INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2003; **42**:465–491 (DOI: 10.1002/fld.523)

CFD simulation of two- and three-dimensional free-surface flow

David Apsley*,† and Wei Hu

Manchester Centre for Civil and Construction Engineering, UMIST, PO Box 88, Manchester, M60 1QD, England, U.K.

SUMMARY

The paper describes the implementation of moving-mesh and free-surface capabilities within a 3-d finite-volume Reynolds-averaged-Navier–Stokes solver, using surface-conforming multi-block structured meshes. The free-surface kinematic condition can be applied in two ways: enforcing zero net mass flux or solving the kinematic equation by a finite-difference method. The free surface is best defined by intermediate control points rather than the mesh vertices. Application of the dynamic boundary condition to the piezometric pressure at these points provides a hydrostatic restoring force which helps to eliminate any unnatural free-surface undulations. The implementation of time-marching methods on moving grids are described in some detail and it is shown that a second-order scheme must be applied in both scalar-transport and free-surface equations if flows driven by free-surface height variations are to be computed without significant wave attenuation using a modest number of time steps. Computations of five flows of theoretical and practical interest—forced motion in a pump, linear waves in a tank, quasi-1d flow over a ramp, solitary wave interaction with a submerged obstacle and 3-d flow about a surface-penetrating cylinder—are described to illustrate the capabilities of our code and methods.

KEY WORDS: finite-volume method; moving grids; free-surface flows; waves

1. INTRODUCTION

The rapid expansion in available and affordable computer power and a greater awareness of the utility of numerical simulation as a design tool has led to computational fluid dynamics (CFD) being used to tackle increasingly complex fluid-flow problems, and simulation of three-dimensional, time-varying flows is now within the range of desktop computers. In this paper we describe the development and application of moving-mesh and free-surface capabilities within a general-purpose finite-volume code and illustrate it with flow calculations undertaken on a personal computer.

* Correspondence to: D. D. Apsley, Manchester Centre for Civil and Construction Engineering, UMIST, P.O. Box 88, Manchester, M60 1QD, U.K.
† E-mail: d.apsley@umist.ac.uk

Contract/grant sponsor: EPSRC grant; contract/grant number: GR/R06717

Received 14 August 2002
Accepted 22 January 2003

Numerous engineering problems involve flow with moving boundaries. The movement of *solid* boundaries may be externally imposed (for example, a piston in a reciprocating engine) or it may be a dynamic response to fluid forcing (e.g. vortex-induced vibrations of bridge decks and oil risers). A moving *free* surface—usually an air–water interface—is encountered in many areas of hydraulic engineering; examples include waves and tidal flows, where the fluctuating loads on coastal and offshore structures are of considerable importance. Even where a free surface is stationary its shape is often not easily determined (for example, flow over weirs and submerged structures). In these cases a standard technique is to employ a time-marching procedure to step towards steady state.

The vast range of physical problems involving free-surface flows has generated a variety of CFD approaches for specific applications. Potential-flow/boundary-element methods [1] are used for wave dynamics and water-entry problems. The shallow-water equations, incorporating depth-averaging and a hydrostatic-pressure approximation, are frequently used in tidal flows and to simulate non-breaking wave propagation and run-up [2]. These methods are powerful tools in their own areas: however, the approximations embodied in the governing flow equations make them unsuitable for general-purpose flow solvers. The neglect of viscous and turbulent transport eliminates a whole range of flow phenomena associated with boundary-layer separation and recirculating flow and the transport and deposition of sediment. The hydrostatic approximation is untenable when vertical accelerations are comparable with that of gravity; an important example is wave impact on structures. For codes aspiring to be general-purpose flow solvers a free-surface capability must be imbedded within a solution procedure for the full Navier–Stokes—or, at least, the Reynolds-averaged Navier–Stokes (RANS)—equations. This is the objective of the present work.

CFD approaches for moving boundaries tend to fall into two categories: fixed-mesh and moving-mesh. In fixed-mesh methods the fluid-containing fraction of each cell must be specified or computed. In the context of free-surface flows, popular techniques include the *volume-of-fluid* (VOF) method [3] where an equation is solved for the void fraction, and the *marker-and-cell* (MAC) method [4] where the free surface is tracked by following the motion of particles on the interface. The VOF method has been touted as the only method of simulating breaking waves. This is not true. The problem of handling breaking waves with a moving mesh is one of finding a suitable algorithm for the movement and removal or addition of cells around the multiply-connected region and the calculations of Reference [5] go some way to showing that this is feasible. By contrast, the VOF method must resolve a sharp interface (between void fractions 0 and 1) and for free surfaces which are rapidly varying in time or space this requires an extremely fine mesh.

The second practice—which is that adopted by the present work—is that of dynamically adapting the mesh in such a way that it is always *surface-conforming*, i.e. mesh cells always contain fluid (in contrast to VOF) whilst impermeable solid walls and free surfaces coincide with cell faces. The practice complements the finite-volume approach because of the latter's natural relationship with the fundamental *integral* forms of the governing conservation equations. Finite-volume, moving-mesh methods and their application to free-surface flow have been described by, for example, [5–8].

The numerical problem can be divided into two parts: the extension of the time-dependent finite-volume technique to include a moving mesh, and the motion of the free surface (which governs the evolution of the mesh). The relationship between the finite-volume method and the integral conservation laws of fluid mechanics makes the first of these natural

and straightforward. The second is more problematic, especially in three dimensions. Numerical methods are described in Section 2.

A number of applications are described in Section 3. These are: (1) a simple mechanical pump, which tests the moving-mesh capabilities without the complication of free-surface movement; (2) oscillation of small-amplitude waves in a tank, for which linear wave theory is a good approximation; (3) quasi-1-d channel flow over a ramp, which has a (steady-state) theoretical solution; (4) passage of a 2-d solitary wave over a submerged obstacle; (5) 3-d shear flow about a surface-penetrating cylinder. Despite the fact that the free-surface movement in the last of these was comparatively small (and has been almost universally neglected in other computations of flow about a cylinder), this proved exceptionally challenging and was the only test case which distinguished the relative viability of the various free-surface-moving algorithms which we tried. In addition, (5) is representative of the high-Reynolds-number problems that are the norm in hydraulics and for which turbulence modelling is an important issue. A summary of our findings and an outline of areas for future research is given in Section 4.

2. NUMERICAL METHODS

2.1. Starting point

Our objective was to implement moving-mesh and free-surface capabilities within a general-purpose university research code called STREAM. The code is a 2- or 3-d finite-volume solver which uses the SIMPLE pressure-correction algorithm to solve the Reynolds-averaged Navier–Stokes (RANS) equations on multi-block structured curvilinear meshes. The basic numerical procedures have been described by Reference [9] and the multi-block meshing technique (which involves forming a 2-cell overlap with adjacent structured blocks) in Reference [10]. The code continues to undergo considerable development, and it has been used extensively for the testing of advanced turbulence models in aerodynamics [11].

The control-volume geometry is defined by specifying the vertices of hexahedral cells. Storage of flow variables is collocated and cell-centred. For the mass fluxes cell-face velocities are determined by the standard Rhie-Chow interpolation technique to eliminate the 'odd-even' decoupling of pressure values resulting from centred differencing of pressure gradients on a collocated grid. Advective fluxes may be prescribed by a number of second- and third-order upwind-biased schemes. An iterative solution of the Navier–Stokes equations is achieved by the SIMPLE pressure-correction technique. Here, the relationship between small pressure and velocity changes (a relationship determined by the *momentum* equations) enables the rephrasing of the *continuity* equation for each cell as a pressure-correction equation, the solution of which is used to increment pressure and velocity fields in such a way as to satisfy simultaneously both mass and momentum equations. A large number of turbulence closures are available [11], ranging from linear and non-linear eddy-viscosity models to full second-moment closure. Each category includes high- and low-Re variants to compute wall-bounded flows using either wall functions or a direct integration through the semi-viscous sublayer.

In the subsections below, Section 2.2 describes the extension of the finite-volume method to include moving meshes, Section 2.3 describes the time-stepping procedures and Section 2.4 deals with the free-surface movement algorithm.

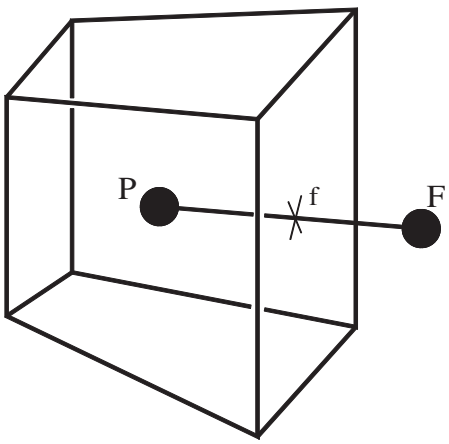

Figure 1. Notation for a general control volume and face.

2.2. *Moving meshes*

The finite-volume method is based on the discretization of the *integral* conservation equations of fluid mechanics. From a programming perspective there are but two canonical equations: *mass conservation* (*continuity equation*):

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_V \rho\,\mathrm{d}V + \int_{\partial V} \rho(\mathbf{U}-\mathbf{U}_\mathrm{g})\cdot\mathrm{d}\mathbf{A} = 0 \tag{1}$$

generic scalar transport equation:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_V \rho\phi\,\mathrm{d}V + \int_{\partial V} [\rho(\mathbf{U}-\mathbf{U}_\mathrm{g})\phi - \Gamma\nabla\phi]\cdot\mathrm{d}\mathbf{A} = \int_{\partial V} \mathbf{T}\cdot\mathrm{d}\mathbf{A} + \int_V S\,\mathrm{d}V \tag{2}$$

where V is an arbitrary control volume with surface ∂V. ϕ is the amount per unit mass of any conserved quantity, including the individual components of momentum $\phi = U_i$, but also any turbulent scalars such as k and ε. Γ, $\mathbf{T}$ and S are generalized diffusivity, non-diffusive flux and source density, respectively, and are different for each transported quantity. The whole of the RHS is usually collectively referred to as 'the source term'; in general, it may contain both surface contributions (which should be treated conservatively) and volume-integrated parts.

