	17.742	196.38	Neumann
6×4	32,445	1.0×10^{-3}	0.3663	17.415	222.76	Neumann
8×6	39,854	1.0×10^{-3}	0.4092	16.467	266.83	Neumann
12.5×10	107,408	5.0×10^{-4}	0.4513	35.6	273.5	Neumann
24.5×40.5	361,460	1.0×10^{-3}	0.4946	-	-	Neumann
24.5×40.5	485,784	5.0×10^{-4}	0.4882	23.1	170.0	Neumann

Table 11 shows the effect on C_L of different far-field boundary conditions. Similar runs to those shown in Table 10 were performed with the Neumann boundary conditions replaced by Dirichlet boundary conditions, ie. u_x = 1.0 and u_y = 0.0 on the top and bottom of the grid. On the smaller sized grids this has a dramatic effect on the value of C_L, but this improvement does not reflect the true situation. Imposition of the Dirichlet boundary conditions on the smaller sized grids simply means that the calculation is simulating conditions similar to those which cause the "wing in ground" effect, which is well known to increase the value of the lift coefficient. As the far-field boundaries are moved further away from the wing section the calculated value of C_L drops until it appears to converge to a value of approximately 0.5100 at realistic boundary positions. This value is still approximately 7% lower than the theoretical value.

Table 11: Lift coefficient for NACA 0006 – Dirichlet boundary conditions

Grid Size (m^2)	Number of elements	VELCON	C_L	Y^+_{min}	Y^+_{max}	Boundary conditions
4×2	24,887	1.0×10^{-3}	0.5489	16.178	266.24	Dirichlet
6×4	32,445	1.0×10^{-3}	0.5193	16.340	265.11	Dirichlet
8×6	39,854	1.0×10^{-3}	0.5045	16.638	262.76	Dirichlet
8×6	39,854	1.0×10^{-5}	0.5176	16.467	266.83	Dirichlet
20×20	72,445	1.0×10^{-3}	0.5007	16.400	267.27	Dirichlet
20×20	72,445	1.0×10^{-4}	0.5100	16.421	268.64	Dirichlet
20×20	72,445	1.0×10^{-5}	0.5100	16.441	270.00	Dirichlet
24.5×40.5	488,992	1.0×10^{-3}	0.5022	22.2	178.00	Dirichlet

5.2 Calculations using triangular elements

In an attempt to improve the accuracy of the simulated value for C_L a further series of runs was performed using first and second-order triangular elements. The results from these runs are shown in Table 12. As discussed in section 3.4 for the drag calculations on a smooth cylinder, convergence is much more difficult to obtain on unstructured meshes using a combination of quadrilateral elements in the boundary layer and triangular elements for the majority of the mesh. The calculations shown in Table 12 were all made to converge by implementing the continuous pressure approximation and increasing the relaxation parameters significantly over those required for convergence on a regular grid constructed purely from mapped quadrilateral elements.

The two runs using first-order triangular elements show results similar to those from calculations using first-order quadrilateral elements. On the 20m×20m grid with Dirichlet boundary conditions the calculated value of C_L is 0.5114, which is comparable to the value of 0.5100 shown in Table 11. Using Neumann boundary conditions the value is 0.4836, which is also consistent with the results shown in Table 8. Using second-order triangular elements and a convergence criterion of VELCON = 1.0×10^{-4} produces an improvement in the result, ie. C_L = 0.5310. This is the best simulated value obtained, but it is still 3% lower than the theoretical value.

Table 12: Lift coefficient for NACA 0006 – triangular elements.

