Theoret. Comput. Fluid Dynamics (1992) 3: 285–306

Theoretical and Computational Fluid Dynamics

Direct Simulation of Fluid Particle Motions[1]

Howard H. Hu and Daniel D. Joseph

Department of Aerospace Engineering and Mechanics, University of Minnesota, Minneapolis, MN 55455, U.S.A.

Marcel J. Crochet

Unite de Mecanique Appliquee, Universite Catholique de Louvain, 1348 Louvain-la-Neuve, Belgium

Communicated by M.Y. Hussaini

Received 2 July 1991 and accepted 9 October 1991

Abstract. Continuum models of two-phase flows of solids and liquids use constitutive assumptions to close the equations. A more fundamental approach is a "molecular dynamic" simulation of flowing "big" particles based on reliable macroscopic equations for both solid and liquid. We developed a package that simulates the unsteady two-dimensional solid–liquid two-phase flows using the Navier–Stokes equations for the liquid and Newton's equations of motion for the solid particles. The Navier–Stokes equations are solved using a finite-element formulation and Newton's equations of motion are solved using an explicit–implicit scheme. We show that the simplest fully explicit scheme to update the particle motion using Newton's equations is unstable. To correct this instability we propose and implement an Explicit–Implicit Scheme in which, at each time step, the positions of the particles are updated explicitly, the computational domain is remeshed, the solution at the previous time is mapped onto the new mesh, and finally the nonlinear Navier–Stokes equation and the implicitly discretized Newton's equations for particle velocities are solved on the new mesh iteratively. The numerical simulation reveals the effect of vortex shedding on the motion of the cylinders and reproduces the drafting, kissing, and tumbling scenario which is the dominant rearrangement mechanism in two-phase flow of solids and liquids in beds of spheres which are constrained to move in only two dimensions.

1. Introduction

Two-phase flows of solids and liquids are usually studied using interpenetrating mixtures. The terms which arise in mixture theories that represent interactions between the liquid and solid are not known. Some assumptions about the form of these interaction terms must be made but these assumptions are typically not testable in any explicit sense. The nature of the detailed interactions between solids and fluids cannot be understood from application of the mixture theories.

One important scenario is called drafting, kissing, and tumbling. In the experiments of Fortes *et al.* (1987) with spheres in a two-dimensional fluidized bed, shown in Figure 1, they described this scenario

[1] This work was supported by the National Science Foundation, the Department of Energy, and the Army Research Office, Mathematics.

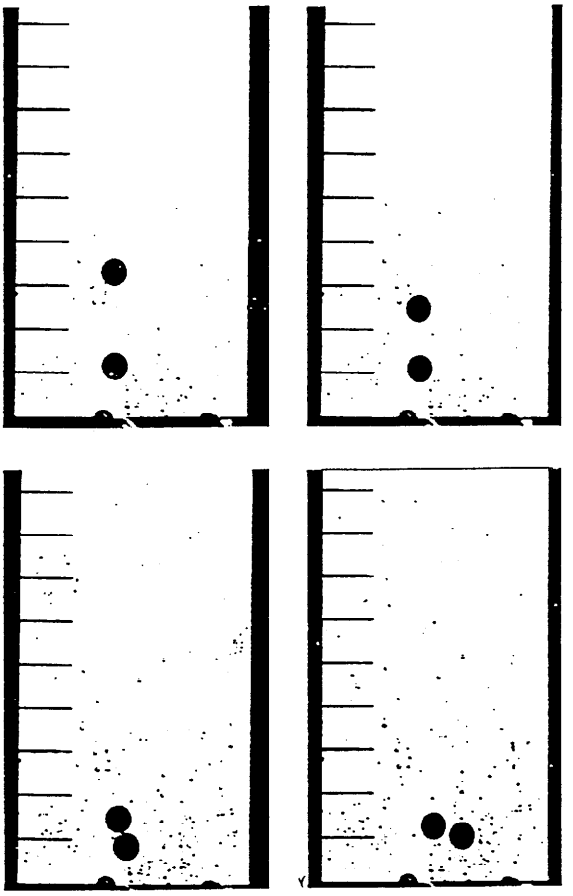


Figure 1. Drafting, kissing, and tumbling of two spheres in two-dimensional fluidized bed. Flow is from bottom to top.

as "... one sphere is captured in the wake of the other. The kissing spheres are aligned with the stream. The streamwise alignment is massively unstable and kissing spheres tumble into more stable cross-stream pairs of doublets which can aggregate into larger relative stable horizontal arrays."

