A Fourth Order Accurate Method to Compute the 3D Flow Around a Ship

Johan F. Malmliden*
Center for Computational Mathematics and Mechanics
Royal Institute of Technology, Stockholm, Sweden

August 19, 1994

Abstract

A fourth order accurate method to compute the three dimensional flow around a ship hull is presented. The time dependent flow around a ship is very complex, especially around and after the stern. This flow is modeled by the incompressible Navier–Stokes' equations. We simplify the problem by neglecting free surface effects and replacing the surface by a lid where a symmetry condition is imposed.

We discretize the geometry by using a composite overlapping grid. The building blocks of the composite grid are a number of simple component grids, that can be constructed using various grid generators. Each component grid covers a simple part of the geometry. The union of the component grids cover the whole computational domain and they overlap where they meet. We base the component grids on a description of the hull consisting of a number of overlapping surface patches. These patches are constructed to be smooth even for complicated geometries by a projection method. By the use of a set of smooth surface patches we easily generate a set of smooth body covering component grids that efficiently and accurately discretize the vicinity of the ship hull. To represent the outer parts of the flow, we use Cartesian or simple cylindrical grids. We use the software CMPGRD to generate a composite overlapping grid around the ship. This enables efficient resolution of both the boundary layer and various flow phenomena around the ship hull, such as trailing vortices astern of the ship.

We use the incompressible Navier–Stokes' solver CGINS to compute a numerical solution to the problem. For high Reynolds number flow, we use a Baldwin Lomax or a k–ϵ turbulence method. The equations are discretized using second or fourth order accurate finite differences. We use explicit, time accurate timestepping by utilizing a second order accurate predictor corrector method. The computational method described in this paper is carefully tested and validated on two and three dimensional examples.

*Partially supported by ONR grant N-00014-90-J-1382, NSF grant DMS 90-61311 and by the U.S. Department of Energy through Los Alamos National Laboratory.

Contents

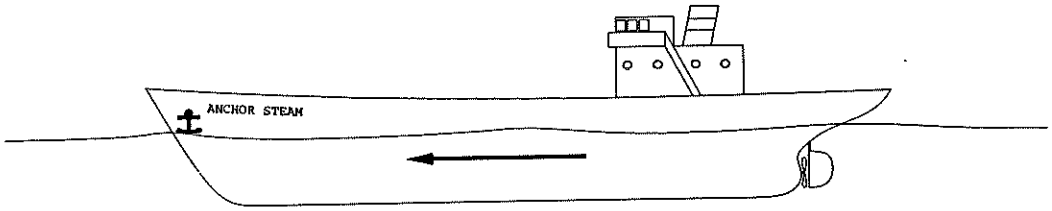


Figure 1: A cargo ship in motion.

1 Introduction

The subject of this paper is a fourth order accurate method to compute the flow around a large ship for example a tanker.

There has been a great effort spent during the last century on devising analytical and computational methods for ship flow. The obvious reason is to use mathematical and numerical methods as a complement to experimental measurements in the design process of a ship. It is extremely expensive to conduct experiments at full scale and even model scale experiments are quite costly. Furthermore, the two similarity parameters, the Reynolds number, Re, and the Froude number, Fr, do not scale identically between model scale and full scale. Experimental design optimization can often be time consuming. Hence there is a need to improve the computational methods for ship flow, both to reduce the cost and time to optimize a ship design.

The time-dependent flow around a cargo ship (for example, a tanker) is very complex in some domains of the flow. The relatively simple flow around the bow and along the hull can be computed to satisfactory accuracy from an engineering point of view. This is done using various simplifications of the Navier–Stokes' equations. One of the common simplifications is to use a a boundary layer approximation close to the hull. Further away from the hull the flow is stationary and irrotational, therefore the flow can be described by potential flow. The solutions of the boundary layer approximation and the outer potential approximation are then matched by enforcing some kind of continuity condition, for references on these methods cf. [37, 40]

However, these approximations cease to be valid in the domain around and after the stern, where it is necessary to use the incompressible Navier–Stokes' equations to model the physics. The stern flow has been extensively studied, but a satisfactory solution has not yet been computed. There are a number of reasons for this: The flow around a large tanker has a very high Reynolds number $O(10^9)$. The flow is therefore quite complicated and fully turbulent. There is also an interaction between the unsteady wake and the free surface. Mostly second-order accurate spatial discretization has been used. It is furthermore a nontrivial task to generate a grid for the complex geometry of a ship that both resolves the solution in the boundary layer and optimizes the distribution of gridpoints in the outer flow domain.

Current methods to compute the flow around a ship can mainly be divided

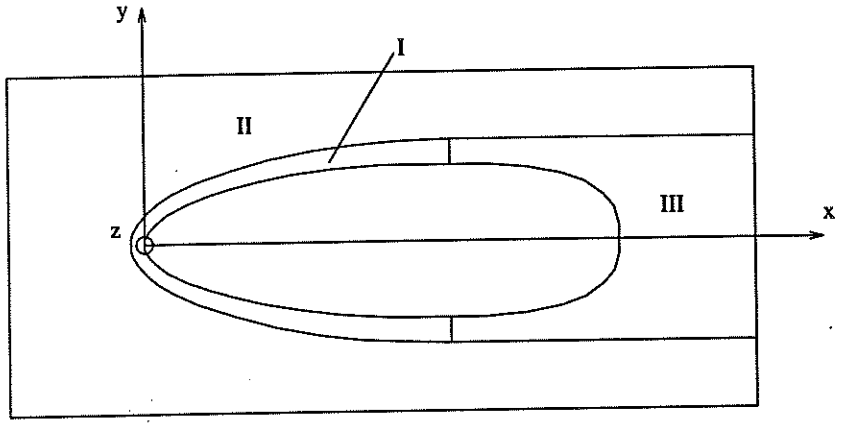


Figure 2: The three flow regime domains. I Boundary layer flow. II Potential flow. III Viscous wake flow.

into two classes:

Methods of class 1. depend on domain decomposition, see figure 2. The flow domain is partitioned into two or three subdomains. Domain I cover the boundary layer and the immediate vicinity of the hull from the bow to a plane placed approximately 1/3 of the ship's total length from the stern. Most of the methods model the flow in this subdomain with some kind of boundary layer approximation incorporated into the Navier–Stokes' equations. This results in a parabolic equation that is relatively inexpensive to solve. Domain II covers the fluid volume behind volume 1 and includes the wake. In this subdomain the flow is fully three dimensional and oftenly time dependent. Therefore the full Navier–Stokes' equations are considered in this subdomain, usually with some kind of turbulence model. This subproblem is very expensive to solve. Domain III consists of the remainder of the fluid. Here the flow is considered to be essentially steady and irrotational. This gives that the velocity can be described by a potential function which is governed by Laplace's equation. The solutions in these subdomins are then coupled by enforcing certain interface conditions, usually by utilizing a Schwarz type iterative method. This iteration converges in a few cycles. For a more thorough description of this method cf. [36]. Some methods in this class only use two domains to model the flow. The flow in both domain I and III is modeled by the incompressible Navier–Stokes' equations and the flow in domain II is modeled by potential flow, cf. [11]. Y. Tahara et. al. [54] has also developed a method that uses desomposition into two subdomains.

Methods of class 2 usually model the flow in the whole computational domain with the Navier–Stokes' equations, with or without turbulence model. The most common turbulence models used in this field are the Baldwin–Lomax and the k–ϵ model. These methods require much more computer time than the domain decomposition methods.

In this work we simplify the problem by neglecting free surface effects and

replace the surface by a lid where a symmetry condition is imposed. In the future we will incorporate a free surface into the model. We also consider the flow to be symmetric with respect to the x–z-plane, see figure 2.

It is generally a difficult task to discretize both the geometry of a ship hull and the wake efficiently. We do this using composite overlapping grids in three steps: First, we generate a nonsingular description of the surface of the hull using a number of overlapping two-dimensional patches. These patches are constructed such that they are smooth, even for complicated geometries, by a projection method. Second, for each of these body-covering patches a component grid is generated. Having a set of smooth, nonsingular surface patches it is easy to generate a set of smooth component grids without singularities. The component grids can be constructed using appropriate existing grid generation techniques. Finally, we use CMPGRD [8] to generate a composite overlapping grid around the ship from a number of component grids. This enables efficient resolution of both the boundary layer and various flow phenomena around the hull such as the unsteady wake and trailing vortices astern of the ship.

We use the incompressible Navier–Stokes' solver CGINS [22] to compute a solution to the problem. The Baldwin Lomax and k–ϵ turbulence models are included in CGINS. The latter turbulence model is found to be the better. It is easy to enter more refined turbulence models into the present method. The equations are discretized using second or fourth order accurate finite differences. The numerical solutions on the different component grids are coupled by enforcing sufficiently accurate interpolation relations. We use explicit, time accurate timestepping by utilizing a second order accurate predictor corrector method.

This solution procedure can be used in methods of both classes, described above. Either as a method for subdomain II (the wake) in a domain decomposition method, or as the entire solver in a Navier–Stokes' method. We choose to implement this method as a part of a domain decomposition method, mainly due to the lesser computational time and storage space needed by this method to obtain a satisfactory solution.

The contents of this paper are organized as follows: In §2 we discuss previous work on this subject and in §3.1 we define the incompressible Navier–Stokes' equations that describe the flow around a ship, especially the formulation of the equations that is employed in the CGINS solver. The different turbulence models are described in §3.3. In §4 we present a theoretical discussion of the errors introduced by the outflow boundary conditions. In §5 we describe the numerical method. We elaborate on the discretization of the complex 3D geometry of the flow using the composite overlapping grid-technique in §5.1. In §5.2.1–5.2.2 we describe the mapping method used to discretize the incompressible Navier–Stokes' equations on a composite overlapping grids and in §5.2.2, we describe the boundary conditions, especially the farfield boundary conditions. The solver CGINS is described briefly in §5.3. Finally, in §6 we present numerical results. A number of example geometries are presented in §6.1 and numerical solutions of flow around the example geometries are displayed. We also validate the method by forcing the solution to be a choosen function. A concluding discussion including an objective for future work is given in §7.

2 Previous Work on the Subject

In this section we describe previous related work. First in §2.1, we elaborate on some of the important work on constructing computational methods for ship flow. Second, in §2.2 we discuss some of the work done in flow computation using overlapping grids.

2.1 Computing the Flow Around a Ship

It is known that a complex interaction exists between the waves, generated by a ship, and the viscous effects in the boundary layer and in the wake. As engineer's strive to improve the ship design, there follows a need to predict both wave drag and viscous drag accurately. Up to a couple of decades ago this work was conducted by analytical methods, very simple computations and - more important - experimental research on model scale ships in towing tanks. Unfortunately the two similarity parameters, the Froude number (Fr) and the Reynolds number (Re) do not scale identically between model scale and full scale. A great effort has therefore been spent to develop numerical methods to solve this problem, in order to construct complementary methods to the experimental design development tools.

The state of computation of ship flow has been assessed in a series of workshops. Some of these are mentioned here. In 1980 SSPA held a workshop in Gothenburg on "Ship Boundary Layers", cf. [35]. There it was found that the boundary layer flow around the bow and the fore part of the hull could be predicted accurately. However the boundary layer approximation is not valid on the aft part of the hull and in the wake so the stern flow predictions were not satisfactory. To investigate the development of methods for computing stern flow a workshop on "Ship Viscous Flow" was held in Gothenburg in 1990, cf. [37]. The computed results predicted the bow flow and the flow on the fore part of the boundary layer with acceptable accuracy, just as in the 1980 SSPA workshop. However, despite a rather wide variety of numerical schemes, grid resolutions and turbulence models, most of the methods produced very similar wake distributions and all of them failed in simulating the hook shape observed in the experimental wake contours. The trailing vortices were badly predicted by most participants.

In 1994 a CFD workshop for "Improvement of Hull Form Designs" was held in Tokyo, cf. [40]. There it was stated that some of the problems in predicting the flow around the stern and in the wake were due to inadequate turbulence modeling. There was, however, a disagreement on how the turbulence models should be developed. Some contributors argued that the use of non-isotropic second moment closure of the Navier–Stokes' equation would enhance the performance of the turbulence model. Others argued that simpler turbulence models should be tailored to each class of ship flow.

One of the main methods used for computing the flow around a ship is represented by "SHIPFLOW" by. L. Larsson et al. cf. [36]. In this method the flow is modeled by different mathematical models in the different flow regions, cf. figure 2. In domain I the flow is considered to be of boundary layer character and

it is modeled by inserting a boundary layer approximation into Navier–Stokes' equations. This results in a parabolic equation that can be efficiently solved by marching from the bow and downstream along the hull. In domain II, that includes the wake, the flow is modeled by inserting a turbulence model into the Navier–Stokes' equations. The equations are discretized by finite differences on a structured grid. This problem is then solved using an implicit line relaxation method, see L. Broberg [6]. The flow in domain III. is considered to be essentially steady and irrotational. This enables the velocity field to be described as the gradient of a potential. The potential is governed by Laplace's equation. This problem is solved by using a the Kelvin source boundary integral method. The solutions in the three domains are usually coupled by some kind of iteration. In "SHIPFLOW" the potential solution is first computed for the whole flow domain. This solution is used to compute the pressure distribution on the hull. This pressure distribution is then used as a forcing for the boundary layer solver. The potential flow solution and the boundary layer solution are matched at a common surface well outside the boundary layer. The boundary layer method computes the boundary layer flow to some point downstream the hull. The method then uses the boundary layer solution as inflow and the potential solution as a side boundary condition for the Navier–Stokes' solver in the wake. This matching is done at some apriori located boundary. Systematic testing has confirmed that it is accurate to locate this boundary half a ships length sidewise from the hull.

One class of methods uses the Navier–Stokes' equations to model the flow in both domain I and II. This type of methods is examplified by F. Stern. [54]. They solve the viscous flow in a thin boundary layer and the wake. Then they compute the displacement thickness and compensate the hull for this. the potential flow is now solved for the outer domain using the perturbed hull . The two solutions are coupled by iterating a number of times.

A new domain decomposition method for viscous free surface ship flow was presented by E. Campana et al., cf. [11]. They employ a linearized version of the free surface condition on the mean surface. Instead of using a classical boundary layer–potential flow iteration method based on displacement thickness, they define a fixed decomposition of the fluid domain with matching surfaces located a priori. They couple the different solutions using a well established procedure, cf. [60]. However the use of this zonal approach for solving free surface flows still needs to be investigated further. They use a finite volume discretization on a structured grid for the viscous flow.

Another of the important methods used to solve incompressible viscous flow problems can be illustrated by the method of T. Hino cf. [27]. This method models the flow in the whole computational domain using the incompressible Navier–Stokes' equations. The method takes the divergence of the momentum equation and iterates the pressure and velocity fields at each time step until continuity is satisfied. Hino uses a finite difference scheme expressed in body fitting curvilinear coordinates. Another large domain method for nonlinear free surface ship flow was designed by Y. Tahara and F. Stern, cf. [53]. They solve the incompressible Navier–Stokes' equations with the Baldwin-Lomax turbulence model. It is worth noting that they apply the nonlinear inviscid free

surface boundary condition on the physical free surface using a body/free surface conforming grid.

A step towards computing a realistic flow around a whole ship including the effects of a free surface was taken by J. Farmer L. Martinelli and A. Jameson, cf. [17, 18, 19]. They use a single, structured grid discretization of the domain and utilize a finite volume method to solve the incompressible Navier–Stokes' equations. The rate of convergence to a steady state is accelerated by using a multigrid scheme. The solution of the bulk-flow is then coupled to the solution of the free surface boundary condition by an iterative method. They transform the free surface onto the mean surface where they apply the free surface boundary condition.

A new method by T. Hino, J. Farmer, L. Martinelli and A. Jameson can be found in [28]. The method employs a single unstructured grid discretization, a finite volume technique and an efficient multigrid method is used to accelerate convergence. The nonlinear free surface condition is applied on the physical surface. The method is tested for two dimensional free surface flow around a wing profile.

In a recent paper by E. Rood, cf. [50] the future needs for computational methods for ship flow was reviewed. They address the need for a numerical towing tank to optimize the ship-construction further. A number of areas are pointed out, where improvements in predicting the flow around a ship are needed. Some of these areas are: Turbulence modeling in the wake has to be improved. There is need for a method to predict the turbulent effective wake into the propeller. Modeling of unsteady cavitation at the propeller also needs enhancements. Furthermore there is a demand for better understanding of the turbulent nonlinear free surface effects. It is pointed out that there is a general need for unsteady, time accurate Navier–Stokes' solvers, to study phenomena that are essentially instationary and include free surfaces.

2.2 Computing Flow using Composite Overlapping Grids

Two dimensional free surface flow around wing profiles using composite overlapping grids have been computed by A. Lungu and K. Mori, [38]. The three-dimensional Navier–Stokes' equations have been solved using overlapping grids by C. Kiris et al. [31]. They have computed the flow in an artificial heart using the artificial compressibility approach and second order accurate finite differences. D. Brown has computed transsonic flow, using a staggered composite grid finite volume method, cf. [7]. K. D. Brislawn et al. has computed flow with large gradients around various geometries using adaptive composite grids; this is described in [5]. P. G. Buning et al. [10] computed the supersonic flow around the space shuttle in ascent using overset grids generated by the CHIMERA grid generator, cf. [3]. E. Pärt and B. Sjögreen [44] have computed two-dimensional flow around the Hermes shuttle using overlapping grids. N. A. Petersson et al. [48, 39], have computed the linearized free surface potential flow around a submerged obstacle using composite overlapping grids. Furthermore N. A. Petersson used moving composite grids to compute nonlinear steep free surface waves, cf. [46], and computed the oscillations of a free jet, cf. [2, 47]. In a

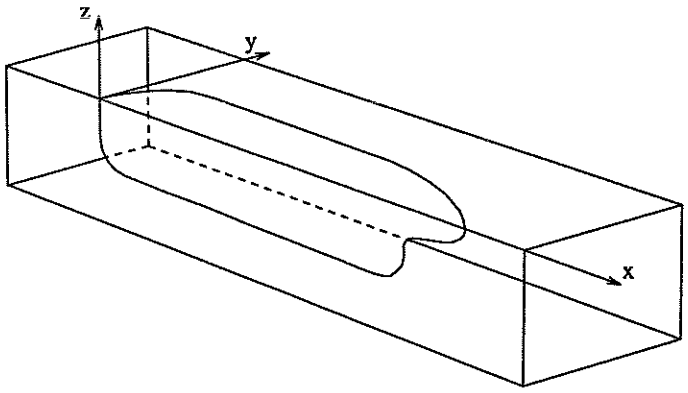


Figure 3: Definition of the coordinate system (x,y,z). The ship bow is located at the origin and the stern at $x = l$, y is directed sideways and z vertically upwards.

recent work, J. Tu and L. Fuchs solve the three dimensional incompressible Navier–Stokes equations on overlapping grids for flow in internal combustion engines with moving pistons and valves [56]. They used a second order accurate method coupled with a multigrid algorithm. Time dependent visco-elastic flow in two dimensions has been computed by F. Olsson et al., using a composite overlapping grid and a second order accurate artificial compressibility method, cf. [41, 42].

High order methods on overlapping grids have been used successfully for a number of problems. G. Browning, [9] used fourth and sixth order accurate methods on overlapping grids to solve the shallow water equations. W. Henshaw and G. Chesshire cf. [13] use fourth order accurate methods to solve elliptic problems on composite overlapping grids. M. J. Ward et al. [58] used fourth order accurate methods to solve nonlinear eigenvalue problems on composite overlapping grids.

The subject of conservation in the interpolation relations was considered by M. Berger [4]. The matter is studied by W. Henshaw et al. [14]. They derive interpolation relations that conserve the unknown quantites for two and three dimensional problems. E. Pärt and B. Sjögreen has also treated this in [45].

3 Governing Equations of the Flow

In this section, we discuss the governing equations of the flow. The momentum and continuity equations are described in §3.1. The specific boundary conditions for ship flow are discussed in §3.2.

3.1 Momentum and Continuity Equations

The flow around a ship is clearly incompressible, we therefore use the following standard form of the incompressible Navier–Stokes' equations to model the physics of the flow cf. [34]. The Cartesian coordinate system we use is defined in figure 3.

$$
\left.\begin{aligned} \mathbf{u}_t + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p - \nu\Delta\mathbf{u} - \mathbf{f} &= \mathbf{0} \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned}\right\} \quad \mathbf{x}\in\Omega, \qquad (1)
$$

$$
B(\mathbf{u},p) = \mathbf{0} \qquad \mathbf{x}\in\partial\Omega,
$$

$$
\begin{aligned} \mathbf{u}(\mathbf{x},0) &= \mathbf{u}_0(\mathbf{x}) \\ p(\mathbf{x},0) &= 0 \end{aligned} \qquad \text{at } t = 0.
$$

Here p is the pressure and ν the kinematic viscosity, $\nu > 0$. The domain Ω lies in $\mathbf{R}^{n_d}$ where n_d, the number of space dimensions, is 2 or 3. There are n_d boundary conditions denoted by $B(\mathbf{u},p) = \mathbf{0}$. On a fixed wall, for example, the boundary conditions are $\mathbf{u} = \mathbf{0}$. Hereafter system (1) will be called *the velocity-divergence form* of the equations.

To get a more tractable form of the continuity equation - an equation for the pressure, we take the divergence of the momentum equation and enter $\nabla\cdot\mathbf{u} = 0$. This gives the following form of the initial-boundary value problem, which is hereafter called *the velocity-pressure formulation*

$$
\left.\begin{aligned} \mathbf{u}_t + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p - \nu\Delta\mathbf{u} - \mathbf{f} &= \mathbf{0} \\ \Delta p + \nabla u\cdot\mathbf{u}_x + \nabla v\cdot\mathbf{u}_y + \nabla w\cdot\mathbf{u}_z - \nabla\cdot\mathbf{f} &= 0 \end{aligned}\right\} \quad \mathbf{x}\in\Omega \qquad (2)
$$

$$
\left.\begin{aligned} B(\mathbf{u},p) &= \mathbf{0} \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned}\right\} \quad \mathbf{x}\in\partial\Omega,
$$

$$
\begin{aligned} \mathbf{u}(\mathbf{x},0) &= \mathbf{u}_0(\mathbf{x}) \\ p(\mathbf{x},0) &= 0 \end{aligned} \qquad \text{at } t = 0.
$$

This is the form of the equations that will be discretized by the solver CGINS, the discretization is described in §5. For a more thorough treatment cf. [22]. The velocity pressure formulation requires an extra boundary condition. Here the condition $\nabla\cdot\mathbf{u} = 0$ for $\mathbf{x}\in\partial\Omega$ is added as the extra boundary condition. W. Henshaw proved the wellposedness and stability for a 2D model problem corresponding to this problem in [20].

3.2 The Boundary Conditions for Modeling Shipflow

In this subsection we elaborate on how we choose the boundary condition $B(\mathbf{u},p) = \mathbf{0}$. At an inflow boundary we prescribe the momentum components ($\mathbf{u} = (u,v,w)$) and enforce the continuity equation.

$$
\left\{\begin{aligned} u &= u_0 \\ v &= v_0 \\ w &= w_0 \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned}\right. ,
$$

where (u_0, v_0, w_0) are functions of (x,z). On the wetted part of the ship hull, we apply a no slip boundary condition.

$$\begin{cases} u & = & 0 \\ v & = & 0 \\ w & = & 0 \\ \nabla \cdot \mathbf{u} & = & 0 \end{cases}.$$

On the symmetry plane, (x–z-plane, where $y = 0$), we apply a symmetry boundary condition. For the bottom- and side-farfield boundaries we can enforce a slip boundary condition if these boundaries are located sufficiently far from the ship. The free water surface is set to be the mean surface (the x–y-plane, where $z = 0$). Here we also apply a slip boundary condition. These boundary conditions can be written in the form.

$$\begin{cases} \mathbf{u}^n & = & 0 \\ \partial \mathbf{u}^{t1}/\partial n & = & 0 \\ \partial \mathbf{u}^{t2}/\partial n & = & 0 \\ \nabla \cdot \mathbf{u} & = & 0 \end{cases},$$

where $\mathbf{u}^n$ denotes the velocity component normal to the boundary, $\mathbf{u}^{t1}$ and $\mathbf{u}^{t2}$ denotes the two velocity components tangential to the boundary and $\partial \mathbf{u}/\partial n$ denotes the normal derivative. Provided that the farfield boundaries are located sufficiently far from the ship, this is a serious restriction only on the free surface boundary. In future works we will incorporate free surface effects into the model of the flow.

At the outflow boundary, we apply the following boundary conditions

$$\begin{cases} \partial u/\partial n & = & 0 \\ \partial v/\partial n & = & 0 \\ \partial w/\partial n & = & 0 \\ \alpha p_n + \beta p & = & \gamma \end{cases},$$

where α, β and γ are constants.

