INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, VOL. 20, 989–1001 (1995)

LAGRANGIAN INCOMPRESSIBLE FLOW COMPUTATIONS IN THREE DIMENSIONS BY USE OF SPACE–TIME FINITE ELEMENTS

PETER HANSBO

Department of Mathematics, Chalmers University of Technology, S-412 96 Göteborg, Sweden

SUMMARY

In this paper we describe a space–time finite element method, with elements aligned along the computed characteristics in space–time, for the computation of incompressible free surface flows in three dimensions.

KEY WORDS streamline diffusion; finite element; Lagrangian method; space–time discretization; 3D flow

1. INTRODUCTION

Lagrangian fluid flow computations (or, for that matter, large-deformation analyses) have traditionally been performed using time differencing along computed characteristics.[1,2] This idea is somewhat conceptually unclear: how are we to compute the spatial differential operators? On what domain? And on what domain should the integration be performed? And what about higher-order time-stepping schemes? We do not suggest that these questions cannot be satisfactorily answered in the classical framework, but if a space–time finite element method is used, they are answered already at the outset! This conceptual elegance of the space–time approach leads to both simpler and more flexible computer implementations.

The numerical examples herein are restricted to Lagrangian computations, but the methodology is directly applicable to the case of an arbitrary Lagrangian–Eulerian (ALE) description. The 'reference' domain is then identical with the domain covered by the mesh, which in one stroke discards the notational plethora of standard ALE algorithms. The implementation is straightforward, with the Jacobian of transformation from the spatial domain to the material or referential domain being identical to the Jacobian of elemental transformation, standard in finite element analysis.

This paper is an extension of previous work by the author[3] concerning two-dimensional free surface problems. Similar ideas have also been proposed independently by Tezduyar *et al.*[4] For an early example of the use of space–time elements for free surface flows, see Reference 5. An alternative approach in the case of systems of conservation laws is given in Reference 6.

2. FINITE ELEMENT FORMULATION

We will consider the Navier–Stokes (NS) equations for an incompressible fluid in a variable domain $\Omega(t) \subset \mathbb{R}^3$, $0 \leq t \leq T$, with boundary $\Gamma(t)$ consisting of the disjoint parts $\Gamma_u(t)$ (where the velocity is

CCC 0271–2091/95/090989–13

prescribed) and $\Gamma_\sigma(t)$ (where the surface traction is known). In order to simplify the statement of the equations, we introduce the four-dimensional space–time domain

$$\mathscr{D} = \bigcup_{0<t\le T} \{(\mathbf{x},\ t) :\ \mathbf{x} \in \Omega(t)\},$$

where $\mathbf{x} = (x_1, x_2, x_3)$, with boundary $\partial\mathscr{D} = \partial\mathscr{D}_u \cup \partial\mathscr{D}_\sigma$, where $\partial\mathscr{D}_u$ and $\partial\mathscr{D}_\sigma$ are the parts of $\partial\mathscr{D}$ arising from Γ_u and Γ_σ respectively.

With this notation the NS equations may be written as

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u} - \operatorname{div}\boldsymbol{\sigma} &= \mathbf{f} \quad \text{in } \mathscr{D},\\ \operatorname{div}\mathbf{u} &= 0 \quad \text{in } \mathscr{D},\\ \mathbf{u} &= \mathbf{g} \quad \text{on } \partial\mathscr{D}_u,\\ \boldsymbol{\sigma}\cdot\mathbf{n} &= \mathbf{h} \quad \text{on } \partial\mathscr{D}_\sigma,\\ \mathbf{u} &= \mathbf{U}_0 \quad \text{in } \Omega(0). \end{aligned}$$

Here $\mathbf{u}$ is the velocity in a spatial co-ordinate system $\mathbf{x}$, $\mathbf{f}$ is the body force, $\mathbf{h}$ is a prescribed boundary tension and $\mathbf{g}$ is a prescribed velocity. Furthermore, $\mathbf{n}$ is the outward-pointing normal to $\Gamma_\sigma(t)$ and $\boldsymbol{\sigma}$ denotes the stress tensor

$$\boldsymbol{\sigma} = -p\mathbf{I} + 2\mu\boldsymbol{\varepsilon}(\mathbf{u}),$$

where p is the kinematic pressure, μ is the kinematic viscosity, $\mathbf{I}$ is the identity tensor and $\boldsymbol{\varepsilon}$ is the symmetrical part of the velocity gradient, with components

$$\varepsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right).$$

The motion of the free boundary $\Gamma_\sigma(t)$ is unknown in advance and thus the geometry of the space–time domain $\mathscr{D}$ is a part of the problem which must be considered in the discretization of (1). For clarity we will use the symbols $\Omega^h(t)$ and $\Gamma_\sigma^h(t)$ when referring to the *computed approximation* of the exact domain.

We will describe the proposed finite element method loosely; a more precise definition is given in Reference 3. Consider first a triangulation of $\Omega^h(t_n)$, which we choose as being defined by linear ('constant strain') tetrahedra. We extend the spatial mesh into a space–time mesh by considering tensor product elements on a reference space–time 'slab' $\hat{S}_n \overset{\text{def}}{=} \Omega^h(t_n) \times (t_n,\ t_{n+1})$. The temporal approximation is continuous on the interval $(t_n,\ t_{n+1})$ and discontinuous between the time intervals. Here we choose constant approximation on each reference slab, which leads to a backward Euler-type method.[3]

Next the space–time elements are mapped from the reference slab to the deforming physical space–time domain using a bilinear map

$$(\mathbf{x},\ t) = F_n(\boldsymbol{\xi} + (\tau - t_n)\mathbf{u}[^*(\boldsymbol{\xi}),\ \tau),$$

where $(\boldsymbol{\xi},\ \tau)$ refers to the reference domain, $(\mathbf{x},\ t)$ refers to the deforming domain and $\mathbf{u}^*$ defines the geometry of the space–time domain. In particular we may set $\mathbf{u}^*$ equal to our computed velocity field, in which case we recover a Stokes problem on a deforming domain, see below.