The main advantage of the integral formulation in the present context is that it does not matter whether the control volumes are stationary or moving, the only concession to mesh movement being the presence of the grid velocity $\mathbf{U}_\mathrm{g}$ in the advection term. $\int_{\partial V}[\rho(\mathbf{U}-\mathbf{U}_\mathrm{g})\phi\cdot\mathrm{d}\mathbf{A}$ is the *net* rate at which the quantity represented by ϕ is advected across a moving surface.

In semi-discrete form (2) becomes

$$\frac{\mathrm{d}}{\mathrm{d}t}(\rho V\phi_P) + \sum_f \{\dot{m}_f\phi_f - D_f(\phi_F-\phi_P)\} = \text{source} \tag{3}$$

where the summation is over the faces of a control volume (Figure 1). Here, subscript P denotes the node at the centre of the control volume and F the node on the opposite side

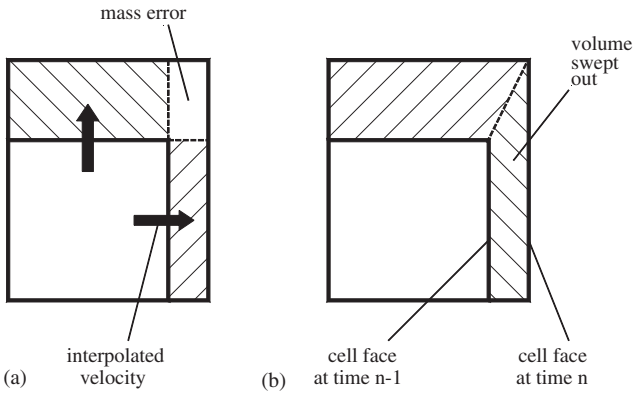

Figure 2. Illustrating the need to satisfy the space-conservation law: (a) $\mathbf{U}_g$ based on centroid velocity; (b) $\int \mathbf{U}_g \cdot d\mathbf{A}$ determined from the rate at which a cell face sweeps out volume.

of the general face f. $\dot{m}_f$ and D_f are net mass fluxes and diffusive transport coefficients, respectively, and '*source*' is a repository for the discretised RHS of (2) plus contributions transferred from the 'non-diagonal' diffusion terms and departures of the advective fluxes from first-order upwind differencing ('deferred corrections'). The *net* mass flux $\dot{m}_f$ is

$$\dot{m}_f = C_f - \rho Q_f \tag{4}$$

being the difference between the mass flow rate across the *instantaneous* cell face,

$$C_f = \int_{\text{face } f} \rho \mathbf{U} \cdot d\mathbf{A}$$

and the rate at which mass is swept out by the moving cell face:

$$\rho Q_f = \int_{\text{face } f} \rho \mathbf{U}_g \cdot d\mathbf{A}$$

If $(\Sigma \dot{m}_f)\phi_P$ is subtracted from both sides of (3) we obtain a standard linearised form:

$$\frac{d}{dt}(\rho V \phi) + a_P \phi_P - \Sigma a_F \phi_F = b_P - (\Sigma \dot{m}_f)\phi_P \tag{5}$$

where the summation is over adjacent nodes or cell faces. For incompressible flows on stationary meshes the net mass flux out of the control volume $(\Sigma \dot{m}_f)$ is zero in the final converged solution and this 'divergence' term may be ignored. In compressible flows or on moving meshes this part is equal to $-d(\rho V)/dt$ and it partially offsets the time derivative.

The rate at which mass is swept out by a cell face, ρQ_f, is determined by the grid velocity $\mathbf{U}_g$. If this is simply based on the velocity of the face centroid (say) then mass errors may arise if the cell is distorted in more than one direction—see Figure 2 and Reference [12]. Instead, we note that $\mathbf{U}_g$ only appears in a product of velocity and area, and $\int \mathbf{U}_g \cdot d\mathbf{A}$ is simply the rate at which volume is swept out by that cell face. Given the co-ordinates of the face vertices at the start and end of each timestep, this hexahedral volume is in practice determined by the same subroutine as that called to evaluate the volume of each cell.

With this definition of $\mathbf{U}_g$, the 'space-conservation law'

$$\frac{d}{dt}\int_V dV - \int_{\partial V} \mathbf{U}_g \cdot d\mathbf{A} = 0 \tag{6}$$

is automatically satisfied, and (assuming ρ is constant) (1) reduces to

$$\int_{\partial V} \rho \mathbf{U} \cdot d\mathbf{A} = 0 \tag{7}$$

Thus, in incompressible flow, the pressure-correction equation is required—as for stationary grids—to enforce zero divergence on the instantaneous mesh at any time level: *it is not necessary to solve a time-dependent mass equation*.

2.3. Time-marching

We have coded three time-marching schemes: backward-differencing, Crank–Nicolson and Gear's scheme. The first of these is first-order accurate in time; the last two are second-order accurate. In our applications we have found that second-order time accuracy is vital for computing genuinely time-dependent problems in a moderate number of timesteps. This is true both for the scalar-transport equation (covered in this section) and the free-surface movement algorithm (Section 2.4). In the context of the finite-volume method, the schemes differ in the approximation for the time derivative and the time level (denoted by a superscript) at which the flux and source terms are evaluated.

2.3.1. Backward differencing ('first-order implicit'). According to this scheme, the flux and source terms in (5) are evaluated at the 'new' time level:

$$\frac{(\rho V \phi_P)^{\text{new}} - (\rho V \phi_P)^{\text{old}}}{\Delta t} = [b_P - a_P \phi_P + \Sigma a_F \phi_F - (\Sigma \dot{m}_f)\phi_P]^{\text{new}} \tag{8}$$

Since the divergence term can be recovered from the continuity equation (compressible or incompressible) as

$$\frac{(\rho V)^{\text{new}} - (\rho V)^{\text{old}}}{\Delta t} = -\Sigma \dot{m}_f \tag{9}$$

these can be rearranged (dropping the 'new' label for convenience) as

$$\left[\frac{(\rho V)^{\text{old}}}{\Delta t} + a_P\right]\phi_P - \Sigma a_F \phi_F = b_P + \frac{(\rho V \phi_P)^{\text{old}}}{\Delta t} \tag{10}$$

Thus, time-marching is easily incorporated via a minor modification to diagonal (a_P) and source (b_P) coefficients. Note that to eliminate the divergence ($\Sigma \dot{m}_f$) term the mass of fluid in the cell, ρV, must, in contrast to the fluxes, be evaluated at the *old* time level.

2.3.2. Crank–Nicolson (*'centred-time/centred-space'*). Here, the flux and source terms in (5) are the averages of those at the 'old' and 'new' time levels:

$$\frac{(\rho V\phi_P)^{\text{new}} - (\rho V\phi_P)^{\text{old}}}{\Delta t} + \sum_f [(\bar{C}_f - \rho Q_f)\bar{\phi}_f - \overline{D_f(\phi_F - \phi_P)}] = \bar{b}_P \tag{11}$$

where an overbar here denotes and average of 'old' and 'new' values; i.e. $\bar{\phi} = \frac{1}{2}(\phi^{\text{new}} + \phi^{\text{old}})$. For a truly time-centred arrangement the advection term requires a very careful treatment, as follows.

(i) The mass flux through an instantaneous cell face, C_f, can be averaged over values at the start and end of the time step. The contribution Q_f, however, represents mass swept out over the course of the time step and is *not* averaged.

(ii) For upwind-biased advection schemes the weighting of $\bar{\phi}_f$ on the adjacent nodal values is not known until the sign of $\dot{m}_f = \bar{C}_f - \rho Q_f$ has been established. Thus, it is *not* possible to include advection in the a_P^{old} and a_F^{old} coefficients, as is commonly done on fixed meshes.

(iii) It is convenient, as before, to simplify the time-dependent and advection terms by subtracting $\bar{\phi}_P$ times the discretised continuity equation (9) from Equation (11), which then becomes, after rearrangement:

$$\frac{\overline{\rho V}}{\Delta t}(\phi_P^{\text{new}} - \phi_P^{\text{old}}) + \sum_f [\bar{\dot{m}}_f(\bar{\phi}_f - \bar{\phi}_P) - \overline{D_f(\phi_F - \phi_P)}] = \bar{b}_P \tag{12}$$

where $\overline{\rho V} = \frac{1}{2}[(\rho V)^{\text{old}} + (\rho V)^{\text{new}}]$. For this time-marching method the mass of fluid in the cell, ρV, appears, unsurprisingly, as the average of values at 'old' and 'new' times.

If $\bar{\phi}_f$ is written, as is common, as the sum of the value at the 'upwind' node, $\bar{\phi}_{\text{U}}$, plus a deferred correction $\bar{\phi}_f - \bar{\phi}_{\text{U}}$, then (12) can be rearranged in canonical form as

$$\left[a_P + 2\frac{\overline{\rho V}}{\Delta t}\right]\phi_P - \Sigma a_F\phi_F = b_P + 2\frac{\overline{\rho V}}{\Delta t}\phi_P^{\text{old}} + \left[b_P + \sum_f D_f(\phi_F - \phi_P)\right]^{\text{old}} - \sum_f \bar{\dot{m}}_f(\phi_{\text{U}}^{\text{old}} - \phi_P^{\text{old}}) - 2\sum_f \bar{\dot{m}}_f(\bar{\phi}_f - \bar{\phi}_{\text{U}}) \tag{13}$$

where

$$a_F = D_f + \max(-\bar{\dot{m}}_f, 0)$$

$$a_P = \sum a_F + source - term\ contributions$$

Again, for convenience, we have dropped the 'new' superscripts. As before, the time-stepping scheme is implemented by modifying matrix coefficients and the whole of the RHS is treated explicitly as a source term. Note, however, the complex form of the advection terms, which involve a combination of time-averaged and 'old' time-level quantities. In principle (if,

apparently, not always in practice) a timestep restriction is implied by the need to keep the net coefficient of ϕ_P^{old} positive.

