Journal of Computational Physics **154,** 497–519 (1999)
Article ID jcph.1999.6324, available online at http://www.idealibrary.com on IDEAL®

The Integrated Space-Time Finite Volume Method and Its Application to Moving Boundary Problems

P. J. Zwart,* G. D. Raithby,* and M. J. Raw†

*Department of Mechanical Engineering, University of Waterloo, Waterloo, Ontario, Canada;
†AEA Technology Engineering Software Limited, Waterloo, Ontario, Canada
E-mail: pjzwart@asc.on.ca, graith@sunwise.uwaterloo.ca, mjraw@asc.on.ca

Received March 9, 1999; revised June 30, 1999

The integrated space-time finite volume method for predicting time-dependent fluid flow problems is developed. By enforcing discrete conservation over space-time control volumes which fill the space-time domain, this method satisfies global conservation in space-time while automatically satisfying the Leibnitz Rule and geometrical conservation law. The method is validated using a variety of two-dimensional problems featuring both prescribed and free boundary motion. Advances in other aspects of cell-centered finite volume discretization—most notably in the modelling of diffusion terms and free surface flows—are also described. © 1999 Academic Press

Key Words: space-time; finite volume; moving boundary; Navier–Stokes; free surface.

1. INTRODUCTION

The finite volume method has proven to be very successful for solving the equations of fluid dynamics. According to this method, the solution domain is filled with a mesh, which is used to define storage locations for each variable: typically these locations are either the mesh vertices (for *vertex-centered* methods) or the cell centroids (for *cell-centered* methods). Finite control volumes are constructed around each storage location, and the governing equations integrated over each control volume. The volume integrals are converted to surface integrals by means of Gauss' divergence theorem, and the surface integrals are approximated in terms of variables defined at the adjacent storage locations. By this process, the differential equations are replaced by algebraic equations: one for each conservation equation for each control volume.

The finite volume method is strictly conservative in the sense that global conservation is satisfied by the discrete equations. This follows provided the discrete transport through each internal face has the same magnitude but opposite sign for the two control volumes which share the face. Consequently, if the algebraic equations for the two control volumes are added together, the terms arising from the surface integral for the face they share must cancel.

For time-dependent problems, the finite volume principle has traditionally been used to discretize the spatial dimensions only. Time has been discretized using a finite difference procedure, such as the Euler or Runge–Kutta methods. If the mesh undergoes motion these methods require the use of the Leibnitz Rule to account for mesh motion. Global conservation is satisfied provided the geometrical conservation law (GCL) [3, 25, 27] is satisfied. If, however, the mesh topology changes with time (for instance by vertex insertion or removal), these methods are not conservative.

In this paper a new approach to enforcing global conservation for time-dependent problems is developed. It is based on discretizing time as well as space with the finite volume principle and is therefore called the *integrated space-time* (IST) finite volume method. With this method, the space-time solution domain is filled with a space-time mesh, which is used to construct space-time control volumes. The governing equations are integrated over each space-time control volume, and the volume integrals are converted to surface integrals using Gauss' divergence theorem. The IST finite volume method is conservative in space-time provided the discrete transport through each internal space-time face has the same magnitude but opposite sign for the two control volumes which share the face. Consequently, the Leibnitz Rule and GCL are *implicitly* satisfied, even if the mesh topology changes with time. The price to be paid for this flexibility is the need to generate a space-time mesh and discretize the equations in space-time. The IST concept was introduced and applied to one-dimensional problems in a previous paper [30]. The present paper generalizes the concept to multi-dimensions.

The IST finite volume method bears some conceptual resemblance to space-time finite element methods, which have existed for some time [6–8, 22, 23]. In order to retain a time-marching algorithm, these methods have introduced the notion of time slabs. The solution is made discontinuous across time planes, which bound the time slabs, using the discontinuous Galerkin method. With IST, we also use time slabs to obtain a time-marching scheme. The discontinuous Galerkin method is not needed, however, because IST is cell-centered in space-time, leading naturally to a discontinuous solution field at the time planes. Moreover, unlike most space-time finite element methods, our space-time meshing algorithm [29] avoids the need for global remeshes and solution projections.

Potential application of the IST finite volume method occurs wherever conservation in time is important. One may identify two particular cases: time-accurate mesh adaptation and moving boundary problems. In this paper moving boundary problems are emphasized. We consider both prescribed boundary motion and and free surface flows. The important additional factor for free surface flows is the kinematic condition. Like a number of other methods [12, 13, 24], we enforce this condition in a surface-adaptive manner by moving boundary vertices in such a way that the mass flows through the boundary are driven to zero; however, the mechanism for doing so is new.

The paper is organized as follows. In Section 2, our finite volume methodology will be described for steady flows, without the complication of space-time. That methodology will then be extended to space-time in Section 3. Section 4 will consider the application of IST to free surface flows, and Section 5 will provide some sample test cases.

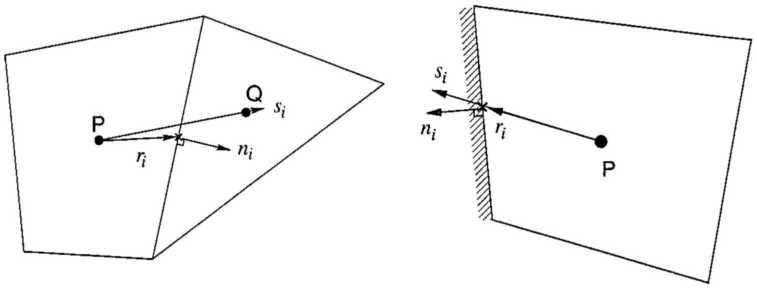


FIG. 1. Typical control volume faces and geometrical nomenclature. Left, internal face; right, boundary face.

2. DISCRETIZATION FOR STEADY FLOWS

In this section, our finite volume methodology for steady flow is described. An unstructured cell-centered method is chosen, where the mesh cells are the control volumes over which discrete conservation is enforced. Typical cells, together with interior and boundary faces, are illustrated in Fig. 1. Important vectors shown on the diagram include n_i (or $\hat{\mathbf{n}}$), the unit outward-directed normal; s_i (or $\hat{\mathbf{s}}$), the unit vector joining cell centroids; and r_i (or $\mathbf{r}$), the vector joining a cell centroid to the face midpoint.

2.1. *The Scalar Conservation Equation*

Before describing the finite volume method for the full Navier–Stokes equations, it is useful to consider the simpler *scalar conservation equation,* which represents the conservation of a generic scalar ϕ transported by advection and diffusion. The differential form of this equation at steady state is

$$\frac{\partial(\rho u_i \phi)}{\partial x_i} + \frac{\partial q_i}{\partial x_i} = 0, \tag{1}$$

where u_i is the (known) velocity in the x_i-coordinate direction, and

$$q_i = -\Gamma \frac{\partial \phi}{\partial x_i}, \tag{2}$$

Γ being the diffusion coefficient. By integrating this over a control volume Ω and using Gauss' divergence theorem, we obtain the integral form

$$\int_S \rho\phi u_i n_i \, dS + \int_S q_i n_i \, dS = 0. \tag{3}$$

For each control volume, the surface integrals are approximated using the midpoint rule, leading to discrete control volume equations of the form

$$\sum_{\mathrm{f}} \left(F_{\mathrm{f}}^{a} + F_{\mathrm{f}}^{d}\right) = 0. \tag{4}$$

F_{f}^{a} and F_{f}^{d} respectively represent the numerical approximations of the advective and diffusive transport through face f. These approximations in general involve solution values and solution gradients from the adjacent cells. Cell gradients are calculated using a least-squares technique [1].

The advective transport is given by $F_{\mathrm{f}}^{a} = J_{\mathrm{f}}\phi_{\mathrm{f}}$, where J_{f} is the mass flow through the face. An upwind-biased discretization is used for ϕ_{f},

$$\phi_{\mathrm{f}} = \phi_{\mathrm{up}} + \Phi\nabla\phi \cdot \mathbf{r}|_{\mathrm{up}}\,, \tag{5}$$

where the subscript up denotes the upwind cell value. One may choose: $\Phi = 0$, yielding a first-order method; $\Phi = 1$, which applies a second-order correction to ϕ_{up}; or a nonlinear expression to enforce boundedness near extrema [2, 26].

The diffusive transport is given by

$$F_{\mathrm{f}}^{d} = -\Gamma\nabla\phi \cdot \hat{\mathbf{n}}\, S_{\mathrm{f}}, \tag{6}$$

S_{f} being the face area. A second-order linearly exact discretization of this term has been developed by decomposing F_{f}^{d} into contributions along two components,

$$F_{\mathrm{f}}^{d} = -\Gamma(\nabla\phi \cdot (\alpha\hat{\mathbf{s}}) + \overline{\nabla\phi} \cdot (\hat{\mathbf{n}} - \alpha\hat{\mathbf{s}}))S_{\mathrm{f}}, \tag{7}$$

where

$$\nabla\phi \cdot (\alpha\hat{\mathbf{s}}) = \alpha\frac{\phi_Q - \phi_P}{\Delta s} \tag{8}$$

and $\overline{\nabla\phi}$ is the average of the adjacent cell gradients. The optimal decomposition is obtained by choosing $\alpha = \hat{\mathbf{n}} \cdot \hat{\mathbf{s}}$, in which case the two vectors $\hat{\mathbf{s}}$ and $\hat{\mathbf{n}} - \alpha\hat{\mathbf{s}}$ become orthogonal [30]. This choice also extends unambiguously to anisotropic continua (such as space-time).