Since we use a constant temporal approximation on the reference slab, while $F_n(\boldsymbol{\xi},\ \tau)$ is linear in τ, this approach amounts to using elements that are superparametric in time (i.e. the polynomial describing the geometry of the elements is of higher degree than the one approximating the solution). This means that the boundaries of $\mathscr{D}$ will be approximated by straight segments over the intervals $I_n = (t_n,\ t_{n+1})$.

In the following we will denote by S_n the transformed counterpart of $\hat{S}_n$ and by ∂S_n^σ the transformed counterpart of $\Gamma_\sigma^h \times I_n$.

The space–time finite element method can now be formulated as follows: for $n = 0, 2, \ldots, N$ find $(\mathbf{u}_{n+1}^h, p_{n+1}^h) \equiv (\mathbf{u}^h, p^h)|_{S_n}$ such that

$$\begin{aligned}
&\int_{S_n} \left[\left(\frac{\partial \mathbf{u}_{n+1}^h}{\partial t} + \mathbf{u}_{n+1}^h \cdot \nabla \mathbf{u}_{n+1}^h \right) \cdot \mathbf{v} - p_{n+1}^h \operatorname{div} \mathbf{v} + 2\mu \boldsymbol{\varepsilon}(\mathbf{u}_{n+1}^h) : \boldsymbol{\varepsilon}(\mathbf{v}) \right] \mathrm{d}\Omega \, \mathrm{d}t \\
&\quad + \int_{S_n} \delta_1 \left(\frac{\partial \mathbf{u}_{n+1}^h}{\partial t} + \mathbf{u}_{n+1}^h \cdot \nabla \mathbf{u}^{n+1} h + \nabla p_{n+1}^h \right) \cdot \left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{u}_{n+1}^h \cdot \nabla \mathbf{v} + \nabla q \right) \mathrm{d}\Omega \, \mathrm{d}t \\
&\quad + \int_{\Omega^h(t_n)} \left(\mathbf{u}_{n+1}^h - \mathbf{u}_n^h \right) \cdot \mathbf{v} \, \mathrm{d}\Omega + \int_{S_n} \operatorname{div} \mathbf{u}_{n+1}^h (q + \delta_2 \operatorname{div} \mathbf{v}) \, \mathrm{d}\Omega \, \mathrm{d}t \\
&\quad + \int_{\partial S_n^\sigma} \delta_3 \left[(2\mu \boldsymbol{\varepsilon}(\mathbf{u}_{n+1}^h) - p_{n+1}^h \mathbf{I}) \cdot \mathbf{n} - \mathbf{h} \right] \cdot \left[(2\mu \boldsymbol{\varepsilon}(\mathbf{v}) - q\mathbf{I}) \cdot \mathbf{n} \right] \mathrm{d}\Gamma \, \mathrm{d}t \\
&= \int_{S_n} \mathbf{f} \cdot \left[\mathbf{v} + \delta_1 \left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{u}_{n+1}^h \cdot \nabla \mathbf{v} + \nabla q \right) \right] \mathrm{d}\Omega \, \mathrm{d}t + \int_{\partial S_n^\sigma} \mathbf{h} \cdot \mathbf{v} \, \mathrm{d}\Gamma \, \mathrm{d}t
\end{aligned} \tag{3}$$

for all possible mesh functions $\mathbf{v}$ and q. In (3), $\delta_1 = C_1 h/(1 + |\mathbf{u}_{n+1}^h|)$ is a stabilizing 'upwind' parameter, $\delta_2 = C_2 h$ is a stabilizing penalty-like term (cf. augmented Lagrangian methods[7]) and $\delta_3 = C_3$ is a penalty term enforcing the boundary condition; here we use positive constants $C_1 \approx 1$, $C_2 \approx 1$ and $C_3 \gg 1$. We also use the notation $\mathbf{u}_0^h = \mathbf{U}_0$.

We note in particular the following.

(i) The method (3) is based on a stabilizing least squares perturbation of the test functions, so that we in fact test with

$$\tilde{\mathbf{v}} = \mathbf{v} + \delta_1 \left(\frac{\partial \mathbf{v}}{\partial t} + \mathbf{u}^h \cdot \nabla \mathbf{v} + \nabla q \right)$$

and

$$\tilde{q} = q + \delta_2 \operatorname{div} \mathbf{v}.$$

(ii) The Neumann boundary condition $\boldsymbol{\sigma} \cdot \mathbf{n} = \mathbf{h}$ is satisfied weakly in the usual sense of the FEM. As noted in previous work,[3] this seems insufficient to ensure fulfilment of the boundary conditions with the present method (at least for inviscid flows; there is an ambiguity in the weak fulfilment of the pressure boundary conditions due to the perturbations). Hence we add a penalty term on the Neumann boundary.

(iii) If the computed velocity $\mathbf{u}_{n+1}^h$ is equal to the nodal velocity $\mathbf{u}^*$, we are in fact solving a Stokes problem (see below) and the definition of δ_1 is no longer valid.[8–10] In such a case the correct choice is $\delta_1 = C_1 h^2$.

(iv) The spatial integral

$$\int_{\Omega^h(t_n)} \left(\mathbf{u}_{n+1}^h - \mathbf{u}_n^h \right) \cdot \mathbf{v} \, \mathrm{d}\Omega$$

is to be computed on the bottom of slab S_n. Since the top of slab S_{n-1} does not necessarily have to match the bottom of slab S_n, we may return to the original grid at each time step. Then the spatial integral defines a built-in L_2-projection, and with $\mathbf{u}^* = \mathbf{u}_{n+1}^h$ the method resembles the characteristic Galerkin methods.[11] However, one important difference is that here all integrals are defined on the same mesh, while the characteristic Galerkin method typically treats the convective term separately.