In our own implementation the computational penalty is the necessity to store certain quantities at the 'old' time level: instantaneous mass fluxes C_f^{old}, plus, for each variable, arrays for ϕ_P^{old} and $[b_P + \sum_f D_f(\phi_F - \phi_P)]^{\text{old}}$. These last are actually assigned at the start of each timestep so, whilst they require core memory, they do not need to be committed to filestore to enable a restart.

2.3.3. Gear's method [*13*]. This uses a second-order finite-difference approximation for the time derivative at time (n):

$$\frac{\mathrm{d}}{\mathrm{d}t}(\rho V\phi_P) \to \frac{\frac{3}{2}(\rho V\phi_P)^{(n)} - 2(\rho V\phi_P)^{(n-1)} + \frac{1}{2}(\rho V\phi_P)^{(n-2)}}{\Delta t} \tag{14}$$

All flux and source terms are evaluated at the 'new' time level (n). The method is fully implicit and stable. It requires storage of all transported variables at two previous time levels. The core memory requirements are similar to those of the Crank–Nicolson scheme, but additional filestore may be needed since all field variables at one previous time level must be stored to disk if a restart is required. The first timestep must obviously be undertaken with a different method.

2.4. Free-surface movement

2.4.1. Overview. For boundary-conforming meshes, the motion of the free surface governs the evolution of the grid. For our own particular applications it is sufficient that the blocks of cells abutting the free surface are simply stretched or compressed in the vertical, with the vertices which define them moving up and down in proportion to their relative position between fixed and free surfaces. However, the surface-moving algorithms that we describe are applicable to more general mesh evolution, provided that suitable re-meshing facilities are available.

Note first of all that the free surface emerges as part of the solution and that its movement is only part of a grand iterative cycle. Within each timestep there are several free-surface updates (accompanied by corresponding mesh adjustments) and for each mesh there are several cycles of the SIMPLE algorithm to update pressure and velocity fields. As the geometry changes continually it is inefficient to solve all equations to a high accuracy on each intermediate mesh and it is our practice to perform a modest number of SIMPLE cycles (up to a maximum number or a suitable reduction in equation residuals) for each adjustment of the free surface. Only when mass, momentum and free-surface kinematic equations are simultaneously satisfied does the solution proceed to the next timestep.

The boundary conditions which must be applied at the free surface are:

kinematic condition—no net flow through the free surface;

dynamic boundary condition—stress is continuous across the interface.

2.4.2. The kinematic boundary condition. It is the kinematic condition that is directly related to the shape and movement of the free surface. There are two distinct ways of enforcing it numerically.

Strategy I: *maintain zero net mass flux.* Move the free-surface (by moving either cell vertices or face-centre control points—see below) in such a way as to maintain zero *net* mass flux through all cell faces on the free surface. This is the strategy adopted by, for example, References [7, 8]: it is very much in tune with the finite-volume methodology.

The free-surface movement is performed *incrementally within each timestep*, the volume swept out at each adjustment being intended to eliminate the current net mass flux for each free-surface cell face. Assuming, for simplicity, vertical motion of cell vertices or control points, the adjustment for each free-surface cell face is of the form

$$\delta\bar{h} = \frac{\dot{m}_s \Delta t}{\rho A_h} \tag{15}$$

where $\delta\bar{h}$ is the average height increment over the cell face, $\dot{m}_s$ is the current net mass flux through the surface control-volume face (which we are aiming to make zero) and A_h is the projected area normal to the direction of movement.

$\delta\bar{h}$ is determined by the change in height of the surface vertices surrounding the control-volume face and direct implementation of the set of equations (15) would lead to a set of simultaneous equations for the vertex heights. In two dimensions each surface vertex is, at most, related to the vertex on either side and the resulting tri-diagonal set is easily solved. In three dimensions, however, since each surface cell face is surrounded by 4 vertices the solution of the equations resulting from (15) requires considerable algebraic effort and, as in Reference [7], it is more efficient to adopt (15) as a *local* approximation for each control point and treat it as an iterative method to achieve zero net mass flux.

Strategy II: *finite-difference solution of a differential equation.* Integrate numerically the first-order differential equation defining the free surface as a material interface ($D(z-h)/Dt = 0$). For example, if the height h is a function of x and y,

$$\frac{\partial h}{\partial t} = (W - \mathbf{U}\cdot\nabla h)_s \tag{16}$$

Multiplying by A_h, (16) becomes, in discrete form

$$A_h \frac{\Delta h}{\Delta t} = (\mathbf{U}\cdot\mathbf{A})_s \tag{17}$$

where $\mathbf{A}$ is the face area vector. Note that we make a distinction between δh, which is a small incremental adjustment within a timestep, and Δh which is the *total* change in height over a timestep (though it will be continually readjusted during the course of a timestep as the RHS of (17) changes). Once again we apply this as a *local* iterative update, rather than solving simultaneous equations for all h values.

Strategies I and II are equivalent in the limit as the grid size goes to zero.

As in the case of the scalar-transport equation, various time-marching schemes may be employed to solve (17). Indeed, our experience from the applications we tried (notably the wave-tank calculation—see Section 3.2), is that for time-accurate solution of motions *driven* by free-surface variations it is vital that a second-order scheme be used for the free-surface movement as well as the transport equations. Thus, for example, with the Crank–Nicolson scheme the RHS of Equation (15) or (17) must be taken as the average of 'old' and 'new' time levels.

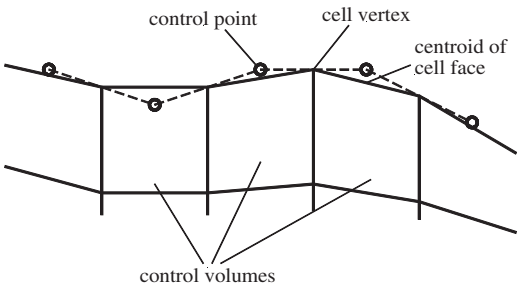

Figure 3. Use of control points to define the free surface.

In addition, we may choose to move either the cell vertices themselves or control points at the cell centres [7]—see Figure 3. In the first case, the RHS of (15) or (17) is evaluated by averaging over cell faces sharing the vertex being moved. In the latter case, (15) or (17) can be evaluated directly: the control points are moved and cell vertices then computed by averaging over the control points of faces meeting at that vertex. In two dimensions this would amount to simple linear interpolation. Experience with our more complex test cases indicates that defining the free surface through intermediate control points yields a more stable algorithm and smoother free surface.

In the last of the applications to be described in Section 3—the surface-penetrating cylinder—the free-surface movement proved especially taxing, despite the variation in surface elevation being comparatively small. This application led us to develop a local under-relaxation technique—outlined below—to stabilise the algorithm.

2.4.3. An under-relaxation technique to stabilise the local free-surface algorithm. Where the free surface is changing rapidly the change in height at each iteration may require under-relaxation. This can be achieved by reducing the change at each iteration by a factor α; i.e.

Strategy I (*mass flux condition*):

$$\delta\bar{h} = \alpha\left(\frac{\dot{m}_f \Delta t}{\rho A_h}\right) \tag{18}$$

or

Strategy II (*finite-difference method*):

$$h^{(n)} = h^* + \alpha\left(h^{(n-1)} + \Delta t\frac{(\mathbf{U}\cdot\mathbf{A})_s}{A_h} - h^*\right) \tag{19}$$

In (19), $h^{(n-1)}$ is the height of the moved point at the previous time level and h^* is the most recent estimate of $h^{(n)}$.

In some applications—notably the start-up motion in the case of the surface-penetrating cylinder—even this under-relaxation proved insufficient to save the free-surface algorithm. One reason for this is that the free-surface algorithms making a *local* update address the change in *height* of the free surface, but not its change in *slope*. This may be seen most clearly in Equation (16): the RHS is essentially maintained constant in evaluating the height change,

but it actually contains ∇h. (In the finite-volume forms (17) or (15) this is built—rather less obviously—into the *orientation* of the surface area vector **A**.) If part of the slope-dependent term can be treated implicitly then some stabilisation of the algorithm should result.

The method is best described first in two dimensions, where it is similar to a very standard implicit treatment of the 1-d wave equation. The kinematic equation for h is

$$\frac{\partial h}{\partial t} = W - U\frac{\partial h}{\partial x}$$

If $h^{(n-1)}$ is the height at the end of the previous timestep and h^* the most recent estimate of $h^{(n)}$, then this discretizes as

$$\frac{h^{(n)} - h^{(n-1)}}{\Delta t} = W - U\underbrace{\frac{\partial h^*}{\partial x}}_{\substack{\text{current}\\ \text{slope}}} - U\underbrace{\frac{\partial}{\partial x}(h^{(n)} - h^*)}_{\substack{\text{adjusted}\\ \text{slope}}} \tag{20}$$

The last term on the RHS is the (implicit) change in slope that will result from free-surface adjustment. If we assume that—(a) this term can be upwind-differenced; (b) the most severe case of slope adjustment occurs when only this point on the surface moves—then we can make the approximation

$$-U\frac{\partial}{\partial x}(h^{(n)} - h^*) \approx -\frac{|U|}{\Delta x}(h^{(n)} - h^*) \tag{21}$$