Grid Size (m^2)	Element order	Number of elements	VELCON	C_L	Y^+_{min}	Y^+_{max}	Boundary conditions
20×20	1st	62,342	1.0×10^{-3}	0.4836	37.28	150.68	Neumann
20×20	1st	62,342	1.0×10^{-3}	0.5114	37.00	158.37	Dirichlet
20×20	2nd	59,098	1.0×10^{-3}	0.5219	53.88	269.3	Dirichlet
20×20	2nd	59,098	1.0×10^{-4}	0.5310	54.00	273.12	Dirichlet

6. Drag on a Sphere

The drag coefficient for a sphere has a similar behaviour as a function of Reynolds number as that of an infinitely long circular cylinder with its axis perpendicular to the flow. Being a three-dimensional body, however, the alternating vortices forming the Von Kármán vortex street are replaced by the formation of a vortex ring. This forms at a Reynolds number of approximately Re =10 and moves further downstream of the body as the Reynolds number increases. For Reynolds numbers between 200 and 2000 the vortex ring may become unstable and move downstream, and its place is immediately taken by a new ring. Separation of the laminar boundary layer begins at the downstream stagnation point and moves upstream as the Reynolds number increases. At approximately Re = 1000 a stable separation point is reached at about 80^0 from the front stagnation point. The drag coefficient then becomes approximately independent of Re until the boundary layer becomes turbulent before separation. This occurs at approximately Re = 3.0×10^5. The separation point then moves further downstream, the wake narrows, and the drag coefficient drops considerably, from around 0.50 to approximately 0.08. A further increase in Re then leads to a slow increase in the drag coefficient again. The exact values are difficult to determine as these depend both on the surface roughness of the sphere and the initial turbulence level in the free stream. Data from Newman [35], Massey [27] and Rouse [36] are shown in Table 12.

Table 13: Drag coefficient for a smooth sphere as a function of Reynolds number.

Reynolds number	0.4×10^6	0.5×10^6	0.6×10^6	1.0×10^6	2.0×10^6	3.0×10^6	4.0×10^6	5.0×10^6
Newman	0.10	0.08	0.10	0.15	0.17	0.19	0.20	0.20
Massey	0.08	0.08	0.10	0.10	0.15	0.16	0.18	0.20
Rouse	0.08	0.10	0.10	0.12	-	-	-	-

6.1 Calculations using hexahedral elements

The sphere had a radius of 1.0 m and was initially located in the middle of a cube having sides of length 6.0 m. The region was meshed using first-order hexahedral elements using the GAMBIT volume meshing command Map. The Map scheme can only be applied to regions having the shape of a logical cube however and so to utilise this scheme for the current problem the region was first divided into six sub-volumes, each of which was logically equivalent to a cube. This was done by creating a second brick shaped volume having a length slightly greater, and a width slightly less, than the diameter of the sphere. The Boolean Split Real Volumes command was then used to intersect the two volumes, which created a four sided cap on the top and bottom of the sphere. The sphere was then intersected by two planes at right angles to each other which resulted in the formation of four further surfaces around the circumference of the sphere. This meant that the surface of the sphere was now divided into six separate areas, each of which was logically equivalent to a square, and each of which could be mapped to one of the six faces on the surrounding cube.

The size of the hexahedral elements was determined by the number of nodes placed along the 12 curves defining the six separate surfaces on the surface of the sphere, the number of nodes placed along the 12 edges defining the cube containing the sphere, and the number of nodes placed along the 8 lines joining the 8 corners of the cube to the 8 logical corners defined by the curves on the surface of the sphere. For the first run each of the 12 curves on the surface of the sphere was divided into 20 equally spaced intervals, each of the 12 edges of the confining cube was divided into 20 equally spaced intervals, and each of the 8 lines joining the corners of the cube to the logical corners on the surface of the sphere was divided into 45 intervals. The node spacing near the surface of the sphere was controlled using the Boundary Layer function within GAMBIT. The thickness of the first row of elements was chosen to be 2.0 mm and the growth rate used was 1.15 for the first 24 rows. The height of the remaining 20 rows was determined by the Mesh Edge command using a Successive Ratio Grading Scheme and a ratio of 1.1.

To ensure that the sphere was far enough away from influences due to flow into and out of the computational domain the mesh was then expanded by adding 6.0 m^3 cubes to both the inflow and outflow faces of the cube containing the sphere. Each of these cubes was meshed by dividing each of the cube edges into 20 equal intervals and filling the volumes with hexahedral elements using the GAMBIT Map command. A total number of 132,000 hexahedral elements was created using this particular meshing scheme. A schematic of the way in which the grid was divided into a number of separate volumes to control the quality of the meshing is shown in Figure 2 below.