In the special case of Stokesian flow we may formulate a simpler method, introduced by Brezzi and Pitkäranta,[10] which is conceptually different but closely related to (3). They introduced a relaxation of the divergence zero condition by letting

$$\operatorname{div} \mathbf{u} = h^2 \nabla^2 p,$$

which Hughes *et al.*[9] later put in the present framework of a least squares stbilization of the pressure. Since there is no need to stabilize the velocities in Lagrangian flow analysis and since we use the simplest linear spatial element, we may use this simpler approach, which gives us the following system of equations to consider (dropping the additional boundary terms):

$$\int_\Omega (\mathbf{u}^h_{n+1} - \mathbf{u}^h_n) \cdot \mathbf{v} \, d\Omega + \int_{S_n} \left(2\mu\varepsilon(\mathbf{u}^h_{n+1}) : \varepsilon(\mathbf{v}) - p^h_{n+1} \operatorname{div} \mathbf{v}\right) d\Omega \, dt + \int_{S_n} h \operatorname{div} \mathbf{u}^h_{n+1} \operatorname{div} \mathbf{v} \, d\Omega \, dt = \int_{S_n} \mathbf{f} \cdot \mathbf{v} \, d\Omega \, dt \tag{4a}$$

and

$$-\int_{S_n} \operatorname{div} \mathbf{u}^h_{n+1} q \, d\Omega \, dt - \int_{S_n} h^2 \nabla p^h_{n+1} \cdot \nabla q \, d\Omega \, dt = 0 \tag{4b}$$

for all mesh functions q and $\mathbf{v}$.

3. LAGRANGIAN COMPUTATIONS BY SPACE–TIME TILT

In Eulerian flow computations the unknown quantities $(\mathbf{u}, p)$ are computed at fixed points in a spatial co-ordinate frame $\mathbf{x}$ as in (1). In contrast, one may use the Lagrangian form of the equations, where the unknown quantities are related to the particles P (labelled by their positions $\mathbf{X}$ in the spatial frame at time $t = 0$), moving through the spatial co-ordinate system with velocity $\hat{\mathbf{u}}(\mathbf{X}, t)$. From a Lagrangian point of view the particles follow paths $\mathbf{x} = \mathbf{x}(\mathbf{X}, t)$ in the spatial frame, paths found by solving the system of equations

$$\frac{\partial \mathbf{x}}{\partial t} = \mathbf{u}(\mathbf{x}, t), \qquad \mathbf{x}(\mathbf{X}, 0) = \mathbf{X}. \tag{5}$$

By the chain rule of calculus one may express the acceleration of a particle in the spatial frame by

$$\frac{\partial \hat{u}_i(\mathbf{X}, t)}{\partial t} = \frac{\partial u_i(\mathbf{x}, t)}{\partial t} + \sum_j \frac{\partial u_i(\mathbf{x}, t)}{\partial x_j} \frac{\partial x_j}{\partial t};$$

consequently, by (5),

$$\frac{\partial \mathbf{u}(\mathbf{x}, t)}{\partial t} + \mathbf{u}(\mathbf{x}, t) \cdot \nabla \mathbf{u}(\mathbf{x}, t) = \frac{\partial \hat{\mathbf{u}}(\mathbf{X}, t)}{\partial t}. \tag{6}$$

Thus the convective derivative is not present in the Lagrangian formulation, which leads to simpler and stabler numerical approximations.

If we set the nodal velocity $\mathbf{u}^*$ equal to the computed solution $\hat{\mathbf{u}}^h$ during at time interval I_n, it is obvious that the path of a particle will be approximated by the motion of the mesh as expressed by the transformation (2). Indeed, since by definition $\mathbf{u}^h(\mathbf{x}, t) = \hat{\mathbf{u}}^h(\boldsymbol{\xi}, t)$ when $\mathbf{x}$ obeys (2), we have with the present superparametric-in-time formulation that

$$\frac{\partial \mathbf{x}}{\partial t} = \mathbf{u}^h(\mathbf{x}, t),$$

so that $\boldsymbol{\xi}$ in fact plays the role of $\mathbf{X}$ in (5) and by (6) the convective derivative will vanish automatically in the discrete scheme.

As an example, consider a one-dimensional element, linear in space and constant in time on the undeformed element. For the mapping to the physical element we use a superparametric-in-time element. The basis functions $\boldsymbol{\varphi}$ and mapping functions $\boldsymbol{\psi}$ are given by

$$\boldsymbol{\varphi} = \begin{bmatrix} \frac{1}{2}(1-\xi) \\ \frac{1}{2}(1+\xi) \end{bmatrix}, \qquad \boldsymbol{\psi} = \begin{bmatrix} \frac{1}{4}(1-\xi)(1-\tau) \\ \frac{1}{4}(1+\xi)(1-\tau) \\ \frac{1}{4}(1+\xi)(1+\tau) \\ \frac{1}{4}(1-\xi)(1+\tau) \end{bmatrix}. \tag{7}$$

The Jacobian of the mapping is given by

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial t}{\partial \xi} \\ \frac{\partial x}{\partial \tau} & \frac{\partial t}{\partial \tau} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} \frac{\partial \psi_i}{\partial \xi} x_i & \sum_{i=1}^{n} \frac{\partial \psi_i}{\partial \xi} t_i \\ \sum_{i=1}^{n} \frac{\partial \psi_i}{\partial \tau} x_i & \sum_{i=1}^{n} \frac{\partial \psi_i}{\partial \tau} t_i \end{bmatrix},$$

which in our case becomes (see Figure 1)

$$\mathbf{J} = \begin{bmatrix} \frac{(x_1 - x_2 + x_3 - x_4)\tau - x_1 + x_2 + x_3 - x_4}{4} & 0 \\ \frac{(x_1 - x_2 + x_3 - x_4)\xi - x_1 - x_2 + x_3 + x_4}{4} & \frac{k_n}{2} \end{bmatrix}, \tag{8}$$

where $k_n = t_{n+1} - t_n$. Note in particular the zero in the upper right corner of $\mathbf{J}$. It is given by the symmetry of the spatial derivatives with respect to opposing nodes, together with the fact that the element has straight horizontal sides on each time level. This quality generalizes to the multidimensional case and thus the Jacobian of the space–time element is no more difficult to invert than the Jacobian of the corresponding spatial element: first we solve for the spatial derivatives of the basis functions, then the temporal derivatives are given as a linear combination of the spatial derivatives.