3.3 Turbulence Modeling

The Reynolds number for a large ship i.e. a crude carrying tanker is of the order of 10^9 ($Re = UL/\nu$, where U is the velocity, L is the length of the ship and ν is the kinematic viscosity). In this case the effects of the laminar viscosity is mainly concentrated to the viscous sublayer or where there are large velocity gradients. The flow close to the stern and in the wake is generally quite complicated and fully turbulent.

To be able to compute turbulent flows in realistic geometries where the Reynolds number is high a turbulence model is necessary. Turbulence models based on Prandtl's mixing length hypothesis aims at mimicking the enhanced mixing of momentum due to the turbulence, this effect results in a higher diffusivity than for a corresponding laminar flow. For a concise and thorough treatment on the subject see [59]. The turbulence model gives a turbulent viscosity, that is larger than the laminar viscosity and due to this we have the possibility to resolve the mean flow.

3.4 The Baldwin-Lomax Turbulence Mdel

The Baldwin Lomax turbulence model, hereafter called B-L, will be described briefly below, for a more thorough treatment cf. [1, 57] or [49]. B-L is an algebraic turbulence model which was originally designed for 2-D flow above a flat plate. However numerical computations, for a variety of more complex geometries has shown the B-L model to give results comparable with those obtained by using other, more sophisticated turbulence models. The turbulence model predicts a turbulent viscosity ν_t that is added to the laminar viscosity ν. The effective viscosity $\nu_0 = \nu + \nu_t$ is entered into the Navier–Stokes' equations according to

$$\left.\begin{aligned} \mathbf{u}_t + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p - \nu_0\Delta\mathbf{u} - \mathbf{f} &= 0 \\ \Delta p - (\nabla u\cdot\mathbf{u}_x + \nabla v\cdot\mathbf{u}_y + \nabla w\cdot\mathbf{u}_z) - C\nabla\cdot\mathbf{u} - \nabla\cdot\mathbf{f} &= 0 \end{aligned}\right\} x\in\Omega,$$

where C is a constant – this is descrived more in detail in §5.2.1, the corresponding term is entered to decrease the divergence in the pressure equation. Note that we are not using conservative formulation of the viscous term. We allow this since we are not interpolating conserative between the component grids. Furthermore, we do not have any shocks in the solution.

How to compute ν_t

B-L is a two layer eddy viscosity model in which the turbulent viscosity ν_t is given by

$$\nu_t = min\begin{cases} (\nu_t)_i \\ (\nu_t)_o \end{cases}.$$

The inner boundary layer viscosity $(\nu_t)_i$ is defined by

$$(\nu_t)_i = l\,|\omega|.$$

The mixing length l uses the Van Driest damping length concept cf. [59]

$$l = \kappa\, y\, (1 - exp(-\frac{y^+}{A^+})),$$

where κ is a constant and the wall length unit y^+ is expressed as

$$y^+ = \sqrt{\frac{u_n}{\nu}}\, y,$$

here, u_n is the normal derivative of the velocity at the solid wall and y is the distance from the wall.
The local vorticity is given by

$$|\omega| = \left\{ \left(\frac{\partial u}{\partial y} - \frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial z} - \frac{\partial w}{\partial y}\right)^2 + \left(\frac{\partial w}{\partial x} - \frac{\partial u}{\partial z}\right)^2 \right\}^{1/2}.$$

The outer viscosity $(\nu_t)_o$ is defined by

$$(\nu_t)_o = \alpha\ C_{cp}\ F_{wake}\ F_{Kleb}(y),$$

where α and C_{cp} are constants, their standard values will be given below. The definition of F_{wake} is

$$F_{wake} = min \begin{cases} y_{max} F_{max} \\ C_{wk} y_{max} u_{dif}^2 / F_{max} \end{cases},$$

where F_{max} is defined as the maximum value of F which occurs in a profile. In 2D, we define a profile to be a plane perpendicular to the free stream velocity vector, In 3D we define a profile to be the set of points that share the same closest point on the ship hull. F is defined by

$$F(y) = y\ |\omega|\ (1 - exp(-\frac{y^+}{A^+})),$$

and y_{max} is the y value for which F_{max} occurs.
The intermittency factor F_{Kleb} cf. [59] is

$$F_{Kleb} = (1 + 5.5(\frac{C_{Kleb}\ y}{y_{max}})^6)^{-1}.$$

The quantity u_{dif} represents a velocity scale equal to the difference between maximum and minimum total velocity in a given profile

$$u_{dif} = (\sqrt{u^2 + v^2 + w^2})_{max} - (\sqrt{u^2 + v^2 + w^2})_{min}.$$

The standard values of the constants in the B-L model are

$$\begin{aligned} A^+ &= 26 \\ C_{cp} &= 1.6 \\ C_{Kleb} &= 0.3 \\ C_{wk} &= 0.25 \\ \kappa &= 0.4 \\ \alpha &= 0.0168 \end{aligned}$$

Discretization of the equations

The B-L equations are discretized using second order accurate finite differences, even when the momentum and pressure equations are discretized to fourth order accuracy. The discretization is done in the same manner as for the momentum equations, this will be described in §5

3.5 The k–ϵ Turbulence Model

The standard k–ϵ turbulence model is described below, for a more thorough treatment cf. [49] or [59]. This is a two-equation, eddy-viscosity model. Two new equations are added, one for the turbulent kinetic-energy k and one for

the turbulence dissipation rate, ϵ. The computed turbulent quantities are then used to calculate a turbulent viscosity ν_0.

The equations we solve are

$$\left.\begin{aligned} \mathbf{u}_t + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p - \nu_0\Delta\mathbf{u} - \mathbf{f} &= 0 \\ \Delta p - (\nabla u\cdot\mathbf{u}_x + \nabla v\cdot\mathbf{u}_y + \nabla w\cdot\mathbf{u}_z) - C\nabla\cdot\mathbf{u} - \nabla\cdot\mathbf{f} &= 0 \\ k_t + (\mathbf{u}\cdot\nabla)k - P - \epsilon - \nabla\cdot(\nu_k\nabla k) &= 0 \\ \epsilon_t + (\mathbf{u}\cdot\nabla)\epsilon - C_{\epsilon 1}(\epsilon/k)P + C_{\epsilon 2}(\epsilon^2/k) - \nabla\cdot(\nu_\epsilon\nabla\epsilon) &= 0 \end{aligned}\right\} x\in\Omega.$$

We are using a non conservative formulation for the viscous terms, for details see §3.4. The boundary- and the initial-conditions for k and ϵ can be written

$$\begin{aligned} B(k,\epsilon) &= 0 \qquad x\in\partial\Omega \\ k(x,0) &= k_0(x) \quad t=0 \\ \epsilon(x,0) &= \epsilon_0(x) \quad t=0 \end{aligned} \quad ,$$

The boundary conditions for the momentum components and the pressure are found in §3.2. The other constants in the equations above are defined by

ν_T	$=$	$C_\nu(k^2/\epsilon)$	turbulent eddy viscosity
ν_0	$=$	$\nu+\nu_T$	total viscosity
ν_k	$=$	$\alpha_k\nu_0$	viscosity coefficient for k
ν_ϵ	$=$	$\alpha_\epsilon\nu_0$	viscosity coefficient for ϵ
P	$=$	$\nu_T\sum_{ij}\partial_i u_j(\partial_i u_j+\partial_j u_i)$	turbulence production term

The standard values of the constants used above are

$$\begin{aligned} C_\mu &= 0.09 \\ C_{\epsilon 1} &= 1.45 \\ C_{\epsilon 2} &= 1.92 \\ \alpha_k &= 1. \\ \alpha_\epsilon &= 0.77 \end{aligned} \quad .$$

3.6 Boundary Conditions for k and ϵ

The turbulent quantities k and ϵ are treated analogously at the boundaries as the momentum components. At an inflow we impose

$$\left\{\begin{aligned} k &= k_0 \\ \epsilon &= \epsilon_0 \end{aligned}\right. .$$

where (k_0,ϵ_0) are functions of (y,z). On the wetted part of the ship hull, we apply a no slip boundary condition

$$\left\{\begin{aligned} k &= k_{min} \\ \epsilon &= \epsilon_{min} \end{aligned}\right. .$$

Where k_{min}, ϵ_{min} are small quantities $O(10^{-4})$, since ϵ has to be larger than some $\delta>0$ to prevent overflow.

The boundary condition at the symmetry-, surface-, bottom- and side-boundary is

$$\begin{cases} \partial k/\partial n &= 0 \\ \partial \epsilon/\partial n &= 0 \end{cases}.$$

At the outflow boundary we set

$$\begin{cases} \partial k/\partial n &= 0 \\ \partial \epsilon/\partial n &= 0 \end{cases}.$$

The k–ϵ model can be used with the boundary conditions as stated above or we can choose to use the wall law boundary condition instead of the boundary condition at a no slip boundary.

Wall law boundary condition

The boundary layers of the turbulent quantites k and ϵ close to a solid surface for example a ship hull are very thin for high Reynolds numbers. Instead of applying the boundary condition (3.6) on a solid surface and resolving the thin boundary layers it is possible to use a wall law that predicts the values of k and ϵ in the boundary layer. For a more thorough discussion and a derivation of the wall-law cf. [57] or [29]. The wall law is used to set the values of k and ϵ one grid step into the flow from the solid wall, it can also be used to set the momentum components at the same location. When the wall law is used to set both momentum components and the turbulent quantities, the first gridpoint is required to be situated between $y^+ = 30$ and $y^+ = 150$, for the wall law to be valid. We currently only use the wall law to set the values of the turbulent quantities, in this case the first point must be placed at $y^+ < 5$. We plan to use the wall law to set the momentum components at a no slip wall too.

For each point on the solid surface, we compute

$$u_\tau = \sqrt{\nu \frac{\partial u}{\partial n}},$$

and one grid step into the flow we set

$$k = C_\mu^{-1/2} \cdot u_\tau^2$$

$$\epsilon = \frac{u_\tau^3}{\kappa y^+}$$

where κ is the von Karman constant mentioned in §3.4, ($\kappa = 0.41$).

The most negative effect of the wall law is that it delays the separation on the ship hull and thus moves it further downstream. This causes the slip vortices to be generated at the wrong place. The flow in the slip vortex can be strongly affected by this effect. Therefore it would be interesting to be able to resolve at least the boundary layer of the momentum components. This is,

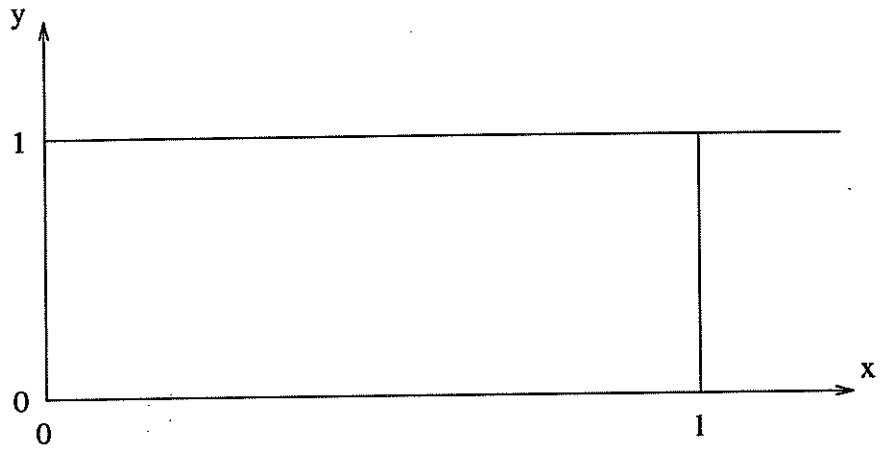


Figure 4: The domain of the modelproblem. We study the effect of the outflow boundary condition by comparing the solution on the halfplane domain ($0 \leq x$) with the solution on a bounded domain ($0 \leq x \leq l$).

however, a hopeless task with todays computers since the Reynolds number of a model scale flow is 10^6 and for a full scale flow 10^8

Discretization of the equations

The equations for k and ϵ are discretized to second-order accuracy (even when the equations for **u** are discretized to fourth-order accuracy). The discretization is done in the same manner as for the momentum equations, this is described in §5.

4 A Study of Outflow Boundary Conditions

In this section, we study the effects of imposing artificial inflow and outflow boundary conditions on a two dimensional model problem, such as the errors introduced by the inflow and outflow boundary conditions in the solution. The domain of the model problem can be found in figure 4, we require the solution to be 1–periodic in y. C. Johansson [30] has studied the effects of either an in- or an out-flow boundary condition on a halfplane problem. Here we study the problem in a finite truncated domain that has both in and outflow boundaries.

The model problem is the 2D incompressible Navier–Stokes' equations:

$$\left.\begin{aligned} u_t + u \cdot u_x + v \cdot u_y + p_x - \nu \Delta u - f_x &= 0 \\ v_t + u \cdot v_x + v \cdot v_y + p_y - \nu \Delta v - f_y &= 0 \\ u_x + v_y &= 0 \end{aligned}\right\} \quad \mathbf{x} \in \Omega \qquad (3)$$

we assume that $f_x = 0$ and $f_y = 0$.

The following inflow boundary condition is used for both the halfplane problem and the finite domain problem

$$\left.\begin{aligned} u &= u_1 \\ v &= v_1 \\ u_x + v_y &= 0 \end{aligned}\right\} \text{ at the inflow boundary, } x = 0. \tag{4}$$

Two similar sets of outflow boundary conditions are studied and compared to the halfplane problem. Boundary condition I is

$$\left.\begin{aligned} \frac{\partial^j u}{\partial x^j} &= 0 \\ \frac{\partial^j v}{\partial x^j} &= 0 \\ p &= 0 \end{aligned}\right\} \text{at the outflow boundary } x = l. \tag{5}$$

and boundary condition II is

$$\left.\begin{aligned} \frac{\partial^j u}{\partial x^j} &= 0 \\ \frac{\partial^j v}{\partial x^j} &= 0 \\ u_x + v_y &= 0 \end{aligned}\right\} \text{at the outflow boundary } x = l, \tag{6}$$

where j is an integer larger than or equal to zero.

We have found that the errors in the velocity components caused by the errors in the velocities at the boundary decrease with increasing j. From this respect boundary condition I is as good as boundary condition II. It is also seen that boundary condition II restricts the flow pattern less at the outflow than boundary condition.

4.1 A Study of the Effects of the Outflow Boundary Conditions

We shall now discuss how the solution to the halfplane problem changes when we introduce outflow boundaries. First, we linearize the model problem around a constant mean flow. Then we study the halfplane case where we only have an inflow boundary condition, cf. (4). The effect of two different outflow boundary conditions is discussed, cf. (5,6). The results clearly shows that these boundary conditions are appropriate when the flow has settled behind the body and that they introduce boundary layers when they are applied too far upstream.

To get an equation for the pressure we differentiate the first equation in (3) with respect to x and the second equation with respect to y and use $u_x + v_y = 0$

$$\Delta p + 2 \cdot (u_x^2 + u_y v_x) = 0,$$

We specify the domain Ω to be: $0 < x < l$, $0 < y < 1$, $t > 0$, cf. figure 4.

We linearize (3) around the constant mean flow, $(U, V) = (U_0, 0)$, this gives us the following equations

$$\left\{\begin{aligned} u_t + U_0 u_x - \nu(u_{xx} + u_{yy}) + p_x &= 0 \\ v_t + U_0 v_x - \nu(v_{xx} + v_{yy}) + p_y &= 0 \\ p_{xx} + p_{yy} &= 0 \end{aligned}\right. .$$

Discussions of the wellposedess of similar linearizations of (3) are performed by C. Johansson [30] and W. Henshaw [20].

To solve this problem, we use Fourier expansion in the y–direction. For the u velocity and the pressure we use a cos–transformation:

$$w(y) = \sum_{k=0}^{\infty} \hat{w}_k \cos(\omega_k y),\ \omega_k = k \cdot \pi,\ k = 0, 1, 2, \ldots,$$

and for the v velocity we use a sin–transformation:

$$w(y) = \sum_{k=0}^{\infty} \hat{w}_k \sin(\omega_k y),\ \omega_k = k \cdot \pi,\ k = 0, 1, 2, \ldots,$$

to simplify the analysis, we assume that $w(\omega_0) = 0$. Furthermore we apply the Laplace transform with respect to time t

$$w(t) = \int_{-\infty}^{\infty} e^{st} \hat{w}(s) ds,$$

and obtain the following system of equations

$$\left\{ \begin{array}{rcl} su + U_0 u_x - \nu(u_{xx} - \omega_k^2 u) + p_x &=& 0 \\ sv + U_0 v_x - \nu(v_{xx} - \omega_k^2 v) - \omega_k p &=& 0 \\ p_{xx} - \omega_k^2 p &=& 0 \end{array} \right. . \tag{7}$$

This is a system of three coupled ordinary differential equations. We determine the general solution by writing the pressure solution in the following way.

$$p = Ae^{\omega_k(x-l)} + Be^{-\omega_k x}, \tag{8}$$

Entering this expression into the equations for the velocity components (u, v), equation (7), yields:

$$\begin{array}{rcl} -\nu u_{xx} + U_0 u_x + (s + \nu\omega_k^2)u &=& -\omega_k\{Ae^{\omega_k(x-l)} - Be^{-\omega_k x}\} \\ -\nu v_{xx} + U_0 v_x + (s + \nu\omega_k^2)v &=& -\omega_k\{Ae^{\omega_k(x-l)} + Be^{-\omega_k x}\} \end{array} . \tag{9}$$

The solution, can be written as the sum of a homogeneous and a particular solution, for u we have $u = u_h + u_p$. We determine the particular solution by employing: $u_p = Ce^{\omega_k(x-l)} + De^{-\omega_k x}$, into equation (9). Similarly, for v, we can write $v = v_h + v_p$, where we choose: $v_p = Ee^{\omega_k(x-l)} + Fe^{-\omega_k x}$. This gives the following result for the constants of the particular solution

$$\left\{ \begin{array}{rcl} C &=& \frac{-A\omega_k}{U_0\omega_k + s} \\ D &=& \frac{B\omega_k}{-U_0\omega_k + s} \\ E &=& \frac{-A\omega_k}{U_0\omega_k + s} \\ F &=& \frac{-B\omega_k}{-U_0\omega_k + s} \end{array} \right. .$$

We find the homogeneous part of the solution by employing the following ansatz

$$\begin{pmatrix} u \\ v \\ p \end{pmatrix} = \sum_{Re\lambda_i<0} \sigma_i e^{\lambda_i x} \begin{pmatrix} u_i \\ v_i \\ p_i \end{pmatrix},$$

into (9). Hence, we get the following eigenvalue problem,

$$\begin{pmatrix} a_1 + b_1\lambda_i - \nu\lambda_i^2 & 0 & \lambda_i \\ 0 & a_1 + b_1\lambda_i - \nu\lambda_i^2 & \lambda_i \\ 0 & 0 & \lambda_i^2 - \omega_k^2 \end{pmatrix} \cdot \begin{pmatrix} u_j \\ v_j \\ p_j \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix},$$

where the constants a_1 and b_1 are given by

$$\begin{cases} a_1 &= \nu(\omega_k^2 + s) \\ b_1 &= U_0 \end{cases}.$$

The eigenvalues are given by

$$\begin{array}{rcl} \lambda_{1,2} &=& \frac{1}{2}\frac{b_1 \pm \sqrt{b_1^2 + 4\nu a_1}}{\nu} \\ \lambda_{3,4} &=& \lambda_{1,2} \\ \lambda_{5,6} &=& \pm\omega_k \end{array}. \tag{10}$$

Knowing the eigenvalues, we can write the expressions for the velocity components (u, v) on the form

$$\begin{array}{rcl} u &=& Ge^{\lambda_1(x-l)} + He^{\lambda_2 x} + \frac{-A\omega_k}{U_0\omega_k+s}e^{\omega_k(x-l)} + \frac{B\omega_k}{-U_0\omega_k+s}e^{-\omega_k x} \\ v &=& Ie^{\lambda_1(x-l)} + Je^{\lambda_2 x} + \frac{-A\omega_k}{U_0\omega_k+s}e^{\omega_k(x-l)} + \frac{-B\omega_k}{-U_0\omega_k+s}e^{-\omega_k x} \end{array}. \tag{11}$$

The half plane problem

First we solve the half plane problem, $0 \le x < \infty$, $0 \le y \le 1, t > 0$. In this case we need to specify a requirement to prevent the solution from growing infinitely when $x \to \infty$

$$\begin{pmatrix} u \\ v \\ p \end{pmatrix} < \begin{pmatrix} \infty \\ \infty \\ \infty \end{pmatrix}, \text{ when } x \to \infty.$$

This condition kills the growing terms in the expressions (8,9), hence, we get the following expressions for u, v, p

$$\begin{cases} u &= He^{\lambda_2 x} + \frac{B\omega_k}{-U_0\omega_k+s}e^{-\omega_k x} \\ v &= Je^{\lambda_2 x} + \frac{B\omega_k}{-U_0\omega_k+s}e^{-\omega_k x} \\ p &= Be^{-\omega_k x} \end{cases}. \tag{12}$$

By entering the expressions in (12) into the inflow boundary condition (4), we obtain the following expressions for the constants (B, H, J)

$$\left\{\begin{array}{lcl} B &=& (\frac{-U_0\omega_k+s}{\omega_k})(\frac{\lambda_2 u_1+\omega_k v_1}{\lambda_2+\omega_k}) \\ H &=& \frac{\omega_k(u_1-v_1)}{\omega_k+\lambda_2} \\ J &=& v_1-(\frac{\lambda_2 u_1+\omega_k v_1}{\lambda_2+\omega_k}) \end{array}\right. .$$

By substituting these expressions back into equation (12), we receive the solution, that is denoted (u_∞, v_∞ and p_∞)

$$\left\{\begin{array}{lcl} u_\infty(x) &=& \frac{\omega_k(u_1-v_1)}{\omega_k+\lambda_2}e^{\lambda_2 x}+(\frac{\lambda_2 u_1+\omega_k v_1}{\omega_k+\lambda_2})e^{-\omega_k x} \\ v_\infty(x) &=& \{v_1-(\frac{\lambda_2 u_1+\omega_k v_1}{\omega_k+\lambda_2})\}e^{\lambda_2 x}+(\frac{\lambda_2 u_1+\omega_k v_1}{\omega_k+\lambda_2})e^{-\omega_k x} \\ p_\infty(x) &=& (\frac{-U_0\omega_k+s}{\omega_k})(\frac{\lambda_2 u_1+\omega_k v_1}{\omega_k+\lambda_2})e^{-\omega_k x} \end{array}\right. . \qquad (13)$$

Outflow boundary condition I

To study how the outflow boundary conditions affect the solution, we now apply the following boundary conditions

$$\left.\begin{array}{rcl} \frac{\partial^j u}{\partial x^j} &=& 0 \\ \frac{\partial^j v}{\partial x^j} &=& 0 \\ p &=& 0 \end{array}\right\} \text{at the outflow boundary } x=l,$$

where j is an integer larger than or equal to zero. By inserting the ansatzes found in equations (8,11) into the boundary conditions we have the following equation system

$$\left\{\begin{array}{lllllllll} -\frac{A\omega}{U_0\omega+s}e^{-\omega l} & + & \frac{B\omega}{-U_0\omega+s} & + & Ge^{-\lambda l} & + & H & = & u_1 \\ -\frac{A\omega}{U_0\omega+s}e^{-\omega l} & - & \frac{B\omega}{-U_0\omega+s} & + & Ie^{-\lambda l} & + & J & = & v_1 \\ -\frac{A\omega^2}{U_0\omega+s}e^{-\omega l} & - & \frac{B\omega^2}{-U_0\omega+s} & + & G\lambda_1 e^{-\lambda l} & + & H\lambda_2 & = & -\omega u_1 \\ -\frac{A\omega^{j+1}}{U_0\omega+s} & + & \frac{B\omega^{j+1}(-1)^j}{-U_0\omega+s} & + & G\lambda_1^j & + & H\lambda_2^j e^{\lambda_2 l} & = & 0 \\ -\frac{A\omega^{j+1}}{U_0\omega+s} & + & \frac{B\omega^{j+1}(-1)^j}{-U_0\omega+s} & + & I\lambda_1^j & + & J\lambda_2^j e^{\lambda_2 l} & = & 0 \\ A & + & Be^{-\omega l} & & & & & = & 0 \end{array}\right.$$

Where the first order approximation of the solution is given by:

$$\left\{\begin{array}{lcl} u &=& u_\infty(x)+u_{err,u}\cdot e^{\lambda_1(x-l)}+u_{err,p}\cdot e^{\omega_k(x-2l)} \\ v &=& v_\infty(x)+v_{err,u}\cdot e^{\lambda_1(x-l)}+v_{err,p}\cdot e^{\omega_k(x-2l)} \\ p &=& p_\infty(x)+p_{err,p}\cdot e^{\omega_k(x-2l)} \end{array}\right. , \qquad (14)$$

where (u_∞,v_∞ and p_∞) denote the solution to the half plane problem. The other terms are errors introduced by errors in the boundary conditions. The unknown quantities in (14) are given by.