The algorithm for implementing the above discretization is as follows:

(1) Assemble an algebraic conservation equation for each cell. The assembly is performed by looping over all faces, linearizing the fluxes through each face in terms of adjacent cell values, and scattering the coefficients and right-hand sides to the linear equations for the adjacent cells. This leads to a matrix equation of the form

$$[A]\{\phi\} = \{b\}. \tag{9}$$

(2) Calculate the residual of the old solution field as

$$\{r\} = \{b\} - [A]\{\phi^{o}\}. \tag{10}$$

Normalize the residuals. For cell P, the normalized residual is defined as

$$\hat{r}_P = \left|\frac{r_P}{a_P(\phi_{\max} - \phi_{\min})}\right|, \tag{11}$$

where a_P is its central coefficient and $\phi_{\max}$ and $\phi_{\min}$ are the maximum and minimum solution values.

(3) If the maximum normalized residual is reduced below its target value, stop. Otherwise solve the system of equations

$$[A]\{\delta\phi\} = \{r\} \tag{12}$$

and update the solution field. The system is solved using the algebraic multigrid solver of Raw [20].

(4) Return to step (1).

2.2. *The Navier–Stokes Equations*

The steady, incompressible Navier–Stokes equations consist of the continuity equation,

$$\frac{\partial u_i}{\partial x_i} = 0, \tag{13}$$

and the momentum equation,

$$\frac{\partial(\rho u_j u_i)}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ji}}{\partial x_j}. \tag{14}$$

In this formulation, p is the modified pressure (having the hydrostatic component removed) and

$$\tau_{ji} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right), \tag{15}$$

μ being the dynamic viscosity.

The integral forms of these equations are

$$\int_S \rho u_i n_i \, dS = 0, \tag{16}$$

$$\int_S \rho u_j n_j u_i \, dS + \int_S p n_i \, dS - \int_S \tau_{ji} n_j \, dS = 0. \tag{17}$$

In discrete form, the equations are

$$\sum_{\mathrm{f}} J_{\mathrm{f}} = 0, \tag{18}$$

$$\sum_{\mathrm{f}} \left(F^a_{\mathrm{f},i} + F^p_{\mathrm{f},i} + F^v_{\mathrm{f},i}\right) = 0, \tag{19}$$

where J_{f}, $F^a_{\mathrm{f},i}$, $F^p_{\mathrm{f},i}$, and $F^v_{\mathrm{f},i}$ respectively represent the numerical approximations of mass flow, advective momentum transport, pressure force, and viscous force at each face.

The advection term of the momentum equation is treated in the same manner as with the scalar equation. The viscous force is also discretized in the same manner as the diffusive flux for the scalar case. For the pressure term, a linearly exact centered discretization is used,

$$F^p_{\mathrm{f},i} = p_{\mathrm{f}} n_i S_{\mathrm{f}}, \tag{20}$$

with

$$p_{\mathrm{f}} = \frac{1}{2}(p_P + p_Q) + \overline{\nabla p} \cdot \mathbf{r}_{\mathrm{c}}, \tag{21}$$

where $\mathbf{r}_c$ is the vector from the midpoint between centroids P and Q to the face midpoint, and $\overline{\nabla p}$ is the average of the adjacent cell pressure gradients. An alternative choice would be to find the point R on the line joining centroids P and Q which is closest to the face midpoint, weight p_P and p_Q according their distance from R, and define $\mathbf{r}_c$ to be the vector from R to the face midpoint. In our experience, this modification has a negligible effect on the solution.

The mass flow through a face is $J_f = \rho u_{f,n} S_f$. It is important to discretize $u_{f,n}$ in a special manner in order to avoid pressure-decoupling [15]. Standard colocated approaches follow the lead of Rhie and Chow [21] in introducing a pressure gradient dependence into $u_{f,n}$. The particular form used here is

$$u_{f,n} = \bar{u}_{f,n} + \alpha d_f \left(\frac{p_Q - p_P}{\Delta s} - \overline{\nabla p} \cdot \hat{\mathbf{s}} \right), \tag{22}$$

where $\overline{u}_{f,n}$ is discretized in the same manner as p_f, $\overline{\nabla p}$ is the average of the adjacent cell pressure gradients, and

$$d_f = -\frac{1}{2}\left(\frac{\Omega_P}{a_P} + \frac{\Omega_Q}{a_Q} \right), \tag{23}$$

a being the average central coefficient of the discrete momentum equations and Ω_P and Ω_Q being the volumes of cells P and Q.

The matrix equation for this system has a block structure and is solved using the same solver as for the scalar case [20].

3. DISCRETIZATION FOR UNSTEADY FLOWS

In this section, the IST finite volume algorithm is presented. Just as discrete conservation equations were derived for spatial control volumes in the previous section, in this section discrete conservation equations will be derived for space-time control volumes. Before doing so, however, we will rewrite the governing equations in a more useful form and briefly describe our space-time mesh generation procedure.

3.1. *Mathematical Formulation*

With the IST finite volume method, the discretizations of space and time are unified. In doing so, it is helpful to unify the space and time terms of the governing equations. We begin with the conventional form of the unsteady scalar equation,

$$\frac{\partial(\rho\phi)}{\partial t} + \frac{\partial(\rho u_i \phi)}{\partial x_i} + \frac{\partial q_i}{\partial x_i} = 0. \tag{24}$$

The IST formulation requires the use of space-time vectors, which are distinguished from purely spatial vectors by a prime. As space-time vectors have an increased span, the subscript t is defined to be one more than the number of spatial dimensions. Thus the time coordinate is x_t'. By defining a “time velocity” $u_t' = 1$, the transient and advection terms of Eq. (24) are combined as

$$\frac{\partial(\rho\phi)}{\partial t} + \frac{\partial(\rho u_i \phi)}{\partial x_i} = \frac{\partial(\rho u_i' \phi)}{\partial x_i'}. \tag{25}$$

We must also ensure that the other terms maintain the special nature of time. For instance, diffusion occurs in space but not in time. The anisotropy of the space-time continuum is reflected by the definition of a space-time metric tensor γ'_{ij}:

$$\gamma'_{ij} = \begin{cases} 1 & \text{if } i = j \text{ and } i, j < t, \\ 0 & \text{otherwise} \end{cases} \tag{26}$$

Then Eq. (24) may be rewritten as

$$\frac{\partial(\rho u'_i \phi)}{\partial x'_i} + \frac{\partial q'_i}{\partial x'_i} = 0, \tag{27}$$

where

$$q'_i = -\gamma'_{ji} \Gamma \frac{\partial \phi}{\partial x'_j}. \tag{28}$$

In the same manner, the continuity and momentum equations may be written in IST form as

$$\frac{\partial u'_i}{\partial x'_i} = 0, \tag{29}$$

$$\frac{\partial(\rho u'_j u'_i)}{\partial x'_j} = -\gamma'_{ji} \frac{\partial p}{\partial x'_j} + \frac{\partial \tau'_{ji}}{\partial x'_j}, \tag{30}$$

where

$$\tau'_{ji} = \gamma'_{ki} \gamma'_{lj}\, \mu \left(\frac{\partial u'_k}{\partial x'_l} + \frac{\partial u'_l}{\partial x'_k} \right). \tag{31}$$

In the momentum equation, the free index i varies over the spatial dimensions. It is fascinating to observe, however, that if the time component is considered, the continuity equation is recovered. It is therefore possible to express the mass and momentum system as a single "space-time momentum equation." For our purposes, however, there does not seem to be any advantage in doing so, and the continuity and momentum equations will be considered separately.

3.2. *Space-Time Mesh Generation*

Just as conventional finite volume methods fill the spatial domain with spatial cells, so also the IST finite volume method must fill the space-time domain with space-time cells. Our space-time meshing algorithm for moving boundary problems is described in detail elsewhere [28, 29]. Briefly, the space-time domain is divided into time slabs, illustrated for one spatial dimension in Fig. 2. Each time slab is tessellated using a four-step procedure, illustrated in Fig. 3. In step (a), the lower spatial mesh is extruded in time. In step (b), the boundary vertices on the new time plane are moved to their new locations. Step (c) modifies the mesh topology next to the boundary, in order to maintain the mesh quality on the new time plane, by adding and removing vertices. This step may introduce new space-time cell shapes (triangles rather than quadrilaterals). The final step (d) is to perform some smoothing of vertices on the new time plane. This algorithm applies also to two dimensions, but with added complexity. The extrusion step produces triangular prism space-time cells.