With the computed velocity as nodal velocity we have that $x_4 = x_1 + k_n u_1^h$, $x_3 = x_2 + k_n u_2^h$, and thus

$$\mathbf{J} = \begin{bmatrix} \frac{k_n\left[u_2^h - u_1^h + \tau\left(u_2^h - u_1^h\right)\right] + 2(x_2 - x_1)}{4} & 0 \\ \frac{k_n\left[u_1^h + u_2^h - \xi\left(u_2^h - u_1^h\right)\right]}{4} & \frac{k_n}{2} \end{bmatrix}, \tag{9}$$

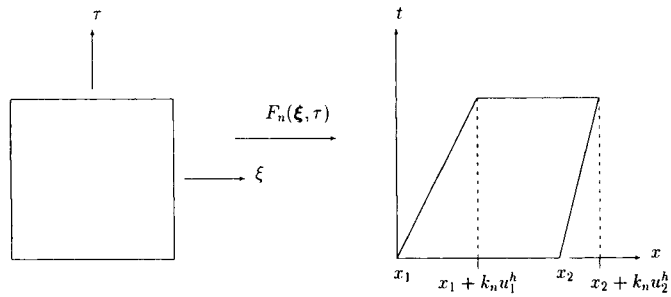

Figure 1. Mapping from the reference element to the physical element

leading to

$$\begin{bmatrix} \dfrac{\partial \varphi_1}{\partial x} \\ \dfrac{\partial \varphi_1}{\partial t} \end{bmatrix} = \frac{1}{\det \mathbf{J}} \begin{bmatrix} -\dfrac{k_n}{4} \\ \dfrac{k_n\left[u_1^h + u_2^h + \left(u_2^h - u_1^h\right)\xi\right]}{8} \end{bmatrix}, \qquad \begin{bmatrix} \dfrac{\partial \varphi_2}{\partial x} \\ \dfrac{\partial \varphi_2}{\partial t} \end{bmatrix} = \begin{bmatrix} -\dfrac{\partial \varphi_1}{\partial x} \\ -\dfrac{\partial \varphi_1}{\partial t} \end{bmatrix},$$

and it follows that

$$\frac{\partial \varphi_i}{\partial t} + u^h \frac{\partial \varphi_i}{\partial x} = \frac{\partial \varphi_i}{\partial t} + \frac{1}{2}\left[(1-\xi)u_1^h + (1+\xi)u_2^h\right] \frac{\partial \varphi_i}{\partial x} \equiv 0$$

as expected. A corresponding effect cannot be achieved with an isoparametric mapping.

4. SOLUTION ALGORITHM

4.1. The discrete system

If the space–time mesh is aligned with the characteristics, the Navier–Stokes problem in effect is reduced to a Stokes problem at each time step. This fact may be used to define a general solution algorithm. We may write (4) in matrix form as

$$\begin{bmatrix} \mathbf{A} & \mathbf{B}^{\mathrm{T}} \\ \mathbf{B} & -\mathbf{C} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{F} \\ \mathbf{0} \end{bmatrix}. \tag{10}$$

Here, denoting by $\boldsymbol{\varphi}_i$ the FE approximation for the velocity and by φ_i the FE approximation for the pressure,

$$\mathbf{A}_{ij} = \int_\Omega \boldsymbol{\varphi}_j(\mathbf{x},\, t_n) \cdot \boldsymbol{\varphi}_i(\mathbf{x},\, t_n)\, \mathrm{d}\Omega + \int_{S_n} 2\mu \boldsymbol{\varepsilon}(\boldsymbol{\varphi}_j) : \boldsymbol{\varepsilon}(\boldsymbol{\varphi}_i)\, \mathrm{d}\Omega\, \mathrm{d}t + \int_{S_n} h\, \mathrm{div}\, \boldsymbol{\varphi}_j\, \mathrm{div}\, \boldsymbol{\varphi}_i,$$

$$\mathbf{B}_{ij} = -\int_{S_n} \varphi_j\, \mathrm{div}\, \boldsymbol{\varphi}_i\, \mathrm{d}\Omega\, \mathrm{d}t, \qquad \mathbf{C}_{ij} = \int_{S_n} h^2 \nabla \varphi_j \cdot \nabla \varphi_i\, \mathrm{d}\Omega\, \mathrm{d}t,$$

$$\mathbf{F}_i = \int_{S_n} \mathbf{f} \cdot \boldsymbol{\varphi}_i\, \mathrm{d}\Omega\, \mathrm{d}t + \int_\Omega \mathbf{u}_n^h(\mathbf{x},\, t_n) \cdot \boldsymbol{\varphi}_i(\mathbf{x},\, t_n)\, \mathrm{d}\Omega.$$

Note the presence of $\mathbf{C}$ corresponding to the slight compressibility. It is easy to see how this matrix stabilizes the pressure: eliminating the velocities, we have to invert $\mathbf{B}\mathbf{A}^{-1}\mathbf{B}^{\mathrm{T}} + \mathbf{C}$ to solve for the pressure. Since $\mathbf{C}$ is positive definite (modulo the 'rigid body' mode which is eliminated, if necessary by artificially prescribing some node), there is in fact no need for $\mathbf{B}\mathbf{A}^{-1}\mathbf{B}^{\mathrm{T}}$ to be invertible (corresponding to the fulfilment of the Babuška–Brezzi condition).

4.2. Solution method

For the solution of the time-dependent Stokes problem we use an Uzawa algorithm with a preconditioned conjugate gradient method applied to the pressure equation.[7,12,13] The system (10) may be written as

$$\left(\mathbf{B}\mathbf{A}^{-1}\mathbf{B}^{\mathrm{T}} + \mathbf{C}\right)\mathbf{p} = \mathbf{B}\mathbf{A}^{-1}\mathbf{F}, \tag{11a}$$

$$\mathbf{u} = \mathbf{A}^{-1}\left(\mathbf{F} - \mathbf{B}^{\mathrm{T}}\mathbf{p}\right). \tag{11b}$$

The method consists of the conjugate gradient method applied to (11a), with the velocity repeatedly updated using (11b). For completeness we give the algorithm.