$$\left\{\begin{array}{lcll} u_{err,u} &=& -\frac{\omega_k^j}{\lambda_1^j}\left(\frac{\lambda_2 u_1+\omega_k v_1}{\lambda_2+\omega_k}\right) & (-1)^j \\ u_{err,p} &=& \frac{-U_0\omega_k+s}{U_0\omega_k+s}\left(\frac{\lambda_2 u_1+\omega_k v_1}{\lambda_2+\omega_k}\right) & e^{\omega l} \\ v_{err,u} &=& \frac{\omega_k^j}{\lambda_1^j}\left(\frac{\lambda_2 u_1+\omega_k v_1}{\lambda_2+\omega_k}\right) & (-1)^j \\ v_{err,p} &=& -\frac{-U_0\omega_k+s}{U_0\omega_k+s}\left(\frac{\lambda_2 u_1+\omega_k v_1}{\lambda_2+\omega_k}\right) & e^{\omega l} \\ p_{err,p} &=& -\frac{-U_0\omega_k+s}{\omega_k}\left(\frac{\lambda_2 u_1+\omega_k v_1}{\lambda_2+\omega_k}\right) & e^{-\omega l} \end{array}\right. .$$

Outflow boundary condition II

We now study a new set of outflow boundary conditions and how they affect the solution. These boundary conditions are when we introduce artificial outflow boundary conditions. For this purpose we consider equation (3).

$$\left.\begin{array}{rcl} \frac{\partial^j u}{\partial x^j} & = & 0 \\ \frac{\partial^j v}{\partial x^j} & = & 0 \\ u_x + v_y & = & 0 \end{array}\right\} \text{at the outflow boundary } x = l, \qquad (15)$$

where j is an integer larger than or equal to zero. Entering the expressions in equation (11) into the boundary conditions (4,15), gives the following equation system

$$\left\{\begin{array}{lllllllll} -\frac{A\omega}{U_0\omega+s}e^{-\omega l} & + & \frac{B\omega}{-U_0\omega+s} & + & Ge^{-\lambda l} & + & H & = & u_1 \\ -\frac{A\omega}{U_0\omega+s}e^{-\omega l} & - & \frac{B\omega}{-U_0\omega+s} & + & Ie^{-\lambda l} & + & J & = & v_1 \\ -\frac{A\omega^2}{U_0\omega+s}e^{-\omega l} & - & \frac{B\omega^2}{-U_0\omega+s} & + & G\lambda_1 e^{-\lambda l} & + & H\lambda_2 & = & -\omega u_1 \\ -\frac{A\omega^{j+1}}{U_0\omega+s} & + & \frac{B\omega^{j+1}(-1)^j}{-U_0\omega+s} & + & G\lambda_1^j & + & H\lambda_2^j e^{\lambda_2 l} & = & 0 \\ -\frac{A\omega^{j+1}}{U_0\omega+s} & + & \frac{B\omega^{j+1}(-1)^j}{-U_0\omega+s} & + & I\lambda_1^j & + & J\lambda_2^j e^{\lambda_2 l} & = & 0 \\ -\frac{A\omega^2}{U_0\omega+s} & - & \frac{B\omega^2}{U_0\omega+s}e^{-\omega l} & + & G\lambda_1 & + & H\lambda_2 e^{\lambda_2 l} & + & \\ \omega(\frac{-A\omega}{U_0\omega+s} & - & \frac{B\omega}{U_0\omega+s}e^{-\omega l} & + & I & + & Je^{\lambda_2 l}) & & \end{array}\right.$$

Where the first order approximation of the solution is given by:

$$\left\{\begin{array}{rcl} u & = & u_\infty(x) + u_{err,u} \cdot e^{\lambda_1(x-l)} + u_{err,p} \cdot e^{\omega_k(x-2l)} \\ v & = & v_\infty(x) + v_{err,u} \cdot e^{\lambda_1(x-l)} + v_{err,p} \cdot e^{\omega_k(x-2l)} \\ p & = & p_\infty(x) + p_{err,p} \cdot e^{\omega_k(x-2l)} \end{array}\right. , \qquad (16)$$

where (u_∞,v_∞ and p_∞) denote the solution to the half plane problem. The other terms are errors introduced by errors in the boundary conditions. The unknown quantities in (16) are given below. (Note that these are the leading terms in a first order approximation.)

$$\left\{\begin{array}{rcl} u_{err,u} & = & (\frac{2\omega_k^{j+1}}{\lambda_1^j})(-1)^j v_1) e^{-\omega_k l} \\ u_{err,p} & = & 0 \\ v_{err,u} & = & (\frac{2\omega_k^{j+1}}{\lambda_1^j})(-1)^j v_1) e^{-\omega_k l} \\ v_{err,p} & = & 0 \\ p_{err,p} & = & 0 \end{array}\right. .$$

We clearly see, that the error, due to velocity error on the boundary, in the velocity components decay as $1/\lambda_1^j$ for both boundary conditions I and II. This error has boundary layer character and decays fast upstream. For boundary condition II the error in the pressure is significantly smaller than for boundary condition I, for which the error, due to pressure error on the boundary, in all solution components is not affected by the order j of the boundary condition.

Thus, by increasing j we suppress the boundary layer in the velocity components. For boundary condition I, we observe that one has to specify the pressure at the outflow boundary to get it correctly. Since the pressure often is unknown, we recommend using boundary condition II with $j = 2$.

If we study the flow in a b units wide canal, the Fourier modes become $\omega_k = k \cdot \pi/b$, $k = 0, 1, 2, \ldots,$. Hence, for a wider canal the error, due to the error in the pressure at the outflow boundary, travels further upstream.

The results obtained for this model problem behave qualitatively as the solution for the Euler equations obtained by G. Kreiss, cf. [33], except for the boundary layers at the outflow.

Therefore a practical method to evaluate if the outflow boundary conditions are disturbing the solution would be to calculate the solution using two computational domains of different length. Check if the flow close to the outflow boundary for the shorter domain is similar to the flow at the same position for the larger domain. If this is the case, we could safely apply the outflow boundary condition for the shorter of these two domains.

For ship flow it is important to have an outflow boundary that lets out longitudinal slip vortices with varying pressure across the vortex. This indicates that bouncary condition II is better that boundary condition I.

5 Numerical Method

5.1 Discretizing the Geometry using a Composite Overlapping Grid

In this section we describe how we discretize the flow domain of interest using composite overlapping grids. This is done in three steps, which will be described in the subsections below.

1. In §5.1.1 we describe how a nonsingular representation of the ship hull surface is constructed by using a number of overlapping 2-dimensional patches.

2. In §5.1.2 we elaborate on how these body covering patches are used to generate component grids such that the union of these component grids cover the vicinity of the ship hull and overlap where they meet. Note that it is easy to generate these component grids so that they do not have singularities in their grid transformations.

3. In §5.1.3 we describe how a composite overlapping grid is constructed from a number of component grids, using the composite grid generator CMPGRD.

This method to discretize a domain is general and applies easily to a complex geometry.

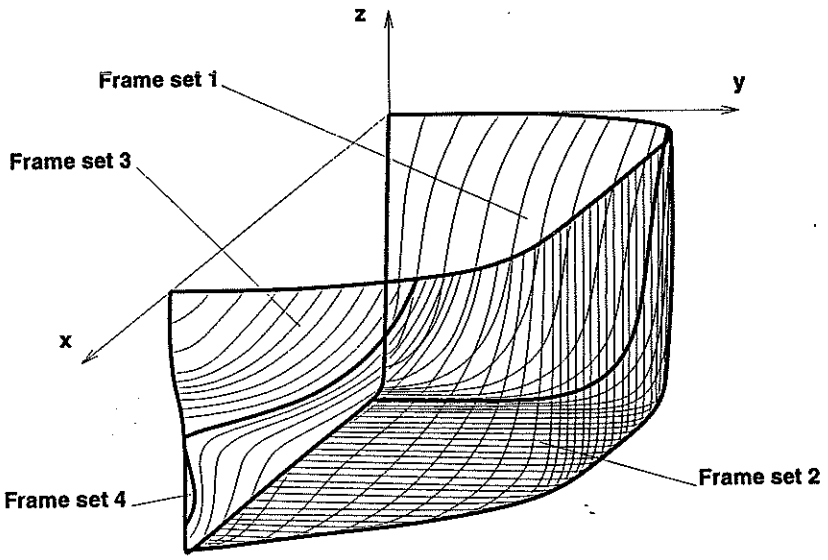


Figure 5: The standard method to describe a ship hull is here represented by four ship frame sets.

5.1.1 Discretizing the Surface of the Ship Hull

The standard method to represent the geometry of a ship hull is by several 2 dimensional patches since the topology of a cargo ship is generally too complicated to be described by a single patch. Each patch is topologically simple and contains a number of ship rib-frames. We consider a right handed Cartesian coordinate system with the x-axis directed in the opposite forward direction of the ship, the y-axis is directed sideways and the z-axis vertically upwards, cf. figure 3. A rib-frame consists of a number of (x, y, z) coordinates on the intersection between the ship hull and a constant-x plane. In figure 5, we show four surface patches – or frame sets that define a typical cargo ship hull. We use symmetry and only represent one half of the ship.

The set of rib-frames can be viewed as one or several topologically rectangular surface patches. We define each patch by its transformation from the unit square parameters to the physical coordinates. This mapping is exemplified in figure 6.

Removing patch singularities by the projection technique.

The rib-frame description often have polar and line singularities in the coordinate transformation. For the frame set in figure 6, there is a singularity at the side $s = 1$. The parameter side maps to a single point in physical space. These singularities can render a poor computational grid where special care often has to be taken when solving the equations. To avoid a typical singularity we define a nonsingular topologically rectangular patch. This patch is then projected onto a domain on the ship hull containing one or several singularities

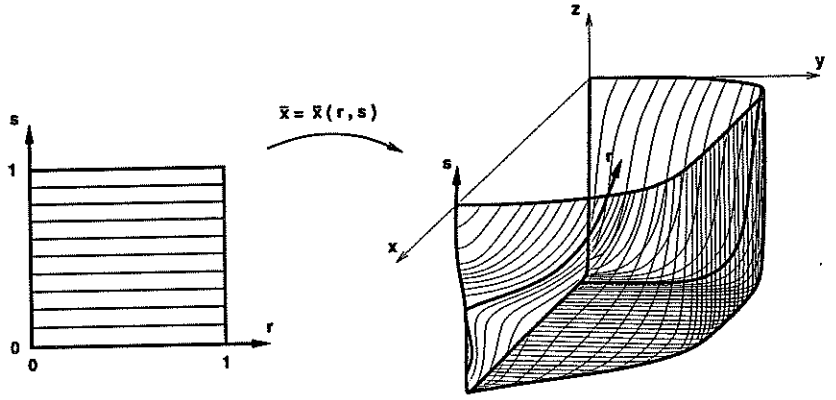


Figure 6: The transformation that defines a rib frame set. The transformation is singular at the line $s = 1$, which maps onto a point. This may render a poor computational grid.

using Newtons method in the parameter space. In figure 7, we demonstrate the projection technique by constructing a smooth representation of a surface that previously was described by a singular frame set.

The patches are projected using various map projection techniques. To get a smooth representation of the geometry it is advisable to take a natural coordinate system for the projection.

By this method we construct a set of topologically rectangular and nonsingular surface patches that overlap each other where they meet and the union of these patches cover the whole ship-hull. We can formally write the representation of the hull as:

$$\bar{x}_{disc} = \bar{x}_{disc}(r_i, s_j, k), \quad 0 \leq r_j \leq 1, \quad 0 \leq s_j \leq 1, \quad 0 \leq k \leq \text{nr. of patches} \quad (17)$$

where $\bar{x}_{disc} = (x, y, z)_{disc}$ denotes discrete representation of the physical coordinates of the hull, r_i is the parameter in each rib-frame. s_j is the frame parameter and k is the patch index. Here $0 \leq i \leq nr_i$, $0 \leq j \leq nr_j$ and $r_i = i/nr_i$ and $s_j = j/nr_j$.

To generate a continuous representation of the coordinates of the hull, we interpolate this discrete representation using a bicubic spline parametrization. This is done using the ESASPLINE package cf. [43]. For a concise treatment on the subject of spline interpolation cf. [15]. For surfaces with very large differences in curvature in the two main axes of curvature and/or very quickly varying curvature it may be necessary to use cubic-linear or linear-linear interpolation. The linear interpolation is used to avoid cusps in the description of the surface. This continuous representation of the hull can be written:

$$\bar{x}_{cont} = \bar{x}_{cont}(r, s, k), \quad 0 \leq r \leq 1, \quad 0 \leq s \leq 1, \quad 0 \leq k \leq \text{nr. of patches}$$

where r and s are the continuous versions of the variables in Eq. (17).

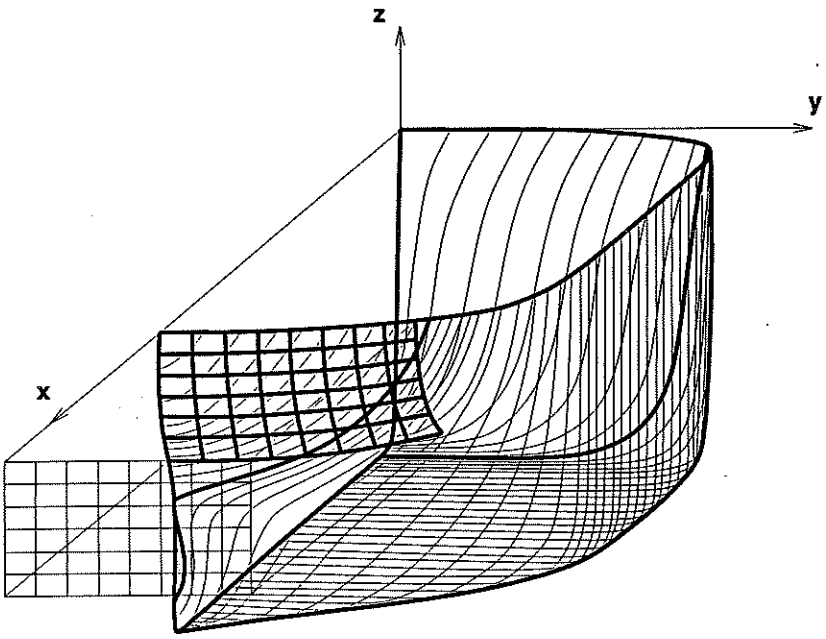


Figure 7: To avoid the transformation singularities we use a projection method. Here we project a nonsingular patch onto a singularity.

5.1.2 Construction of Component Grids that cover the Ship Hull.

As the next step we create one of the body covering component grids by using one of the ship hull patches as the body attached surface in the corresponding grid. The component grids can be generated by the use of various techniques, some of them will be described briefly here, for a more concise treatment cf. [55] or [25, 24].

Many useful techniques require the parameterization of at least two surfaces, that correspond to opposite sides of the grid. The boundary surface is fixed to the geometry of the body, but there is some freedom to specify the opposite surface. The outer surface can be generated by using normals to the surface of the ship hull. This technique enables simple and efficient discretization of the boundary conditions. Another technique is to copy the body covering surface and rescale this surface to construct an outer surface. The outer surface can also be constructed by using a simple surface with analytical geometry or by some kind of blending of known surfaces. The component grid is then generated using some suitable interpolation from the bounding surfaces. It can also be generated by the use of elliptic/hyperbolic grid generation, where some kind of field equation is solved and the solution is used to compute the distribution of gridpoints, for more details see the books [55] or [32]. A generic component grid covering a part of the vicinity of a ship hull is shown in figure 8. The component grid is now defined by the mapping from the unit cube parameters to the physical grid. We can write this continuous transformation.

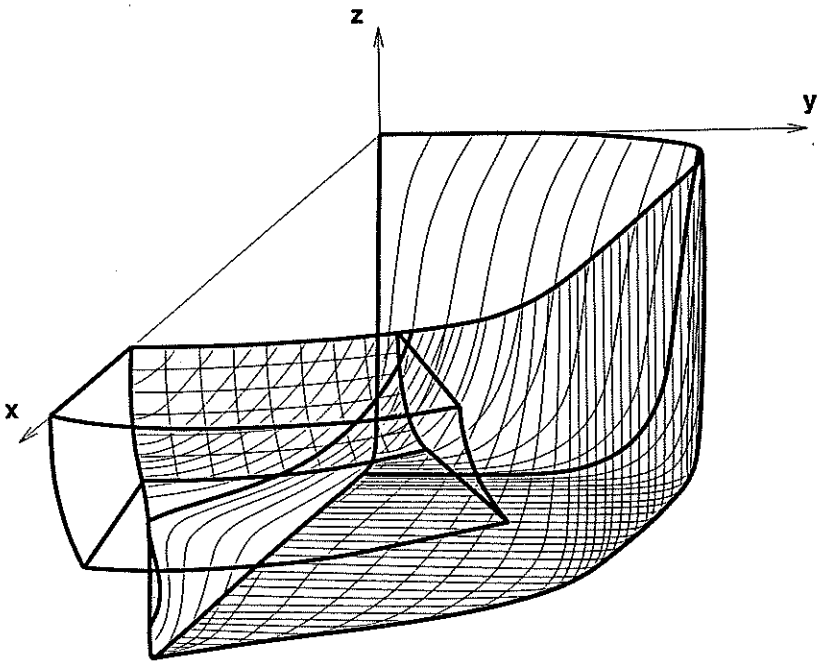


Figure 8: Construction of a component grid, originating from a ship hull patch. This component grid is constructed by transfinite interpolation between the surface on the hull and an outer surface. The outer surface is a rescaled and translated copy of the surface on the hull.

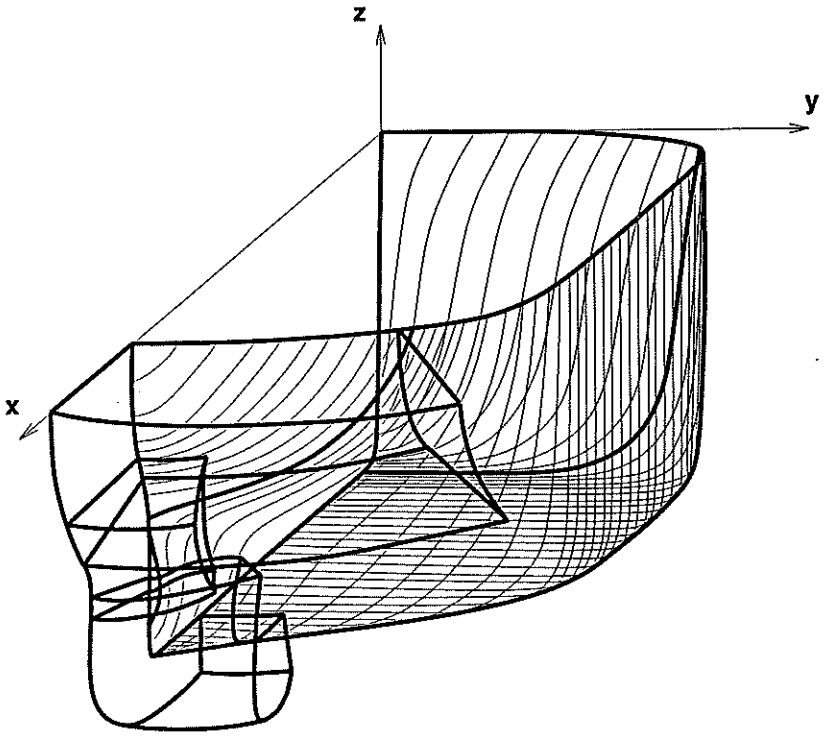


Figure 9: A set of overlapping component grids covering the propeller hub and the overhang of the stern.

$$\bar{x}_{k_1,cont} = \bar{x}_{k_1,cont}(r,s,t),\ 0 \le r \le 1,\ 0 \le s \le 1,\ 0 \le t \le 1,$$

where r, s, t are the grid parameters and k_1 is the component grid index, $1 \le k_1 \le$ nr. of component grids.

It is furthermore easy to generate these component grids so that they do not contain any singularities in the grid transformation, even for very complex geometries. In figure 9 we show a couple of component grids that cover a part of a ship stern.

In this fashion we construct a component grid for each one of the body covering patches. This results in a set of component grids that overlap eachother where they meet and cover the domain around the ship hull. To represent the outer flow domain we construct one or several background grids. The background grid can be e.g. a Cartesian grid, a cylindrical grid or a spherical grid.

5.1.3 Generating a Composite Overlapping Grid using CMPGRD

We then generate a composite overlapping grid using these curvilinear component grids and one or more background grids that covers the whole computational domain. For this purpose we employ the composite grid generator CMPGRD cf. [8, 24, 13]. Once all component grids have been defined, CMPGRD

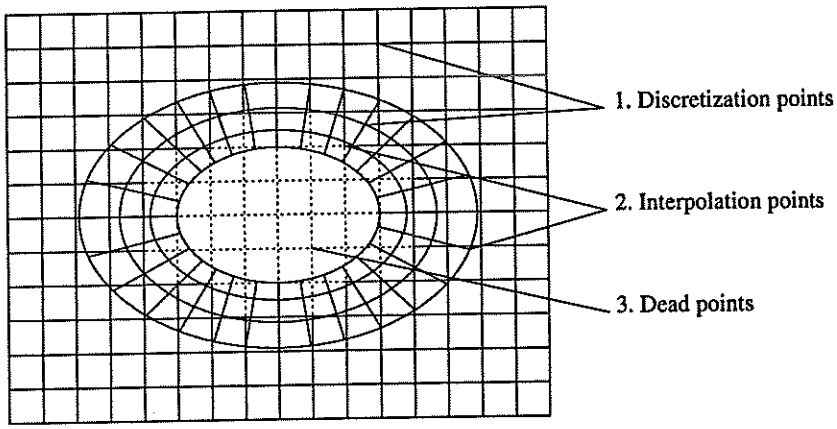


Figure 10: The three types of points in a composite grid.

automatically determines interpolation conditions, of desired order of accuracy, to connect the component grids. CMPGRD has a general algorithm for doing this, which can handle any number of component grids. It can generate the component grids appropriate for higher order interpolation, higher order discretization, cell-center or cell-vertex grids and the sequence of coarser grids needed for the multigrid algorithm [23]. There are three types of gridpoints in a composite overlapping grid.

1. Discretization points, (Inner or boundary points).

2. Interpolation points.

3. Dead, unused points.

The three types of points are exemplified in figure 10

The composite overlapping grid covers the whole computational domain and resolves the computational geometry – in our case a ship and the surrounding sea, cf. 11,12. It is also very simple to increase the resolution in any part of the flow in an efficient manner by adding gridlines where they are needed and/or by adding new component grids. A boundary layer is easily resolved by introducing the proper stretching in the curvilinear grids that attach to the body. A trailing vortex can be effectively resolved by adding a cartesian grid that covers the domain around the vortex. Furthermore a very complex geometry, as a propeller or a rudder can be described by a set of component grids which can be added to the composite grid describing the hull of the ship.

To efficiently resolve flow phenomena, varying in time, adaptive composite grids can be used, for example to compute unsteady trailing vortices. For a description of adaptive composite grids, cf. [5]. Unsteady flow around a moving geometry, eg. a ship with a rotating propeller or a moving rudder, can be computed using moving composite grids to discretize the geometry, for a description

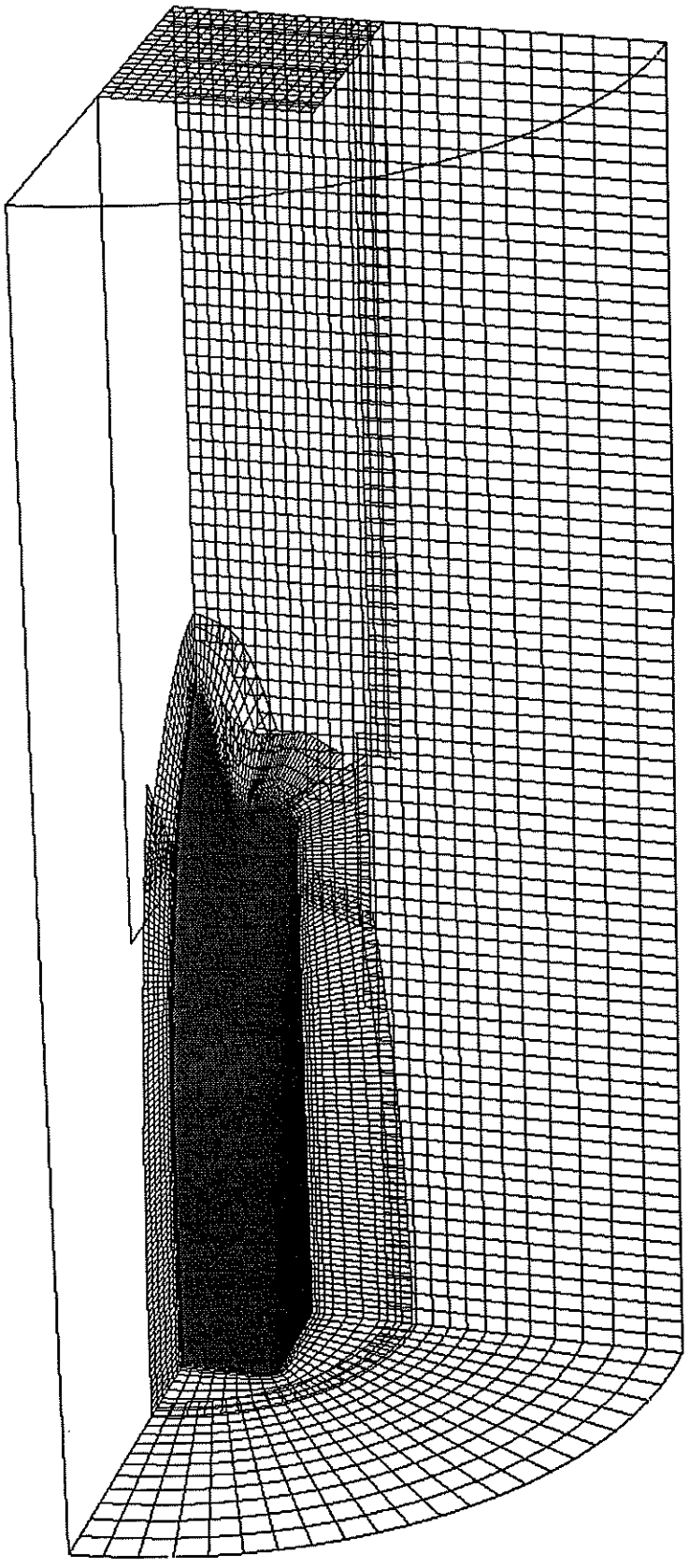


Figure 11: A composite grid covering a ship stern.