The accuracy of this approximation is not important as this term will vanish in the converged solution. Transferring this term to the LHS of (20) gives, after some rearrangement,

$$h^{(n)} = h^* + \frac{h^{(n-1)} + (W - U(\partial h^*/\partial x)) - h^*}{1 + c} \tag{22}$$

where

$$c = \frac{|U|\Delta t}{\Delta x}$$

is the Courant number. Comparison with (19) shows that the change is equivalent to an under-relaxed height change with

$$\alpha \to \frac{1}{1 + c}$$

The changes necessary for three dimensions differ only in detail. Equation (17) becomes

$$h^{(n)} - h^{(n-1)} = \frac{\Delta t}{A_h}[\mathbf{U}\cdot\mathbf{A}^* + \mathbf{U}\cdot\delta\mathbf{A}]$$

where, again, a superscript * denotes the most recent iteration and the last term on the RHS arises from any change in orientation of the free surface. In finite-difference form,

$$\frac{\mathbf{U}\cdot\delta\mathbf{A}}{A_h} \to -\mathbf{U}\cdot\nabla(h - h^*) \to -\frac{\mathbf{U}\cdot\mathbf{A}_\xi}{V}\frac{\partial}{\partial\xi}(h - h^*) - \frac{\mathbf{U}\cdot\mathbf{A}_\eta}{V}\frac{\partial}{\partial\eta}(h - h^*)$$

where ξ and η are the curvilinear co-ordinates indexing control volumes in directions *within* the free surface, $\mathbf{A}_\xi$ and $\mathbf{A}_\eta$ are the area vectors of the corresponding near-surface control-volume faces and V is the cell volume. Making the same upwind-differenced approximation as above leads once again to an under-relaxed update of the form

$$h^{(n)} = h^* + \frac{h^{(n-1)} + (\Delta t \mathbf{U} \cdot \mathbf{A}/A_h) - h^*}{1 + c} \tag{23}$$

where the Courant number is

$$c = \Delta t \left(\frac{|\mathbf{U} \cdot \mathbf{A}_\xi|}{V} + \frac{|\mathbf{U} \cdot \mathbf{A}_\eta|}{V} \right) \tag{24}$$

The last reduces on a cartesian grid to the more recognisable form

$$c = \frac{|u|\Delta t}{\Delta x} + \frac{|v|\Delta t}{\Delta y}$$

The same local under-relaxation formulae may be applied if one decides to pursue Strategy I (maintain zero net mass flux) instead.

2.4.4. The dynamic boundary condition. The dynamic boundary condition insists that the stress tensor be continuous at the interface. Neglecting surface tension and viscous effects, this is usually taken as the gauge pressure vanishing at the surface:

$$P = 0 \quad \text{on} \quad z = h(x, y) \tag{25}$$

However, there are several numerical niceties to observe.

(i) For turbulent flow, where the surface boundary layer is not resolved (so that, for turbulent kinetic energy k for example, we apply $\partial k/\partial n = 0$ at the boundary, rather than the strict condition $k \to 0$), the dynamic boundary condition is

$$P + \rho\overline{u_n^2} = 0 \quad \text{on} \quad z = h(x, y) \tag{26}$$

where $\rho\overline{u_n^2}$ is the turbulent momentum flux in a direction normal to the surface.

(ii) Rather than introducing an additional source term in the W-momentum equation it is better to solve for the piezometric pressure $P^* = P + \rho g z$. The dynamic boundary condition for P^* is

$$P^* + \rho\overline{u_n^2} = \rho g h \quad \text{on} \quad z = h(x, y) \tag{27}$$

(iii) Where the free surface is defined by intermediate control points the dynamic boundary condition on the piezometric pressure should be applied at the control points and not the centroids of free-surface cell faces. This helps to eliminate unnatural free-surface undulations numerically in the same way as it would physically—a hydrostatic pressure distribution providing forces to redistribute fluid and restore a smooth interface.

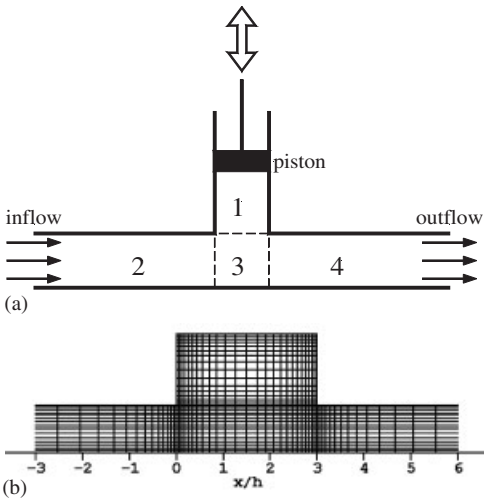

Figure 4. Mechanical pump: (a) 4-block geometry; (b) computational mesh.

3. APPLICATIONS

3.1. Simple model of a pump

This application was studied as a test of the moving-mesh algorithm in the absence of free-surface effects and, whilst the geometry is rather simplistic, the possibilities for more complex designs are clear, given suitable meshing capabilities.

Fluid is pumped by the action of a piston, as shown in Figure 4. Non-return valves are simulated, allowing fluid to enter from the left during the suction part of the cycle (at a flow rate exactly compensating the rate at which volume is swept out by the piston), whilst in the second half of the cycle the inlet is closed and fluid is expelled at the right.

A 4-block structured grid is used, as illustrated in the figure. Only the cells in block 1 are allowed to change dynamically, with uniform vertical compression or expansion in relation to the movement of the piston. Non-dimensional variables are used, such that the height of the inlet/outlet channel is unity (and the width of the piston chamber is 3). The height H of the piston base at time t is described by

$$H = 2.5 + A\sin(2\pi t/T), \quad A = 1, \ T = 64$$

The Reynolds number corresponding to this non-dimensionalisation is 10^5 and a turbulent flow is simulated using a standard high-Re k–ε turbulence closure [14] with no-slip wall boundary conditions being implemented via wall functions. Note that velocities on the walls must be the same as those of the walls themselves, being non-zero on the piston. This presents a problem analogous to that for free-surface (or, for that matter, the much-computed 'lid-driven-cavity' problem): namely, that there is a discontinuity in velocity at the junction of two moving surfaces.

Figure 5 shows a complete cycle of the pump in terms of the instantaneous streamlines and velocity vectors after start-up transients have disappeared (about 5 cycles, as

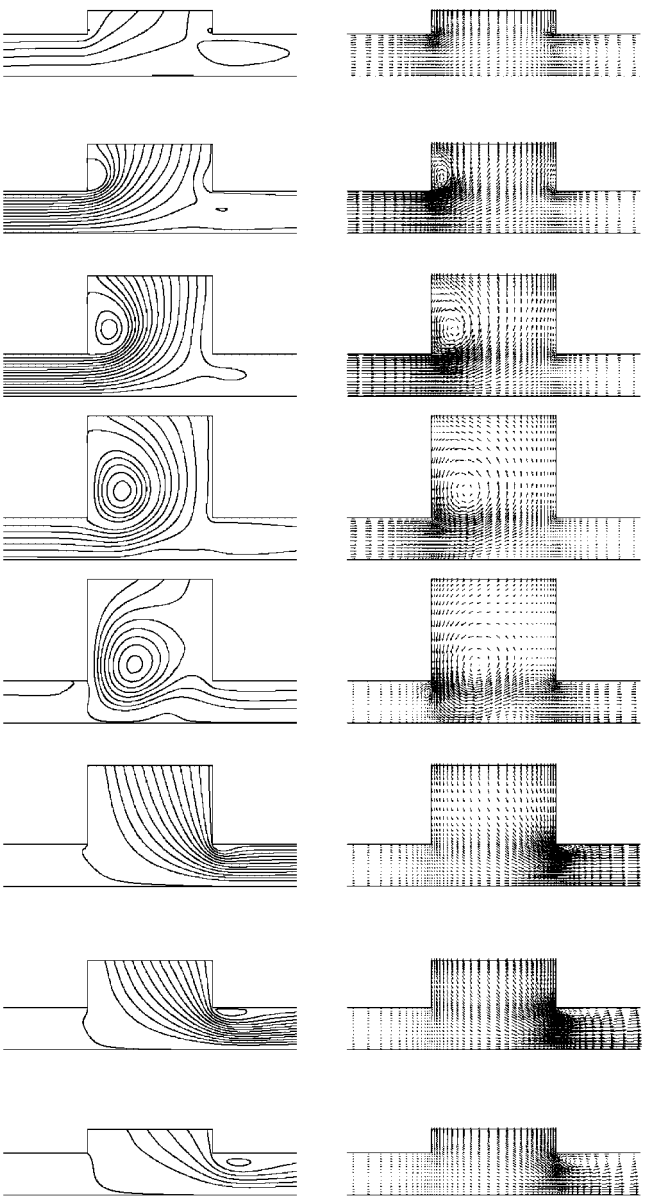

Figure 5. Simulated flow in a pump: (a) streamlines; (b) velocity vectors.

determined by velocity variations at a reference position below the piston chamber). The dominant feature is the large vortex formed initially from the separating flow as flow is drawn into the pump. This extensive feature moves across the piston chamber, being finally expunged during the last part of the cycle. Two smaller vortices are shed from the

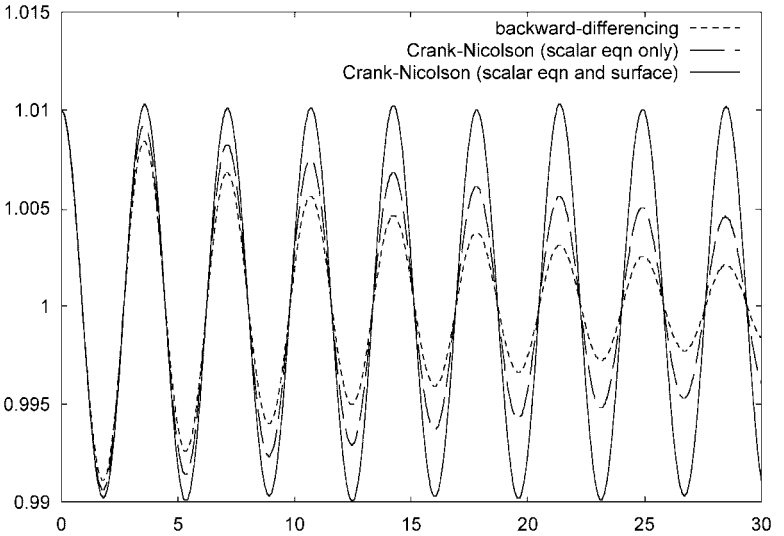

Figure 6. Small-amplitude waves in a tank—height of fluid at one side as a function of time.

corners of the piston chamber as the piston descends, but these are rapidly removed as suction recommences.