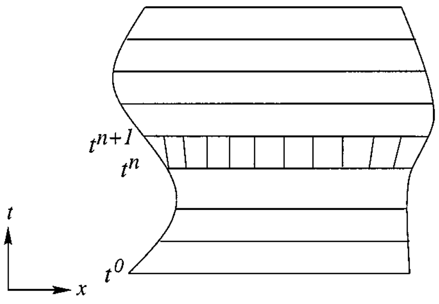


FIG. 2. Division of space-time domain into time slabs.

Adding and removing vertices near and on the boundary generates space-time tetrahedra and pyramids.

We anticipate still more complexity in extending the algorithm to three-dimensional problems, where four-dimensional space-time cells are required. In fact, it is our experience that space-time mesh generation is the most difficult aspect of IST. Before it can find widespread use, a more general space-time meshing strategy must be developed.

Typical faces which bound space-time control volumes for one spatial dimension are illustrated in Fig. 4. Important vectors shown on the diagram are n'_i (or $\hat{\mathbf{n}}'$), s'_i (or $\hat{\mathbf{s}}'$), and r'_i (or $\mathbf{r}'$), which are space-time extensions of the vectors shown in Fig. 1 for the steady algorithm. We also distinguish between *space-time faces*, which span the distance between time planes; and *time faces*, which lie on time planes.

3.3. *Scalar Conservation Equation*

The IST finite volume method for the scalar conservation equation begins by integrating Eq. (27) over each space-time finite volume Ω'. Since all terms are in divergence form, Gauss' divergence theorem is used to convert them to surface integrals,

$$\int_{S'} \rho u'_i n'_i \phi \, dS' + \int_{S'} q'_i n'_i \, dS' = 0, \tag{32}$$

where S' is the space-time surface bounding Ω' and n'_i is the outward-directed space-time normal to S' .

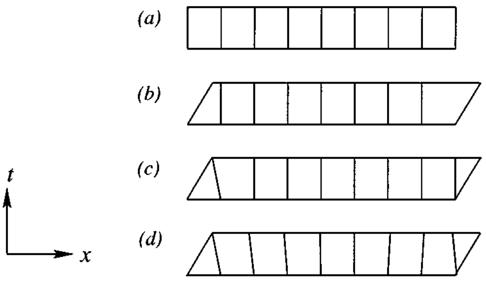


FIG. 3. Generating a mesh for a time slab: (a) extrude in time, (b) move boundaries, (c) add/remove vertices, (d) smooth.

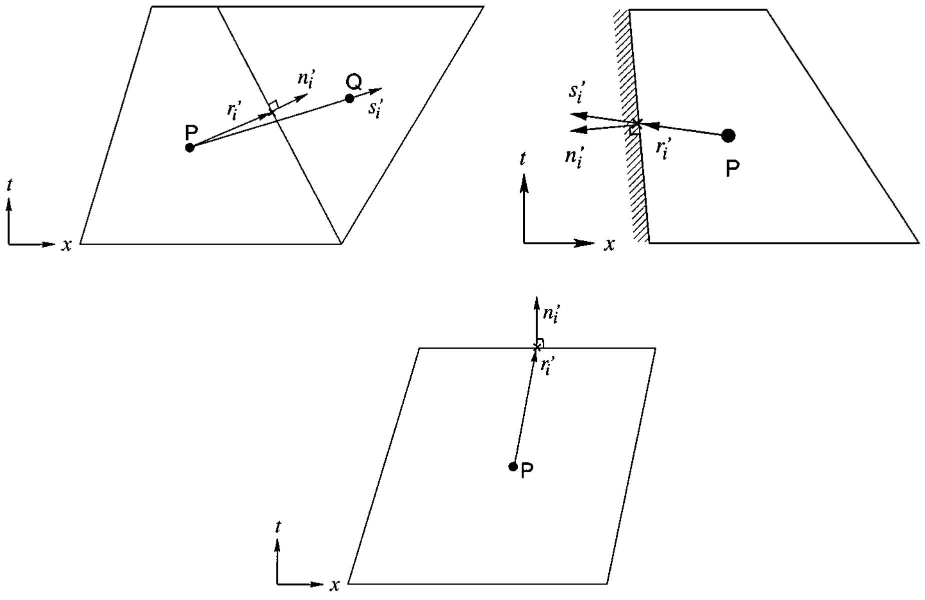


FIG. 4. Typical control volume faces in space-time. Top left, internal space-time face; top right, boundary space-time face; bottom, time face.

The surface integrals ae approximated at the midpoints of the space-time faces bounding the control volume. The resulting discrete equation is

$$\sum_{\mathrm{f}} \left(F_{\mathrm{f}}^{a} + F_{\mathrm{f}}^{d}\right) = 0, \tag{33}$$

where F_{f}^{a} and F_{f}^{d} respectively represent the approximations to the advective and diffusive transport through face f. They are written in terms of solution values and gradients (including the time derivative) at the adjacent cells. Cell gradients are calculated using the least-squares techniques, but with two complications. First, the least-squares stencil is one-sided in time, for the space-time cell neighbours from the next time level are not yet known. Second, the least-squares matrix may be degenerate for some space-time cell shapes and may have to be adapted to include additional points from the previous time slab.

The advective transport through a face is $F_{\mathrm{f}}^{a} = J_{\mathrm{f}}\phi_{\mathrm{f}}$. The mass transport J_{f} has distinctly different interpretations at space-time and time faces. At space-time faces, it represents the quantity of mass which crosses the face during the time slab, whereas at time faces, it represents the quantity of mass at that time level. At both faces, ϕ_{f} is obtained from the same upwind-biased approximation:

$$\phi_{\mathrm{f}} = \phi_{\mathrm{up}} + \Phi\nabla'\phi \cdot \mathbf{r}'|_{\mathrm{up}}. \tag{34}$$

It is interesting to note that choosing $\Phi = 0$ for time faces on orthogonal space-time meshes gives a discretization equivalent to the backward Euler method for the transient term, while $\Phi = 1$ is similar to a three-level second-order backward-difference scheme. The current framework, however, is capable of maintaining second-order spatial and temporal accuracy also on general space-time meshes. Nonlinear expressions for Φ may also be used. For instance, the limiter of Barth and Jesperson [2] may be extended to space-time by requiring that all ϕ_f around a cell be bounded by the space-time neighbours of the cell. The space-time neighbours from the next time slab must be excluded, however, for they are not yet known. As a result, spatial and temporal accuracy is reduced to first order not only near extrema but also wherever temporal gradients are large compared with spatial gradients.

The value of ϕ_f must be specified at all inflows into the space-time domain. This includes not only traditional inflow boundaries but also the initial time plane t^0, where $\mathbf{u} \cdot \hat{\mathbf{n}}' = -1$. The values of ϕ_f specified at these time faces are precisely the same as initial conditions specified in traditional methods.

The diffusive transport through a face is

$$F_\text{f}^d = \mathbf{q}' \cdot \hat{\mathbf{n}}' \, S_\text{f}, \tag{35}$$

where $\mathbf{q}'$ is a vector having components

$$q_i' = -\gamma_{ji}' \Gamma \frac{\partial \phi}{\partial x_j'}. \tag{36}$$

Unlike the corresponding term for steady flow, this expression involves an anisotropic medium. However, the discretization developed for steady flow may be extended to anisotropic situations in an unambiguous manner by recognizing that the diffusive flux is driven by the component of the gradient vector in the direction of the spatial component of $\hat{\mathbf{n}}'$. By defining a vector $\mathbf{m}'$ having components $m_i' = \gamma_{ji}' n_j'$, the diffusive transport becomes

$$F_\text{f}^d = -\Gamma \nabla' \phi \cdot \mathbf{m}' \, S_\text{f}. \tag{37}$$

This expression has the same form as Eq. (6) and may be decomposed in the same manner:

$$F_\text{f}^d = -\Gamma (\nabla' \phi \cdot (\alpha \hat{\mathbf{s}}') + \overline{\nabla' \phi} \cdot (\mathbf{m}' - \alpha \, \hat{\mathbf{s}}')) S_\text{f}. \tag{38}$$

The natural choice for the scaling factor α is $\alpha = \mathbf{m}' \cdot \hat{\mathbf{s}}'$. But at some highly nonorthogonal space-time faces, this choice may produce negative α, leading to convergence difficulties. Instead we choose $\alpha = \text{Max}(\mathbf{m}' \cdot \hat{\mathbf{s}}', 0)$. This diffusion discretization is valid not only for space-time, but also for general anisotropic diffusivities. Because of the special nature of γ_{ji}' in space-time, however, F_f^d is zero at time faces.

The algorithm used to implement this discretization is as follows:

(1) Generate the space-time mesh for the current time slab.