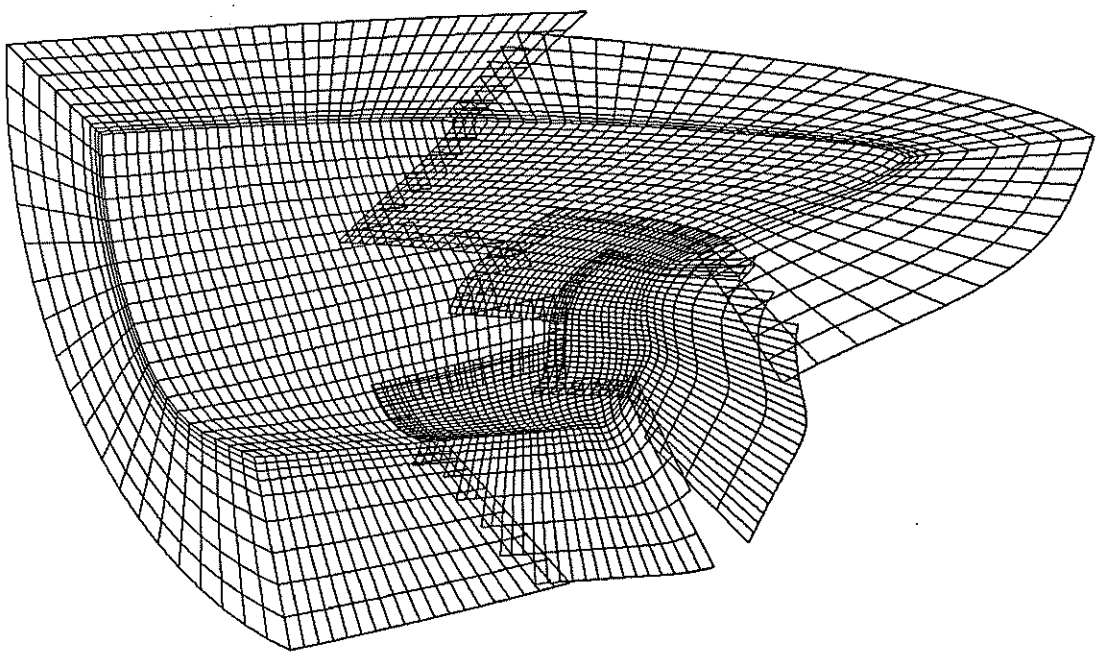


Figure 12: A composite grid covering a ship stern, blowup of the curvilinear component grids covering the ship stern.

of this technique cf. [46]. The two techniques of adaptive composite grids and moving composite grids, combined can be used to compute the nonlinear free surface flow, close to the water surface. For a description of a finite volume method that computes free surface flow around a ship cf. [17].

5.2 Discretizing the Equations on a Composite Overlapping Grid

In this section we elaborate on how the incompressible Navier–Stokes' equations are discretized on a composite overlapping grid. In §5.2.1 we describe the spatial discretization of the PDE. in some detail. We describe the extra numerical boundary conditions needed for fourth order accuracy in §5.2.2. The interpolation between the component grids is discussed in §5.2.3. We describe the artificial dissipation introduced in the momentum equations in §5.2.4. Furthermore in §5.2.5, we describe the timestepping scheme.

5.2.1 Spatial Discretization

We use the method of lines to discretize the incompressible Navier–Stokes' equations (2), ie. we discretize the system of PDEs in space which yields a large system of ODEs that can be written in the form

$$\frac{d\mathbf{U}}{dt} = \mathcal{F}(t, \mathbf{U}, \mathbf{P}),$$

where $\mathbf{U}$ is a vector of all solution values at all grid points. For the purpose of discussing time-stepping methods it is often convenient to think of the pressure as simply a function of $\mathbf{U}$, $\mathbf{P} = \mathcal{P}(\mathbf{U})$.

Let $\mathbf{U}_i$ and P_i denote the discrete approximations to $\mathbf{u}$ and p so that

$$\mathbf{U}_i \approx \mathbf{u}(\mathbf{x}_i) \quad , \quad P_i \approx p(\mathbf{x}_i) \; .$$

Here $\mathbf{U}_i = (U_{1i}, U_{2i}, U_{3i})$ and $i = (i_1, i_2, i_3)$ is a multi-index denoting the three coordinates. After discretizing in space, the equations (2), are of the form.

$$\left.\begin{array}{rcl} \frac{d}{dt}\mathbf{U}_i + (\mathbf{U}_i \cdot \nabla_h)\mathbf{U}_i + \nabla_h P_i - \nu\Delta_h \mathbf{U}_i - \mathbf{f}(\mathbf{x}_i, t) &=& 0 \\ \Delta_h P_i - \sum_m \nabla_h U_{m,i} \cdot D_{m,h}\mathbf{U}_i - C_d(\nu)\, \mathcal{V}_i \nabla_h \cdot \mathbf{U}_i - \nabla_h \cdot \mathbf{f}(\mathbf{x}_i, t) &=& 0 \end{array}\right\} \quad \mathbf{x} \in \Omega$$

$$\left.\begin{array}{rcl} B(\mathbf{U}_i, P_i) &=& 0 \\ \nabla_h \cdot \mathbf{U}_i &=& 0 \end{array}\right\} \quad \mathbf{x}_i \in \partial\Omega_h$$

$$\mathbf{U}(\mathbf{x}_i, 0) = \mathbf{U}_0(\mathbf{x}_i) \qquad \text{at } t = 0$$

where

$$\mathcal{V}_i = \frac{1}{\Delta x_{1,i}^2} + \frac{1}{\Delta x_{2,i}^2} + \frac{1}{\Delta x_{3,i}^2}$$

The subscript "h" denotes a second or fourth-order centered finite difference approximation,

$$D_{m,h} \approx \frac{\partial}{\partial x_m} \quad , \quad \nabla_h = (D_{1,h}, D_{2,h}, D_{3,h}) \quad , \quad \Delta_h \approx \sum_m \frac{\partial^2}{\partial x_m^2}$$

Extra numerical boundary conditions are also added, see [22] for further details. An artificial diffusion term is added to the momentum equations. This is described in §5.2.4.

Approximation of the derivatives by using the mapping method

The derivatives with respect to (x, y, z) are approximated by transforming them to the computational parameters (r, s, t) by use of the chain rule. For simplicity, we exemplify this with the first and second order derivatives with respect to (x), on component grid k.

$$\begin{aligned}
\left(\frac{\partial}{\partial x}\right)_k &= \frac{\partial r_k}{\partial x}\frac{\partial}{\partial r_k} + \frac{\partial s_k}{\partial x}\frac{\partial}{\partial s_k} \\
\left(\frac{\partial^2}{\partial x^2}\right)_k &= (\frac{\partial r_k}{\partial x})^2 \frac{\partial^2}{\partial r_k^2} + 2\frac{\partial r_k}{\partial x}\frac{\partial s_k}{\partial x}\frac{\partial^2}{\partial r_k \partial s_k} \\
&+ (\frac{\partial s_k}{\partial x})^2 \frac{\partial^2}{\partial s_k^2} + \frac{\partial^2 r_k}{\partial x^2}\frac{\partial}{\partial r_k} + \frac{\partial^2 s_k}{\partial x^2}\frac{\partial}{\partial s_k}
\end{aligned}$$

On the discrete side, we have the first and second order central finite differences.

$$\begin{aligned}
D^0_{m,x_k} &= \frac{\partial r_k}{\partial x} D^0_{m,r_k} + \frac{\partial s_k}{\partial x} D^0_{m,s_k} \\
D^+_{m,x_k} D^-_{m,x_k} &= (\frac{\partial r_k}{\partial x})^2 D^+_{m,r_k} D^-_{m,r_k} + 2\frac{\partial r_k}{\partial x}\frac{\partial s_k}{\partial x} D^0_{m,r_k} D^0_{m,s_k} \\
&+ (\frac{\partial s_k}{\partial x})^2 D^+_{m,s_k} D^-_{m,s_k} + \frac{\partial^2 r_k}{\partial x^2} D^0_{m,r_k} + \frac{\partial^2 s_k}{\partial x^2} D^0_{m,s_k}
\end{aligned}$$

The other differences are mapped in an analogous way.
Δ_{m+}, Δ_{m-} are the forward and backward undivided difference operators in parameter direction m.

$$\begin{aligned}
|\nabla_h \mathbf{U}_i|_1 &= n_d^{-2} \textstyle\sum_{m=1}^{n_d} \sum_{n=1}^{n_d} |D_{m,h} U_{ni}| \\
\Delta_1^+ \mathbf{U}_i &= \mathbf{U}_i(r_1 + 1, r2) - \mathbf{U}_i(r_1, r2) \\
\Delta_1^- \mathbf{U}_i &= \mathbf{U}_i(r_1, r2) - \mathbf{U}_i(r_1 - 1, r2) \\
\Delta_2^+ \mathbf{U}_i &= \mathbf{U}_i(r_1, r2 + 1) - \mathbf{U}_i(r_1, r2) \\
\Delta_2^- \mathbf{U}_i &= \mathbf{U}_i(r_1, r2) - \mathbf{U}_i(r_1, r2 - 1) \quad \text{etc.}
\end{aligned}$$

The discrete differences are examplified by the second order accurate discretization.

$$D^0_{2,r_k} \mathbf{U}_i = (\Delta_1^+ \mathbf{U}_i - \Delta_1^- \mathbf{U}_i)/(2dr)$$

The other differences are defined analogously.
As stated above, the component grids are defined by the transformation from the parameter space (r, s, t), where $0 < r, s, t < 1$, to the physical space. This transformation can be used to extract the physical coordinates of a gridpoint (x, y, z) and the derivatives of the transformation with respect to the physical coordinates, e.g. $(\partial r/\partial x,\ \partial r/\partial y,\ \partial r/\partial z,\ \partial s/\partial x)$ and higher order derivatives, if needed. This information is provided by the composite grid generator CMPGRD.

5.2.2 Numerical Boundary Conditions

The numerical boundary conditions are applied on extra lines of ghostcells, one line of ghostcells is required for second order accurate discretization and two lines for fourth order. Numerical boundary conditions are derived by applying the momentum equations and the pressure equation on the boundary. The normal derivative of the divergence is also specified on the boundary. The numerical boundary conditions are:

$$\left\{ \begin{array}{rcl} \mathbf{u}_t + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p = \nu\Delta\mathbf{u} + \mathbf{f} & = & 0 \\ \Delta p + \nabla u\cdot\mathbf{u}_x + \nabla v\cdot\mathbf{u}_y + \nabla w\cdot\mathbf{u}_z - \nabla\cdot\mathbf{f} & = & 0 \quad \mathbf{x}\in\partial\Omega\,. \\ \frac{\partial}{\partial n}(\nabla\cdot\mathbf{u}) & = & 0 \end{array} \right. \tag{18}$$

On the second line of ghostcells the pressure and the tangential components of the velocity are extrapolated. The second-order accurate version of this method does not require these extra numerical boundary conditions. The boundary conditions are discretized analogously as the PDE.

5.2.3 Interpolation between the Component Grids

The grid functions on the different component grids are coupled by using sufficiently accurate interpolation formulas. The function values of the unknown variables of a component grid k_1, that overlaps another component grid k_2 are interpolated from the gridfunction on grid k_2. The interpolation is carried out in the unit cube parameters, (r, s, t) of grid k_2 using a Lagrangian interpolation formula of sufficiently high order. In one dimension, the interpolation formula can be written.

$$f(r) = \sum_j \prod_{j\neq i} \frac{r - r_j}{r_i - r_j} f(r_i), \tag{19}$$

we denote $\Phi_i(x) = \prod_{j\neq i} \frac{r-r_j}{r_i-r_j}$

We write biquadratic interpolation in 2 dimensions using this notation, see figure 13.

$$f(r, s) = \sum_i \sum_i \Phi_i(r)\Phi_j(s) f(r_i, s_j), \tag{20}$$

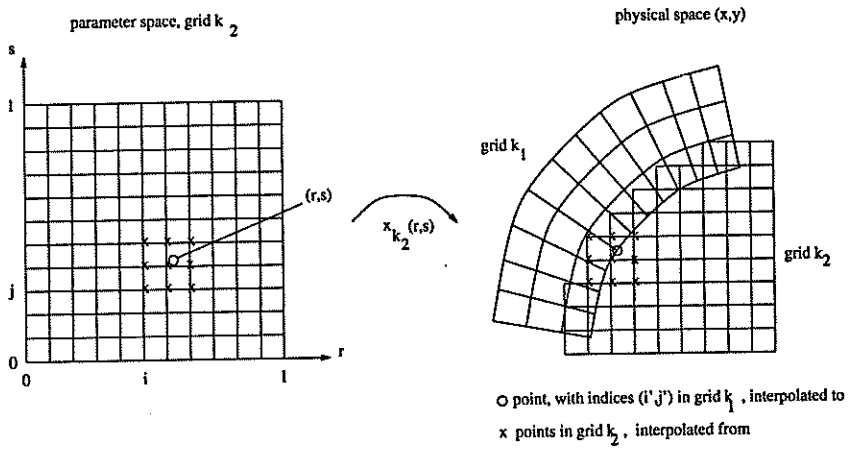


Figure 13: The interpolation from grid k_2 to grid k_1 is performed in the parameter space of grid k_2.

Chesshire et al. cf. [14] has shown how to choose the order of interpolation to obtain the same overall accuracy as for the discretization of the PDE. For a second order accurate discretization of the PDE, in 3 dimensions it is necessary to use a tri-cubic interpolation. Furthermore, for a fourth order accurate discretization of the PDE. in 3 dimensions a tri-quintic interpolation formula is required. These conditions require a larger overlap for higher order methods which results in stronger restrictions on how the component grids can be constructed. For a more detailed discussion cf. [13]. The subject of conservative interpolation is treated in [14].

5.2.4 Artificial Dissipation

For the second order accurate discretization we use an artificial diffusion based on a second-order undivided difference. In the fourth-order case the artificial diffusion is based on a fourth-order undivided difference. This type of artificial viscosity is based on the results on least scales of incompressible fluid flow by W. D. Henshaw, L. Reyna and H. O. Kreiss [26]. The idea is that if the constants in the expressions for the artificial dissipation, (ad21 and ad22 or ad41 and ad42), are properly chosen, the size of the smallest scale in the solution could be controlled.

In the second order case the artificial diffusion is:

$$\mathbf{d}_{2,i} = (\mathtt{ad21} + \mathtt{ad22}|\nabla_h \mathbf{U}_i|_1) \sum_{m=1}^{n_d} \Delta_{m+}\Delta_{m-}\mathbf{U}_i$$

and in the fourth order case:

$$\mathbf{d}_{4,i} = -(\mathtt{ad41} + \mathtt{ad42}|\nabla_h \mathbf{U}_i|_1) \sum_{m=1}^{n_d} \Delta^2_{m+}\Delta^2_{m-}\mathbf{U}_i.$$

Here $|\nabla_h \mathbf{U}_i|_1$ is the magnitude of the gradient of the velocity.

The artificial diffusion is introduced in the momentum equations according to:

$$\frac{d}{dt}\mathbf{U}_i + (\mathbf{U}_i \cdot \nabla_h)\mathbf{U}_i + \nabla_h P_i - \nu \Delta_h \mathbf{U}_i - \mathbf{f}(\mathbf{x}_i, t) - \mathbf{d}_{m,i} = 0$$

however, it does not alter the pressure equation. The artificial diffusion does not affect the order of accuracy for the method. This is described in detail in [22].

5.2.5 Time Stepping

To discretize the large system of ODE's obtained by the method of lines an efficient Adams PC Time-Stepping scheme is used. The second-order Adams predictor-corrector method uses fewer evaluations of the pressure equation than of the momentum equations. Given the solution and its time derivative at time t, $(\mathbf{U}(t), \mathbf{F}(t))$, the solution and time derivative at the next time $(\mathbf{U}(t + \Delta t), \mathbf{F}(t + \Delta t))$ are determined from the following modified Adams predictor-corrector method

$$\begin{aligned}
\mathbf{U}^* &= \mathbf{U}(t) + \Delta t\,(\mathbf{F}(t)) \\
\mathbf{P}^* = \mathcal{P}(\mathbf{U}^*) \quad &, \quad \mathbf{F}^* = \mathcal{F}(t + \Delta t, \mathbf{U}^*, \mathbf{P}^*) \\
\mathbf{U}(t + \Delta t) &= \mathbf{U}(t) + \Delta t \left(\frac{1}{2}\mathbf{F}^* + \frac{1}{2}\mathcal{F}(t, U, P) \right) \\
\mathbf{P}(t + \Delta t) = \mathbf{P}^* \quad &, \quad \mathbf{F}(t + \Delta t) = \mathcal{F}(t + \Delta t, \mathbf{U}(t + \Delta t), \mathbf{P}^*)
\end{aligned}$$

Note that when computing $\mathbf{F}(t+\Delta t)$ the pressure is fixed to be the predicted value $\mathbf{P}^*$. $\mathbf{P}^*$ is still a second order accurate approximation to $\mathbf{P}(t + \Delta t)$ so the overall method is second order. We have observed that the stability of this new method is unchanged since the pressure terms acts to damp the divergence and it is not necessary in stabilizing the momentum equations.

5.3 The Second/Fourth Order Accurate Solver CGINS

In this section, we give a brief description of the equation solver CGINS, used in this work and some of the experiences we have had using the code. Specifically §5.3.1 contains a general description of the solver and references to further information about this program. Following in §5.3.2 we describe how the program solves the Poisson equation for the pressure equation. Then, in §5.3.3 the various time stepping schemes available are discussed.

5.3.1 General Description

CGINS is a program for the solution of the incompressible Navier–Stokes' equations on regions that have been discretized with a compressible overlapping grid created by the grid generation program CMPGRD [8]. The spatial discretization of the equations is described in §5.2.1, further details of the method of

discretization used by CGINS are described in [22]. A discussion of solving PDEs on overlapping grids is given in [13], including a description of the way that CGINS interpolates between component grids. CGINS use the DSK subroutines, cf. [12] to manage its data structures.

The code can solve problems in two and three space dimensions to second and fourth order accuracy. Two turbulence models are currently available, a Baldwin-Lomax model and a k–ϵ model. A number of time stepping procedures are also available, this will be described in more detail in §5.3.3.

CGINS solves the incompressible Navier–Stokes' equations written in the pressure divergence form, analogous to equation (3)

$$\left.\begin{aligned} \mathbf{u}_t + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p - \nu\Delta\mathbf{u} - \mathbf{f} &= 0 \\ \Delta p - (\nabla u\cdot\mathbf{u}_x + \nabla v\cdot\mathbf{u}_y + \nabla w\cdot\mathbf{u}_z) - C_d(\nu)\nabla\cdot\mathbf{u} - \nabla\cdot\mathbf{f} &= 0 \end{aligned}\right\} \quad \mathbf{x}\in\Omega \quad (21)$$

$$\left.\begin{aligned} B(\mathbf{u},p) &= \mathbf{0} \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned}\right\} \quad \mathbf{x}\in\partial\Omega$$

$$\mathbf{u}(\mathbf{x},0) = \mathbf{u}_0(\mathbf{x}) \qquad \text{at } t = 0$$

There are n_d boundary conditions, $B(\mathbf{u},p) = \mathbf{0}$, where n_d is the number of space dimensions. On a no-slip wall, for example, $\mathbf{u} = \mathbf{0}$. For numerical solution, an extra boundary condition is required for the pressure. The continuity equation is added as an implicit boundary conditions for the pressure. The boundary condition $\nabla\cdot\mathbf{u} = 0$ is added. With this extra boundary condition it follows that the above problem is equivalent to the formulation with the Poisson equation for the pressure replaced by $\nabla\cdot\mathbf{u} = 0$ everywhere. The term $C_d(\nu)\nabla\cdot\mathbf{u}$ appearing in the equation for the pressure is used to damp the divergence [22].

5.3.2 Solving the Pressure Equation

CGINS solves the pressure equation (21) by using the Composite Grid Elliptic equation Solver, CGES, cf. [21]. CGES is an interface to a number of standard linear algebra package solvers, among others: a Yale sparse matrix direct solver, which is efficient for 2-dimensional problems, and iterative solvers as: Bi-conjugate gradient squared cf. [52, 16] and GMRES, [51] for 3-dimensional problems. For large 3D problems the Composite Grid Multigrid Solver, CGMG. cf. [23] can also be used.

We have found the restarted GMRES-iterative method, utilized with an Incomplete LU preconditioner, to be the most effective method for our computations. In the beginning of a run GMRES needs a large number of iterations to compute the pressure in each timestep, but when the solution is converging to steady state, the pressure changes slower and slower. GMRES therefore converges to the pressure solution faster and faster during a run, since it uses the pressure solution from the previous timestep as an initial guess for the next timestep. In the end of the convergence phase GMRES needs $O(3)$ iterations to compute the pressure. Furthermore it is not necessary to compute the pressure field every timestep, we have experimentally found that it is sufficient to update the pressure every fifth timestep in the end of a run.

5.3.3 Temporal Discretization

Currently there are three time-stepping procedures available.

- A fourth-order Runge-Kutta method.
- A second-order accurate predictor-corrector method, cf. §5.2.5.
- A second-order mid-point rule.

We have used the second order accurate predictor corrector method. The midpoint rule is unstable for purely hyperbolic problems but is good if there is dissipation.

The time-step is chosen automatically. The time step is chosen so that

$$\Delta t = \texttt{cfl}/\lambda_{\max}$$

where $\lambda_{\max}$ is a bound on the largest eigenvalue of the discrete time stepping matrix normalized by the size of the stability region. With this definition for $\lambda_{\max}$ a choice of $\texttt{cfl} = 1$ should usually be appropriate.

6 Numerical Results

6.1 Computational Examples

In this section, we describe the computational examples, that has been chosen for validation and testing of the numerical method. The testing consists of accuracy tests and "real flow" computations. For this task four test geometries have been chosen, one 2D geometry, cf. §6.1.1, and three 3D geometries of increasing complexity in §6.1.1–6.2.1.

The accuracy tests validate the order of accuracy of the method. This is carried out for the test geometries described in §6.1.1–6.1.2. These accuracy tests are performed by adding a forcing term to the equations. The forcing can be chosen so that the continuous solution is a desired function. We choose this function to be a second order polynomial in all coordinate directions. By using this forcing in the computation of the solutions for a number of different gridstep-sizes we can see if the order of accuracy in the numerical solution obtained corresponds to the order of accuracy of the discretization scheme. In all test cases, we have found that the numerical scheme has the desired order of accuracy.

"Real flow" computations are performed for all of the four test geometries in §6.1.1–6.2.1. The results from these computations are displayed in the corresponding sections.

In §6.1.1, we present results for a simple 2D test geometry - a circle in a box. In §6.1.2 we present results for a 3D test geometry, consisiting of a cylinder mounted to a wall. Furthermore in §6.1.3 we present results for a simple 3D model stern geometry and finally in §6.2.1 we present results for a 3D HSVA tanker stern geometry.