3.2. *Small-amplitude waves in a tank*

This is a common test case for the development of free-surface codes since inviscid theory gives exact solutions for fluid motions and surface profiles in the limit as the wave amplitude tends to zero. For harmonic motions proportional to $e^{i(kx-wt)}$ in water of undisturbed depth d, linearized theory gives for the dispersion (frequency vs wave number) relation

$$\omega^2 = gk \tanh kd$$

where the wavelength is $\lambda = 2\pi/k$ and the period is $T = 2\pi/\omega$. Standing-wave motion in a tank of length L can be obtained from the superposition of two such waves, the surface displacement being of the form $A\cos(kx - \omega t)$, provided that L is an integral number of half wavelengths. We considered only the longest of such waves, initiating the motion from a fluid at rest with surface profile $y_s = d + A\cos(\pi x/L)$, where $A = 0.01d$. The wave period predicted by theory then gives

$$T\sqrt{\frac{g}{L}} = 2\sqrt{\frac{\pi}{\tanh(\pi d/L)}}$$

or a value of 3.55 for the particular case where $d = L$.

For the computations, numerical experiments confirmed that a uniform grid of 40×40 control volumes gave satisfactory spatial resolution, but the calculations proved very sensitive to the time-marching scheme for the free surface. Figure 6 shows the height of water at the left hand side of the tank over a number of oscillations using $\Delta t/T = 0.01$ (i.e. 100 time steps per period) whilst Figure 7 shows the typical near-surface flow. Although the wave

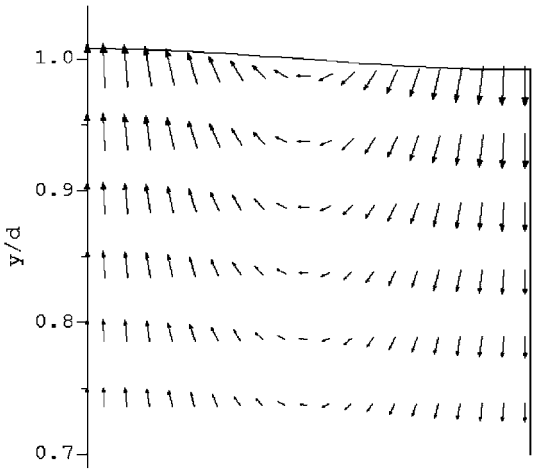

Figure 7. Small-amplitude waves in a tank—near-surface flow.

Figure 8. Flow over a ramp—geometry definition.

period was entirely in line with theoretical predictions, wave attenuation (which should not occur since the case is inviscid) was considerable with the first-order time-stepping scheme. Applying a second-order time-stepping scheme to the scalar-transport equation alone improved matters slightly (Gear's method and the Crank–Nicolson scheme gave almost indistinguishable results), but only when the second-order scheme was applied to the free-surface movement also was wave attenuation eliminated. Indeed, a calculation of total kinetic+potential energy confirmed that negligible energy was lost over 10 oscillations in the last case. We conclude that for flows driven by free-surface height variations a second-order time-marching scheme for both scalar-transport equations and free-surface motion is vital if the flow is to be simulated numerically with a modest value of the timestep.

Both strategies for updating the free-surface height—enforcing zero net mass flux or solving the kinematic equation—gave indistinguishable results for this test case.

3.3. Quasi 1-d flow over a ramp

A second inviscid test case for which theory provides useful comparisons is quasi-1d flow over a bump. For the simple ramp shown in Figure 8 the flow is uniform at distances sufficiently far from the height transition and the classical combination of Bernoulli's theorem and continuity gives for the downstream height h:

$$\frac{h}{h_0} + \frac{1}{2}\mathrm{Fr}^2\left(\frac{h_0}{h}\right)^2 = 1 + \frac{1}{2}\mathrm{Fr}^2 - \frac{z_b}{h_0} \tag{28}$$

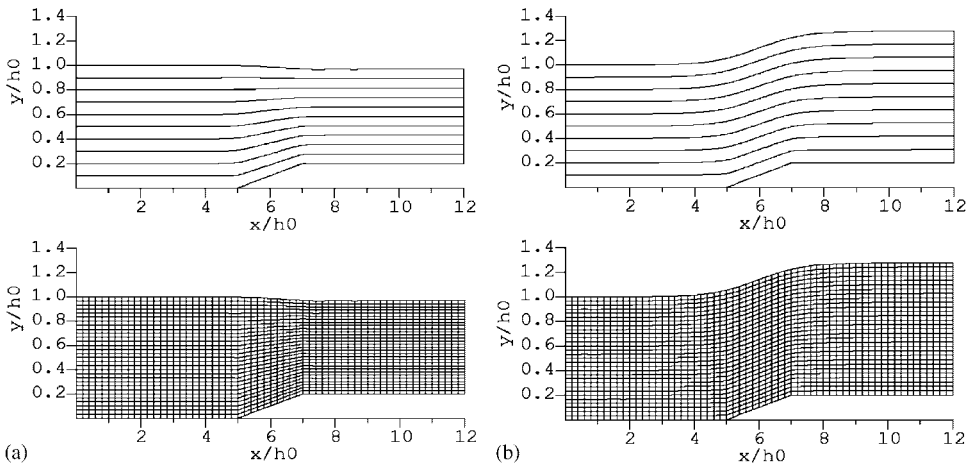

Figure 9. Flow over a ramp—streamlines and final grids: (a) subcritical ($Fr = 0.4$); (b) supercritical ($Fr = 2$).

where a subscript 0 denotes upstream flow conditions,

$$\mathrm{Fr} = \frac{u_0}{\sqrt{g h_0}} \tag{29}$$

is the upstream Froude number and z_b is the local bed height. (28) is easily inverted numerically by the Newton–Raphson method.

We examined both subcritical ($Fr = 0.4$) and supercritical ($Fr = 2$) approach flows, with a step height $z_b = 0.2 h_0$. A mesh of 60×30 control volumes was found to give a grid-independent solution. Since the final flow is steady, a simple time-marching procedure may be used to iterate toward steady state. First-order time-marching, with $\Delta t = 0.3\ h_0/u_0$, was employed, with a uniform adjustment to the entire surface profile at the end of each time step to maintain the inlet height at h_0. A convective boundary condition at outflow was found useful in minimising wave reflection and reducing the time required to achieve steady state.

The final solutions for subcritical and supercritical flow calculations are shown in Figure 9. The downstream heights in these two cases are 0.7687 and 1.0792 respectively, compared with values of 0.7689 and 1.0776 from 1-d theory.

3.4. Passage of a solitary wave over a submerged obstacle

Our fourth application example concerns the interaction of a solitary wave with a submerged bluff body. The application is useful both in confirming that our numerical method can pass a nonlinear wave without significant dispersion or loss in amplitude and as a practical problem for marine structure design. On encountering a submerged obstacle the wave is subjected to both inviscid effects (wave steepening, an increase in amplitude and partial reflection) and purely viscous phenomena (large vortices formed by the roll-up of separated shear layers from the obstacle corners; these vortices may, in practice, be responsible for significant scour). Such a test case has been computed elsewhere by finite differences; notably by Reference [15] using a streamfunction-vorticity formulation on a surface-conforming mesh and by Reference [16]

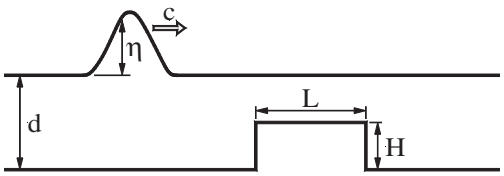

Figure 10. Interaction of a solitary wave with a submerged obstacle—geometry definition.

using a primitive-variable formulation on a Cartesian mesh, with free-surface tracking using a modified MAC scheme. In the former, dye-streak photographs from a laboratory flume are also available, revealing the formation and development of vortices in the lee of the obstacle.

The geometry of a solitary wave passing over a submerged rectangular obstacle of height H and length L is shown in Figure 10. d is the still water depth. The *inviscid* equations of motion admit finite-amplitude solitary waves ('solitons') of amplitude A, with surface elevation η and longitudinal velocity given by

$$\eta = A\,\mathrm{sech}^2(kx - \omega t)$$

$$U = \frac{c\eta}{d+\eta}$$

Here, the wave number $k = \sqrt{3A/4d^3}$ and phase speed $c \equiv \omega/k = \sqrt{g(d+A)}$. In our computations these profiles were applied at the inlet. Quantities are non-dimensionalized by the undisturbed water depth d, velocity scale $u_0 = \sqrt{gd}$ (the phase speed of long waves of vanishing amplitude) and the density of the fluid. In practice, viscosity will gradually attenuate the solitary wave. The Reynolds number is defined as $Re = u_0 d/\nu$. All calculations assume laminar flow.

Computations were performed on a 5-block grid, illustrated in Figure 11, where only the upper 3 blocks were allowed to deform in response to the motion of the free surface. The mesh was refined near the surface of the obstacle, with minimum cell dimensions $\Delta x/d = 0.008$ and $\Delta y/d = 8 \times 10^{-5}$ and some lesser refinement $\Delta y/d = 0.002$ near the free surface. The total number of cells varied between cases but was about 24000. A non-dimensional timestep of 0.01 was used throughout, although subsequent tests showed that, provided a second-order scheme was used for the free-surface update, a much larger value could have been used without introducing non-physical wave attenuation. The free-surface movement scheme adopted strategy I: seek to enforce zero net mass flux. If the alternative strategy of solving the kinematic equation was pursued a very small time step was necessary to prevent undershoot leading to a significant wave train behind the solitary wave.

Two cases were examined, corresponding to the experimental observations of Reference [15] and the computations of Reference [16], respectively.

Case (*a*): ('Short obstacle'): wave amplitude $A/d = 0.4$; obstacle dimensions $H/d = 0.5$, $L/d = 2$; Reynolds number 82 000.

Case (*b*): ('Long obstacle'): wave amplitude $A/d = 0.15$; obstacle dimensions $H/d = 0.5$, $L/d = 20$; Reynolds number 210 000.