1. Initialization; given $\mathbf{p}_0$.

$$\mathbf{u}_0 = \mathbf{A}^{-1}(\mathbf{F} - \mathbf{B}^T\mathbf{p}_0)$$
$$\mathbf{r}_0 = \mathbf{B}\mathbf{A}^{-1}\mathbf{F} - (\mathbf{B}\mathbf{A}^{-1}\mathbf{B}^T + \mathbf{C})\mathbf{p}_0 = \mathbf{B}\mathbf{u}_0 - \mathbf{C}\mathbf{p}_0$$
$$\mathbf{s}_0 = \mathbf{S}^{-1}\mathbf{r}_0$$
$$\mathbf{d}_0 = \mathbf{s}_0$$
$$\mathbf{w}_0 = \mathbf{A}^{-1}\mathbf{B}^T\mathbf{d}_0$$

2. For $k = 0, 1, 2, \ldots$

$$\alpha_k = \mathbf{r}_k^T\mathbf{s}_k / \mathbf{d}_k^T(\mathbf{B}\mathbf{w}_k + \mathbf{C}\mathbf{d}_k)$$
$$\mathbf{p}_{k+1} = \mathbf{p}_k + \alpha_k\mathbf{d}_k$$
$$\mathbf{u}_{k+1} = \mathbf{A}^{-1}[\mathbf{F} - \mathbf{B}^T(\mathbf{p}_k + \alpha_k\mathbf{d}_k)] = \mathbf{u}_k - \alpha_k\mathbf{w}_k$$
$$\mathbf{r}_{k+1} = \mathbf{B}(\mathbf{u}_k - \alpha_k\mathbf{w}_k) - \mathbf{C}(\mathbf{p}_k + \alpha_k\mathbf{d}_k) = \mathbf{r}_k - \alpha_k(\mathbf{B}\mathbf{w}_k + \mathbf{C}\mathbf{d}_k)$$

3. Check the size of $|r_{k+1}|$; if below tolerance, stop. Otherwise:

$$\mathbf{s}_{k+1} = \mathbf{S}^{-1}\mathbf{r}_{k+1}$$
$$\beta_k = \mathbf{r}_{k+1}^T\mathbf{s}_{k+1} / \mathbf{r}_k^T\mathbf{s}_k$$
$$\mathbf{d}_{k+1} = \mathbf{s}_{k+1} + \beta_k\mathbf{d}_k$$
$$\mathbf{w}_{k+1} = \mathbf{A}^{-1}\mathbf{B}^T\mathbf{d}_{k+1}$$

Next k.

The question is how to select the preconditioning matrix $\mathbf{S}$. The natural choice for $\mathbf{S}$ was given by Cahouet and Chabard[12] as a linear combination of a discretized Laplacian and the mass matrix, depending on the appearance of the matrix $\mathbf{A}$ (see also Reference 13 and, for a related method, Reference 14). In this way one obtains a preconditioner which is easy to construct and which is spectrally equivalent to the pressure matrix $\mathbf{B}\mathbf{A}^{-1}\mathbf{B}^T$. Since $\mathbf{C}$ is small compared with the pressure matrix (its eigenvalues ranging between $O(h^2)$ and $O(1)$), we expect, and indeed find in practice, that the pressure equation converges rapidly with a good preconditioner for the pressure matrix.

A drawback of Uzawa-type schemes is that the velocity matrix $\mathbf{A}$ must be inverted in each loop. The inversion has to be accurate in order for the method to converge,[14] and if $\mathbf{A}$ is ill-conditioned, this may make the method expensive. In our application, however, we assume that the viscosity is small and thus $\mathbf{A}$ is close to the mass matrix which is easy to invert (this is not quite true, since we also augment $\mathbf{A}$ with a penalty term which makes things a little less perfect). With this $\mathbf{A}$ it also follows that the natural choice for $\mathbf{S}$ is a discrete Laplacian. In the numerical examples presented herein, we use a direct solver for the inversion of $\mathbf{S}$, with back substitution at each conjugate gradient step. We update the geometry three times at each time step, which also means updating the system matrix. However, we do not update the preconditioning matrix: the geometry does not change enough to make a difference. One could in principle use the same preconditioning matrix as long as the topology of the mesh is not changed and just check whether the effect of preconditioning is deteriorating, but here we update $\mathbf{S}$ when changing time step.

A faster approach to the inversion of the discrete Laplacian is of course to use multigrid acceleration of an appropriate iterative method. Note, however, that for Lagrangian flows this would mean the use of a more complicated non-nested multigrid implementation, since the nodes asociated with the finest level can move independently of the nodes on the coarser levels.

5. NUMERICAL EXAMPLES

5.1. *Collapsing liquid column*

A column of inviscid liquid, originally contained in a cylinder of radius 1 and height 2, collapses under the influence of gravity, modelled by setting $\mathbf{f} = (0, 0, -1)$. The initial domain is shown in Figure 2. We use a constant time step $k_n = 0{\cdot}1$. The domain and central cut-outs showing velocity and pressure isolines are given in Figures 3–6 after 5, 10, 15 and 20 time steps. The corresponding maximum velocity and pressure were $(u_{\max}, p_{\max}) = (0{\cdot}87, 0{\cdot}67)$, $(1{\cdot}4, 1{\cdot}1)$, $(1{\cdot}6, 1{\cdot}5)$ and $(1{\cdot}7, 1{\cdot}0)$ respectively. The mesh consists of 6823 elements and there are 1417 nodes. Note that the pictures are not shown on the same scale.