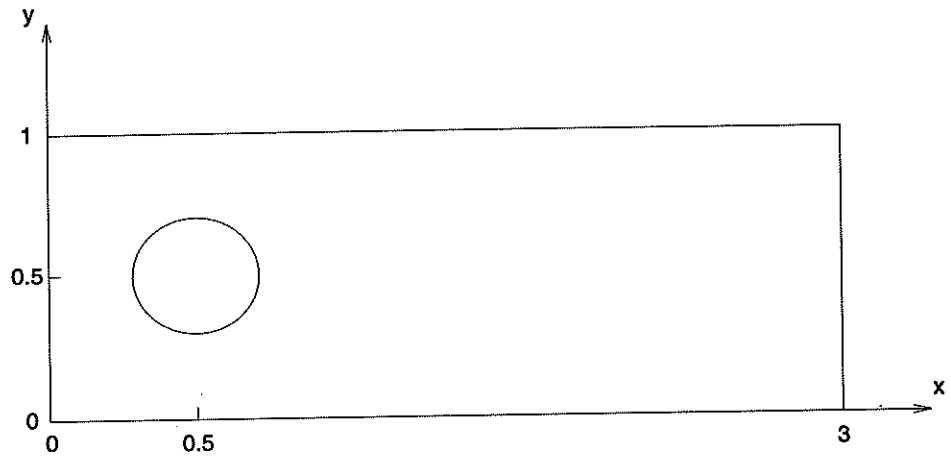


Figure 14: Test geometry 1: a circle in a rectangle.

mounted to a wall. Furthermore in §6.1.3 we present results for a simple 3D model stern geometry and finally in §6.2.1 we present results for a 3D HSVA tanker stern geometry.

6.1.1 A Simple 2D Test Geometry

As a test geometry for 2D flow we have chosen a circle in a box, see figure 14, for accuracy tests the height of the box is 1.0, the length 1.0 and the radius $r = 0.2$. The accuracy tests for this geometry are presented in tables 1–4. We have tested laminar and turbulent flow using both second and fourth order accurate discretization. The accuracy tests indicate that the solution has the correct order of accuracy. Note that the k and ϵ equation are discretized to second order accuracy also when the momentum equations are discretized to fourth order accuracy.

For the 2D "real flow" computations, we use the large box showed in figure 14, the following boundary conditions are used for 2D flow. We set the divergence to be zero at all walls. At the upper and lower walls, we impose symmetry boundary conditions.

$$\begin{cases} \partial u/\partial n &= 0 \\ \partial v/\partial n &= 0 \end{cases} \tag{22}$$

At the circle no slip boundary conditions.

$$\begin{cases} u &= 0 \\ v &= 0 \end{cases} \tag{23}$$

At the inflow, we prescribe:

$$\begin{cases} u &= u_0 \\ v &= 0 \end{cases} \tag{24}$$

Nr. of gridpts.	erru	errv	errp
$14 \times 14 + 25 \times 4$	0.023	0.014	0.074
quota	4.04	4.37	4.11
order	2.01	2.13	2.04
$27 \times 27 + 49 \times 7$	0.0057	0.0032	0.018
quota	4.38	4.44	4.39
order	2.13	2.14	2.13
$53 \times 53 + 97 \times 13$	0.0013	0.00072	0.0041

Table 1: Accuracy test: 2-dimensional laminar flow around a circle, second order accurate discretization

Nr. of gridpts.	erru	errv	errp	errk	erreps
$14 \times 14 + 25 \times 4$	0.015	0.014	0.076	0.020	0.082
quota	3.85	5.19	3.80	3.77	4.32
order	1.95	2.37	1.93	1.92	2.11
$27 \times 27 + 49 \times 7$	0.0039	0.0027	0.020	0.0053	0.019
quota	3.90	3.55	3.64	3.53	3.80
order	1.96	1.83	1.85	1.83	1.93
$53 \times 53 + 97 \times 13$	0.0010	0.00076	0.0055	0.0015	0.0059

Table 2: Accuracy test: 2-dimensional turbulent flow around a circle, second order accurate discretization

and for the outflow boundary we set:

$$\begin{cases} u_{xx} &= 0 \\ v_x &= 0 \\ p &= p_0 \end{cases} \tag{25}$$

We compute the "real flow" around the circle for a number of different laminar and turbulent viscosities. A comparison between experiments and computed flows for low Reynolds numbers is displayed in figures 16–23. The corresponding part of the composite grid used for the calculations is found in figure 15. We observe good agreement between the measurements and the calculated flows.

Nr. of gridpts.	erru	errv	errp
$25 \times 25 + 39 \times 8$	0.0019	0.0012	0.30
quota	27	21.82	10.0
order	4.75	4.45	3.33
$49 \times 49 + 77 \times 15$	0.000070	0.000055	0.031
quota	23.33	27.50	9.68
order	4.54	4.78	3.27
$97 \times 97 + 173 \times 29$	0.0000031	0.0000020	0.0032

Table 3: Accuracy test: 2-dimensional laminar flow around a circle, fourth order accurate discretization

Nr. of gridpts.	erru	errv	errp	errk	erreps
$25 \times 25 + 39 \times 8$	0.0013	0.0012	0.17	0.0056	0.020
quota	50.98	18.46	4.15	3.73	3.64
order	5.67	4.20	2.06	1.91	1.86
$49 \times 49 + 77 \times 15$	0.000051	0.000065	0.041	0.0015	0.0055
quota	21.25	32.50	9.53	3.33	3.23
order	4.41	5.02	3.25	1.74	1.70
$97 \times 97 + 173 \times 29$	0.0000024	0.0000020	0.0043	0.00045	0.0017

Table 4: Accuracy test: 2-dimensional turbulent flow around a circle, fourth order accurate discretization

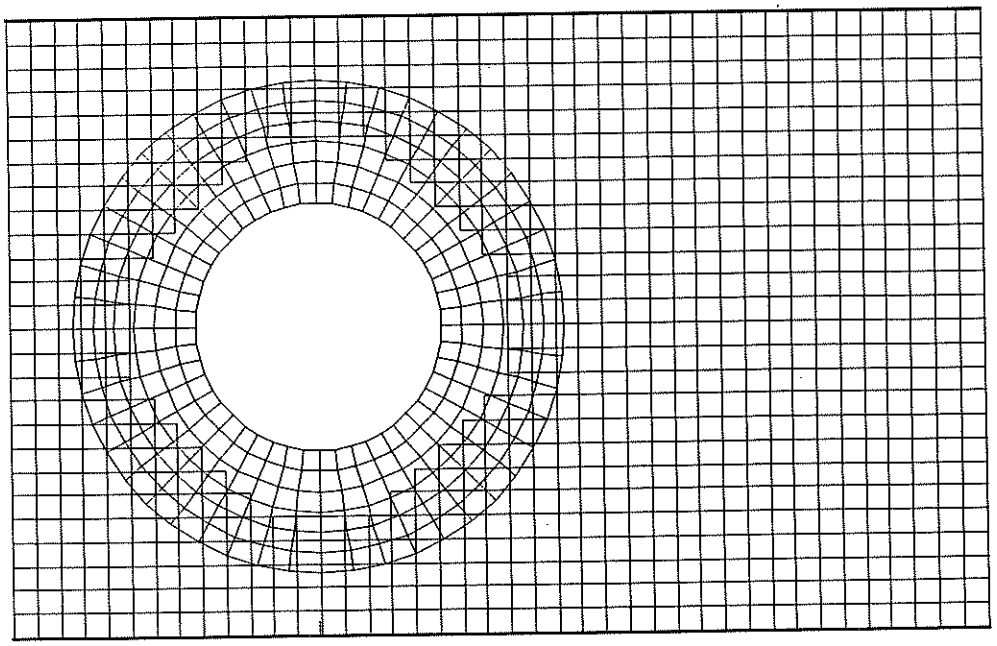


Figure 15: Composite grid for comparison between experiments and computations for laminar 2D flow around a circle.

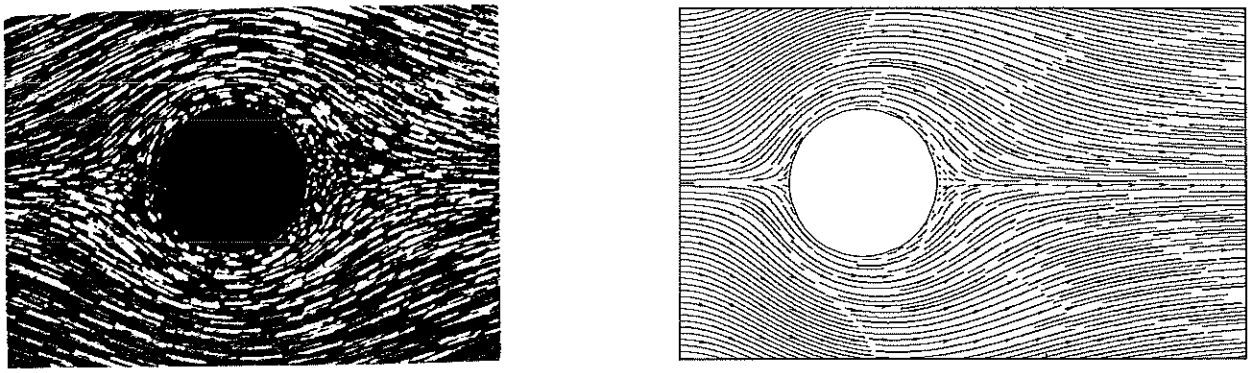


Figure 16: Comparison between experiments and computations $Re = 0.25$.

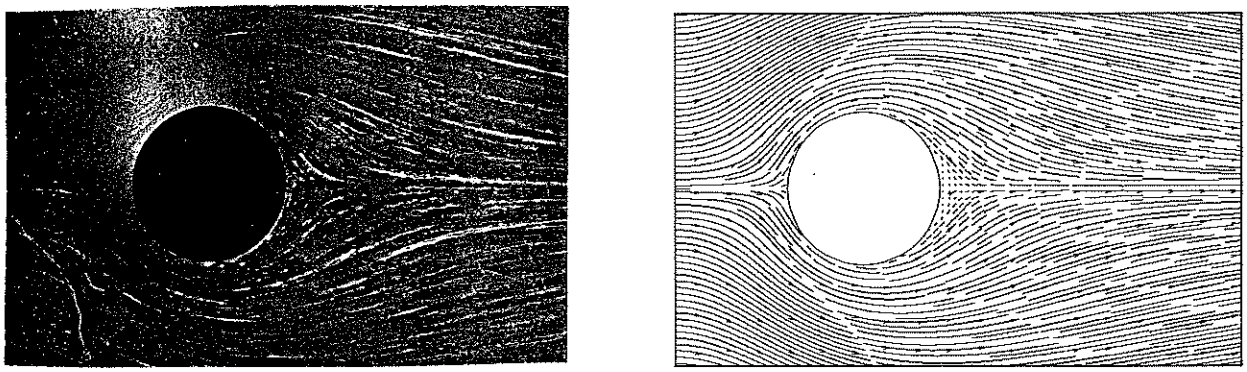


Figure 17: $Re = 3.64$.

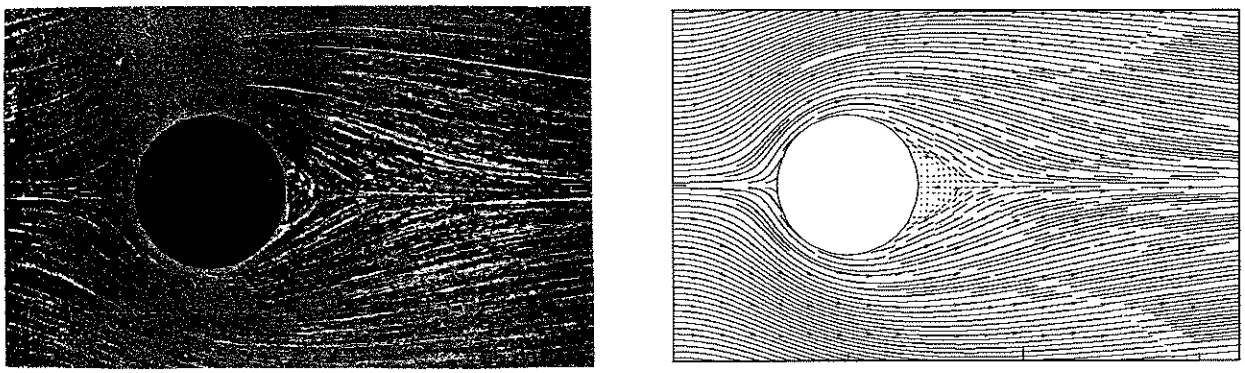


Figure 18: $Re = 9.10$.

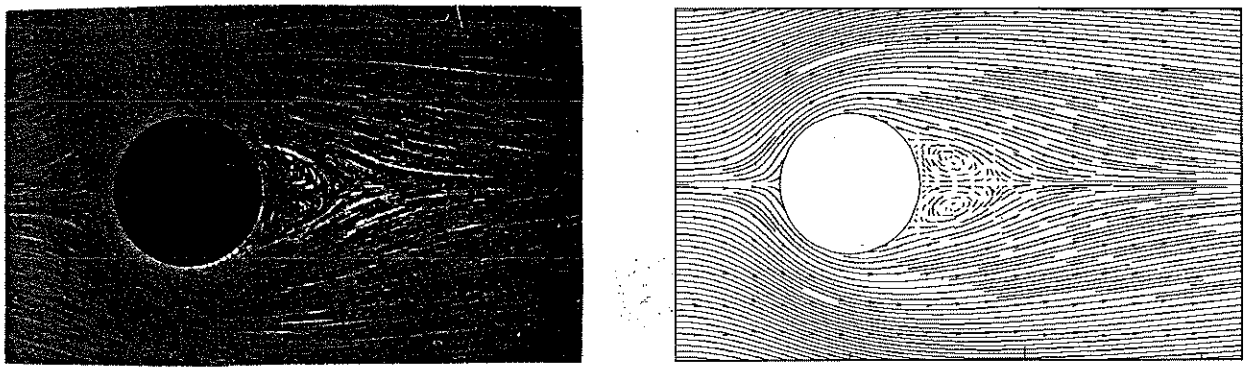


Figure 19: $Re = 13.09$.

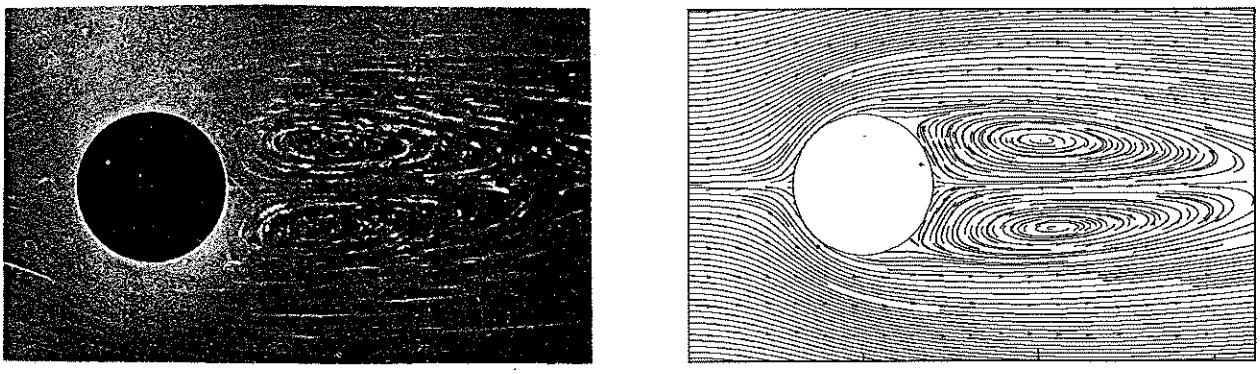


Figure 20: $Re = 39.9$.

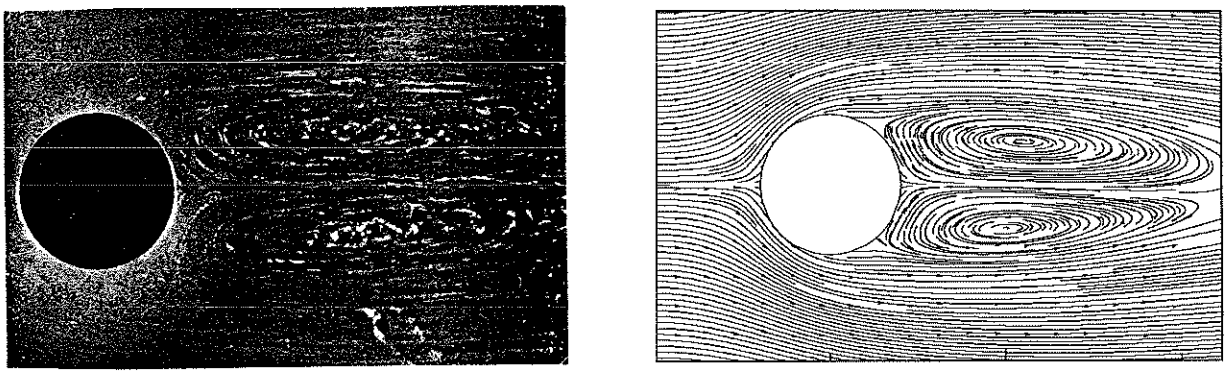


Figure 21: $Re = 57.7$.

Figure 22: Experimental flow at $Re = 65.0$.

We compare the solutions obtained by using the Baldwin Lomax– and the k–ϵ–turbulence model in figures 24–35. The Baldwin Lomax turbulence model gives a stronger wake.

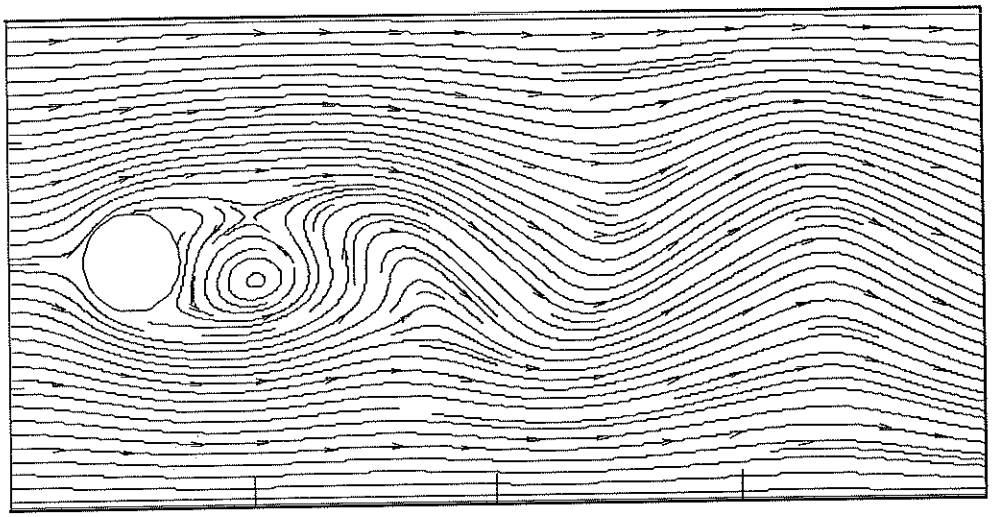


Figure 23: Numerically computed flow at $Re = 65.0$.