In order to understand the interactions between the solid and the liquid, we take a more fundamental approach with the Navier–Stokes equations for the liquid and Newton's equations of motion for the particles. Numerical simulation of these equations can give us all the details of the flow and the fluid–solid interactions together with clear understandings of the mechanisms involved. Direct simulation of fluid–particle motions is applicable to many other problems of interest; we are interested in stable arrangements of particles in flows, Segre and Silberberg effects, lubricated transport of solid particulates, particle clustering in flows, etc.

These numerical simulation involves three major difficulties. The first is automatic remeshing. A new mesh needs to be generated at each time step, according to the positions of the particles. The second difficulty is projection. The flow field at the old time step must be mapped onto the new mesh in order to evaluate the unsteady term in the Navier–Stokes equations in Eulerian form. Finally, the numerical scheme that discretizes the coupled Navier–Stokes equations and particle equations has to be stable and efficient.

There are few numerical works on these types of simulations. Avalosse *et al.* (1990) developed a very flexible finite-element procedure to calculate flows in spatial domains which vary strongly with time. They successfully simulated the mixing in a twin-screw extruder with triangular cams, and the motion of a sedimenting cylinder in a closed container. However, in their work the Reynolds number is assumed to be small; thus the inertia of the fluid is neglected, which greatly simplifies the simulation.

Tezduyar *et al.* (1990a, 1990b) developed a DSD/ST (Deforming-Spatial-Domain/Space-Time) procedure for finite-element computations involving moving boundaries and interfaces. They tested their method for the flows with a drafting cylinder. This procedure, however, seems unable to handle

situations where the spatial domain changes strongly with time and which apply to problems in three dimensions.

2. Description of the Problem

Consider two-dimensional flow in an infinite channel of width W. The fluid in the channel is assumed to be Newtonian and incompressible with density ρ_f and viscosity μ_f. The coordinate x is taken in the direction of the gravity and y is in the width of the channel, as shown in Figure 2. There are N infinitely long rigid cylinders of diameter d and density ρ_s moving inside the channel. Let time $t \in (0, T)$, let Ω_t denote the region occupied by the fluid, and let Γ_t be the boundary of this region. Assume that the center of cylinder i $(i = 1, 2, \ldots, N)$ is located at (X_i, Y_i), its rotational angle with respect to the initial angle is Θ_i, its velocity components in the x, y direction are (U_i, V_i), its angular velocity is Ω_i. It will be convenient to regard (X_i, Y_i, Θ_i) and (U_i, V_i, Ω_i) as vectors. Figure 2 shows the situation when there are two cylinders $(N = 2)$ in the channel.

The motion of the fluid and the solid cylinders is coupled. The fluid exerts forces and torques on the cylinders, thus changing the motion of the cylinders. The motion of the cylinders induces flow in the fluid by changing the position of the boundary and the velocities on the boundary. The velocity $\mathbf{u}(\mathbf{x}, t)$ and the pressure $p(\mathbf{x}, t)$ in the fluid are governed by the Navier–Stokes equations

$$\left.\begin{aligned} \rho_f\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) &= \rho_f \mathbf{g} + \nabla\cdot\sigma \\ \nabla\cdot\mathbf{u} &= 0 \end{aligned}\right\} \quad \text{on } \Omega_t, \quad \forall t \in (0, T), \tag{1}$$

where $\mathbf{g}$ is the gravity vector and σ is the stress tensor given by

$$\sigma = -p\mathbf{1} + \mu_f[\nabla\mathbf{u} + (\nabla\mathbf{u})^T]. \tag{2}$$

The motion of the cylinders satisfies the Newton's law

$$\mathbf{M}\frac{d\mathbf{U}_i(t)}{dt} = \mathbf{F}_i \tag{3}$$

and

$$\frac{d\mathbf{X}_i(t)}{dt} = \mathbf{U}_i(t), \qquad i = 1, 2, \ldots, N, \quad \forall t \in (0, T) \tag{4}$$

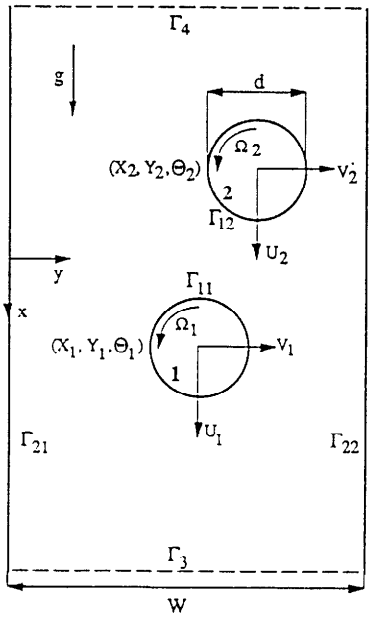


Figure 2. Configuration of the problem. Two cylinders sedimenting in a channel.