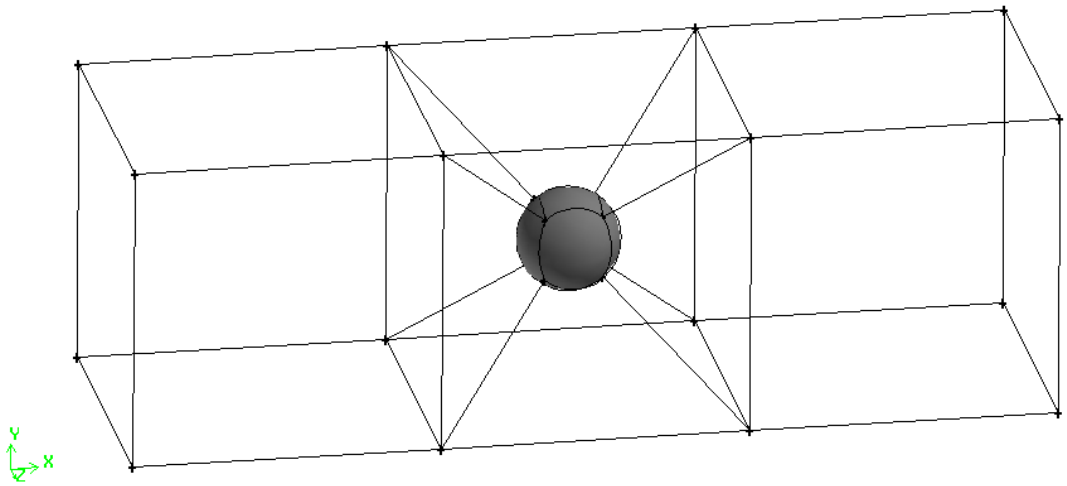

Figure 2: Schematic of the way the grid for the mapped sphere is divided into separate areas to control the quality of the meshing.

The segregated solution algorithm with the recommended combination of iterative solvers was again used to solve the equations. The initial velocity had u_x = 1.0 m/s and u_y = u_z = 0.0, giving a Reynolds number 2.0×10^6. The initial assumed free stream turbulent intensity was 5.0%, the initial values for k and ε were 0.004 and 0.006 respectively and the default convergence criteria were used. The calculated drag coefficient had a value of 0.210. Table 13 shows that the approximate experimental value at this Reynolds number is 0.16±0.01.

To examine any dependency of this result on the flow speed the same sphere, with identical meshing, was then run with a free stream velocity of u_x = 0.25 m/s and u_y = u_z = 0.0, giving a Reynolds number of 0.5×10^6. The data in Table 13 indicate an experimental value somewhere between 0.08 and 0.10 for this value of the Reynolds number. The free stream turbulence intensity was kept at 5.0%, which lead to k and ε values of 0.000234 and 0.00001 respectively. The calculated drag coefficient was only marginally lower and had a value of 0.20, indicating again, as in section 3.3, that the standard k-ε model is incapable of capturing the changes in the drag coefficient as a function of flow speed in the critical region.

To check the grid dependency of this result a further run was made with finer gridding. For this run each of the 12 curves on the surface of the sphere was divided into 30 equally spaced intervals, each of the 12 edges of the confining cube was divided into 30 equally spaced intervals, and each of the 8 lines joining the corners of the cube to the logical corners on the surface of the sphere was divided into 60 intervals. The thickness of the first row of elements on the surface of the sphere was 1.0 mm and the growth rate used was 1.15 for the first 30 rows. The height of the remaining 30 rows was controlled by the Mesh Edge command using a Successive Ratio Grading Scheme

and a ratio of 1.05. This resulted in a total of 396,000 elements, or three times the number used in the previous run. The calculated drag coefficient for this run was 0.15, which is much closer to the experimental value of 0.16±0.01. Hence a threefold increase in the number of elements has resulted in a 25% decrease in the value of the drag coefficient. The total time taken for solution on this relatively large mesh was 22 hours on an SGI Octane using one of two R10,000 processors running at 225 Mhz. Solution time for the runs using the more coarsely grided mesh was approximately 5.5 hours.