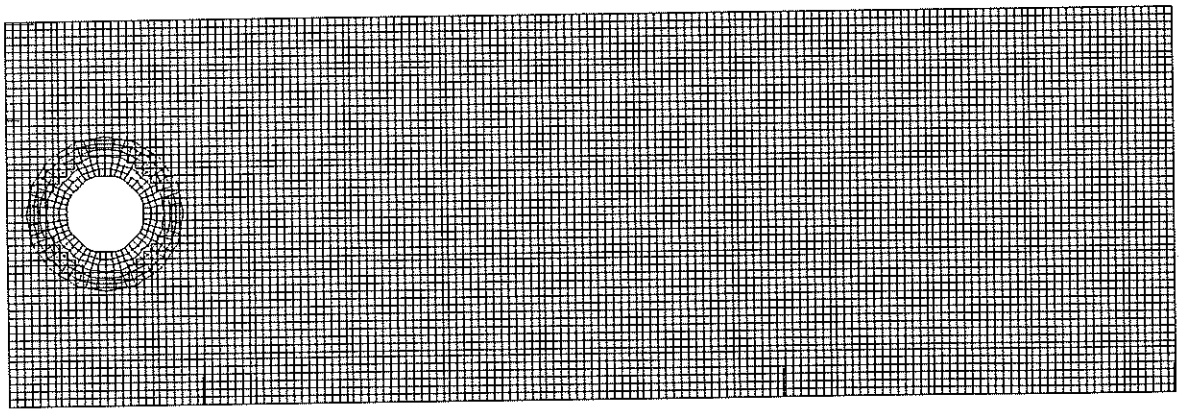


Figure 24: Composite grid, second order accurate discretization

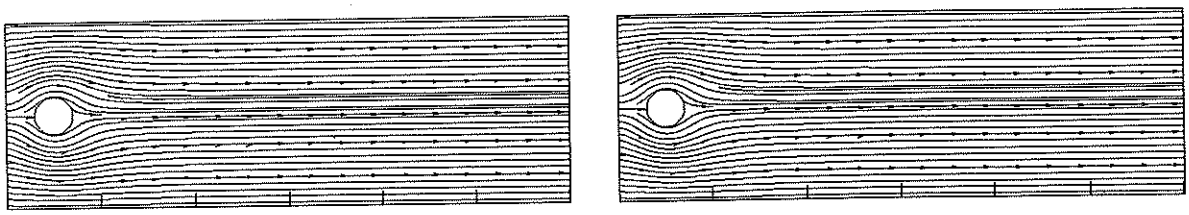


Figure 25: Baldwin Lomax / k–ϵ –turbulence model, $t = 0.0$

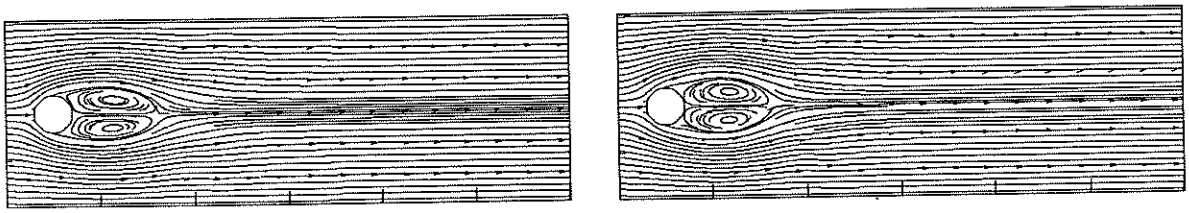


Figure 26: Baldwin Lomax / k–ϵ –turbulence model, $t = 1.5$

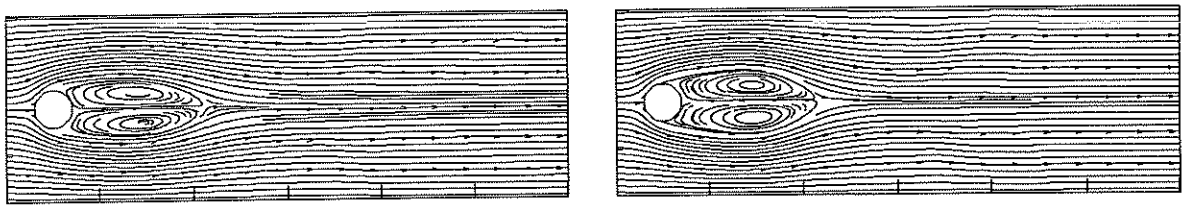


Figure 27: Baldwin Lomax / k–ϵ –turbulence model, $t = 3.0$

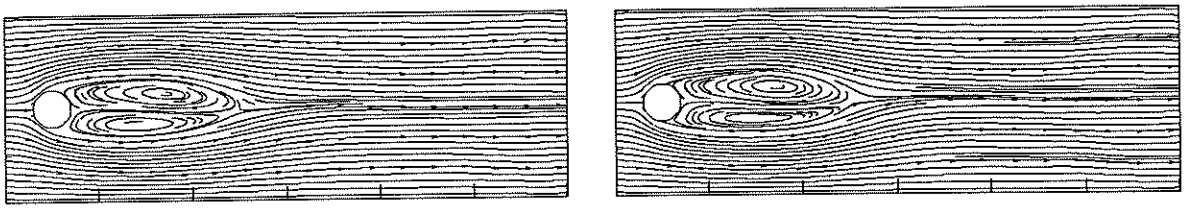


Figure 28: Baldwin Lomax / k–ϵ –turbulence model, $t = 4.5$

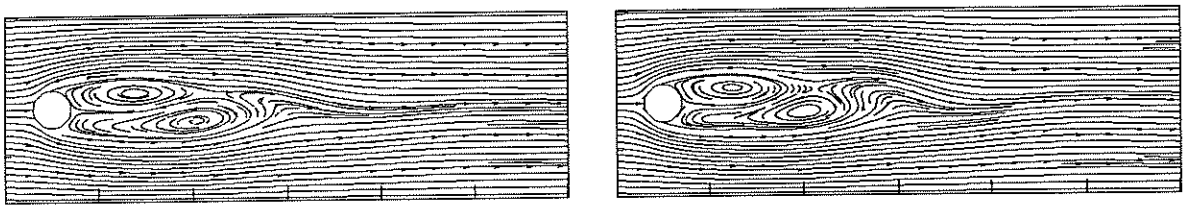


Figure 29: Baldwin Lomax / k–ϵ –turbulence model, $t = 6.0$

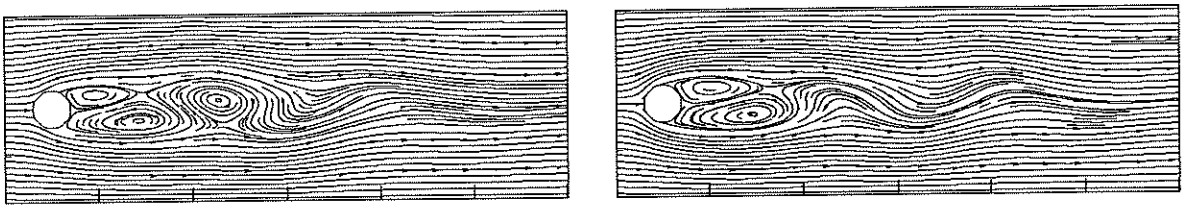


Figure 30: Baldwin Lomax / k–ϵ –turbulence model, $t = 7.5$

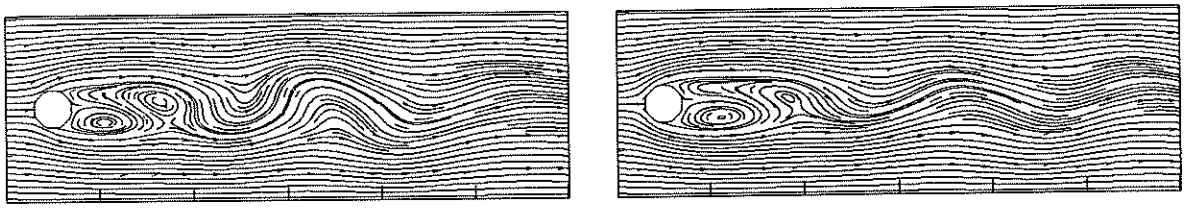


Figure 31: Baldwin Lomax / k–ϵ –turbulence model, $t = 9.0$

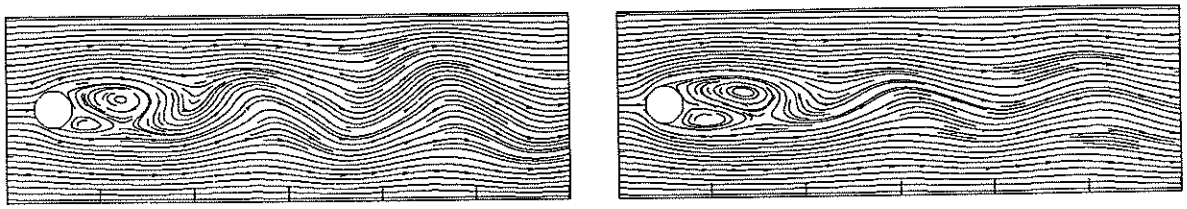


Figure 32: Baldwin Lomax / k–ϵ –turbulence model, $t = 10.5$

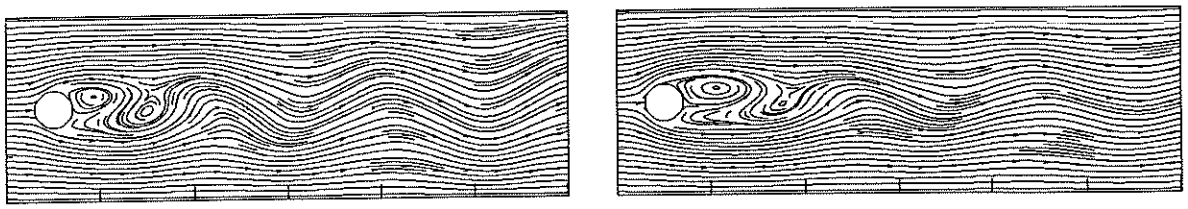


Figure 33: Baldwin Lomax / k–ϵ –turbulence model, $t = 12.0$

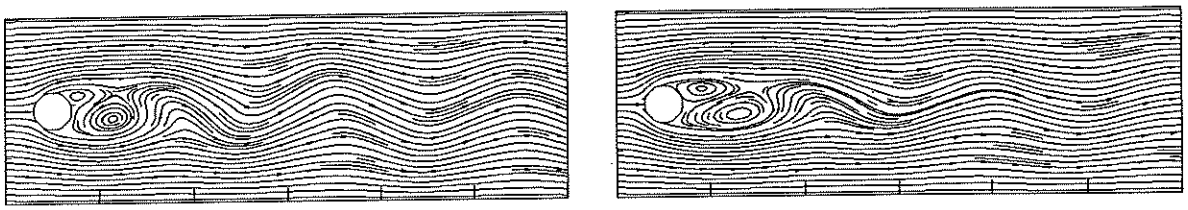


Figure 34: Baldwin Lomax / k–ϵ –turbulence model, $t = 13.5$.

We test the fourth order solution by comparing with a second order solution, computed using roughly the same grid. The difference between the fourth and the second order grids is that the second order grid has less interpolation overlap and only one row of ghostcells instead of two. The grid is found in figure 36 and the solutions in figure 37. Enlargements of the circle and the wake area in these two figures are found in figures ??. We observe that the differences between the solutions are small. For the fourth order accurate solution, the recirculation bubbles have sharper "corners" at the separation points and at the point where the wake rejoins downstream. Furthermore the center of the recirculation bubbles are further from the symmetry line. These obsevatins indicate that the fourth order accurate solution have at least slightly higher accuracy. This is especially useful in 3D computations, since memory requirements often are severe.

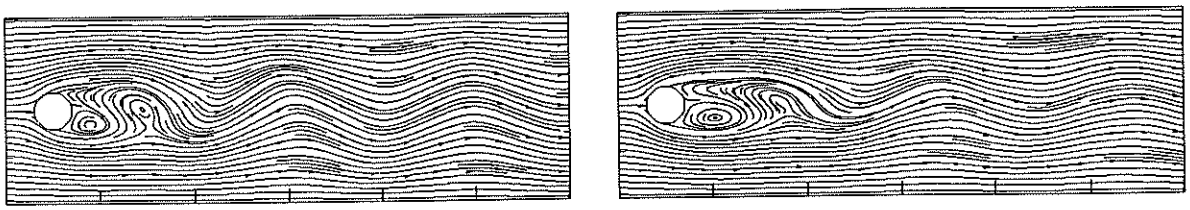


Figure 35: Baldwin Lomax / k–ϵ –turbulence model, $t = 15.0$

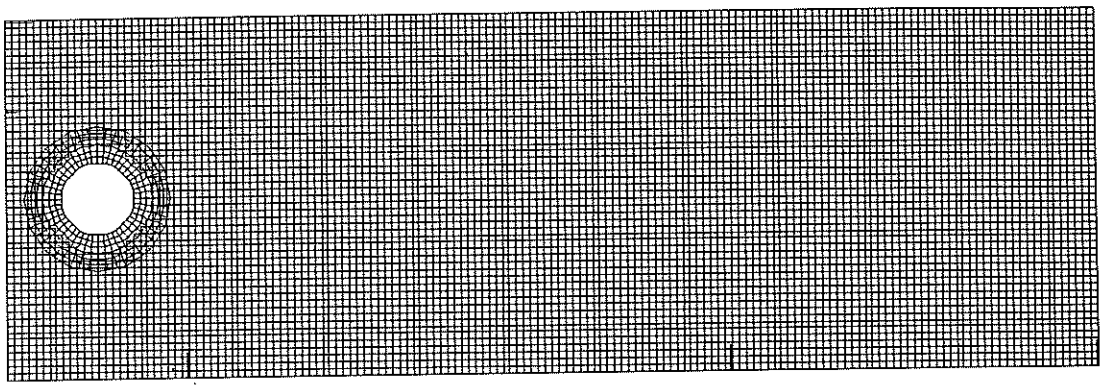


Figure 36: Composite grid, second order accurate discretization

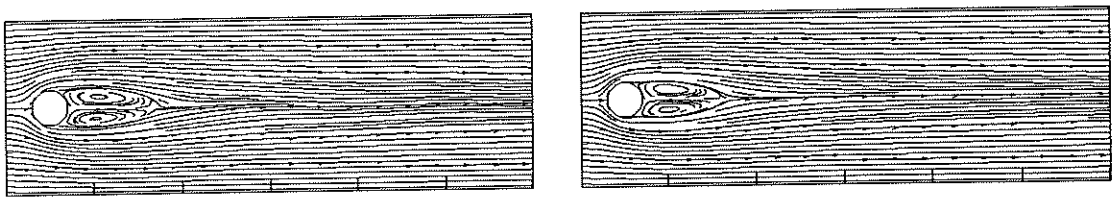


Figure 37: Comparison between second / fourth order accurate discretization

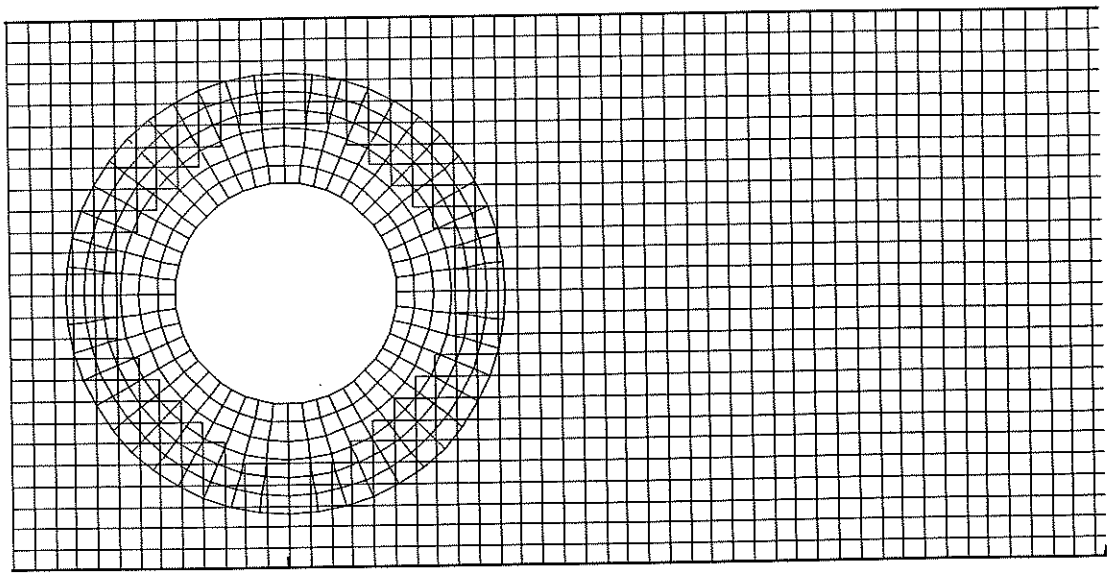


Figure 38: Enlargement of the composite grid above

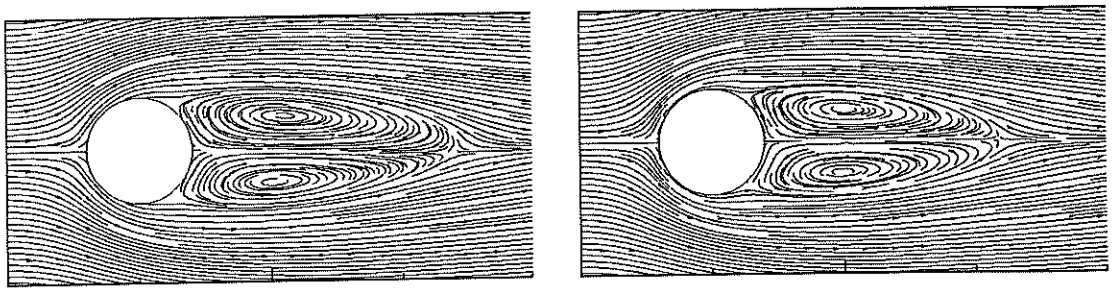


Figure 39: Enlargement of the comparison between second / fourth order accurate discretization

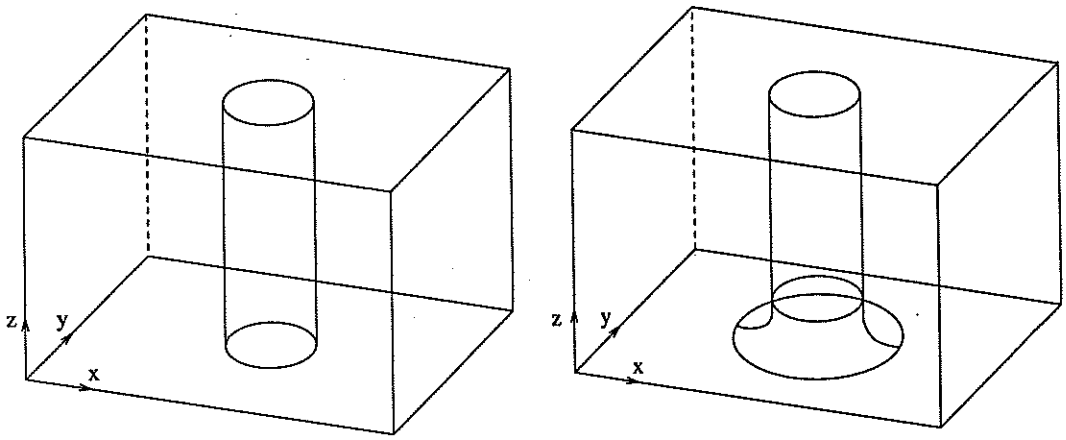


Figure 40: Test geometry 2: a) a cylinder in a box, b) a cylinder in a box smoothly fitted to the wall.

6.1.2 A 3D Cylinder

As a simple example 3D geometry to validate and check the solver, we have chosen a cylinder mounted to a wall in a box, see figure 40. The side length of the box is 1.0 and the radius of the cylinder is $r = 0.15$. We study both a cylinder mounted sharply to a wall and a cylinder mounted smoothly by the use of a collar - like fillet. The smooth mounting is chosen for demonstration of some of the capabilities of composite overlapping grids. It is quite simple to create an accurate discretization of the mounting of a wing to a body by using this technique.

We have performed accuracy tests computing both laminar and turbulent flow, using both second order and fourth order accurate discretization of the equations. These results can be found in tables (5–6) and they indicate that the solution has the correct order of accuracy. The tests are performed as described in §6.1.1.

Nr. of gridpts.	erru	errv	errw	errp
$21 \times 21 \times 17 + 9 \times 25 \times 9$	0.12	0.18	0.45	0.13
quota	2.67	3.75	6.08	1.81
order	2.83	3.81	5.21	1.71
$29 \times 29 \times 24 + 12 \times 35 \times 12$	0.045	0.048	0.074	0.072
quota	1.96	1.92	1.80	2.12
order	1.94	1.88	1.70	2.17
$41 \times 41 \times 33 + 17 \times 49 \times 17$	0.0229	0.025	0.041	0.034

Table 5: Accuracy test: Laminar 3-dimensional flow around a cylinder, 2nd order accurate discretization

Nr. of gridpts.	erru	errv	errw	errp	errk	erreps
$21 \times 21 \times 17 + 9 \times 25 \times 9$	0.068	0.093	0.051	0.27	0.34	0.70
quota	1.74	2.33	1.76	1.80	1.62	2.80
order	1.60	2.44	1.63	1.70	1.39	2.97
$29 \times 29 \times 24 + 12 \times 35 \times 12$	0.039	0.040	0.029	0.15	0.21	0.25
quota	1.90	2.11	1.81	1.79	2.10	1.92
order	1.85	2.15	1.71	1.68	2.14	1.88
$41 \times 41 \times 33 + 17 \times 49 \times 17$	0.020	0.019	0.016	0.084	0.10	0.13

Table 6: Accuracy test: Turbulent 3-dimensional flow around a cylinder, second order accurate discretization

Nr. of gridpts.	erru	errv	errw	errp
$20 \times 20 \times 18 + 10 \times 25 \times 10$	0.14	0.14	0.19	0.39
quota	4.52	4.38	6.78	3.00
order	4.35	4.26	5.52	3.17
$28 \times 28 \times 25 + 14 \times 35 \times 14$	0.031	0.032	0.028	0.13
quota	3.44	3.56	4.24	3.42
order	3.56	3.66	4.17	3.55
$39 \times 39 \times 35 + 19 \times 49 \times 19$	0.0091	0.0091	0.0066	0.038

Table 7: Accuracy test: Laminar 3-dimensional flow around a cylinder, fourth order accurate discretization

Nr. of gridpts.	erru	errv	errw	errp	errk	erreps
$21 \times 21 \times 17 + 9 \times 25 \times 9$	0.071	0.090	0.053	0.24	0.31	0.60
quota	3.76	3.56	3.72	3.18	1.64	2.40
order	3.82	3.67	3.79	3.34	1.43	2.53
$29 \times 29 \times 24 + 12 \times 35 \times 12$	0.019	0.025	0.0014	0.075	0.19	0.25
quota	3.86	3.77	3.61	2.93	2.09	1.93
order	3.90	3.83	3.71	3.10	2.13	1.90
$41 \times 41 \times 33 + 17 \times 49 \times 17$	0.0049	0.0066	0.00039	0.026	0.089	0.13

Table 8: Accuracy test: Turbulent 3-dimensional flow around a cylinder, fourth order accurate discretization

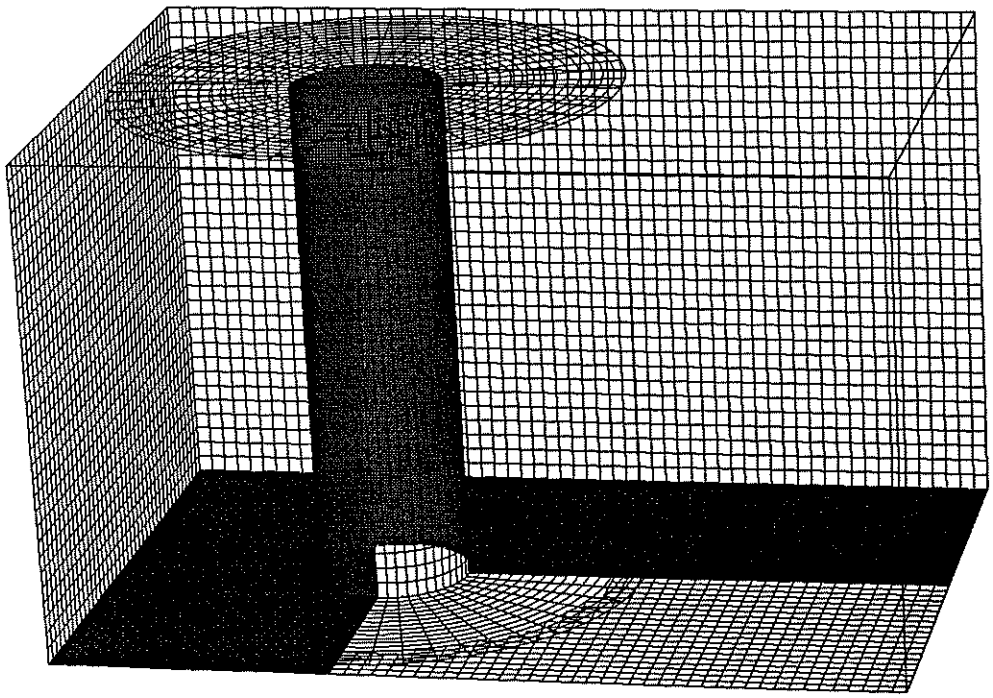


Figure 41: Discretization of the cylinder in testcase 2 a)

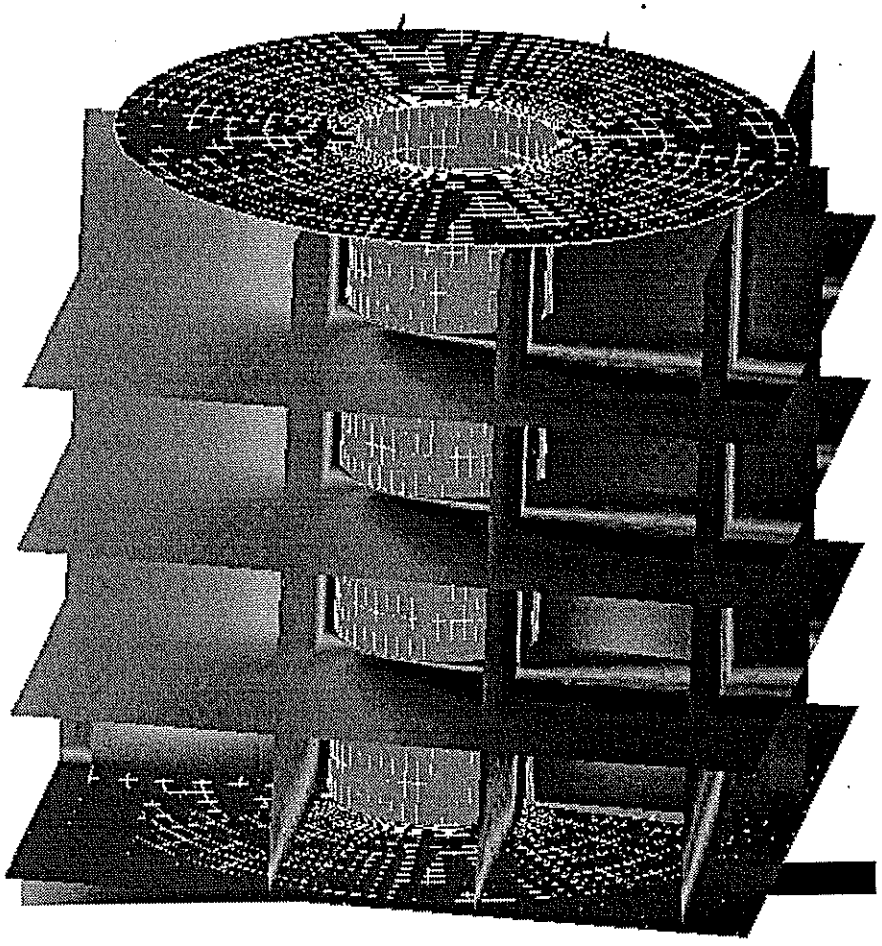


Figure 42: Flow around the cylinder in testcase 2 a), isocontours for the pressure

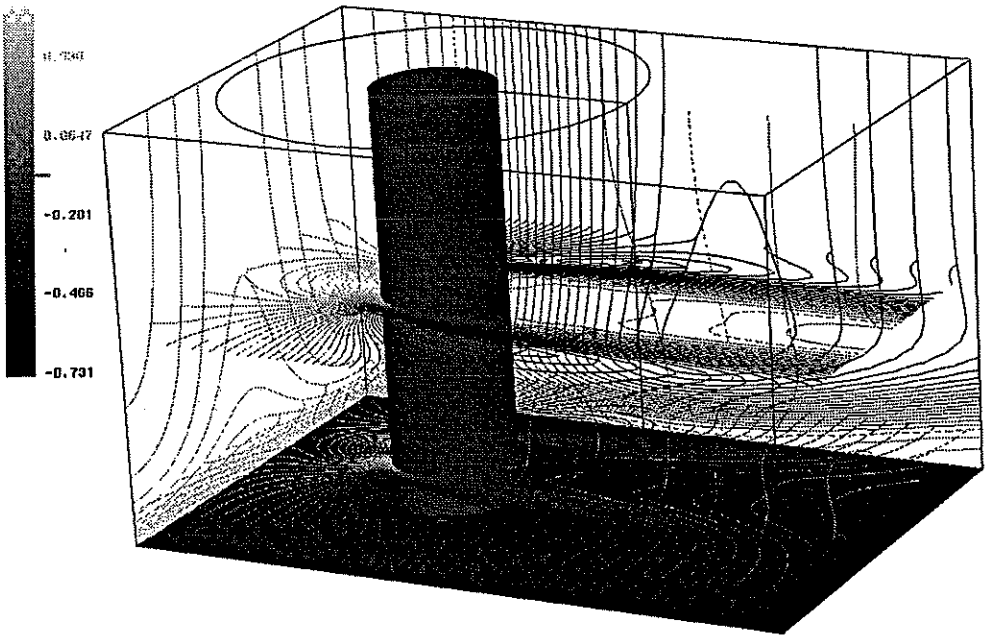


Figure 43: Flow around the cylinder in testcase 2 a), isocontours for the axial velocity