(2) Obtain the solution for the current time slab using the same algorithm as described for steady flows in Subsection 2.1. (Note that the face loop must include both time faces and space-time faces.)

(3) Return to step (1) until the specified stopping time has been reached.

3.4. *Navier–Stokes Equations*

Upon integration over the space-time control volumes, Eqs. (29) and (30) become

$$\int_{S'} \rho u_i' n_i' \, dS' = 0, \tag{39}$$

$$\int_{S'} \rho u_i' u_j' n_j' \, dS' + \int_{S'} \gamma_{ji}' \, p n_j' \, dS' - \int_{S'} \tau_{ji}' n_j' \, dS' = 0. \tag{40}$$

In discrete form, the equations are

$$\sum_{\mathrm{f}} J_{\mathrm{f}} = 0, \tag{41}$$

$$\sum_{\mathrm{f}} \left(F^{a}_{\mathrm{f},i} + F^{p}_{\mathrm{f},i} + F^{v}_{\mathrm{f},i} \right) = 0. \tag{42}$$

J_{f}, $F^{a}_{\mathrm{f},i}$, $F^{p}_{\mathrm{f},i}$, and $F^{v}_{\mathrm{f},i}$ respectively represent the mass flow, advective momentum transport, pressure force, and viscous force at each face.

The advection and viscous terms of the momentum equation are treated in the same manner as the corresponding terms from the scalar equation. The pressure force at space-time faces is a straightforward generalization of Eq. (20) to space-time,

$$F^{p}_{\mathrm{f},i} = \gamma'_{ji} p_{\mathrm{f}} n'_{j} S_{\mathrm{f}}, \tag{43}$$

where

$$p_{\mathrm{f}} = \frac{1}{2}(p_P + p_Q) + \overline{\nabla' p} \cdot \mathbf{r}'_{\mathrm{c}}. \tag{44}$$

The discretization of the mass flow J_{f} presented for steady flow also generalizes to space-time, but is slightly more involved. It is defined as $J_{\mathrm{f}} = \rho u'_{\mathrm{f},n} S_{\mathrm{f}}$, where $u'_{\mathrm{f},n} = \mathbf{u}'_{\mathrm{f}} \cdot \hat{\mathbf{n}}'$ is the perpendicular component of the space-time velocity vector. By defining $u_{\mathrm{f},n}$ to be the dot product of the spatial components of $\mathbf{u}'_{\mathrm{f}}$ and $\hat{\mathbf{n}}'$, n'_t to be the time component of $\mathbf{n}'$, and using the identity $u_t = 1$, the mass flow may be expressed as $J_{\mathrm{f}} = \rho(u_{\mathrm{f},n} + n'_t) S_{\mathrm{f}}$. The first term represents the spatial contribution to the mass flow, i.e., the mass which exits (or enters) the control volume due to fluid flow. The second term represents the temporal contribution, i.e., the mass left behind (or swallowed) as the face moves with time. At time faces, the spatial contribution vanishes, so $J_{\mathrm{f}} = \rho S_{\mathrm{f}}$. At space-time faces, $u_{\mathrm{f},n}$ is discretized in a similar manner as with steady flow, but modified slightly to ensure time-step independence at steady state [10].

4. FREE SURFACE FLOWS

Free surface flow problems differ from those with prescribed boundary motion in the conditions which must be applied to the free surface. The momentum balance at the free surface is closed with the *dynamic conditions*, which state that the forces at the interface are in equilibrium. They are not difficult to apply.

The greater difficulty lies with the *kinematic condition*, which is used to determine the interface position. It specifies that no mass flows through the free surface faces:

$$J_{\mathrm{f}} = \rho \mathbf{u}' \cdot \mathbf{n}' S_{\mathrm{f}} = 0. \tag{45}$$

We implement this condition in a surface-adaptive framework, wherein the mesh conforms to the free surface. By satisfying the kinematic condition directly, rather than switching to a Lagrangian formulation at the free surface, overall conservation is strictly enforced.

This approach is similar to that developed by others [12, 17, 24] in a conventional finite volume context. Enforcing this form of the kinematic condition for each face is awkward in a cell-centered context, however, because the number of vertices which define the free surface may differ from the number of free faces. Workarounds have been developed by staggering the control volumes below the free surface [24] or adding control points to the free faces [12]. We propose a new approach, wherein the kinematic condition is not enforced for the faces, but rather for the vertices. Define ξ_i to be the set of free faces which touch a free vertex and w_{ji} to be a weighting factor expressing the fraction of the mass flow through face j which is apportioned to vertex i. Then the condition we enforce is $F_i = 0$, where

$$F_i = \sum_{j \in \xi_i} J_j w_{ji} \tag{46}$$

is the mass which crosses the portion of the free surface associated with vertex i. We choose $w_{ji} = 1/n$, n being the number of vertices which touch face j.

The algorithm used to link the solution of the kinematic condition and the hydrodynamic equations is:

(1) Solve the continuity and momentum equations, treating the free surface as a pressure boundary.

(2) Calculate the residual mass flow $r_i = F_i$ for all vertices on the free surface.

(3) Move the vertices on the free surface so that $F_i = 0$. The procedure for this is described below.

(4) Return to step (1) until the solution for the current time slab is converged.

Step (3) involves assembling and solving a matrix for the vertex displacements. Define Δs_k to be the displacement of vertex k in a specified direction (which we take to be the perpendicular direction) which will drive the residual mass flows (defined in Step (2)) to zero. Newton's method is used to obtain the displacements as

$$\frac{\partial F_i}{\partial s_k} \Delta s_k = -r_i. \tag{47}$$

The Jacobian matrix $\partial F_i / \partial s_k$ is constructed numerically by shifting each vertex by a small amount and calculating the change in the mass flows for the surrounding vertices. In two-dimensions, the matrix is tridiagonal.

On its own, this algorithm is not stable. In particular, the case of there being fewer free faces than vertices results in a singular matrix, which manifests itself as unconstrained wiggles in the free surface profile. Problems also exist when there are enough free faces. An analysis of the dominant effects on the matrix indicates that a typical row has the form

$$-\frac{\rho l}{4}(\Delta s_{i-1} + 2\Delta s_i + \Delta s_{i+1}) = -r_i, \tag{48}$$

where l is the length of the edges on the free surface. The resulting solution is prone to wiggles because the homogenized system admits solutions such as $\Delta s_i = (-1)^i$.

Both types of wiggles are high-frequency (having the same frequency as the mesh) and are eliminated using a new procedure called *mass redistribution* along the free surface. Consider the wiggly free surface shown in Fig. 5. The wiggle will be damped if the mass flux F_P for vertex P is decreased by an amount J^r and the mass flux F_Q for vertex Q is increased by the same amount. In essence, mass is redistributed between vertices across

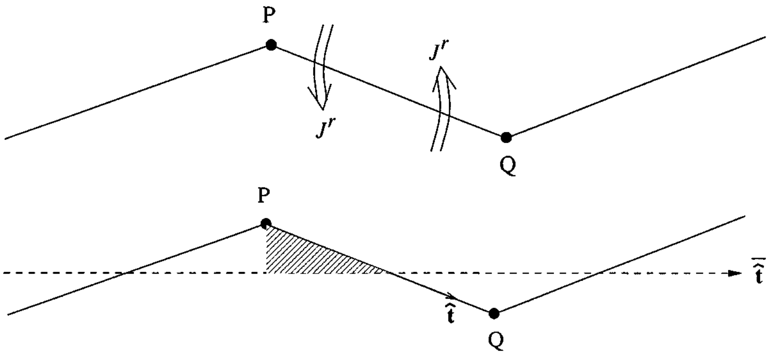


FIG. 5. Top, a free surface wiggle; bottom, proposed damping mechanism.

free surface faces. If J^r_{ji} represents the redistributed mass to vertex i across face j, the mass flow through the portion of the free surface associated with vertex i is redefined as

$$F_i = \sum_{j\in\xi_i} \left(J_j w_{ji} + J^r_{ji}\right). \tag{49}$$

It remains to find an expression for the redistributed mass J^r_{ji}. From Fig. 5, it is clear that a wiggle leads to a difference between the local edge tangent vector $\hat{\mathbf{t}}$ and the mean edge tangent vector $\bar{\hat{\mathbf{t}}}$. (The mean edge tangent is determined as the average of the vertex tangents which touch the edge, where each vertex tangent is itself the average of the edge tangents which it touches.) The wiggle will be eliminated when the mass associated with the area of the cross-hatched triangle is redistributed from vertex P to vertex Q. If A is the triangle area, then

$$\begin{aligned} J^r &= \rho A \\ &\approx \rho \frac{l^2}{8}(\hat{\mathbf{t}} \times \bar{\hat{\mathbf{t}}}). \end{aligned} \tag{50}$$

Then $J^r_{jP} = -J^r$ and $J^r_{jQ} = J^r$.