where $\mathbf{M} = \text{diag}\{m, m, I\}$ is the mass matrix with m the mass and I the polar moment of inertia of the cylinders per unit length ($m = \rho_s(\pi d^2/4)$ and $I = m(d^2/8)$), $\mathbf{U}_i(t) = (U_i, V_i, \Omega_i)$ is the velocity vector, $\mathbf{X}_i(t) = (X_i, Y_i, \Theta_i)$ is the position vector, $\mathbf{F}_i = (F_{ix}, F_{iy}, F_{im})$ is the force vector which consists of the x and y components of the force and the torque acting on the cylinder i. The force vector should be the total force acting on the cylinder, which includes a body term resulting from gravity and a surface term resulting from tractions of the fluid on the solid. We also frequently use the acceleration vector $\mathbf{f}_i$ of the cylinder i in describing the numerical schemes, $\mathbf{f}_i = (f_{ix}, f_{iy}, f_{im}) \overset{\text{def}}{=} \mathbf{M}^{-1}\mathbf{F}_i = (F_{ix}/m, F_{iy}/m, F_{im}/I)$.

The motion in the fluid phase should also satisfy some conditions at the boundary Γ_t. In general Γ_t can be divided into two parts $(\Gamma_t)_g$ and $(\Gamma_t)_h$ where the Dirichlet and Neumann type of boundary conditions are satisfied:

$$\mathbf{u} = \mathbf{u}_g \qquad \text{on } (\Gamma_t)_g, \quad \forall t \in (0, T), \tag{5}$$

and

$$\sigma\mathbf{n} = \mathbf{h} \qquad \text{on } (\Gamma_t)_h, \quad \forall t \in (0, T), \tag{6}$$

where $\mathbf{u}_g$ and $\mathbf{h}$ are prescribed values of the velocity and traction, some known functions, and $\mathbf{n}$ is the outward normal on $(\Gamma_t)_h$.

More specifically, we consider the case of cylinders sedimenting ($\rho_s > \rho_f$) in an initially tranquil fluid. Referring to Figure 2, we let Γ_{1i} be the boundary on the surface of cylinder i ($i = 1, 2, \ldots, N$) at time $t \in (0, T)$. Then, applying no-slip condition, the fluid velocity at a point (x, y) on Γ_{1i} is equal to the velocity of the cylinder at the same point:

$$\mathbf{u} = [U_i - \Omega_i(y - Y_i)]\mathbf{e}_x + [V_i + \Omega_i(x - X_i)]\mathbf{e}_y, \qquad \forall (x, y) \text{ satisfying } (x - X_i)^2 + (y - Y_i)^2 = d^2/4. \tag{7}$$

On the two stationary channel walls, Γ_{21} and Γ_{22}, the no-slip condition requires the fluid velocity to be zero. Theoretically the channel is infinitely long $x \in (-\infty, \infty)$, but the computational domain is cut off far away from the cylinders. Thus far ahead of the first cylinder, on boundary Γ_3, we assume that the disturbance caused by the motion of the cylinders is very small. Therefore

$$\mathbf{u} = \mathbf{0} \qquad \text{on } \Gamma_{21}, \Gamma_{22}, \text{ and } \Gamma_3. \tag{8}$$

Far behind the last cylinder, on boundary Γ_4, we apply the usual traction-free condition

$$\sigma\mathbf{n} = \mathbf{0} \qquad \text{on } \Gamma_4. \tag{9}$$

Furthermore, imagine that the fluid is initially at rest. Its velocity is zero, the pressure is hydrostatic, and the cylinders are held up by some external forces at certain positions,

$$\mathbf{u}(x, 0) = \mathbf{0} \qquad \text{on } \Omega_0, \tag{10}$$

$$\mathbf{X}_i(0) = \mathbf{X}_{i0} \quad \text{and} \quad \mathbf{U}_i(0) = \mathbf{0} \qquad \text{for } i = 1, 2, \ldots, N. \tag{11}$$

Then, at $t = 0^+$, we set the cylinders free, and we want to know the motion of these cylinders and the flow of the fluid thereafter.