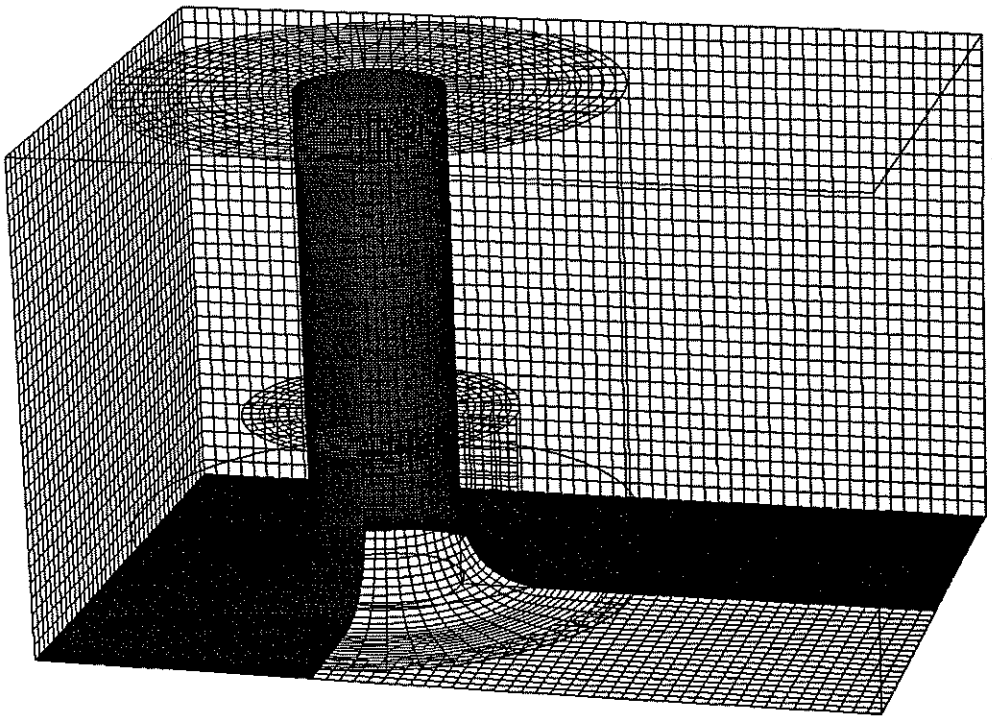


Figure 44: Discretization of the cylinder with fillet in testcase 2 b)

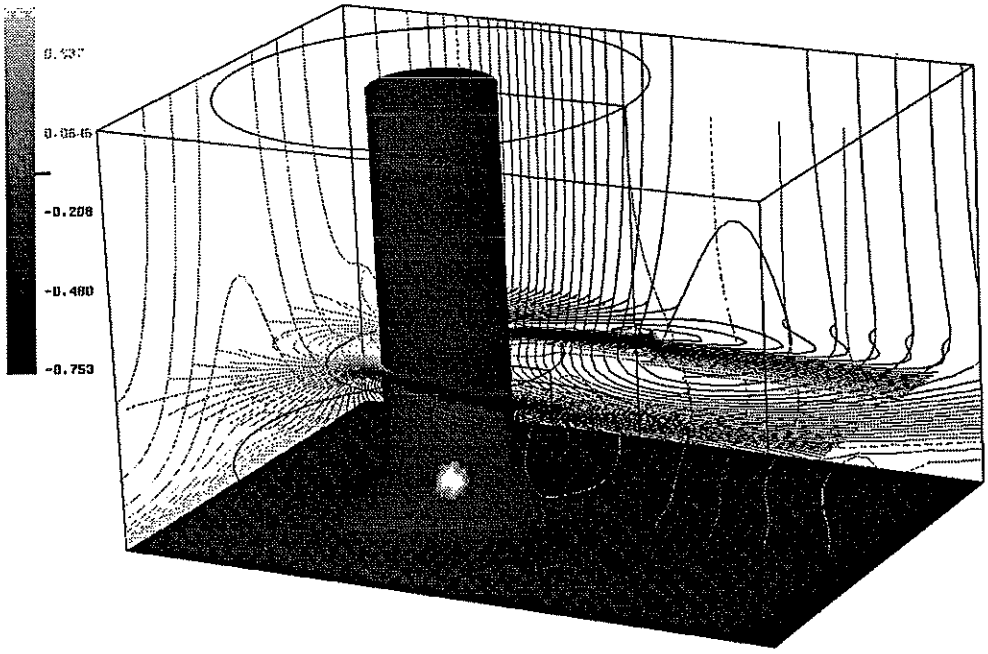


Figure 45: Flow around the cylinder with a fillet in testcase 2 b), isocontours for the pressure

6.1.3 A Simple 3D Ship Hull

We model a simple 3D ship stern by a half of a circular cylinder mounted to the fourth of a sphere, see figure 47. We consider the flow to be symmetric around this model stern. The composite grid used to discretize half of the test geometry is found in figure 48. We have performed a series of computations of both laminar and turbulent flow.

We show a plot of the solution in figure 49–50.

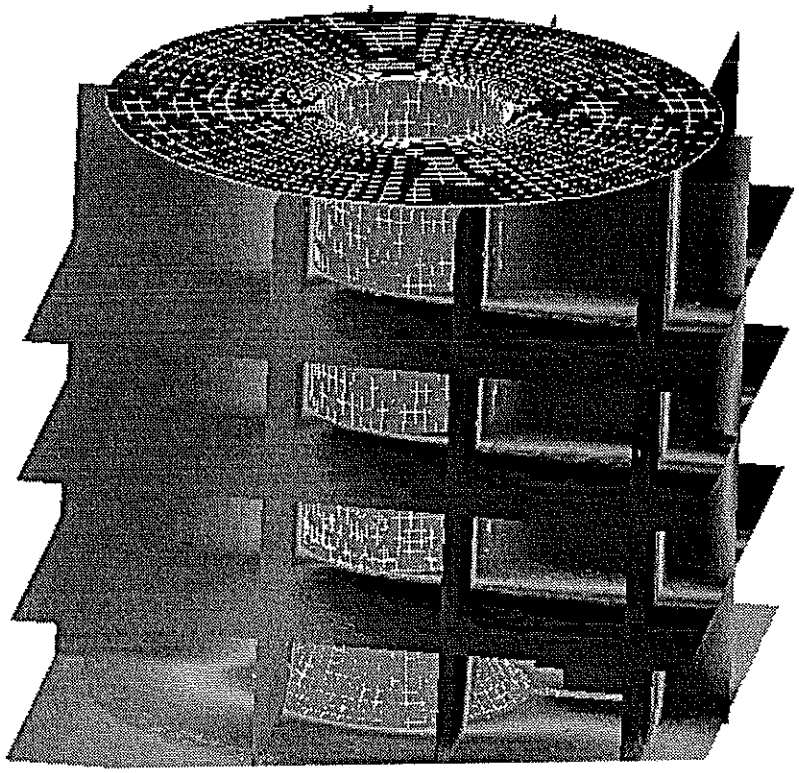


Figure 46: Flow around the cylinder with a fillet in testcase 2 b), isocontours for the axial velocity

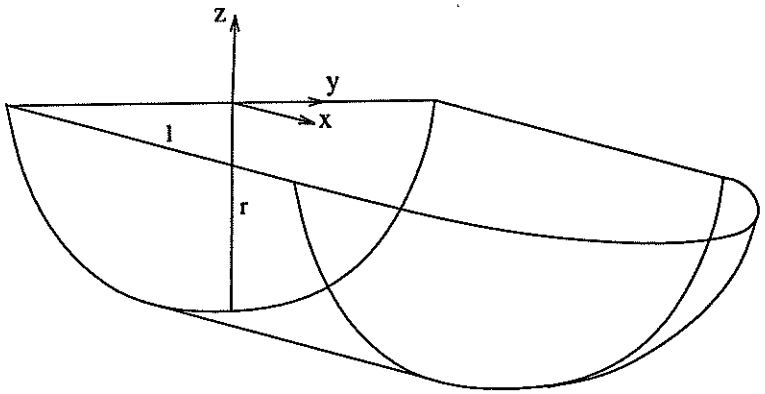


Figure 47: Test geometry 3: A cylinder fitted to a quarter-sphere.

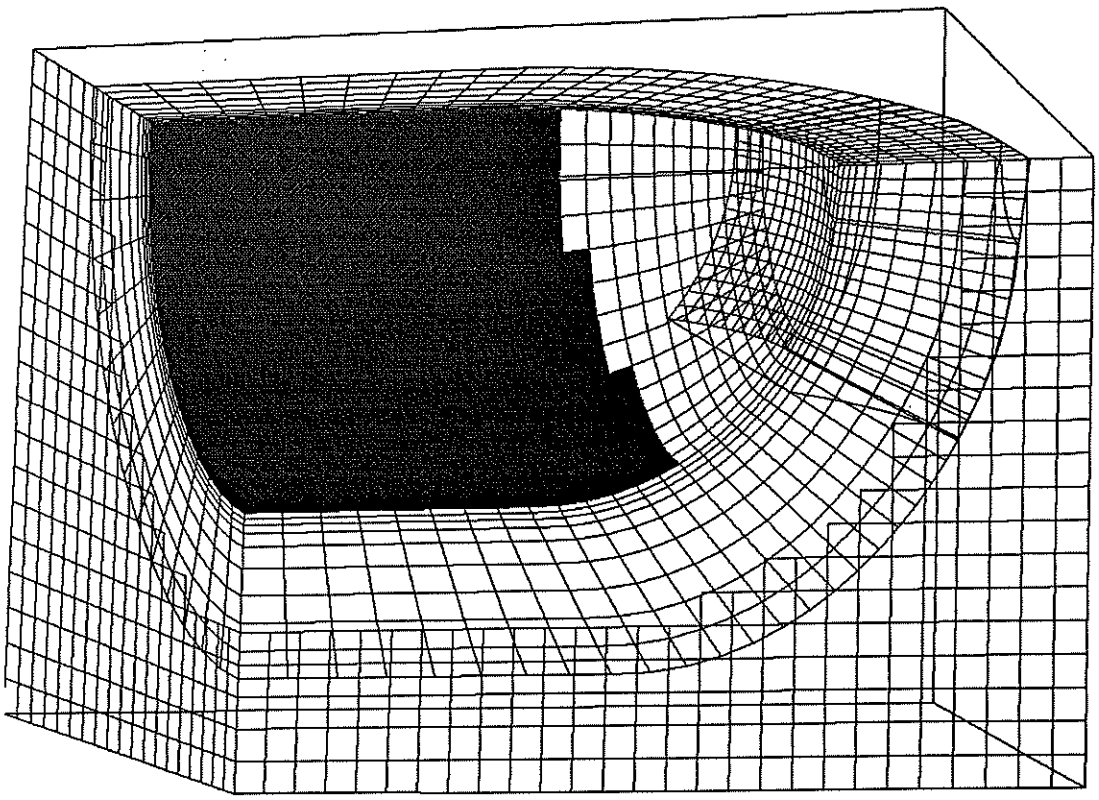


Figure 48: Composite grid discretization of test geometry 3, a part of a sphere mounted to a part of a circular cylinder.

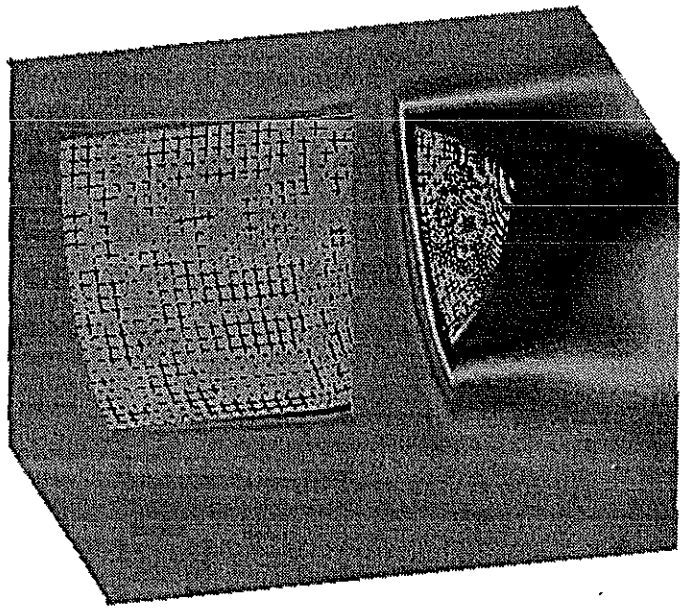


Figure 49: Flow around test geometry 3. Isocontours of axial velocity, $Re_D = 10^6$, k–ϵ turbulence model

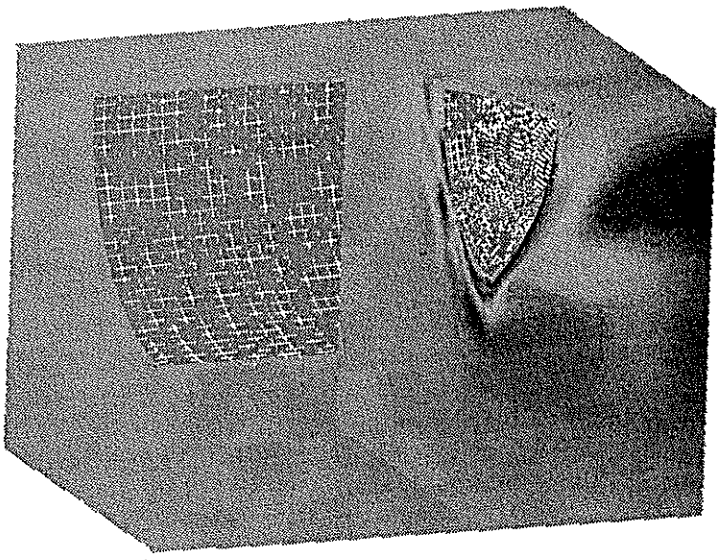


Figure 50: Flow around test geometry 3. Isocontours of sidewise velocity, $Re_D = 10^6$, k–ϵ turbulence model

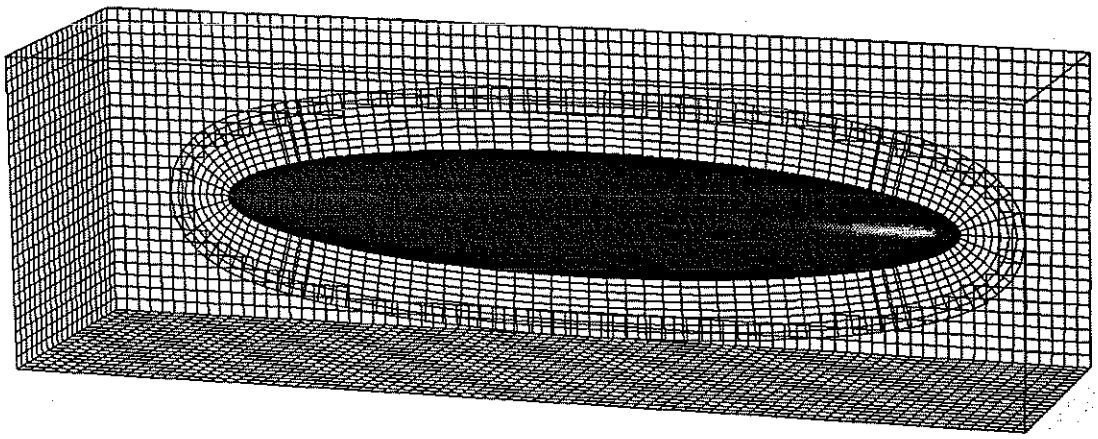


Figure 51: A submerged simple ellipsoid in a box.

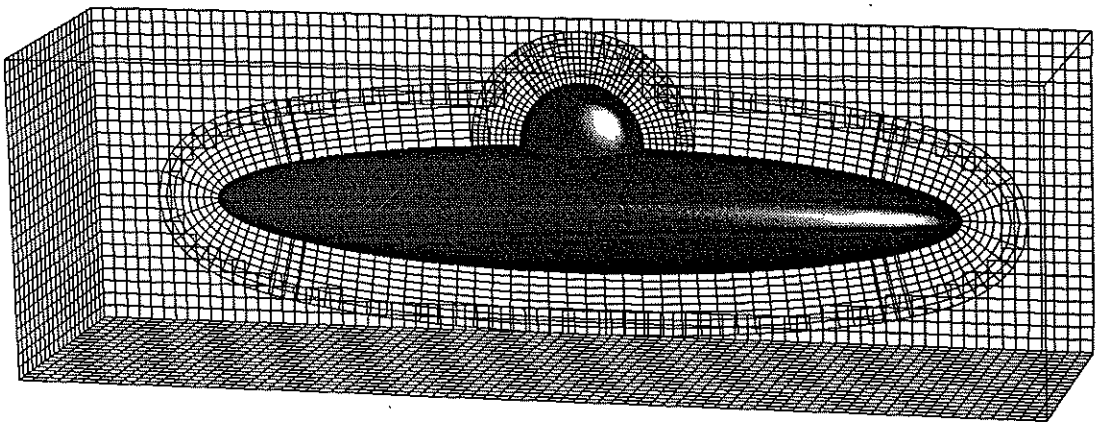


Figure 52: A submerged simple ellipsoid in a box.

6.2 3D Submerged Obstacles

In this section we examplify flow around submerged obstacles. We have chosen a single ellipsoid and a double ellipsoid for this task, see figures 51,52. Dr's G. Chesshire and N. A. Petersson has generated the double ellipse grid. We then display a figure showing computed flow around the double ellipse.

Further details of the composite overlapping grids used to discretize the geometries are displayed in figures 53,54.

We show a plot of the solution in figure 55.

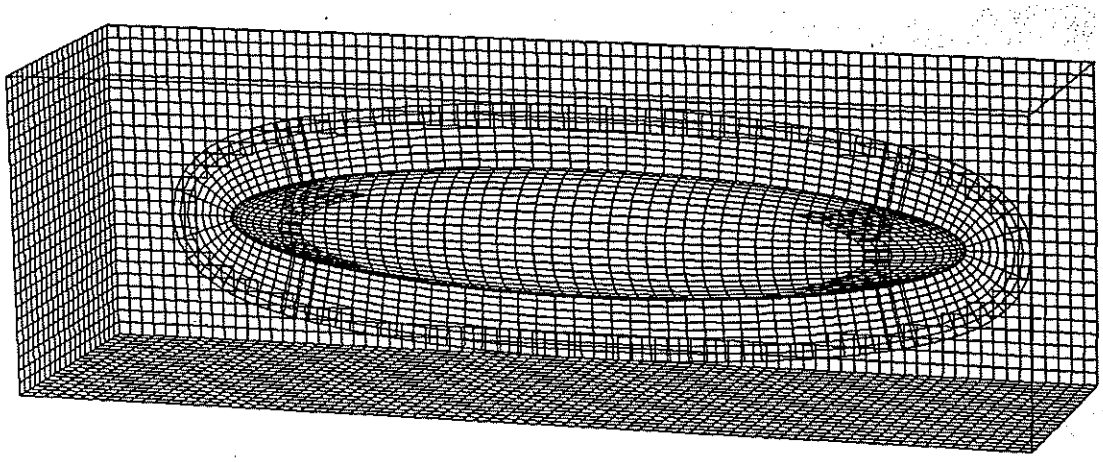


Figure 53: A composite overlapping grid, discretizing a submerged simple ellipsoid in a box.

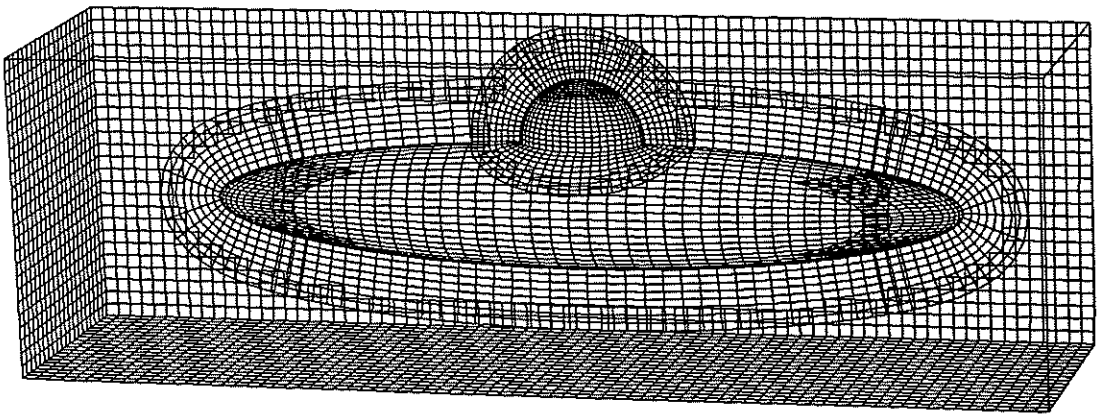


Figure 54: A composite overlapping grid, discretizing a submerged simple ellipsoid in a box.

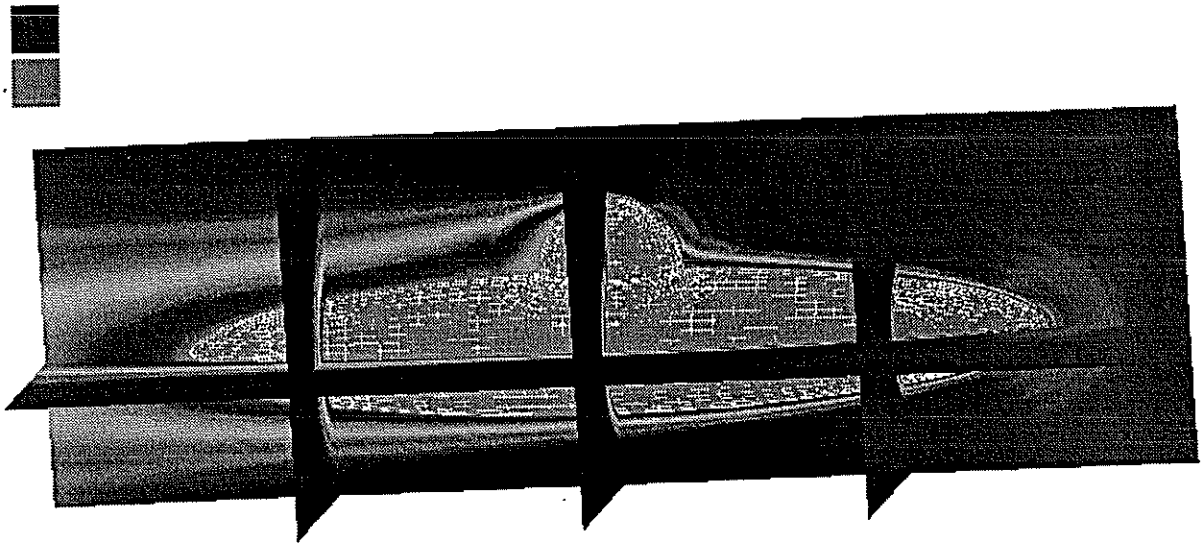


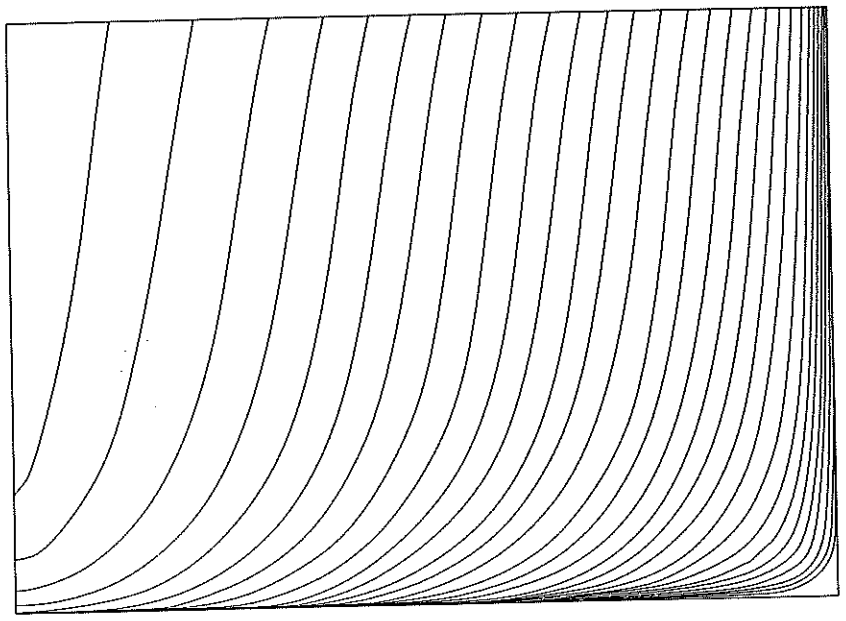
Figure 55: Flow around a submerged double ellipse. Contourplot of the axial velocity.

6.2.1 A 3D Tanker Hull

We use a HSVA, see figure 56, tanker as a reference geometry for our computations. This tanker has been extensively studied and there are experimental data available for this ship hull. It has also been used as a reference geometry in several workshops on the subject, cf. [35, 37, 40].

The discretization of the stern is viewed in figures (11,12). An isosurface plot of flow around the hsva stern is found in figure 57. The Reynolds number for this flow is 10^6 and the k–ϵ turbulence model is used.

In figures 58–61 we compare the axial (longitudinal) velocity obtained by our method with both experimental results and computational results. The experimental and computational results are taken from [37]. The computational results we compare with were computed using the method described in [36]. We observe that the boundary layers computed using the present method are far to thick. This is due to the fact that we use an explicit method to solve the momentum equations. The very small gridsteps, needed to resolve a boundary layer, forces the timestep to be prohibitively small. This is one of the flaws with the present method and we are working on local timestepping and/or using an implicit method to update the momentum components.