With this algorithm, the kinematic condition is not satisfied for the faces. It is satisfied, however, both globally and in the neighbourhood of each vertex provided that, for every face j, $\sum_{i\in\psi_j} w_{ji} = 1$ and $\sum_{i\in\psi_j} J^r_{ji} = 0$, where ψ_j is the set of free vertices which touch the face. A simple proof of this is found in [28].

5. TEST CASES

The IST solver was tested using a number of problems having analytical solutions [28, 30]. Mesh refinement studies using these problems showed second-order accuracy in space and time if no limiting of the advection terms was performed and first-order accuracy if limiting was performed. The following test cases illustrate the behaviour on more difficult problems. These cases were executed on a SPARC Ultra 10 processor running at 300 MHz with 1 GB RAM.

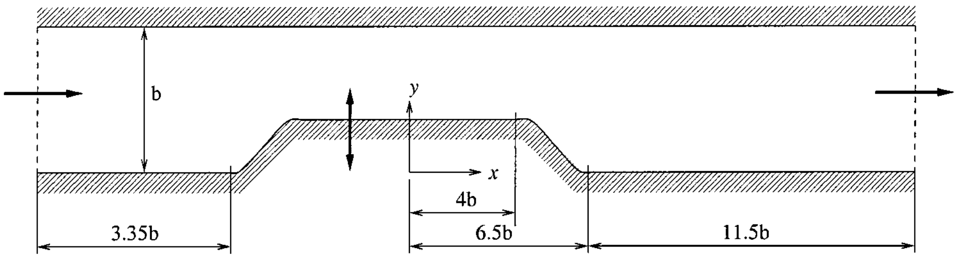


FIG. 6. Geometry of moving indentation test case, with $b = 1$.

5.1. *Duct with a Moving Indentation*

This section describes a test case involving prescribed boundary motion. Consider a channel featuring a moving indentation in one wall. Experimental studies of this type of flow have been carried out by Pedley and Stephanoff [16], and numerical studies have been performed using a vorticity-stream function approach [18] and a finite volume method [4]. The geometry is shown in Fig. 6. The oscillation period is T and the normalized time is $t^* = t/T$. The height of the indentation at a particular time is $h = .19(1 - \cos(2\pi t^*))$, and the curved portion of the lower wall is described by

$$y = \begin{cases} 0.5h(1 - \tanh(4.14(x - 5.25))) & \text{if } 4b < x < 6.5b, \\ 0.5h(1 + \tanh(4.14(x + 5.25))) & \text{if } -6.5b < x < -4b. \end{cases} \tag{51}$$

As with the previous numerical studies, only the first cycle of the flow is solved. Fully developed conditions are assumed at $t = 0$. Solutions have been obtained on a coarse mesh (having initially 6622 triangles and $\Delta t^* = 0.02$) and a fine mesh (having initially 25,896 triangles and $\Delta t^* = 0.01$). The spatial meshes on various time planes around the downstream end of the indentation for the coarse run are given in Fig. 7. The coarse mesh requires an average of 15 iterations per time step to reduce all normalized residuals below 10^{-4}, leading to a total CPU time of 35 minutes. The fine mesh requires an average of 14 iterations per time step, leading to a total CPU time of 5 h.

Normalized shear stresses along the lower channel wall are plotted for both the coarse and fine mesh runs in Fig. 8. The plot shows that a mesh-independent result has not yet been attained, although the qualitative behaviours are similar. These solutions have roughly the same accuracy as other computations [4]. It should be noted that the high-frequency kinks in the shear stress profiles are induced by the irregular space-time cells where vertices have been added or removed.

5.2. *Breaking Dam*

The next test case involves the collapse of a two-dimensional column of fluid. Experiments of this nature using several configurations were performed some time ago by Martin and Moyce [11]. It is also a common numerical test case [8, 9, 14,19]. We consider the case having an initial aspect ratio $h_0/w_0 = 2$, where h_0 is the initial column height and w_0 is the initial width. As viscous effects are negligible, the flow is modelled as inviscid. A dimensionless time is defined as $t^* = \sqrt{2g/w_0}t$ and a dimensionless front position as $w^* = w/w_0$. The problem is solved in the time range $0 < t^* < 5$ using a coarse mesh (having initially 130 cells and $\Delta t^* = 0.05$) and a fine mesh (having 500 cells and $\Delta t^* = 0.025$). The

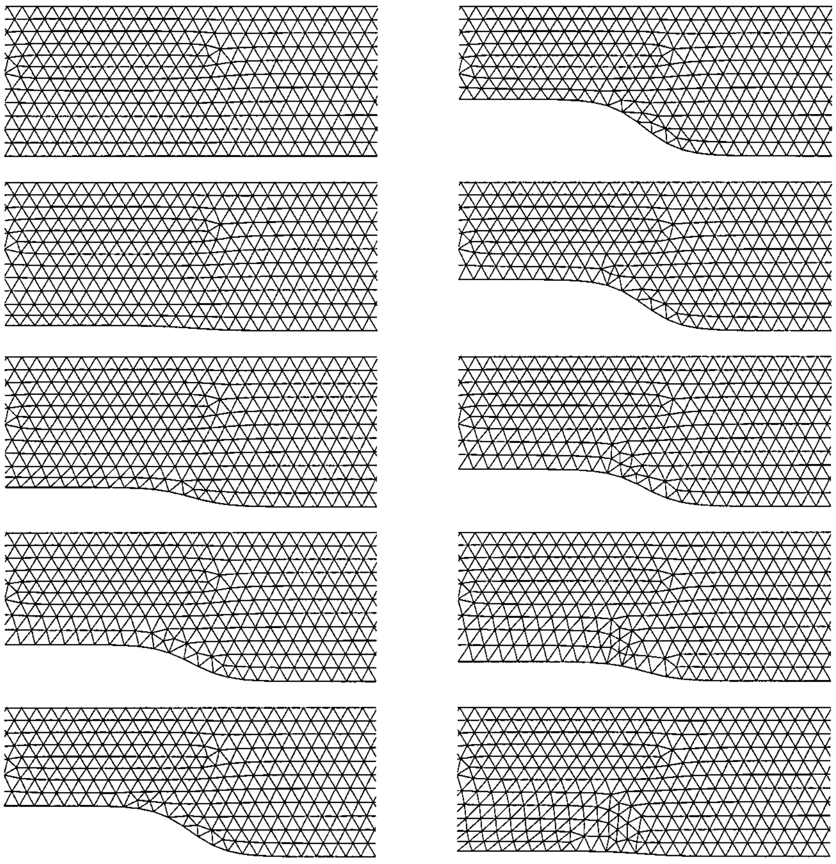


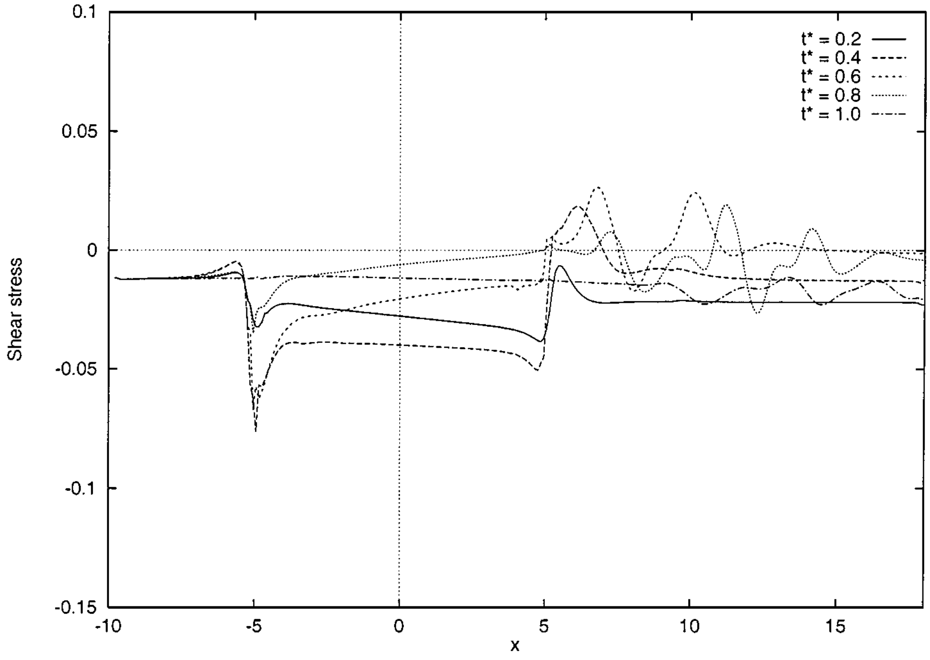
FIG. 7. Mesh in the downstream vicinity of the indentation for the moving indentation test case on time planes $t^* = 0.1, 0.2, \ldots, 1.0$. The sequence runs by column.

spatial mesh for the fine grid is shown for various time levels in Fig. 9. The figure clearly demonstrates the capacity of the method to handle large changes in the free surface while maintaining mesh quality. The coarse mesh requires an average of 8 iterations per time step to reduce all normalized residuals below 10^{-3}, leading to a total CPU time of 37 s. The fine mesh requires an average of 6 iterations per time step, leading to a total CPU time of 4.3 minutes.