5.2. *Splashing wave*

A disc of inviscid liquid, $x_1^2 + x_2^2 \leq 1$ and $0 \leq x_3 \leq 0{\cdot}25$ is furnished with a cosine hump according to

$$x_3 = \tfrac{1}{4}\{1 + \tfrac{1}{2}[1 + \cos(\pi R)]\},$$

where $R = (x_1^2 + x_2^2)^{1/2}$ Initially the fluid is at rest. Suddenly it collapses under the influence of gravity (modelled as in the previous example). We show a side view of the original mesh in Figure 7. It consists of 8887 elements and has 1965 nodes. In Figure 8 we show the velocity and a cut-out (the (x_1, x_3) plane at $x_2 = 0$) of the pressure at time $t = 0{\cdot}25$. Finally we show a sequence of velocity cut-outs after 15, 25, 30, 40, 45 and 50 time steps of size $k_n = 0{\cdot}05$ in Figure 9, from top to bottom. At the final time the mesh has collapsed owing to extreme deformation.

6. CONCLUDING REMARKS

The space–time finite element method presented herein is very simple to implement and use, since nothing has to be worked out analytically; all transformations between material, reference and spatial domains are performed using standard superparametric finite element mappings from the parent

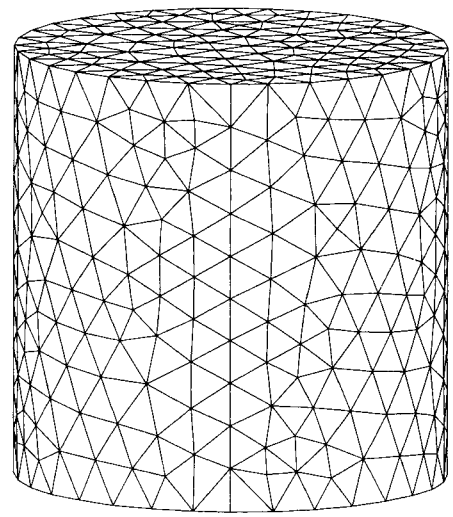

Figure 2. Initial domain for collapsing column

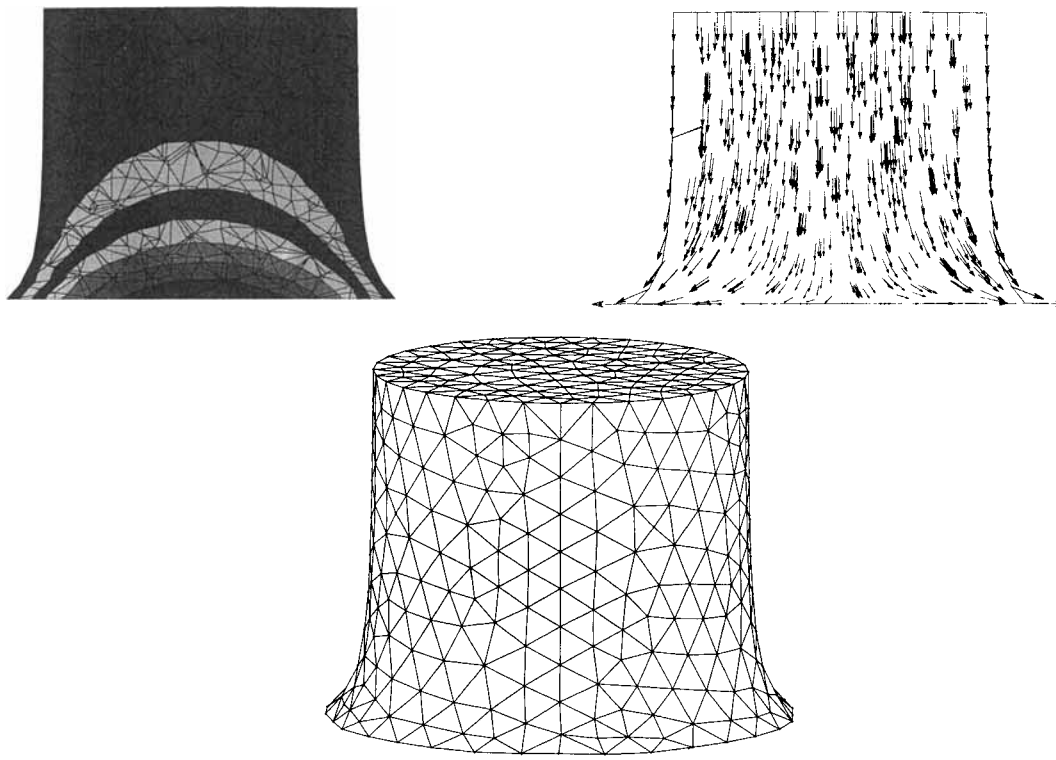

Figure 3. Column at time $t = 0{\cdot}5$

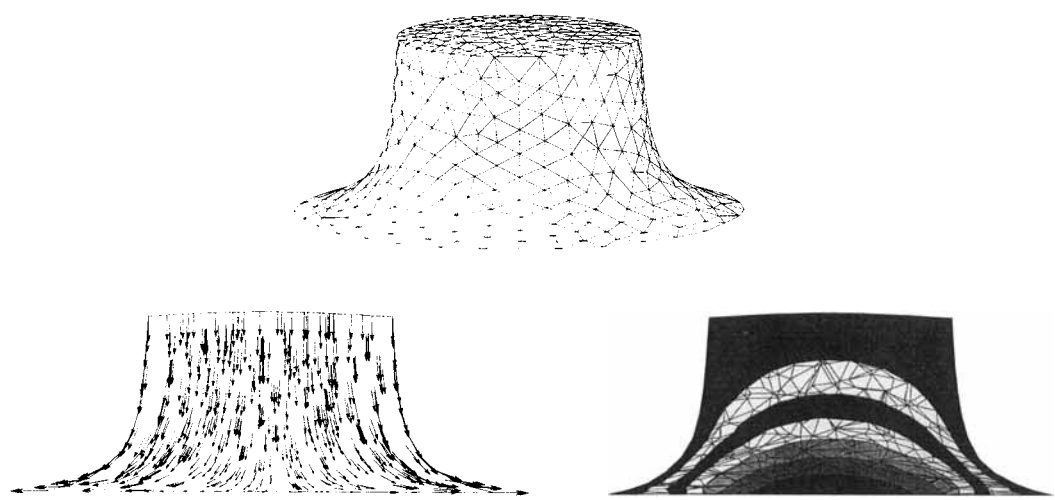

Figure 4. Column at time $t = 1{\cdot}0$

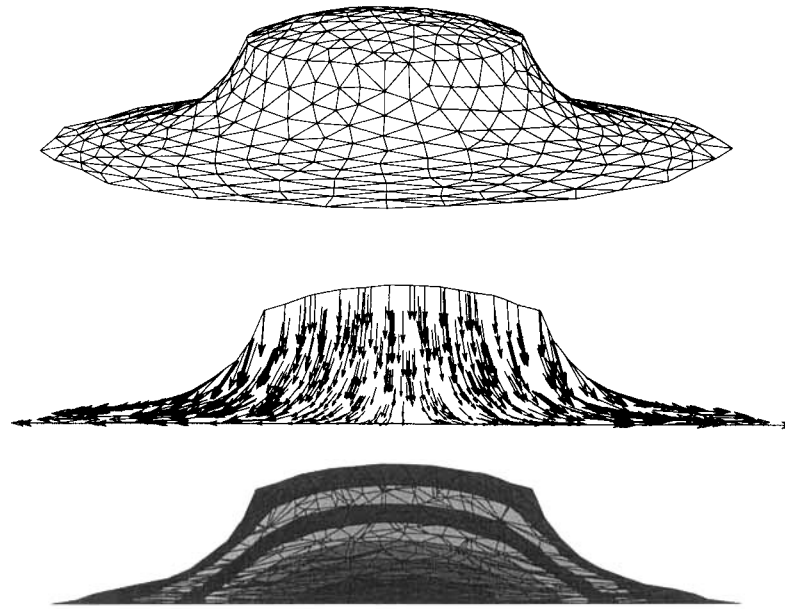

Figure 5. Column at time $t = 1{\cdot}5$

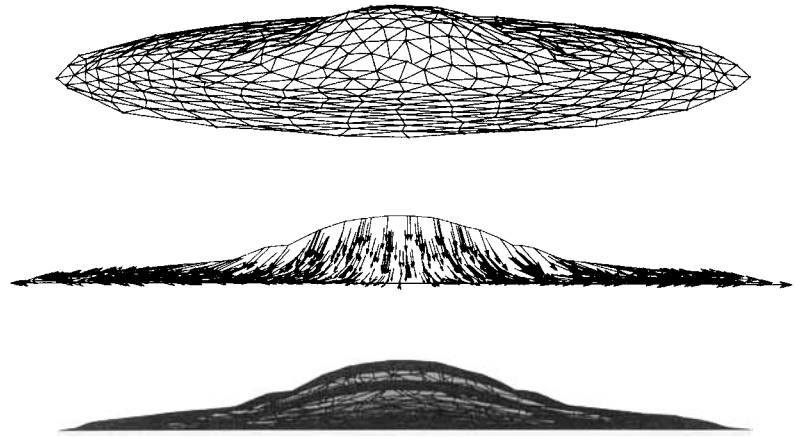

Figure 6. Column at time $t = 2{\cdot}0$

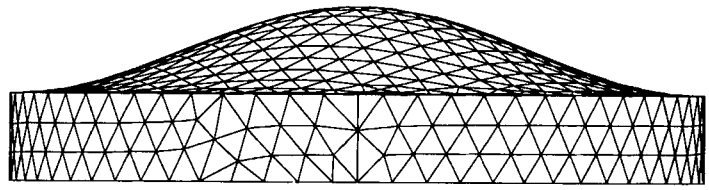

Figure 7. Initial domain for splashing wave

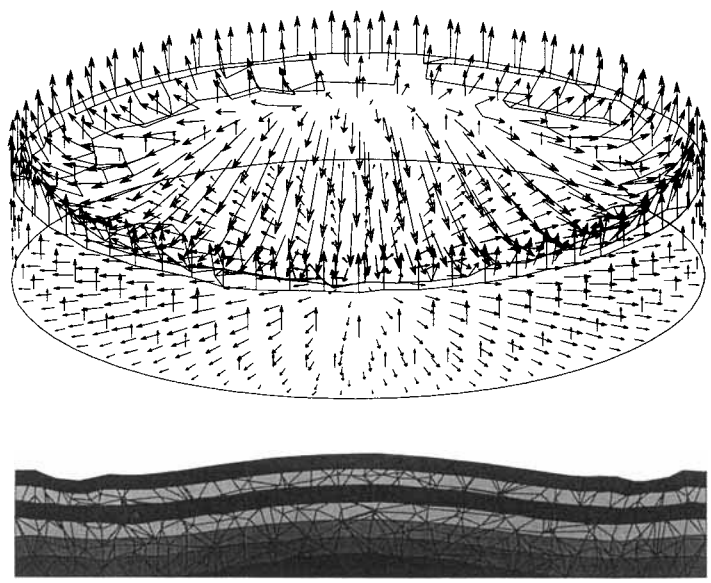

Figure 8. Velocity and pressure cut-out at time $t = 0{\cdot}25$

element to the physical element on which the computations are performed. There is no problem if one wishes to increase the temporal accuracy: simply increase the polynomial order. Change one subroutine!