7. Discussion and Conclusion

The simulation results presented here indicate that the FIDAP code has several problems with regard to accurate simulations of turbulent flow around underwater bodies. The main problem appears to be the inability of the turbulence models incorporated in the code to fully capture the relevant physics required to simulate flow separation effects at high Reynolds numbers. The simulation results showed that the models were unable to accurately calculate drag coefficients for two-dimensional cylinders and three-dimensional spheres. In particular, the models were unable to reproduce the strong variation of drag coefficient for each of these shapes in the critical region just after transition from laminar to turbulent flow in the boundary layers. There was also very little difference between the results calculated using the standard k-ε model and the three variants of this model, ie. the RNG k-ε model, the Extended k-ε model, and the Anisotropic k-ε model.

Another problem with the code, unrelated to the turbulence models, was the difficulty found in calculating accurate values for the lift coefficient of a flat plate or a NACA wing profile. Modern CFD codes should be able to routinely calculate lift coefficients for wings at low angles of attack, where the flow is fully laminar, to within an accuracy of 1% [37]. The values calculated using FIDAP were consistently around 8% too low, even when excessively large numbers of elements were used in the calculations. This problem, as well as the evident failure of the turbulence models to accurately predict flow separation effects, prevented us from pursuing more interesting calculations, such as the angle of stall for different wing shapes. This problem in particular is of current interest to MPD with regard to other problems related to underwater flow.

A further problem with the code has been the difficulty in obtaining converged solutions on unstructured hybrid grids containing mixtures of quadrilateral and triangular elements (in two-dimensions), or hexahedral and tetrahedral elements (in three-dimensional flows). This problem has been noted by other FIDAP users [38], who were unable to find acceptable solutions to these problems. As was noted in section 3.4, considerable effort was required to obtain convergence on unstructured grids containing both quadrilateral and triangular elements, and all efforts to obtain solutions on grids containing tetrahedral elements were unsuccessful.

Because of the above mentioned problems it is recommended that FIDAP should be replaced by a CFD code containing both more advanced turbulence models, as well as the ability to more easily obtain converged solutions on hybrid unstructured meshes. One such code which is currently used extensively in DSTO is the finite-volume code Fluent [9]. This code is used routinely in Air Operations Division for both standard and non-standard aeronautical applications, in Air Vehicles Engines Division for simulation of combustion flows, and also in Maritime Platforms Division for multi-phase fluid flow problems.

Fluent would appear to be an excellent replacement for FIDAP for the class of problems considered here because it has been shown to have superior convergence properties on unstructured hybrid meshes [37], as well as a variety of advanced turbulence models [9]. Fluent users are able to perform Large Eddy Simulations of turbulent flow using two subgrid-scale stress models (Smagorinsky-Lilly and an RNG-based subgrid scale model), as well as a Reynolds Stress Model (RSM), neither of which are available in FIDAP.

Fluent also contains the standard, RNG, and realizable k-ε models, but their implementation in Fluent is enhanced by more accurate near-wall treatments for wall-bounded flows. Fluent uses both the standard wall functions, plus more advanced non-equilibrium wall functions which partly account for the effects of pressure gradients and give improved results in flows involving separation. The good agreement which McKenzie [20] has found using the RNG k-ε model in the Fluent code to model flow separation around submarine shapes is probably due to the use of non-equilibrium wall functions with this model.

8. References

1. Jones, D.A., Clarke, D.B., Brayshaw, I.B., Barillon, J.L. and Anderson, B. “The Calculation of Hydrodynamic Coefficients for Underwater Vehicles”. DSTO *Technical Report*, DSTO-TR-1329, 30pp, July 2002.
2. Clarke, D.B. and Jones, D.A. “HYGUESS - A Computer Code for the Calculation of Hydrodynamic Coefficients for Underwater Vehicles”, DSTO *Technical Report*, in preparation.
3. Aage, C. and Smitt, L.W., “Hydrodynamic Manoeuvrability Data of a Flatfish Type AUV”, IEEE Proceedings, OCEANS 94, pp. III-425 to III-430.
4. Marker, M., Anderson, B. Gaston, D., and Kelly, P., "Design of the Underwater Vehicle MODROV" Presented at the Undersea Defence Technology Pacific Conference, (this paper contains a description of the Wayamba vehicle) Sydney, 1997.
5. Peterson, R.S., “Evaluation of semi-empirical methods for predicting linear static and rotary hydrodynamic coefficients”, NCSM TM 291-80.