To compare the numerical results with the experiments, a plot of front position w^* against time is given in Fig. 10. It is important to point out that the experimental results have undergone a time shift of $\Delta t^* = 0.3$ in order to compensate for uncertainties in the time origin. This shift is consistent with what is performed in other numerical studies in the literature. With this shift, there is excellent agreement between the experimental and computed results. The plot also shows that the solution is nearly mesh-independent.

5.3. *Overturning Wave*

A final test case illustrates the capability of this free surface algorithm on a challenging flow—an overturning wave. The wave is generated in a water channel by a piston wavemaker, as described by Dommermuth *et al.* [5]. The piston has a time-varying frequency, amplitude, and phase carefully chosen to generate a plunging breaker some distance downstream. These



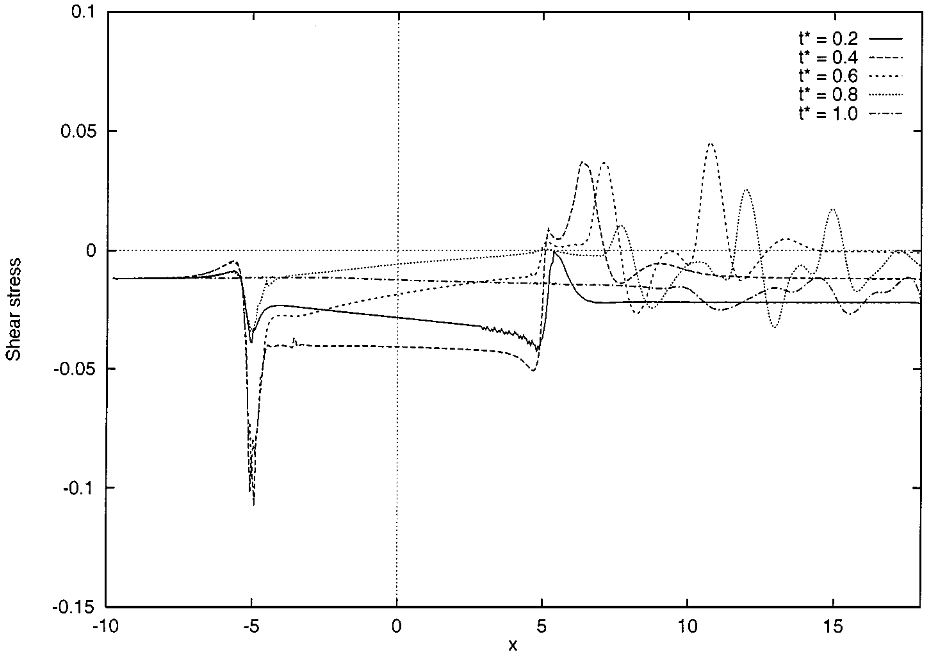


FIG. 8. Shear stress profiles along the lower channel wall for the moving indentation test case. Top, coarse grid; bottom, fine grid.

authors performed both an experimental study and numerical calculations using a nonlinear panel method. The wave channel geometry is illustrated in Fig. 11. The dimensions are normalized by the undisturbed water height, so that the height is 1 m and the length 20 m. Inviscid flow is assumed, and the density and gravitational constant are set to unity.

In the present calculations, the mesh has initially 23,170 triangles, which are concentrated near $x = 12$ where the wave overturns. The spacing also decreases with time in order to

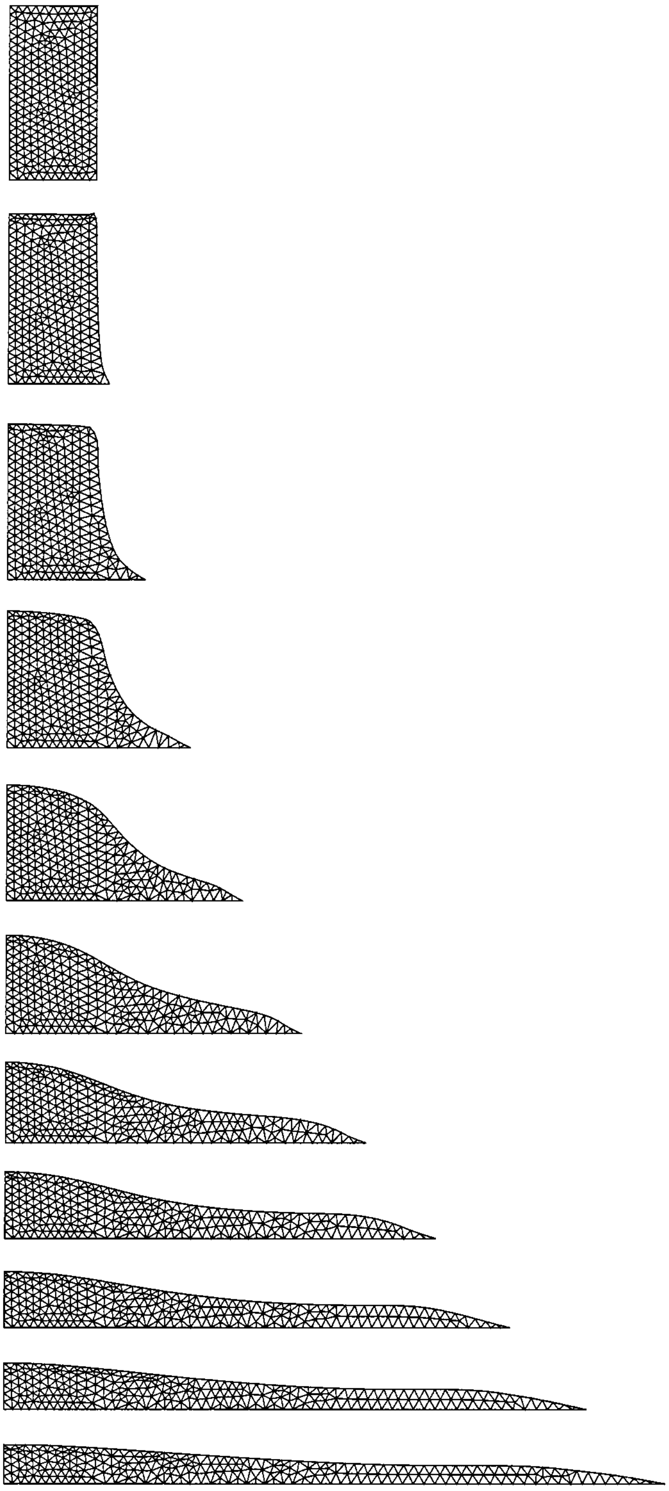


FIG. 9. Mesh development for breaking dam test case on time planes $t^* = 0, .5, 1, 1.5, \ldots, 5.0$.

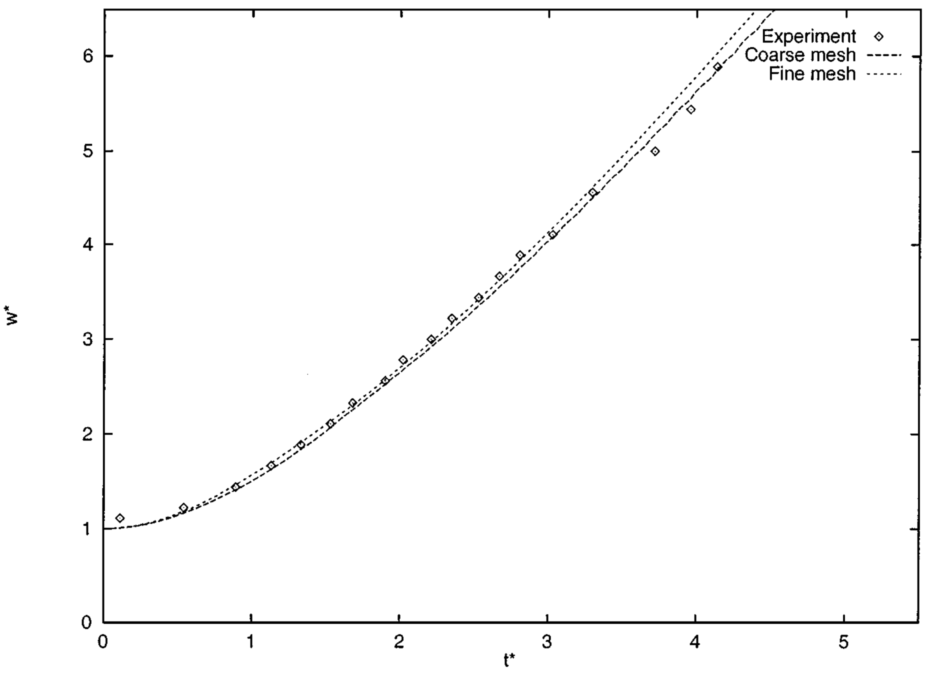


FIG. 10. Comparison between calculated and experimental results for the breaking dam test case. The plot shows the evolution of the dimensionless front position w^* with dimensionless time t^*.

obtain adequate resolution of the overturning wave: at $t = 51.76$, when the computations end, there are over 60,000 triangles. The initial time step is 0.1 and decreases with time in an adaptive manner, where no boundary vertex may move more than a specified fraction of its local spacing in one time step. The time step at the end of the computations is about 0.00035, and a total of 3200 time slabs are required. An average of four iterations per time slab is required to reduce all normalized residuals below 10^{-3}, leading to a total CPU time of about 49 h. Most of this effort is required for the overturning phase. The collision phase is not simulated because the space-time meshing algorithm does not allow for surface reconnection.