3. Mesh Generation

Assume that at time instant t, based on the solution at a previous time, we are able to predict new positions of the cylinders. Then we have to generate a new mesh on which we can discretize the Navier–Stokes equations, solve the flow, and compute the forces acting on the cylinders. Since we are anticipating rather complicated interactions of the cylinders, the geometry of the computation domain can change drastically from time to time. We therefore choose to use unstructured meshes in time, in which there is no one-to-one correspondence between the nodes of two meshes at two different times. This method enables us to handle problems with large deformation. A problem of the method is that at each time step we are required to project the velocities from one mesh to another.

The core of our remeshing package is the subroutine MSHPTG from the interactive two-dimensional mesh generator Emc^2, developed by Hecht and Saltel (1989) of INRIA. MSHPTG requires information about the segments on the boundary of the computational domain, like the coordinates

of all the points on the boundary, the connections of these points and the mesh size near these points, etc. It generates the elements into the interior of the domain, and outputs a regulated triangular (triangles with about equal sides) mesh with elements of three vertices.

In our remeshing package we need to specify the positions of the cylinders and the general information about the computation domain (e.g., the width of the channel, the diameter of the cylinder, the number of the cylinders, etc.). The package first locates all the boundary points on the computational domain and assigns different parts of the boundary with indices for different boundary conditions. Then it calls MSHPTG to generate a regulated mesh on an artificial domain (squeezed geometrically at two ends). Next it goes to a routine that smoothly stretches the mesh on the artificial domain at two ends and fits the mesh onto the real computational domain, in this way we can reduce the number of total elements used in the computation. After that all the midnodes in each element are generated to form P2/P1 elements used in our finite-element Navier–Stokes solver. Finally, the package reorders all the elements to minimize the frontwidth using a variant of Cuthill and McKee's algorithm (Cuthill and McKee, 1969), since a frontal method is used in our Navier–Stokes solver. This remeshing package is quite reliable and efficient.

Some typical meshes used in the computation are displayed in Figure 3. In each window, on the left is an overall view of the mesh and on the right is a closer view near the cylinders. The remeshing package also has some geometric-based refinement capability in the regions where the cylinders are very close together and where the cylinders are very close to the channel walls, as indicated in Figure 3(b) and (d). The mesh size in these regions is of the size of the gap between the cylinders or between the cylinder and the channel wall.