6. Peterson, R.S., "Hydrodynamic Analysis of Submersibles – The HYSUB System", March, 1997.

7. Nahon, M., Department of Mechanical Engineering, University of Victoria, Victoria, B.C., Canada. Private communication, June, 1999.

8. FIDAP– Commercially available CFD software package based on the Finite Element method. A product of Fluent Inc., Centerra Resource Park, 10 Cavendish Court, Lebanon, NH, USA. Distributed in Australia by CFD-RES, Sydney.

9. Fluent – Commercially available CFD software package based on the Finite Volume method. A product of Fluent Inc., Centerra Resource Park, 10 Cavendish Court, Lebanon, NH, USA. Distributed in Australia by CFD-RES, Sydney.

10. CFX– Commercially available CFD software package based on the Finite Volume method. A product of AEA Technology Engineering Software, Didcot, Oxfordshire, UK. Distributed in Australia by ATD International Pty. Ltd., Hawthorn, Vic.

11. Acheson, D.J. "*Elementary Fluid Dynamics*", Clarendon Press, Oxford, 1990.

12. *IDAP 8 Theory Manual*, Fluent, Inc., December 1998.

13. Givler, R.C., Gartling, D.K., Engleman, M.S. and Haroutunian, V. "Navier-Stokes Simulations of Flow Past Three-Dimensional Submarine Models", *Computer Methods in Applied Mechanics and Engineering*, **87**, 175-200 (1991).

14. Williams, J. and Hajiloo, A., "Aerodynamic Flow Field Around Two Car-Like Shapes, Computational vs Experimental Fluid Dynamics", available from the Fluent website, www.fluent.com, as paper number 276.

15. Hajiloo, A., Williams, J., Hackett, J.E. and Thompson, S.A. "Limited Mesh Refinement Study of the Aerodynamic Flow Field around a Car-Like Shape, Computational vs. Experimental Fluid Dynamics", SAE Congress, Detroit, Michigan, February 26-29, 1996.

16. White, F.M., "*Viscous Fluid Flow*", McGraw-Hill, 2nd Edition, 1991

17. Wilcox, D.C. "*Turbulence Modelling for CFD*", 2nd edition, DCW Industries, (2000).

18. Jones, W.P. and Launder, B.E. "The Prediction of Laminarization with a Two-Equation Model of Turbulence", *Internat. J. Heat Mass Transfer*, **15**, pp. 301-314 (1972).

19. Yakhot, V., Orszag, S.A., Thangam, S., Gatski, T.B. and Speziale, C.G. "Development of Turbulence Models for Shear Flows by a Double Expansion Technique", *Physics of Fluids A*, **4**, (7), pp. 1510 – 1520 (1992).

20. McKenzie, G., AMRL CFD Users Group Meeting, Fishermens Bend, 11 April, 2000.

21. Chen, Y.S. and Kim, S.W., "Computation of Turbulent Flows Using an Extended k-ε Turbulence Closure Model", NASDA CR-179204, 1987.

22. Piquet, J., "*Turbulent Flows – Models and Physics*", Springer-Verlag, 1999.

23. Cresswell, R., MSC.Software Australia, 271 William St, Melbourne, Vic 3000. Formerly Compumod, and former distributers of the FIDAP code. Private communication, July 1999.

24. Anderson, J. D., Jr., "*Computational Fluid Dynamics – The Basics with Applications*", McGraw-Hill, Inc., 1995.

25. Peyret, R. and Taylor, T.D., "*Computational Methods for Fluid Flow*", Springer Series in Computational Physics, Springer-Verlag, New York, Inc., 1983.

26. Haroutunian, V, Engelman, M.S. and Hasbani, I., *International Journal for Numerical Methods in Fluids*, **17**, 323-348 (1993).