The experimental and numerical results provided by Dommermuth *et al.* [5] include surface elevations at various locations along the wave channel. The elevations at the same locations using the current results are plotted in Fig. 12. The agreement with the experimental data, also included in the plot, is excellent. Outlines of the predicted surface profile during the overturning stage are plotted for various times in Fig. 13. Figure 14 shows a close-up of the mesh at $t = 51.75$, just before collision.

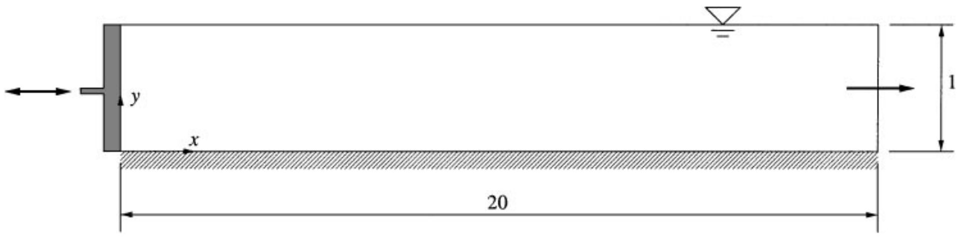


FIG. 11. Wave channel geometry used in overturning wave test case.

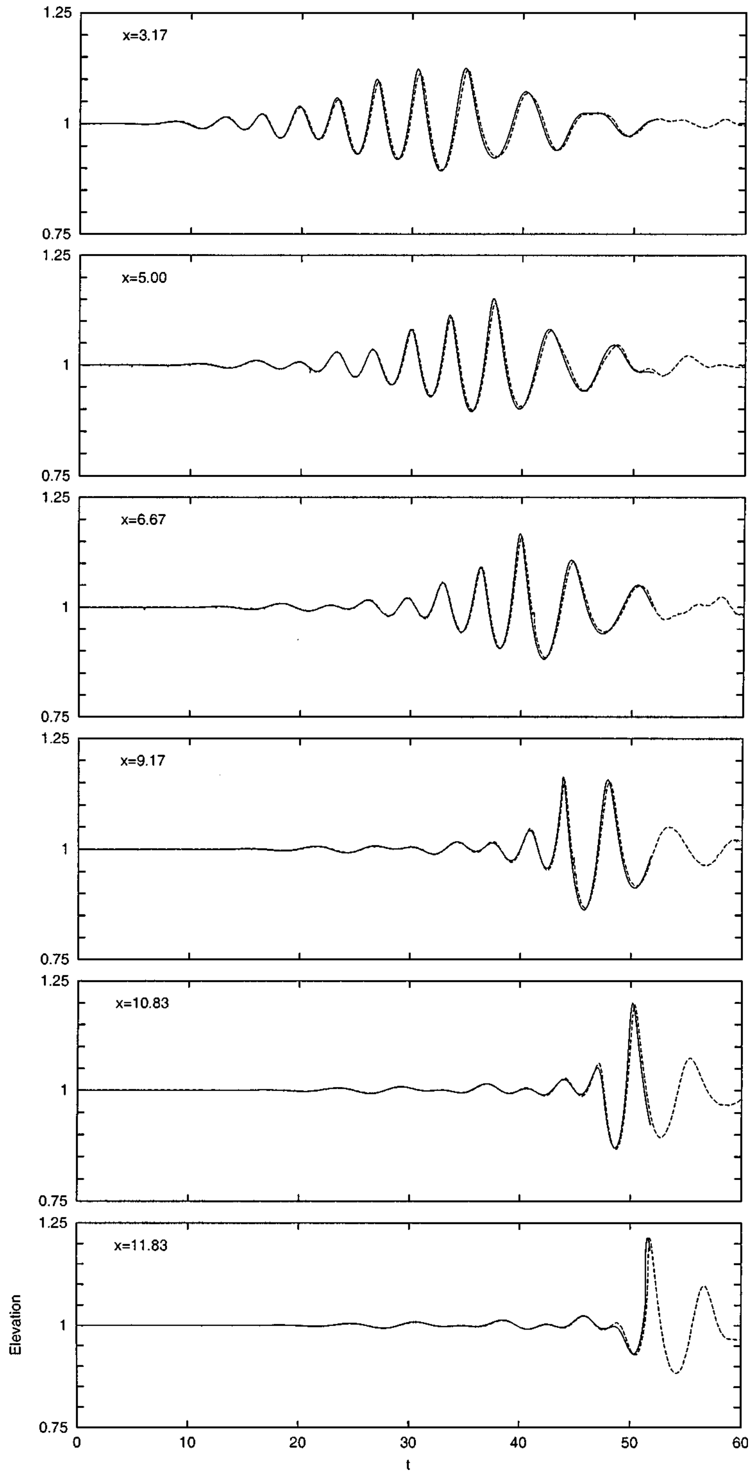


FIG. 12. Free surface elevations against time at various locations in the wave channel for the overturning wave test case. —, computed; ---, experimental.

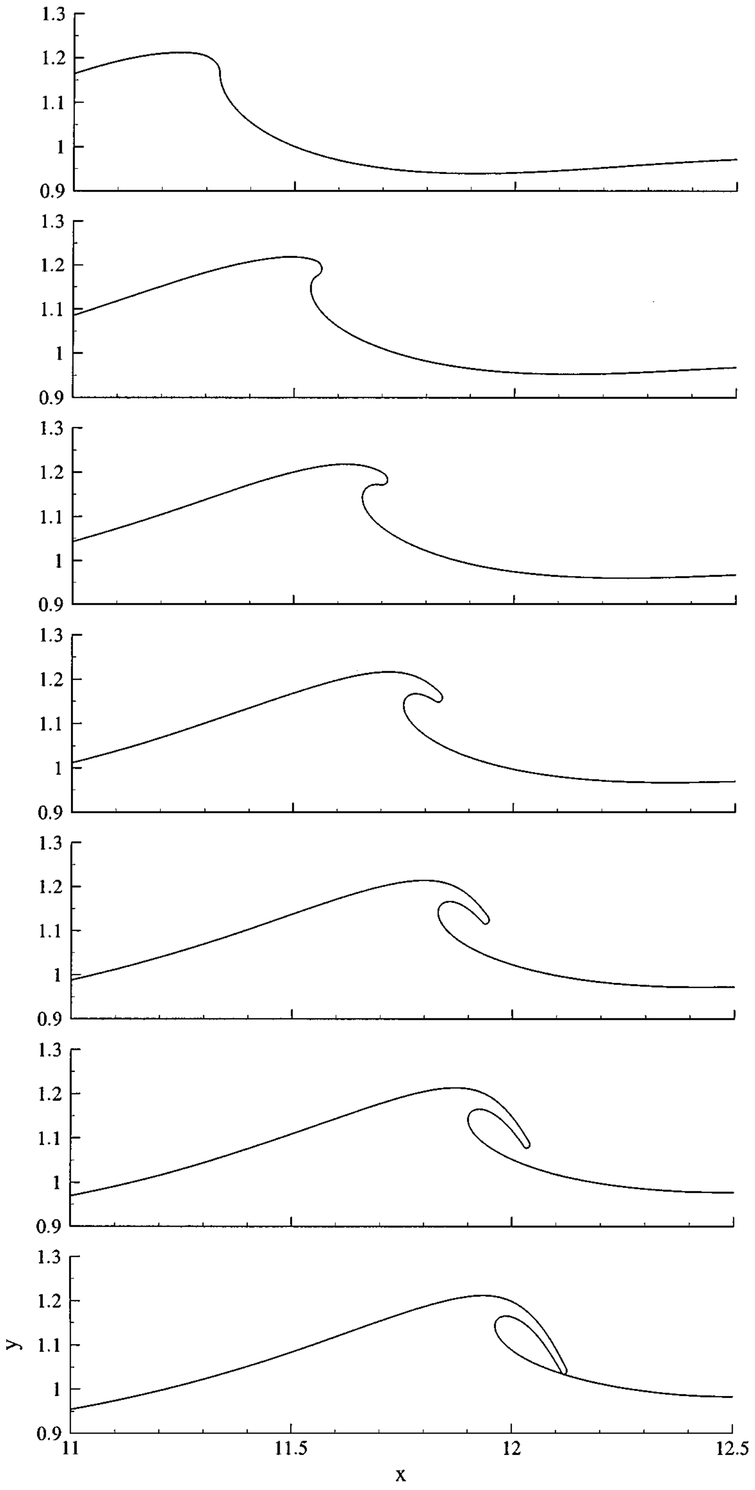


FIG. 13. Outlines of the overturning wave at times $t = 50.70, 51.05, 51.24, 51.40, 51.54, 51.65, 51.76$.