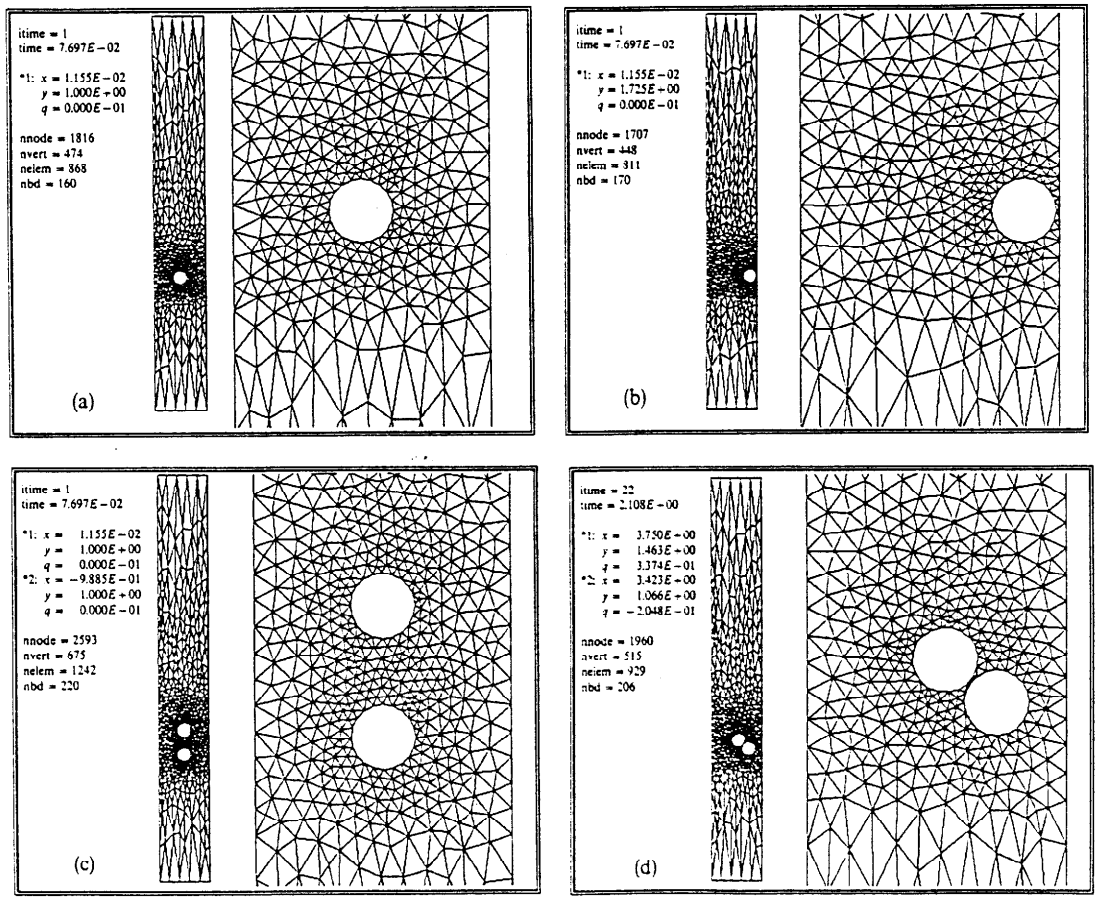


Figure 3. Some typical meshes used in the computation. In each window "itime" is the number of the time step, "time" is the actual time in seconds, (x, y, θ) are the positions of the cylinders, and "nnode, nvert, nelem, nbd" are the mesh information corresponding to the number of nodes, the number of the vertices, the number of elements, and the number of the boundary nodes in the mesh.

4. Projection

After calling the remeshing package we generate a mesh which is not related to the old mesh. We have to interpolate the fluid velocity from the old mesh onto the new mesh in order to evaluate the time derivative term in the Navier–Stokes equations, $\partial \mathbf{u}/\partial t$, which is, by definition, the derivative holding the position $\mathbf{x}$ fixed, which in discretized form is $(\mathbf{u}(\mathbf{x}, t_{n+1}) - \mathbf{u}(\mathbf{x}, t_n))/(t_{n+1} - t_n)$ for $t_{n+1} - t_n$ small. More precisely, assume that we have a new mesh with nodes at $\mathbf{x}(t_{n+1})$ and we have already calculated $\mathbf{u}(\mathbf{x}(t_n), t_n)$, the values of fluid velocity at the mesh nodes $\mathbf{x}(t_n)$ of the previous time t_n. We want to project this velocity field onto the new mesh nodes $\mathbf{x}(t_{n+1})$ to get $\mathbf{u}(\mathbf{x}(t_{n+1}), t_n)$.

In our projection package, for each node $\mathbf{x}$ in the new mesh (at present time) on which the velocity is being interpolated, we initiate a search in the old mesh (at previous time) to locate the element where the node $\mathbf{x}$ lies. Then the local coordinates for this node $\mathbf{x}$ are calculated. Finally, the values of the velocity components are interpolated using the known velocity on the local nodes and the interpolation functions on that element.

To search for the element where a given point lies, the easiest way is to look over all the elements in the mesh until the right one is found. This method is not efficient. An efficient searching algorithm requires easy access to the adjacency information of the finite elements. This information is hard to retrieve in the usual finite-element mesh data structure. Here we introduce a new mesh data structure based on segments. The usual mesh data information can be converted into this new data structure. A segment is a line in the mesh connecting two vertices from a vertex of low number to a vertex of high number. For segment i in the mesh, there are three two-dimensional arrays, "nod, elt, lnk", representing node, element, and linkage, respectively:

$$\begin{aligned} \mathrm{nod}(1, i) &= a, \qquad \mathrm{nod}(2, i) = b, \\ \mathrm{elt}(1, i) &= \ell, \qquad \mathrm{elt}(2, i) = r, \\ \mathrm{lnk}(1, i) &= j, \qquad \mathrm{lnk}(2, i) = k, \end{aligned}$$

where a and b are the vertex number on the segment i ordered so that $a < b$ or $\mathrm{nod}(1, i) < \mathrm{nod}(2, i)$. ℓ and r are the element number on the left and the right of the segment $i(ab)$, $\mathrm{elt}(1, i)$ is the element on the left of ab and $\mathrm{elt}(2, i)$ is the element on the right of ab. j and k are the segments that link with the segment ab, segment j belongs to the element on the right of ab and connects vertex a according to the ordinary rule and segment k connects vertex b, as shown in Figure 4. This mesh data structure is used in the package of Couniot *et al.* (1989) for simulation of injection molding. This mesh data structure has already been used in our remeshing package to generate the midnodes and to renumber the elements, since these tasks are extremely easy to fulfil with this data structure.