27. Massey, B.S., "Mechanics of Fluids", 4th Edition, Van Nostrand Reinhold Company Ltd., 1979.

28. Delany, N.K. and Sorensen, N.E. "Low-Speed Drag of Cylinders of Various Shapes", NACA Technical Note 3038, November 1953.

29. Roshko, A. "Experiments on the flow past a circular cylinder at very high Reynolds number", J. Fluid Mech. **10**, 345-356, 1961.

30. Bulbeck, J., AMRL CFD Users Group Meeting, Fishermens Bend, 11 April, 2000.

31. *FIDAP 8 Tutorial Manual*, Fluent, Inc., December 1998.

32. Fluent AUSSEA Users Group Meeting, Australian Technology Park, Sydney, November 11-12, 1999.

33. Acheson, D.J., "*Elementary Fluid Mechanics*", Clarendon Press, Oxford, 1990.

34. Abbott, I.A and Von Doenhoff, A.E. "*Theory of Wing Sections*", Dover Publications Inc. 1959.

35. Newman, J.N., "*Marine Hydrodynamics*", MIT Press, 1977.

36. Rouse, H. "*Elementary Mechanics of Fluids*", John Wiley and Sons, 1946.

37. Fairlie, B. AMRL CFD Users Group Meeting, Fishermens Bend, 11 April, 2000.

38. Seeling, C. Kodak Australasia Pty. Ltd., Coburg, Victoria. Private communication, March, 2000.

DISTRIBUTION LIST

An Evaluation of the FIDAP Computational Fluid Dynamics Code for the Calculation of Hydrodynamic Forces on Underwater Platforms

D.A. Jones and D.B. Clarke

AUSTRALIA

DEFENCE ORGANISATION

	No. of copies
Task Sponsor	
CSO (Cap Dev) COMAUSNAVSUBGRP, Stirling	1
S&T Program	
Chief Defence Scientist FAS Science Policy AS Science Corporate Management Director General Science Policy Development	shared copy 1
Counsellor Defence Science, London	Doc Data Sheet
Counsellor Defence Science, Washington	Doc Data Sheet
Scientific Adviser to MRDC, Thailand	Doc Data Sheet
Scientific Adviser Joint	1
Navy Scientific Adviser	1
Scientific Adviser - Army	Doc Data Sht & Dist List
Air Force Scientific Adviser	Doc Data Sht & Dist List
Scientific Adviser to the DMO M&A	Doc Data Sht & Dist List
Scientific Adviser to the DMO ELL	Doc Data Sht & Dist List
Platforms Sciences Laboratory	
Chief of Maritime Platforms Division	Doc Data Sht & Dist List
Research Leader: Ms Janis Cocking	Doc Data Sht & Dist List
Dr D.A. Jones	3
Mr D.B. Clarke	1
Mr B. Anderson	1
DSTO Library and Archives	
Library Maribyrnong	Doc Data Sheet
Library Edinburgh	1
Australian Archives	1
Capability Systems Division	
Director General Maritime Development	1
Director General Aerospace Development	Doc Data Sheet
Director General Information Capability Development	Doc Data Sheet
Office of the Chief Information Officer	
Deputy CIO	Doc Data Sheet
Director General Information Policy and Plans	Doc Data Sheet
AS Information Strategies and Futures	Doc Data Sheet

AS Information Architecture and Management Doc Data Sheet
Director General Australian Defence Simulation Office Doc Data Sheet

Strategy Group
Director General Military Strategy Doc Data
SheetDirector General Preparedness Doc Data Sheet

HQAST
SO (Science) (ASJIC) Doc Data Sheet

Navy
SO (SCIENCE), COMAUSNAVSURFGRP, NSW Doc Data Sht & Dist List
Director General Navy Capability, Performance and Plans, Navy Headquarters Doc Data Sheet
Director General Navy Strategic Policy and Futures, Navy Headquarters Doc Data Sheet

Army
ABCA National Standardisation Officer, Land Warfare Development Sector, Puckapunyal e-mailed Doc Data Sheet
SO (Science), Deployable Joint Force Headquarters (DJFHQ) (L), Enoggera QLD Doc Data Sheet
SO (Science) - Land Headquarters (LHQ), Victoria Barracks NSW Doc Data & Exec Summ