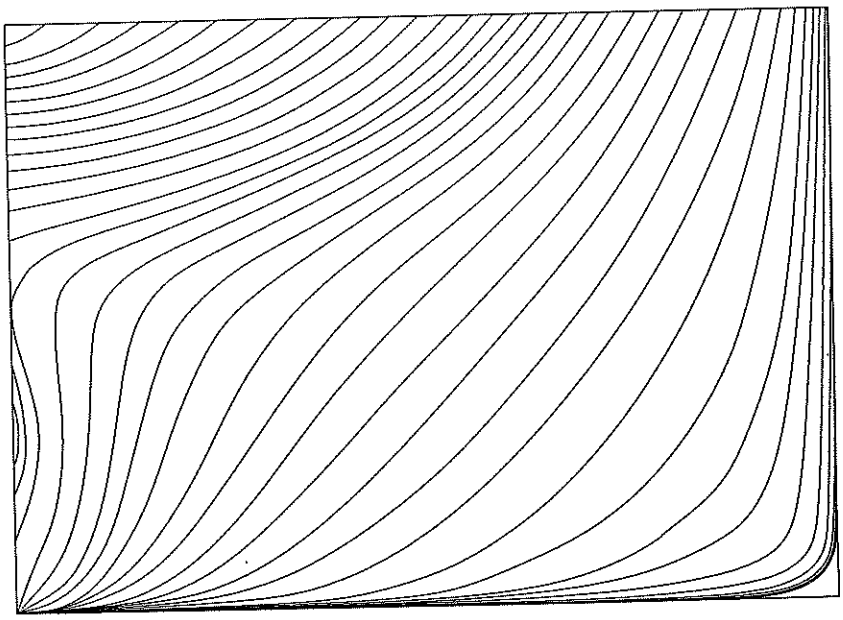


Figure 56: Test geometry 4: A HSVA tanker, described by ship rib-frames. a) The bow, b) The stern

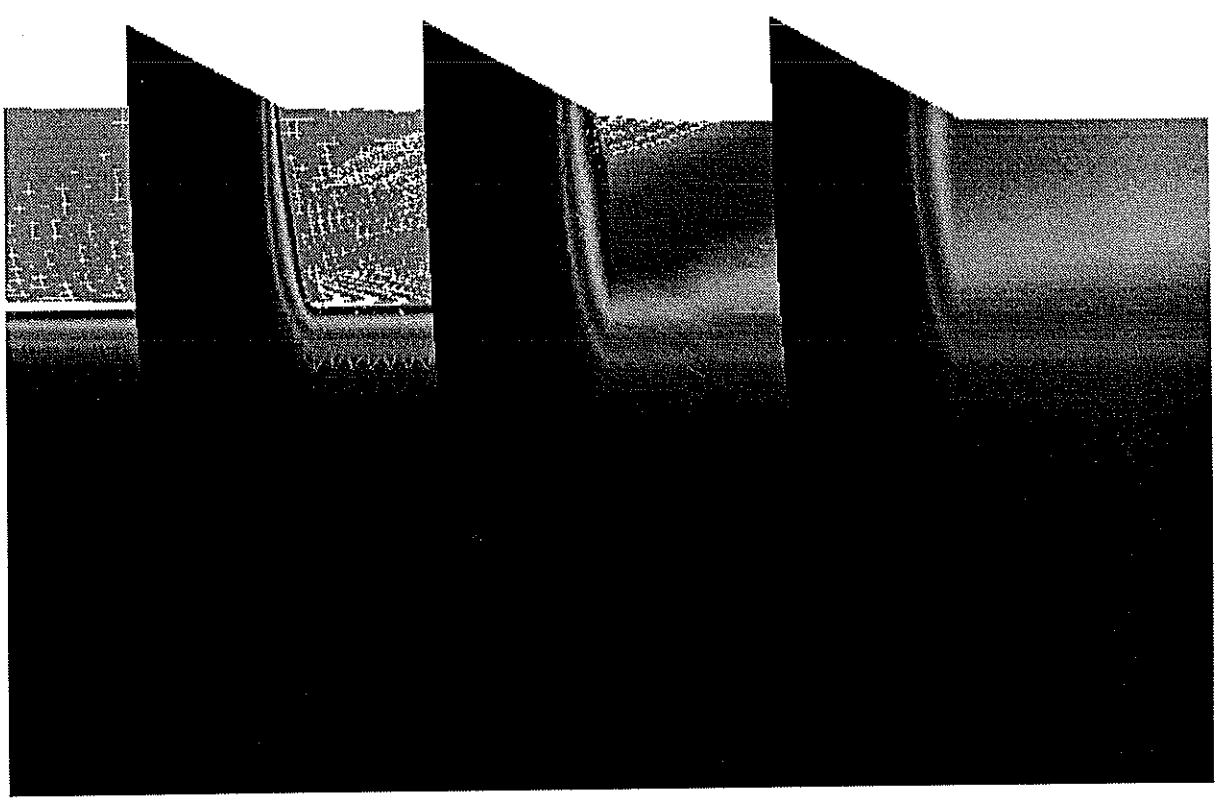


Figure 57: Flow around a ship stern. Contourplot of the axial velocity.

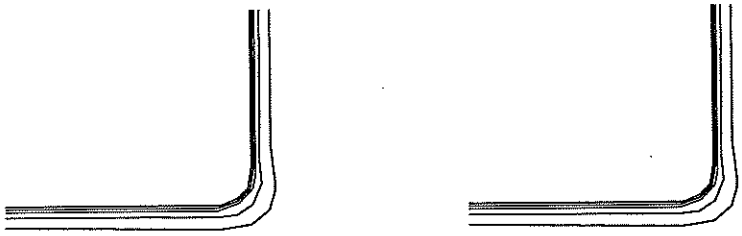


Figure 58: A comparison between results obtained by our method, Other calculations and experiments

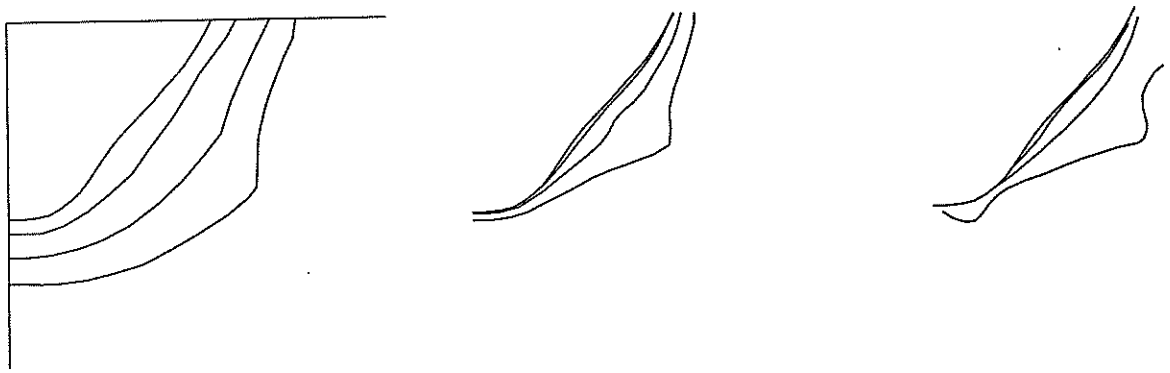


Figure 59: A comparison between results obtained by our method, Other calculations and experiments

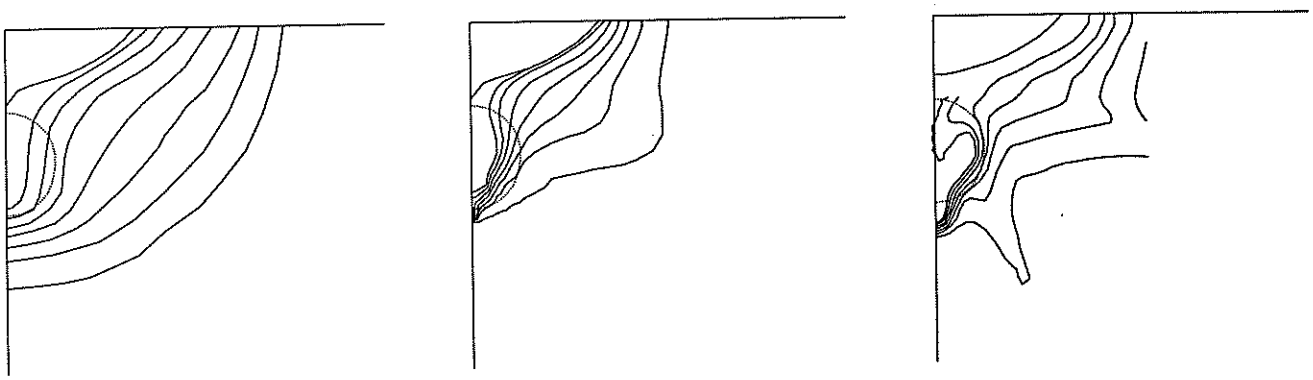


Figure 60: A comparison between results obtained by our method, Other calculations and experiments

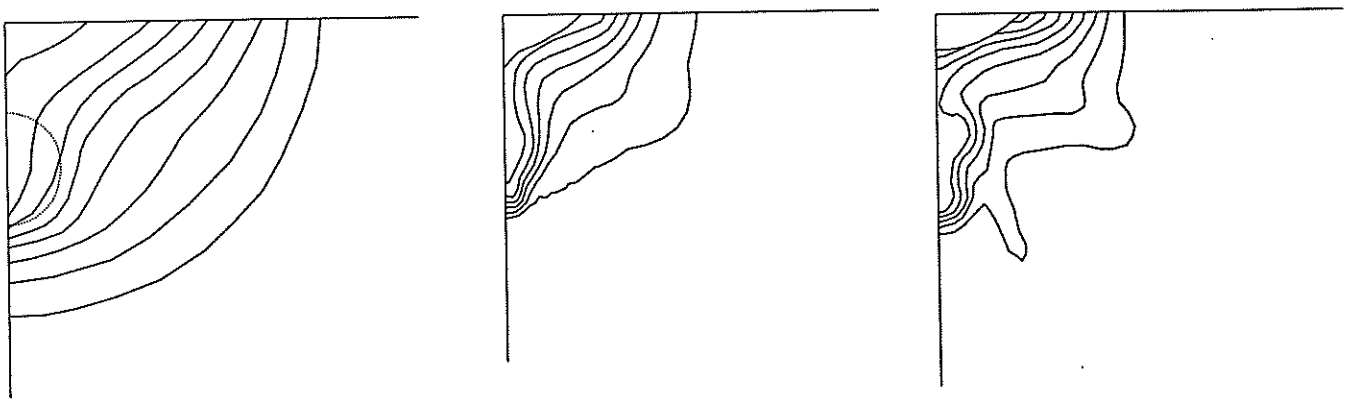


Figure 61: A comparison between results obtained by our method, Other calculations and experiments

7 Discussion and Objectives of Future Research

We have presented a fourth order accurate method for calculation of viscous ship flow. In this paper, we are considering double model flow - ie. assuming that the flow is symmetric with respect to the water surface. However, we are currently working to incorporate free surface effects into this method. The computational method developed in this paper is carefully tested and validated on two and three dimensional examples. In this method we use composite overlapping grids to discretize the geometry. The physical equations governing the flow are discretized by the use of finite difference methods. This enables accurate and efficient resolution of the geometry and the flow phenomena.

A drawback of the present method is that the timestepping of the momentum equations is explicit, this prohibits decreasing the size of the gridcells below a certain limit since the timestep becomes too small for practical computations. To be able to increase the resolution in general and specifically to increase the resolution of boundary layers one idea is to use local timestepping for time accurate solution of the momentum equations. An alternative would be to use implicit timestepping for the momentum equations.

Another problem is the fact that a the number of gridpoints needed to generate a fourth order accurate grid around a ship hull is very large. It is not feasible to perform fourth order flow computations for a ship grid using the current method on a modern vector computer. Therefore an effective parallelization of the solver would be fruitful for solving large problems with high order accuracy. We would like to use the multigrid method for solution of the pressure equation, where GMRES could be used to solve the pressure equation on the coarsest grid.

We plan to study viscous free surface flow around submerged obstacles and surface piercing ships. Here, we aim at discretizing the fluid up to the actual free surface by the use of an adaptive moving composite overlapping grid. To study unsteady phenomena, we need to refine and speed up the grid generation process. An interesting application is to study the interaction between the flow around the hull and the flow through the propeller and past the rudder. To accomplish this we need to utilize a moving composite overlapping grid technique. Another interesting application is to model the vibrations in the ship structure and the acoustic sound field generated by ship, after having computed the flow field around the hull.

Acknowledgement

Prof. H. O. Kreiss is gratefully acknowledged for many fruitful discussions on the present problem. Furthermore he arranged visits at UCLA for the author during the springs of –93 and –94.

The author is also indebted to *Dr. N. A. Petersson* that contributed to this paper with many valuable suggestions.

Dr. D. L. Brown is acknowledged for providing computer resources at Center for NonLinear Studies, LANL.

Finally the author wants to thank *Dr's. G. Chesshire* and *W. Henshaw* for sharing their expertise on composite grids and the utility software, they have developed.

References

[1] B. S. Baldwin and H. Lomax. Thin layer approximation and algebraic model for separated turbulent flows. *AIAA paper 78-257*, 1978.

[2] S. E. Bechtel, J. A. Cooper, M. G. Forest, N. A. Petersson, D. L. Reichard, A. Saleh, and V. Venkataraman. A new model to determine dynamic surface tension and elongational viscosity using oscillating jet measurements. LA–UR 93–3500, Los Alamos National Laboratory, NM, 1993. (Submitted to J. Fluid Mech.).

[3] J. A. Benek, P. G. Buning, and J. L. Steger. Proceedings of the 7th aiaa computational fluid dynamics conference, cincinnati. In *A 3-D Chimera Grid Embedding Technique*, pages 322–331. AIAA, 1985.

[4] M. J. Berger. On conservation at grid interfaces. *SIAM J. of Numer. Anal.*, 24:967–984, 1987.

[5] K. D. Brislawn, D. L. Brown, G. Chesshire, and J. S. Saltzman. Adaptive composite overlapping grids for hyperbolic conservation laws. To appear as unclassified report, LANL, 1994.

[6] L. Broberg. *Numerical Calculation of Ship Stern Flow.* PhD thesis, SSPA, Chalmers Institute of Technology, 1988.

[7] D. L. Brown. A finite volume method for solving the Navier-Stokes equations on composite overlapping grids. In B. Engquist and B. Gustafsson, editors, *Third International Conference on Hyperbolic Problems*, pages 141–158. Chartwell-Bratt, 1991.

[8] D. L. Brown, G. Chesshire, and W. D. Henshaw. Getting started with CMPGRD, introductory user's guide and reference manual. report LA-UR-90-3729, Los Alamos National Laboratory, 1989.

[9] G. Browning. A comparison of three numerical methods for solving differential equations on the sphere. *Monthly Weather Review*, 117:1058–1075, 1989.

[10] P. G. Buning, I. T. Chiu, S. Obayashi, Y. M. Rizk, and J. L. Steger. Numerical simulation of the integrated space shuttle vehicle in ascent. paper 88-4359-CP, AIAA, 1988.

[11] E. Campana, A. Di Mascio, P. G. Esposito, and F. Lalli. Domain decomposition in free surface viscous flows. In *Abstracts and papers. Sixth International Conference on Numerical Ship Hydrodynamics*, pages 6.16–6.26. Iowa Institute of Hydraulic Research, 1994.

[12] G. Chesshire and W. D. Henshaw. The DSK package, a data structure for efficient fortran array storage, (reference guide for the DSK package). IBM Research Report RC 14353, IBM Research Division, Yorktown Heights, NY, 1988.

[13] G. Chesshire and W. D. Henshaw. Composite overlapping meshes for the solution of partial differential equations. *JCP*, 90(1):1–64, 1990.

[14] G. Chesshire and W. D. Henshaw. Conservation on composite overlapping grids. IBM Research Report RC 16531, IBM Research Division, Yorktown Heights, NY, 1991.

[15] C. de Boor. *Splinefunktionen, Lectures in Mathematics.* Birkhauser, ETH, Zurich, 1978.

[16] H. A. Van der Vorst. Bi-cgstab: A fast and smoothly converging variant of bi-cg for the solution of nonsymmetric linear systems. *SIAM J. Sci. Stat. Comp.*, 13(2):631–644, 1992.

[17] J. Farmer, L. Martinelli, and A Jameson. A fast multigrid method for solving the nonlinear ship wave problem with a free surface. In *Abstracts and papers. Sixth International Conference on Numerical Ship Hydrodynamics*, pages 4.1–4.13. Iowa Institute of Hydraulic Research, 1993.

[18] J. Farmer, L. Martinelli, and A Jameson. Multigrid solutions of the euler and navier-stokes equations for a series 60 $c_b = 0.6$ ship hull for froude numbers 0.160, 0.220 and 0.316 (program 1: Navier-stokes formulation). In *Proceedings. CFD Workshop Tokyo 1994*, pages 56–65. Ship Research Institute Japan, 1994.

[19] J. Farmer, L. Martinelli, and A Jameson. Multigrid solutions of the euler and navier-stokes equations for a series 60 $c_b = 0.6$ ship hull for froude numbers 0.160, 0.220 and 0.316 (program 2: Euler formulation). In *Proceedings. CFD Workshop Tokyo 1994*, pages 156–165. Ship Research Institute Japan, 1994.

[20] W. D. Henshaw. Analysis of a difference approximation for the incompressible navier stokes equations. IBM Research Report RC 19333, IBM Research Division, Yorktown Heights, NY, 1993.

[21] W. D. Henshaw. CGES user guide, version 1.00, a solver for steady state boundary value problems on overlapping grids. IBM Research Report RC 19361, IBM Research Division, Yorktown Heights, NY, 1994.

[22] W. D. Henshaw. A fourth-order accurate method for the incompressible Navier-Stokes equations on overlapping grids. *Journal of Computational Physics*, 113(1):13–25, 1994.

[23] W. D. Henshaw and G. Chesshire. Multigrid on composite meshes. *SIAM J. Sci. Stat. Comput.*, 8(6):914–923, 1987.

[24] W. D. Henshaw, G. Chesshire, and M.E. Henderson. On constructing three dimensional overlapping grids with CMPGRD. IBM research report, IBM Research Division, Yorktown Heights, NY, 1992.

[25] W. D. Henshaw, G. Chesshire, and M.E. Henderson. Three dimensional overlapping grid generation techniques. Ibm research report, IBM Research Division, Yorktown Heights, NY, 1993.

[26] W.D. Henshaw, H.O. Kreiss, and L.G. Reyna. Smallest scale estimates for the incompressible Navier-Stokes equations. *Arch. Rational Mech. Anal.*, 112:21–44, 1990.

[27] T. Hino. Computation of free surface flow around an advancing ship by the navier stokes equations. In *Proceedings. Fifth International Conference on Numerical Ship Hydrodynamics*, pages 103–117. Pineridge Press, 1993.

[28] T. Hino, L. Martinelli, and A. Jamesson. A finite-volume method with unstructured grid for free surface flow simulations. In *Abstracts and papers. Sixth International Conference on Numerical Ship Hydrodynamics*, pages 4.18–4.37. Iowa Institute of Hydraulic Research, 1994.

[29] B. C. V. Johansson. Boundary conditions for the $k - \epsilon$ model for turbulent flow. Technical Report 115, Dept. of Scientific Computing, Uppsala University, 1988.

[30] B. C. V. Johansson. Boundary conditions for open boundaries for the incompressible navier stokes equation. *J. Comp. Phys.*, 105, No 2:233–251, 1993.

[31] C. Kiris, S. Rogers, D. Kwak, and I.D. Chang. Computation of incompressible viscous flow through artificial heart devices with moving boundaries. *ASME J. of Biofluidmechanical Engineering*, to appear, 1992.

[32] P. M. Knupp. *Fundamentals of grid generation.* Boca Raton, Philadelphia, 1993.

[33] G. Kreiss. The dependence on the outflow boundary condition of the solution of steady, incompressible euler equations. *SIAM J. Numer. Anal.*, 28, No. 5:1242–1264, 1991.

[34] H.-O. Kreiss and J. Lorenz. *Initial-Boundary Value Problems and the Navier-Stokes Equations.* Academic Press, 1989.

[35] L. Larsson. *SSPA-ITTC Workshop on Ship Boundary Layers 1980 Proceedings.* SSPA-Report, No. 90., 1981.

[36] L. Larsson, L. Broberg, K. K. Kim, and D. H. Zhang. A method for resistance and flow prediction in ship design. Transactions Vol. 99, Society of Naval Architects and Marine Engineers, 1991.

[37] L. Larsson, V. C. Patel, and Gilbert Dyne. *Ship Viscous Flow, Proceedings of 1990 SSPA-CTH-IIHR Workshop.* Flowtech Research-Report, No. 2., 1991.

[38] A. Lungu and K. Mori. Developments of the finite difference schemes for free-surface flow computation. In *Proceedings. CFD Workshop Tokyo 1994*, pages 331–340. Ship Research Institute of Japan, 1994.

[39] J. F. Malmliden and N. A. Petersson. A fast iterative method to compute the flow around a submerged body. Technical Report LA-UR-93-0839, (to appear in Math. Comput.), Center for NonLinear Studies, Los Alamos National Laboratory, 1993.

[40] K. Mori. *Proceedings. CFD Workshop Tokyo 1994.* Ship Research Institute, Japan, 1994.

[41] F. Olsson. A solver for time dependent viscoelastic fluid flow. *J. Non-Newtonian Fluid Mech.*, 51:309–340, 1994.

[42] F. Olsson and P. Isaksson. Effects of an elastic component of the coating colour rheology on the blade coating process as revealed by numerical methods. *Rheologica Acta*, 1994. to appear.

[43] J. Oppelstrup. Users guide to the esaspline package. Technical report, Kungliga Tekniska Hogskolan, STOCKHOLM, 1980.

[44] E. Part-Enander and B. Sjogreen. Solving the Euler equations for high Mach numbers on overlapping grids. Technical Report 114, Dept. of Scientific Computing, Uppsala University, 1988.

[45] E. Part-Enander and B. Sjogreen. Shock waves and overlapping grids. Technical Report 131, Dept. of Scientific Computing, Uppsala University, 1991.

[46] N. A. Petersson. Computing gravity waves on water by using moving composite overlapping grids. CAM Report 92-01, UCLA, 1992.

[47] N. A. Petersson. Fast numerical computation of 2–D free surface jet flow with surface tension. LA–UR 93–3938, Los Alamos National Laboratory, NM, 1993. (Submitted to J. Comput. Phys.).

[48] N. A. Petersson and J. F. Malmliden. Computing the flow around a submerged body using composite grids. *J. Comput. Physics*, 105(1):47–57, 1993.

[49] W. Rodi. Turbulence models and their application in hydraulics. state of the art paper, IAHR, 1987.

[50] E. P. Rood. Ship hydrodynamics basic research in the united states. paper, International Conference on Hydrodynamics, 1994.

[51] Y. Saad and M. H. Schultz. Gmres, a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM. J. Sci. Stat. Comput.*, 7(3):856–869, 1986.

[52] P. Sonneveld. Cgs: A fast lanczos-type solver for nonsymmetric linear systems. *SIAM. J. Sci. Stat. Comput.*, 10(1):36–52, 1989.

[53] Y. Tahara and F. Stern. A large domain approach for calculating ship boundary layers and wakes for nonzero froude number. In *Proceedings. CFD Workshop Tokyo 1994*, pages 45–55. Ship Research Institute of Japan, 1994.

[54] Y. Tahara, F. Stern, and B. Rosen. An interactive approach for calculating ship boundary layers and wakes for nonzero froude number. *J. Comput. Phys.*, 98(1):33–53, 1992.

[55] J. F. Thompson, Z. U. A. Warsi, and C. W. Mastin. *Numerical Grid Generation.* North-Holland, New York, 1985.

[56] J. Tu. *Three Dimensional Overlapping Grids and Multigrid Methods for Flow Calculations in Complex IC. Engine Geometries.* PhD thesis, Royal Institute of Technology, Stockholm, 1992.

[57] D. Vandromme. Introduction to the modeling of turbulence. turbulence modeling for compressible flows and implementation in navier stokes solvers. Lecture Series 02, von Karman Institute for Fluid Dynamics, 1991.

[58] M. J. Ward, W. D. Henshaw, and J. B. Keller. Summing logarithmic expansions for strong localized perturbations of linear and nonlinear eigenvalue problems. Ibm research report, IBM Research Division, Yorktown Heights, NY, 1993.

[59] D. C. Wilcox. *Turbulence modeling for CFD.* DCW Industries Inc., La Cañada, California, 1993.

[60] W. M. Yue and D. K. P. Yue. Numerical solutions for large amplitude ship motions in the time domain. In *Proceedings. 18 th Symposium on Naval Hydrodynamics.* University of Michigan, Ann Arbor, MI, 1990.