Suppose $\mathbf{x}$, $\mathbf{x}_a$, $\mathbf{x}_b$ are position vectors for the fluid point $\mathbf{x}$, and the nodal points a and b. Then if

$$[(\mathbf{x} - \mathbf{x}_a) \times (\mathbf{x} - \mathbf{x}_b)] \cdot \mathbf{k}$$

is greater than zero, the point $\mathbf{x}$ lies on the left side of the segment ab, otherwise it lies on the right side of the segment. We are looking for the element containing $\mathbf{x}$. Using this cross product the search scheme becomes quite easy. Given a point $\mathbf{x}$, start from an arbitrary segment i. If $\mathbf{x}$ is on the *left* side of the present segment i, choose the segment linked with the *end vertex* of the segment i as the next segment; if $\mathbf{x}$ is on the *right* side of the present segment i, choose the segment linked with the *start vertex* as the next segment. Repeat the process until the searching is looping inside an element. This element is the element where the given point lies. Figure 5 shows an example of the search scheme. In the figure the number in the circle represents the element number, the number in normal form is the vertex number, and the number in italic form indicates the search path. The search

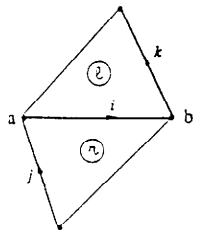


Figure 4. Mesh data structure based on segments. Segment i starts at vertex a and ends at vertex b $(a < b)$, it connects with segment j at a and connects with segment k at b. On the left of the segment is element ℓ and on the right is element r.

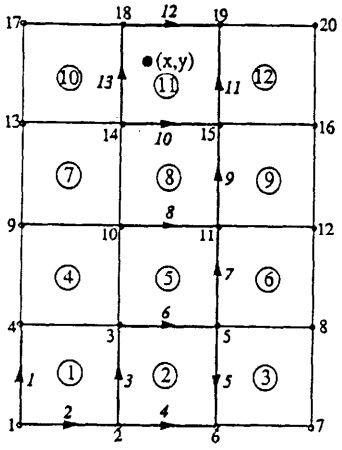


Figure 5. Diagram of a search to find the element where a given point (x, y) lies. The italic numbers indicate the search path. There are 20 vertices, 31 segments, and 12 elements in this mesh.

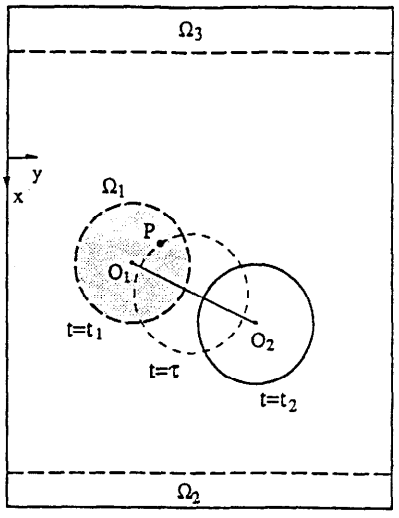


Figure 6. Regions, Ω_1 (shaded), Ω_2, and Ω_3, which need special treatments in the projection process. The computation domain at time t_1 is indicated with thick dashed lines, and the computation domain at time t_2 is indicated with the solid lines.

scheme can go around obstacles (like cylinders) in the domain, and can be made more efficient by starting with a better initial guess.

For a given point in an element, it is possible to get the local coordinates of the point analytically, if the sides of the element are straight. We find that it is easier to obtain the local coordinates by a Newton iteration. The Newton iteration can be easily extended to the case of curved boundaries and converges extremely fast. After we get the local coordinates for the point, it is straightforward to do the interpolation in the element. The velocities are interpolated quadratically using all the nodes in the element.

For two subsequent time steps, t_1 and t_2, the projection process described above works for most of the nodes in the new mesh at t_2. Since, however, the cylinders move in the channel, some nodes in the new mesh are found to be outside the old mesh at t_1, for example, in Figure 6, the nodes in the shaded region Ω_1 which was occupied by a cylinder at the previous time step and in the extra regions Ω_2 and Ω_3 created due to the adjustment of the boundaries Γ_3 and Γ_4 of the computational domain. We place the boundary Γ_3 10 diameters ahead of the first cylinder and Γ_4 20 diameters behind the last cylinder. As the cylinders move, these two boundaries move accordingly. The occurrence of region Ω_3 is rare in the sedimenting cylinder case, but it is still possible as we will see in the numerical results. In the projection, the velocities on the nodes in region Ω_2 are simply put to zero, $\mathbf{u}(\mathbf{x}, t_1) = 0$, since the fluid disturbance there is very small. The velocities on the nodes in region Ω_3 are determined by extrapolation using the velocities on the nodes inside the mesh. However, for the nodes in region Ω_1, projection of the velocity is done differently. We realize that the aim of projection is to calculate the time derivative for the fluid particles at the present time t_2. For any point $P(\mathbf{x})$ in Ω_1, see Figure 6, we can always find an intermediate time $\tau \in [t_1, t_2]$, such that P is on the surface of the cylinder at time τ. Then physically at time $t = t_2$, the time derivative for the fluid particle at point P should be discretized as $(\mathbf{u}(\mathbf{x}, t_2) - \mathbf{u}(\mathbf{x}, \tau))/(t_2 - \tau)$, where $\mathbf{u}(\mathbf{x}, \tau)$ is the velocity at P on the solid cylinder surface at $t = \tau$. If the motion of the cylinder at t_1 and t_2 is known, it is not hard to calculate τ and $\mathbf{u}(\mathbf{x}, \tau)$ by direct interpolation.