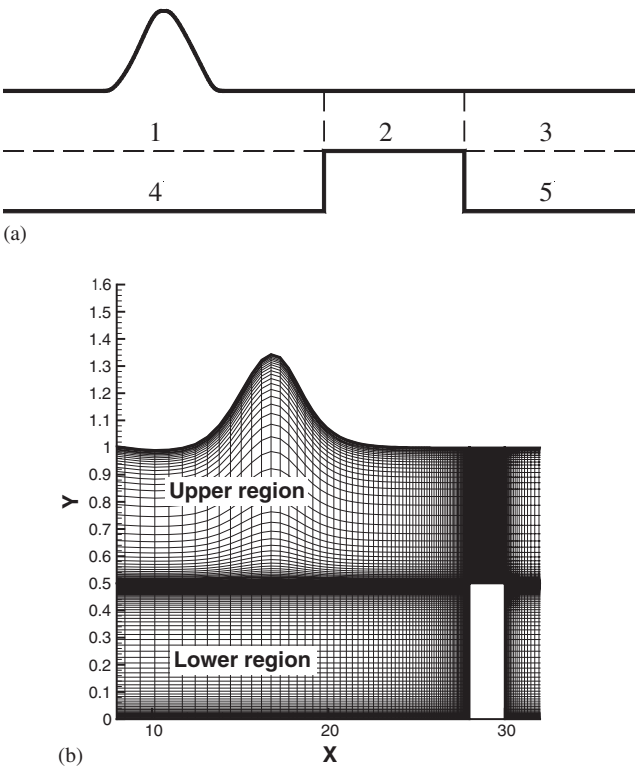

Figure 11. Interaction of a solitary wave with a submerged obstacle—computational mesh: (a) multi-block structure; (b) deformation under the wave.

The general behaviour of a solitary wave encountering each obstacle is illustrated in Figure 12. In Case (a) the obstacle is short compared with the wavelength λ (taken as $2\pi/k$, which corresponds roughly to the distance between points with elevation $1/100$ of the maximum): when $A/d = 0.4$, $\lambda/d \approx 11$. There is some wave reflection from the front face of the obstacle, but the wave passes largely unchanged. In Case (b) the wavelength is more closely matched to the length of the obstacle ($\lambda/d \approx 19$) and there is greater interaction. Wave reflection occurs from the upstream end of the obstacle with a smaller reflection (downward-going wave) from the downstream end. The wave steepens asymmetrically as its elevation increases over the obstacle. The distortion is such that two peaks of differing amplitude, and hence different phase speed, emerge and propagate downstream as distinct solitons. This confirms the findings of [16].

The vortical flow patterns around the obstacle in Case (a) are illustrated in Figure 13. The streaklines (Figure 13a) are directly comparable with the dye-streak photographs of Tang and Chang and reveal a lee vortex of size and location in good agreement with that visualisation. The instantaneous streamlines (Figure 13b) reveal a complex pattern of vortices. Secondary vortices at the base of a marine structure can give rise to considerable scour.

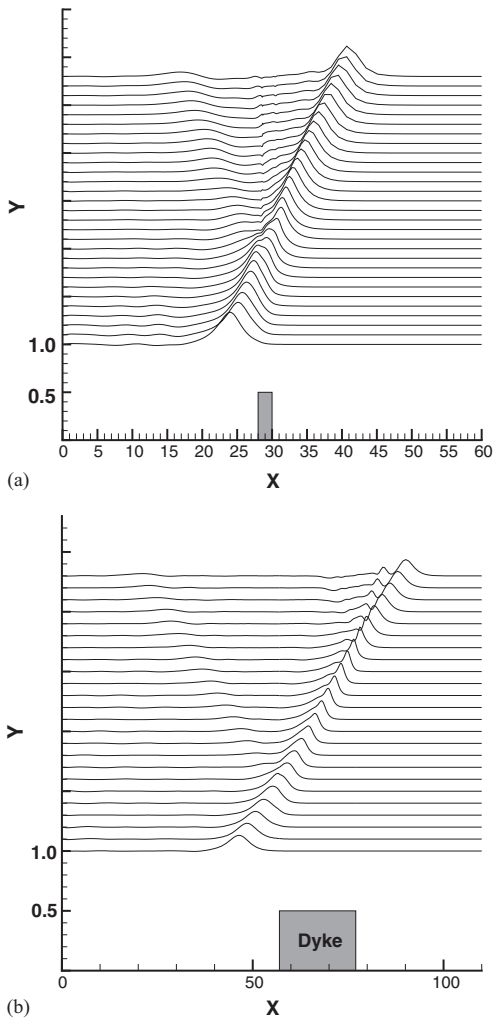

Figure 12. Passage of a solitary wave over a submerged rectangular obstacle—surface profile: (a) short obstacle; (b) long obstacle.

3.5. 3-d flow about a surface-penetrating cylinder

Whilst much computational effort has been expended on 2-d calculations of flow about a cylinder, focusing on the fluctuating drag and lift forces in flow about a body of effectively infinite length, the case of shear flow about a surface-penetrating cylinder has received comparatively little attention from the CFD community, despite its obvious practical application to structures such as bridge piers. Where it has been computed, the focus has tended to be on the scouring action of the horseshoe vortex that forms at the junction with the channel bed; the free surface has generally been treated by the rigid lid approximation (e.g. Reference [17]). More recently, LES calculations by Reference [18] investigated the effect of

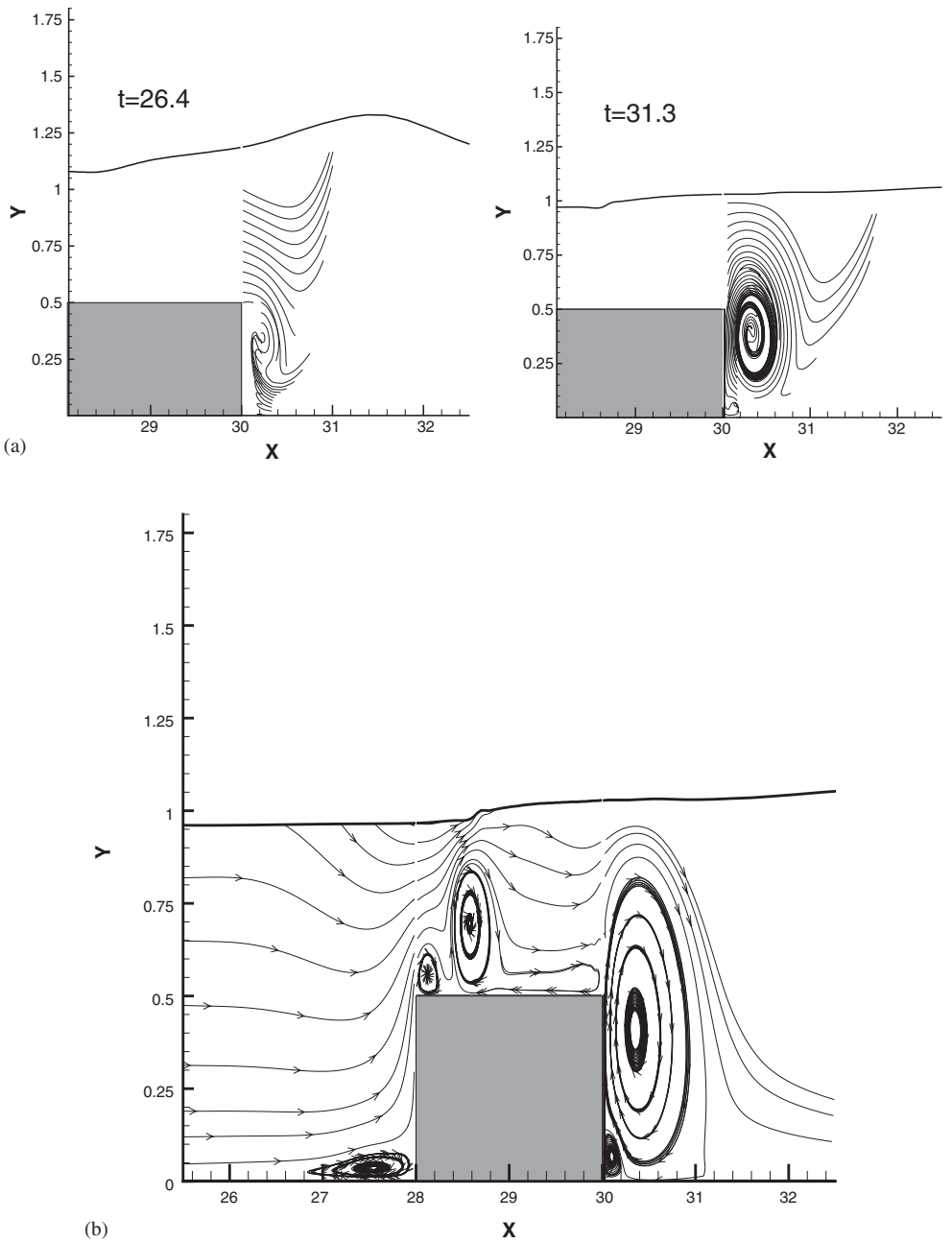

Figure 13. Passage of a solitary wave over a submerged rectangular obstacle—flow pattern: (a) streaklines; (b) instantaneous streamlines.