Intelligence Program
DGSTA Defence Intelligence Organisation 1
Manager, Information Centre, Defence Intelligence Organisation 1
Assistant Secretary Corporate, Defence Imagery and Geospatial Organisation Doc Data Sheet

Defence Materiel Organisation
Head Airborne Surveillance and Control Doc Data Sheet
Head Aerospace Systems Division Doc Data Sheet
Head Electronic Systems Division Doc Data Sheet
Head Maritime Systems Division Doc Data Sheet
Head Land Systems Division Doc Data Sheet
Head Industry Division Doc Data Sheet
Chief Joint Logistics Command Doc Data Sheet
Management Information Systems Division Doc Data Sheet
Head Materiel Finance Doc Data Sheet

Defence Libraries
Library Manager, DLS-Canberra Doc Data Sheet
Library Manager, DLS - Sydney West Doc Data Sheet

OTHER ORGANISATIONS
National Library of Australia 1
NASA (Canberra) 1

UNIVERSITIES AND COLLEGES
Australian Defence Force Academy
Library 1

Head of Aerospace and Mechanical Engineering	1
Serials Section (M list), Deakin University Library, Geelong,	1
Hargrave Library, Monash University	Doc Data Sheet
Librarian, Flinders University	1

OUTSIDE AUSTRALIA

INTERNATIONAL DEFENCE INFORMATION CENTRES

US Defense Technical Information Center	2
UK Defence Research Information Centre	2
Canada Defence Scientific Information Service	e-mail link to pdf
NZ Defence Information Centre	1

ABSTRACTING AND INFORMATION ORGANISATIONS

Library, Chemical Abstracts Reference Service	1
Engineering Societies Library, US	1
Materials Information, Cambridge Scientific Abstracts, US	1
Documents Librarian, The Center for Research Libraries, US	1

SPARES	5
Total number of copies:	**34**

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION
DOCUMENT CONTROL DATA

1. PRIVACY MARKING/CAVEAT (OF DOCUMENT)

2. TITLE

An Evaluation of the FIDAP Computational Fluid Dynamics Code for the Calculation of Hydrodynamic Forces on Underwater Platforms

3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED REPORTS THAT ARE LIMITED RELEASE USE (L) NEXT TO DOCUMENT CLASSIFICATION)

Document	(U)
Title	(U)
Abstract	(U)

4. AUTHOR(S)

D.A. Jones and D.B. Clarke

5. CORPORATE AUTHOR

Platforms Sciences Laboratory
506 Lorimer St
Fishermans Bend Victoria 3207 Australia

6a. DSTO NUMBER	6b. AR NUMBER	6c. TYPE OF REPORT	7. DOCUMENT DATE
DSTO-TR-1494	AR-012-893	Technical Report	August 2003

8. FILE NUMBER	9. TASK NUMBER	10. TASK SPONSOR	11. NO. OF PAGES	12. NO. OF REFERENCES
510/207/1295	NAV 00/206	Navy- CANSG	38	38

13. URL on the World Wide Web

http://www.dsto.defence.gov.au/corporate/reports/DSTO-TR-1494.pdf

14. RELEASE AUTHORITY

Chief, Maritime Platforms Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved for public release

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SA 5111

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CITATION IN OTHER DOCUMENTS Yes

18. DEFTEST DESCRIPTORS

Computational Fluid Dynamics, Underwater Vehicles, Numerical Simulation, Fluid Flow, Lift, Drag

19. ABSTRACT

Maritime Platforms Division within DSTO is currently studying the science and technology of autonomous underwater vehicles for defence applications. Part of this work involves a study of the hydrodynamics and manoeuvrability of these vehicles and the development of methods to determine the hydrodynamic coefficients of submerged bodies as a function of their shape. This report describes the application of the FIDAP Computational Fluid Dynamics package to the calculation of lift and drag forces on relatively simple underwater vehicle shapes including cylinders, spheres, flat plates and wing profiles. The degree to which FIDAP accurately reproduces known experimental data on these shapes is described and the applicability of other Computational Fluid Dynamics packages is discussed.