5. Discretization of the Equations

The Navier–Stokes solver in our program is based on a finite-element package POLYFLOW developed by Marcel Crochet *et al.* of the MEMA group at the Universite Catholique de Louvain, Belgium (see Crochet *et al.*, 1991). The Navier–Stokes equations are discretized with a Galerkin finite-element formulation in velocity and pressure form with P2/P1 elements in space, and with a

fully implicit finite difference scheme in time. At each time step, the resulting nonlinear matrix equations are handled by Newton iteration, and the linear equations in the iterations are solved with a frontal method. The forces acting on each part of the boundary are calculated by evaluating volume integrals on the elements adjacent to the boundary. In the nonlinear Newton iterations the forces acting on the cylinders and the residuals of the iteration are monitored to guarantee the convergence of the flow field at each time instant.

Initially, the fluid is at rest, the cylinders are held up at certain positions and the force acting on each cylinder is its weight minus buoyancy, $\mathbf{F}_i(0) = ((\rho_s - \rho_f)(\pi d^2/4), 0, 0)$. To simulate the unsteady motion of the cylinders and the flow of the fluid after the cylinders are set free, we first came up with a very simple explicit scheme which decouples the motion of the cylinders and the motion of the fluid at each time step.

Fully Explicit Scheme

(1) Initialization:

$$t_0 = 0, \qquad n = 0 \qquad \text{(index for time step)},$$

$$\mathbf{u}(\mathbf{x}(t_0), 0) = 0, \qquad p(\mathbf{x}(t_0), 0) = \rho_f g x(t_0), \tag{12}$$

and

$$\mathbf{X}_i(t_0) = \mathbf{X}_{i0}, \qquad \mathbf{U}_i(t_0) = 0, \qquad \mathbf{f}_i(t_0) = ((\rho_s - \rho_f)/\rho_s, 0, 0), \qquad \text{for } i = 1, \ldots, N. \tag{13}$$

(2) Updating: select an appropriate time step Δt_{n+1},

$$t_{n+1} = t_n + \Delta t_{n+1},$$

$$\mathbf{X}_i(t_{n+1}) = \mathbf{X}_i(t_n) + \Delta t_{n+1}\mathbf{U}_i(t_n), \tag{14}$$

and

$$\mathbf{U}_i(t_{n+1}) = \mathbf{U}_i(t_n) + \Delta t_{n+1}\mathbf{f}_i(t_n) \qquad \text{for } i = 1, 2, \ldots, N. \tag{15}$$

(3) Remeshing and projection: based on the positions of the cylinders $\mathbf{X}_i(t_{n+1})$, a new mesh $\mathbf{x}(t_{n+1})$ is generated, and the velocity $\mathbf{u}(\mathbf{x}(t_n), t_n)$ on the old mesh is projected onto the new mesh to get $\mathbf{u}(\mathbf{x}(t_{n+1}), t_n)$.

(4) Navier–Stokes solver: on the new mesh $\mathbf{x}(t_{n+1})$, with $\mathbf{u}(\mathbf{x}(t_{n+1}), t_n)$ as initial values and $\mathbf{U}_i(t_{n+1})$ prescribed on the cylinder surfaces, the Navier–Stokes solver is called to solve the fluid flow $\mathbf{u}(\mathbf{x}(t_{n+1}), t_{n+1})$, $p(\mathbf{x}(t_{n+1}), t_{n+1})$, and to calculate the forces on the cylinders $\mathbf{F}_i(t_{n+1})$ or $\mathbf{f}_i(t_{n+1})$.

(5) If time is less than T, then $n := n + 1$ and go to (2); otherwise stop.