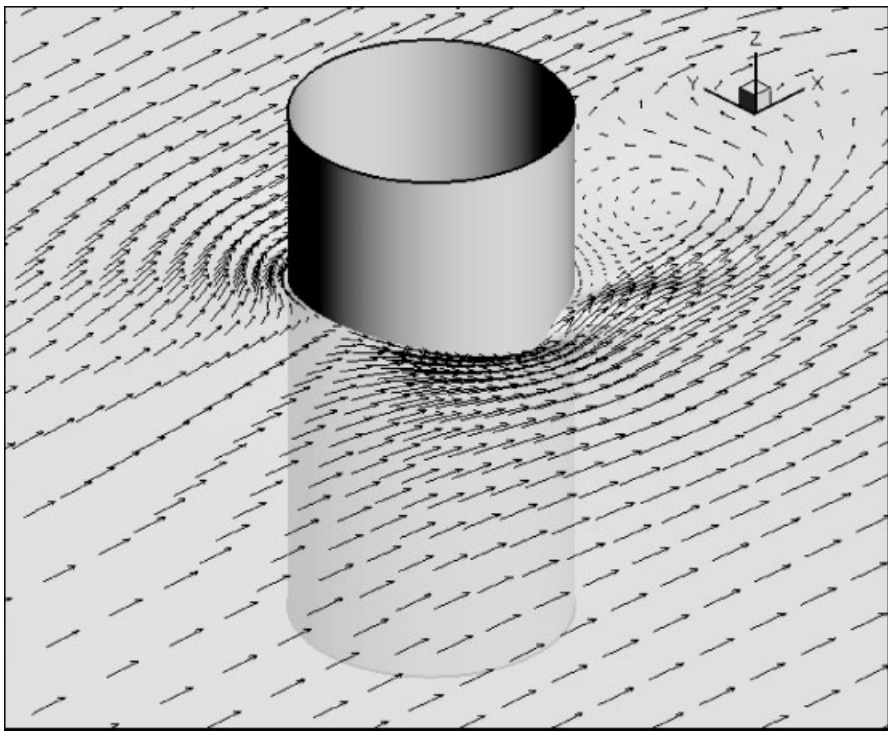

Figure 14. Overview of flow about a surface-penetrating cylinder.

free-surface motions on flow about a surface-penetrating cylinder, concluding that wave motions attenuate the periodic vortex shedding near the free surface to a depth of order one diameter for large (but subcritical) Froude numbers. However, their simulations used a uniform approach flow with symmetry condition on the lower boundary, so eliminating any junction vortex. The combination of three effects—a turbulent shear flow (giving rise to a horseshoe vortex), vortex shedding and free-surface motion—represents a challenge to numerical modellers.

We considered the turbulent shear flow past a surface-penetrating cylinder at a Reynolds number (based on bulk velocity U_b and cylinder diameter D) of 39 000 and Froude number ($Fr = U_b/\sqrt{gd}$, where d is the undisturbed flow depth) of 0.2. At this Froude number the free-surface motions are small and do not significantly affect the vortex-shedding process. A general overview of the computed flow field is shown in Figure 14.

Calculations were performed on the 12-block grid depicted in Figure 15. The mesh incorporated 10 layers of cells in the vertical, giving a total of 44 928 control volumes. Control volumes were stretched in the vertical according to the motion of the free surface. We make no claim to grid independence at this mesh density, our objective here being to develop and test the free-surface movement algorithm. The horseshoe vortex in particular is under-resolved and future work will examine this structure in greater detail.

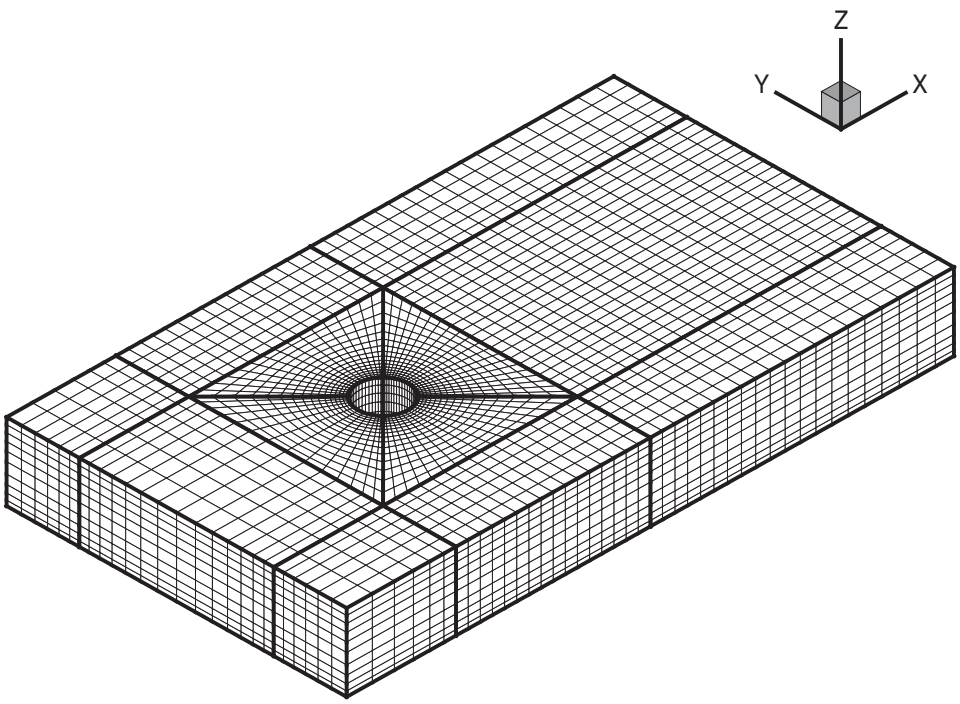

Figure 15. Surface-penetrating cylinder—computational mesh.

Flow variables were initialised at inflow from a preliminary 1-d fully developed flow calculation with the same bulk velocity (allowing a streamwise pressure gradient to drive the flow, rather than the small friction slope in the open-channel situation). The free-surface height was prescribed at the outflow boundary, with a convection boundary condition for other flow variables. Free-slip ('Euler wall') boundary conditions were applied at the side boundaries. Note that this gave a blockage ratio of 1/7, which can have an important impact on vortex shedding. Wall functions were used to apply non-slip boundary conditions at the channel bed and cylinder walls (which were both assumed smooth). We considered two high-Re turbulence models here: the 'standard' k–ε model of Reference [14] that is the mainstay of many commercial codes and the non-linear k–ε model of Reference [19]. The latter was formally developed as a low-*Re* closure, but for this test case was deployed without the near-wall viscous damping terms, since to resolve the semi-viscous sublayer on both channel bed and cylinder surface would have demanded excessive computational resources. Of these two closures, only the non-linear model gave rise to vortex shedding and it is these results which are included here. We discuss reasons for the lack of vortex shedding with the standard k–ε model below.

Experience from calculating 2-d flows about cylinders suggests that whilst periodic vortex shedding will ultimately arise as a natural instability of the flow it can take a considerable time to develop. It is common practice to employ some initial asymmetry to trigger vortex shedding, and a convenient method here was to spin the cylinder at a gradually diminishing rate up to non-dimensional time (tU_b/D) of 2.

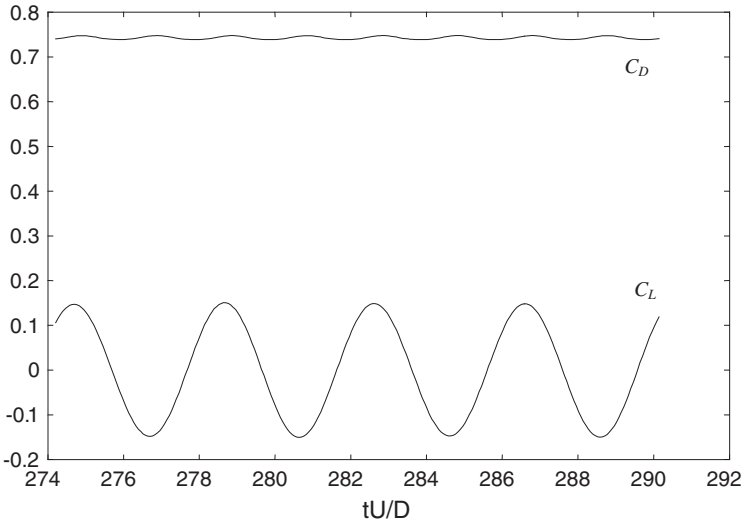

Figure 16. Surface-penetrating cylinder—drag and lift coefficients.

The flow was started impulsively, which led to some very large initial transients in the free-surface elevation. Of the various free-surface movement strategies that we tried, the only one which gave stable and smooth free-surface profiles was solution of the kinematic equation (Strategy II), with the free-surface elevation defined by control points. The relaxation technique discussed in 2.4.3 was vital in stabilising the free-surface movement algorithm for moderate sizes of non-dimensional timestep (here, 0.06 or approximately 1/64th of a period). A second-order time-marching scheme (Crank–Nicolson) was also necessary to produce significant vortex shedding with this timestep.

Figure 16 shows the drag and lift coefficients C_{D} and C_{L} after initial transients have decayed. These quantities are defined by

$$C_{\mathrm{D}} = \frac{\text{drag}}{\frac{1}{2}\rho U_{\mathrm{b}}^2 A}, \quad C_{\mathrm{L}} = \frac{\text{lift}}{\frac{1}{2}\rho U_{\mathrm{b}}^2 A}$$

where '*drag*' and '*lift*' are the integrated pressure+viscous forces in the streamwise and transverse directions and $A = hD$ is the projected area normal to the flow (in still water). These have the characteristic features of vortex shedding—the drag force fluctuates at twice the frequency of the lateral force and with a much smaller amplitude. The coefficients are smaller in amplitude ($C_{\mathrm{D}} \approx 0.74$) than experimental data for an effectively *infinite* cylinder at this Reynolds number and the Strouhal number ($S = fD/U_{\mathrm{b}}$) is slightly larger, at about 0.25. However, such comparisons are difficult for a number of reasons. In this case the approach flow is highly turbulent, incident velocity shear tends to decorrelate motions along the length of the cylinder, the junction vortex (about 15% of the depth of the channel) greatly reduces pressure variations near the bed and there is a significant blockage ratio.