We emphasize the many different possibilities using this unifying space–time approach: for Lagrangian computations as herein; for arbitrary Lagrangian–Eulerian computations as in Reference 4; as a general approach to fast solution of convective problems;[11] for increasing the accuracy of the numerical solution of more general time-dependent problems.[6]

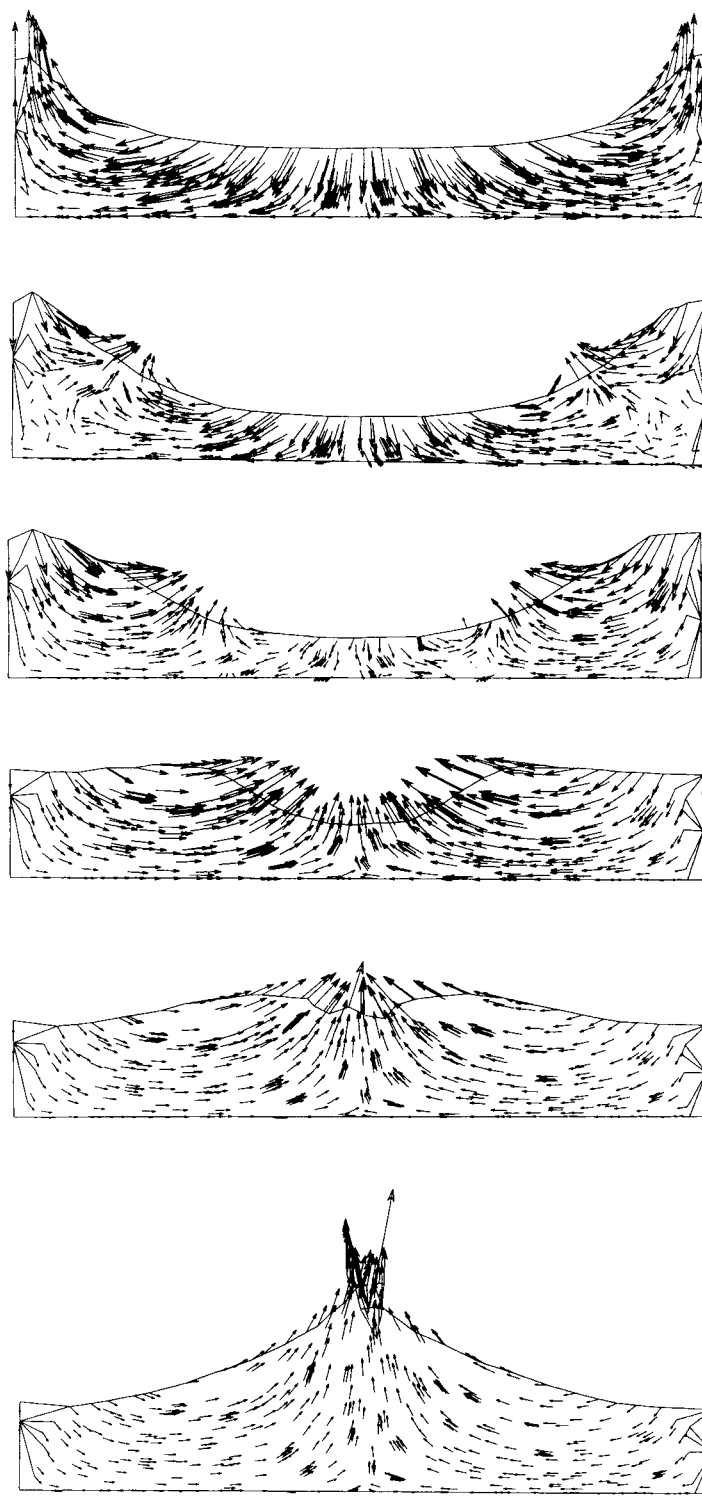

Figure 9. Sequence of velocity cut-outs (time increases from top to bottom)

REFERENCES

1. P. Bach and O. Hassager, 'An algorithm for the use of the Lagrangian specification in Newtonian fluid mechanics and applications to free-surface flow', *J. Fluid Mech.*, **152**, 173–190 (1985).
2. T. Okamoto and M. Kawahara, 'Two-dimensional sloshing analysis by Lagrangian finite element method', *Int. j. numer. methods fluids*, **11**, 453–477 (1990).
3. P. Hansbo, 'The characteristic streamline diffusion method for the time-dependent incompressible Navier–Stokes equations', *Comput. Methods Appl. Mech. Eng.*, **99**, 171–186 (1992).
4. 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**, 339–351 (1992).
5. C. S. Frederiksen and S. M. Watts, 'Finite element method for time-dependent incompressible free surface flow', *J. Comput. Phys.*, **39**, 282–304 (1981).
6. P. Hansbo, 'Space–time oriented streamline diffusion methods for non-linear conservation laws in one dimension', *Commun. numer. methods eng.*, **10**, 203–215 (1994).
7. M. Fortin and R. Glowinski, *Augmented Lagrangian Methods: Application to the Numerical Solution of Boundary-Value Problems*, North-Holland, Amsterdam, 1983.
8. P. Hansbo and A. Szepessy, 'A velocity–pressure streamline diffusion finite element method for the incompressible Navier–Stokes equations', *Comput. Methods Appl. Mech. Eng.*, **84**, 175–192 (1990).
9. T. J. R. Hughes, L. P. Franca and M. Balestra, 'A new finite element formulation for computational fluid dynamics: V. Circumventing the Babuška–Brezzi condition: a stable Petrov–Galerkin formulation of the Stokes problem accommodating equal-order interpolations', *Comput. Methods Appl. Mech. Eng.*, **59**, 85–99 (1986).
10. F. Brezzi and J. Pitkäranta, 'On the stabilization of finite element approximations of the Stokes equations', in W. Hackbusch (ed.), *Efficient Solution of Elliptic Systems*, Vieweg, Braunschweig, 1984, pp. 11–19.
11. C. Johnson, 'A new approach to algorithms for convection problems which are based on exact transport + projection', *Comput. Methods Appl. Mech. Eng.*, **100**, 45–62 (1992).
12. A. Cahouet and J.-P. Chabard, 'Some fast 3D finite element solvers for the generalized Stokes problem', *Int. j. numer. methods fluids*, **12**, 929–946 (1991).
13. R. Q. N. Zhou, 'Preconditioned finite element algorithms for 3D Stokes flows', *Int. j. numer. methods fluids*, **17**, 667–685 (1993).
14. J. H. Bramble and J. E. Pasciak, 'Iterative techniques for time dependent Stokes problems', *Preprint*, Department of Mathematics, Cornell University, Ithaca, NY, 1993.