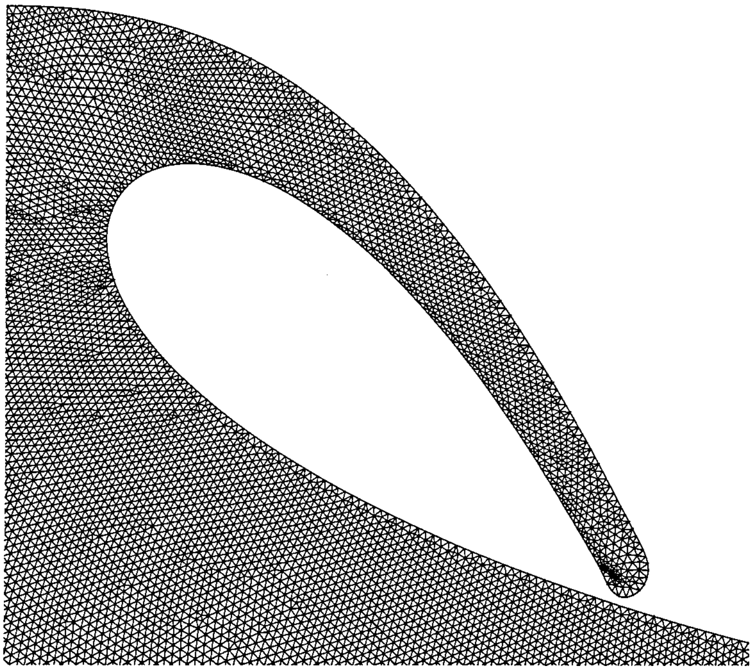


FIG. 14. Close-up of the overturning wave at $t = 51.75$.

6. CONCLUSIONS

The integrated space-time (IST) finite volume method for unsteady flows, based on a unified discretization of space and time, has been developed. The computational benefits of time-marching are retained by subdividing the space-time domain into time slabs. The method is strictly conservative, even when mesh points are added or removed. The method has been tested for moving boundary problems, including free surface flows. For this class of flow, a new procedure for satisfying the kinematic condition has been developed. For the future, IST also holds promise for time-accurate conservative mesh adaptation and three-dimensional moving-boundary problems provided that a suitably general space-time meshing strategy can be developed.

ACKNOWLEDGMENTS

The authors are grateful to the Natural Sciences and Engineering Research Council of Canada (NSERC) and AEA Technology Engineering Software Ltd. for financial support and to Douglas Dommermuth for providing the experimental data for the overturning wave test case.

REFERENCES

1. T. J. Barth, *A 3-D Upwind Euler Solver for Unstructured Meshes*, AIAA Paper 91-1548, 1991.
2. T. J. Barth and D. C. Jesperson, *The Design and Application of Upwind Schemes on Unstructured Meshes*, AIAA Paper 89-0366, 1989.

3. I. Demirdžić and M. Perić, Space conservation law in finite volume calculations of fluid flow, *Int. J. Numer. Methods Fluids* **8**, 1037 (1988).
4. I. Demirdžić and M. Perić, Finite volume method for prediction of fluid flow in arbitrarily shaped domains with moving boundaries, *Int. J. Numer. Methods Fluids* **10**, 771 (1990).
5. D. G. Dommermuth, D. K. P. Yue, W. M. Lin, R. J. Rapp, E. S. Chan, and W. K. Melville, Deep-water plunging breakers: A commparison between potential theory and experiments, *J. Fluid Mech*. **189**, 423 (1988).
6. A. M. Froncioni, P. Labbé, A. Garon, and R. Camarero, Interpolation-free space-time remeshing for the Burgers equation, *Comm. Numer. Methods Eng.* **13**, 875 (1997).
7. P. Hansbo, The characteristic streamline diffusion method for convection-diffusion problems, *Comput. Methods Appl. Mech. Engrg.* **96**, 239 (1992).
8. P. Hansbo, The characteristic streamline diffusion method for the time-dependent Navier–Stokes equations, *Comput. Methods Appl. Mech. Eng*. **99**, 171 (1992).
9. C. W. Hirt and B. D. Nichols, Volume of fluid (VOF) method for the dynamics of free boundaries, *J. Comput. Phys*. **39**, 201 (1981).
10. S. Majumdar, Role of underrelaxation in momentum interpolation for calculation of flow with nonstaggered grids, *Numer. Heat Transfer* **13**, 125 (1988).
11. J. C. Martin and W. J. Moyce, An experimental study of the collapse of liquid columns on a rigid horizontal plane, *Philos. Trans. Roy. Soc. London Ser. A* **244**, 312 (1952).
12. S. Muzaferija and M. Perić, Computation of free-surface flows using the finite-volume method and moving grids, *Numer. Heat Transfer Part B* **32**, 369 (1997).
13. S. Muzaferija and M. Perić, Computation of free-surface flows using interface-tracking and interface capturing methods, in *Nonlinear Water Wave Interaction*, edited by O. Mahrenholtz and M. Markiewicz (Computational Mechanics, Southhampton, 1998).
14. D. Pan, Y.-S. Yang, and C.-H. Chang, Computation of internal flow with free surfaces using artificial compressibility, *Numer. Heat Transfer Part B* **33**, 119 (1998).
15. S. V. Patankar, *Numerical Heat Transfer and Fluid Flow* (Hemisphere, Washington, DC, 1980).
16. T. J. Pedley and K. D. Stephanoff, Flow along a channel with a time-dependent indentation in one wall: The generation of vorticity waves, *J. Fluid Mech*. **160**, 337 (1985).
17. G. D. Raithby, W.-X. Xu, and G. D. Stubley, Prediction of incompressible free surface flows with an element-based finite volume method, *Comput. Fluid Dynam. J*. **4**, 353 (1995).
18. M. E. Ralph and T. J. Pedley, Flow in a channel with a moving indentation, *J. Fluid Mech*. **190**, 87 (1988).
19. B. Ramaswamy and M. Kawahara, Lagrangian finite element analysis applied to viscous free surface flow, *Int. J. Numer. Methods Fluids* **7**, 953 (1987).
20. M. Raw, *Robustness of Coupled Algebraic Multigrid for the Navier–Stokes Equations*, AIAA Paper 96-0297, 1996.
21. C. M. Rhie and W. L. Chow, Numerical study of the turbulent flow past an airfoil with trailing edge separation, *AIAA J*. **21**, 1525 (1983).
22. T. E. Tezduyar, M. Behr, and J. Liou, A new strategy for finite element computations involving moving boundaries and interfaces—The deforming-spatial-domain/space-time procedure. I. The concept and the preliminary numerical tests, *Comput. Methods Appl. Mech. Eng*. **94**, 339 (1992).
23. T. E. Tezduyar, M. Behr, and J. Liou, A new strategy for finite element computations involving moving boundaries and interfaces—The deforming-spatial-domain/space-time procedure. II. Computation of free-surface flows, two-liquid flows, and flows with drifting cylinders, *Comput. Methods Appl. Mech. Eng*. **94**, 353 (1992).
24. J. L. Thé, G. D. Raithby, and G. D. Stubley, Surface-adaptive finite volume method for solving free surface flows, *Numer. Heat Transfer Part B* **26**, 367 (1994).
25. P. D. Thomas and C. K. Lombard, Geometric conservation law and its application to flow computations on moving grids, *AIAA J*. **17**, 1030 (1979).
26. V. Venkatakrishnan, Convergence to steady state solutions of the Euler equations on unstructured grids with limiters, *J. Comput. Phys*. **118**, 120 (1995).

27. H. Zhang, M. Reggio, J. Y. Trépanier, and R. Camarero, Discrete form of the GCL for moving meshes and its implementation in CFD schemes, *Comput. & Fluids* **22**, 9 (1993).
28. P. J. Zwart, *The Integrated Space-Time Finite Volume Method*, Ph.D. thesis, University of Waterloo, 1999.
29. P. J. Zwart and G. D. Raithby, Space-time meshing for two-dimensional moving boundary problems, in *Proceedings of the 7th International Meshing Roundtable, 1988,* p. 187.
30. P. J. Zwart, G. D. Raithby, and M. J. Raw, An integrated space-time finite volume method for moving boundary problems, *Numer. Heat Transfer Part B* **34**, 257 (1998).