Figure 17 shows shaded height contours superimposed on the surface velocity vectors. Surface elevation at the upstream impingement point is in accordance with that predicted from Bernoulli's theorem ($U^2/2g$). At this Froude number the pressure is nearly hydrostatic

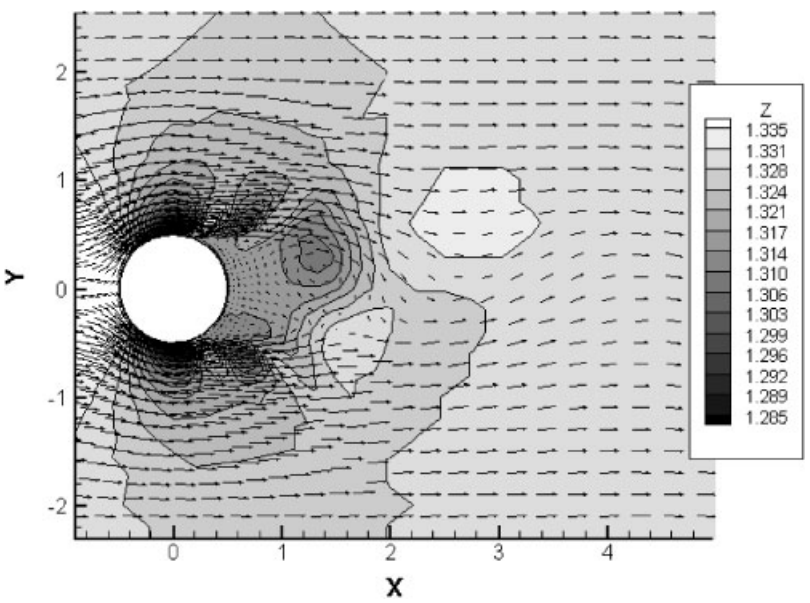

Figure 17. Surface-penetrating cylinder—surface elevation and flow field.

so that elevation changes tend to reflect pressure variations: notably the depressions near the centre of vortices. However, there are significant surface slopes near the separation points in particular and evidence of a sequence of waves generated from these.

Finally, Figure 18 explains why form drag is reduced and vortex shedding absent when the standard k–ε model is used by plotting the near-surface turbulent kinetic energy for the standard and non-linear models. With the standard model there is excessive and unphysical generation of turbulence energy upstream of the cylinder. This additional turbulence in the boundary layer acts to delay separation significantly. The reason for this excessive generation of turbulence may be found by noting that the ratio of turbulence production to dissipation is given, for a linear eddy-viscosity model, by

$$\frac{\text{production}}{\text{dissipation}} = C_\mu \bar{s}^2$$

where C_μ is the main coefficient in the eddy-viscosity relationship ($\nu_t = C_\mu k^2/\varepsilon$) and $\bar{s}^2 = 2\ (k/\varepsilon)^2 S_{ij} S_{ij}$ is an invariant of the mean strain tensor $S_{ij} = 1/2(\partial U_i/\partial x_j + \partial U_j/\partial x_i)$, non-dimensionalized by the turbulent timescale. If you like, $\bar{s}$ reflects the typical relative magnitude of mean velocity gradient. If C_μ is a constant (with a value of 0.09 in the standard model) the ratio of production to dissipation becomes very high in regions of high velocity gradient—such as impingement points. On the other hand, in the non-linear model of Reference [19], C_μ behaves like $1/\bar{s}^{3/2}$ for large irrotational strains, so significantly reducing turbulence generation. In addition, the non-linear stress–strain relation contains cubic terms whose action is (correctly) to depress turbulence in regions of convex curvature, again favouring flow separation from the sides of the cylinder.

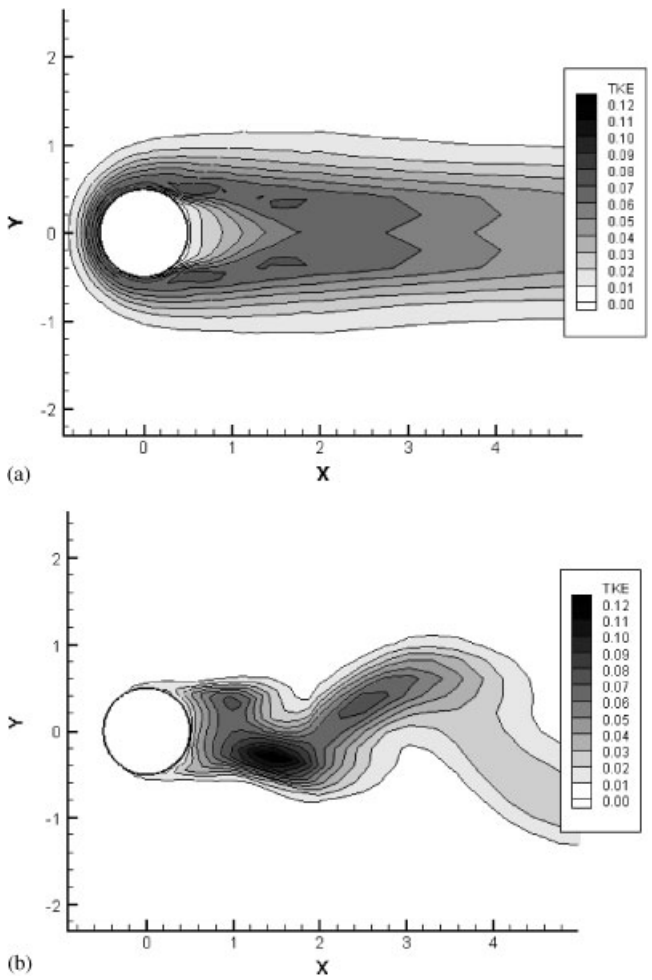

Figure 18. Surface-penetrating cylinder—near-surface turbulent kinetic energy (k/U_b^2): (a) standard k–ε model; (b) non-linear k–ε model.

4. CONCLUSIONS

The implementation of moving-mesh and free-surface algorithms within a finite-volume code has been described and illustrated by a number of two- and three-dimensional, fundamental and applied test cases. Calculations have been validated against simple theory (quasi 1-d flow; linear and non-linear waves), experiments and previous calculations that have used different numerical methods. Two strategies for enforcing the free-surface kinematic condition have been successfully implemented: iterating toward a condition of zero net mass flux or solving the kinematic equation by a finite-difference method. It is found that defining the free surface by intermediate control points is preferable to moving the mesh vertices directly and that a local under-relaxation technique seeking to limit the explicit update at each timestep can be

used to stabilise the free-surface algorithm in more intractable cases. To compute flows driven by free-surface height variations (notably waves on still water) with moderate sizes of time step a second-order time-marching scheme is vital in both scalar-transport and free-surface equations if unphysical wave attenuation is to be eliminated.

The methods which have been developed, coded and tested offer considerable potential for future work. Application to floating bodies and wave forces on offshore or coastal structures are examples of current interest. Computations of the 3-d surface-penetrating cylinder will be extended to higher Froude numbers and the complex interaction between horseshoe vortex, wake vortex and surface waves studied in more detail on finer meshes.

ACKNOWLEDGEMENTS

This work was supported financially by an EPSRC grant (GR/R06717).

REFERENCES

1. Longuet-Higgins MS, Cokelet ED. The deformation of steep surface waves on water. II. A numerical method of computation. *Proceedings of the Royal Society of London Series A* 1976; **350**:1–26.
2. Hibbert S, Peregrine DH. Surf and runup on a beach: a uniform bore. *Journal of Fluid Mechanics* 1979; **95**: 323–345.
3. Hirt CW, Nicholls BD. Volume of fluid (VOF) method for dynamics of free boundaries. *Journal of Computational Physics* 1981; **39**:201–221.
4. Harlow FH, Welsh JE. Numerical calculation of time-dependent viscous incompressible flow with a free surface. *Physics of Fluids* 1965; **8**:2182–2189.
5. Zwart PJ, Raithby GD, Raw MJ. The integrated space-time finite volume method and its application to moving boundary problems. *Journal of Computational Physics* 1999; **154**:497–519.
6. Thé JL, Raithby GD, Stubley GD. Surface-adaptive finite-volume method for solving free surface flows. *Numerical Heat Transfer, Part B* 1994; **26**:367–380.
7. Muzaferija S, Perić M. Computation of free-surface flows using the finite-volume method and moving grids. *Numerical Heat Transfer B* 1997; **32**:369–384.
8. Egelja A, Schäfer M, Durst F. An adaptive grid Eulerian method for the computation of free surface flows. *International Journal of Computational Fluid Dynamics* 1998; **10**:213–224.
9. Lien FS, Leschziner MA. A general non-orthogonal collocated finite volume algorithm for turbulent flow at all speeds incorporating second-moment turbulence-transport closure, part 1: computational implementation. *Computer Methods in Applied Mechanics and Engineering* 1994; **114**:123–148.
10. Lien FS, Chen WL, Leschziner MA. A multiblock implementation of a non-orthogonal, collocated finite volume algorithm for complex turbulent flows. *International Journal for Numerical Methods in Fluids* 1996; **23**: 567–588.
11. Apsley DD, Leschziner MA. Advanced turbulence modelling of separated flow in a diffuser. *Flow, Turbulence and Combustion* 1999; **63**:81–112.
12. Demirdžić I, Perić M. Space conservation law in finite volume calculations of fluid flow. *International Journal for Numerical Methods in Fluids* 1988; **8**:1037–1050.
13. Gear CW. *Numerical Initial Value Problems in Ordinary Differential Equations*. Prentice-Hall: Englewood Cliffs, NJ, 1971.
14. Launder BE, Spalding DB. The numerical computation of turbulent flows. *Computer Methods in Applied Mechanics and Engineering* 1974; **3**:269–289.
15. Tang C-J, Chang J-H. Flow separation during solitary wave passing over submerged obstacle. *Journal of Hydraulic Engineering* 1998; **124**:742–749.
16. Huang C-J, Dong C-M. On the interaction of a solitary wave and a submerged dike. *Coastal Engineering* 2001; **43**:265–286.
17. Tseng M-H, Yen C-L, Song CCS. Computation of three-dimensional flow around square and circular piers. *International Journal for Numerical Methods in Fluids* 2000; **34**:207–227.
18. Kawamura T, Mayer S, Garapon A, Sørensen L. Large eddy simulation of a flow past a free surface piercing circular cylinder. *Journal of Fluids Engineering* 2002; **124**:91–101.
19. Craft TJ, Launder BE, Suga K. Development and application of a cubic eddy-viscosity model of turbulence. *International Journal of Heat and Fluid Flow* 1996; **17**:108–115.