Unfortunately this simple scheme is unstable. The solution blows up in few time steps. The problem is in the explicit discretization of (15). We can see why by considering the very early stages of the motion of one cylinder. At these stages the cylinder is accelerating, the velocity of the cylinder is very small, the nonlinear inertia term $\mathbf{u}\cdot\nabla\mathbf{u}$, and the viscous effect can be ignored. Thus the flow can be treated initially as potential flow. In the theory of potential flow, the equation of motion for an accelerating cylinder driven by a constant force in an infinite domain is written as (see Milne-Thomson, 1960, p. 241)

$$m\frac{dU}{dt} = f - m_v\frac{dU}{dt}, \tag{16}$$

where f is the driven force on the cylinder, m is the mass of the cylinder, and m_v is the virtual mass (or added mass) which is an equivalent mass of the fluid surrounding the cylinder being accelerated by the motion of the cylinder. The explicit discretization of (16) with an equal time increment Δt leads to

$$\begin{aligned} U(t_{n+1}) - U(t_n) &= \frac{f}{m}\Delta t - \frac{m_v}{m}[U(t_n) - U(t_{n-1})] \\ &= \frac{f\Delta t}{m + m_v}\left[1 - \left(-\frac{m_v}{m}\right)^n\right] - \left(\frac{m_v}{m}\right)^{n-1}[U(t_1) - U(t_0)] \qquad \text{for} \quad n \geq 1. \end{aligned} \tag{17}$$

Equation (17) shows that the explicit discretization is always unstable if $m_v \geq m$, whatever the time step Δt. In the ideal flow the virtual mass for a cylinder moving in an infinite domain is

$m_v = \rho_f(\pi d^2/4)$. Since $m_v < m$ the scheme is stable for potential flow. However, in a real fluid, "virtual mass" can be larger due to the adherence of fluid on the cylinder surface which causes a larger mass of fluid to accelerate with the cylinder. In the case of a cylinder with diameter d accelerating in a channel of width $4d$, the numerical tests show that in the viscous case the virtual mass is around $1.5\rho_f(\pi d^2/4)$, which can be larger than the mass $\rho_s(\pi d^2/4)$ of the cylinder.

The use of predictor corrector or high-order Runge–Kutta schemes prevents the above numerical instability, but, at each time step, these methods require us to call the Navier–Stokes solver at least twice, thus greatly increasing the cost. We decided to adopt another approach. As we have seen, the fully explicit scheme is unstable due to the drag force contributed by the acceleration of the mass of fluid around the cylinder. Therefore if this part of the drag force can be separated from the total drag force, then the scheme described above will probably be stable. This is done more generally as follows.

In general the forces acting on the cylinders depend on the positions, the velocities, and the accelerations of the cylinders:

$$\mathbf{F}_i(t) = \mathbf{F}_i(\mathbf{X}_1, \mathbf{U}_1, \mathbf{a}_1, \ldots, \mathbf{X}_N, \mathbf{U}_N, \mathbf{a}_N), \qquad i = 1, 2, \ldots, N, \tag{18}$$

where $\mathbf{a}_j = d\mathbf{U}_j/dt$ is the acceleration of cylinder j. The forces on the cylinders are determined by the Navier–Stokes equations where the positions and velocities of the cylinders act as the boundary and boundary conditions and the accelerations of the cylinders act implicitly as the initial condition. The cylinder velocity equations (3) are then discretized implicitly as

$$\frac{\mathbf{M}}{\Delta t_{n+1}}[\mathbf{U}_i(t_{n+1}) - \mathbf{U}_i(t_n)] = \mathbf{F}_i(t_{n+1}), \tag{19}$$

where

$$\mathbf{F}_i(t_{n+1}) = \mathbf{F}_i(\mathbf{X}_1(t_{n+1}), \mathbf{U}_1(t_{n+1}), \mathbf{a}_1(t_{n+1}), \ldots, \mathbf{X}_N(t_{n+1}), \mathbf{U}_N(t_{n+1}), \mathbf{a}_N(t_{n+1})). \tag{20}$$

An expansion of the above force $\mathbf{F}_i(t_{n+1})$ around $t = t_n$ leads to

$$\mathbf{F}_i(t_{n+1}) \approx \mathbf{F}_i(t_n) + \frac{\partial \mathbf{F}_i}{\partial \mathbf{a}_j}(t_n)[\mathbf{a}_j(t_{n+1}) - \mathbf{a}_j(t_n)] + \frac{\partial \mathbf{F}_i}{\partial \mathbf{U}_j}(t_n)[\mathbf{U}_j(t_{n+1}) - \mathbf{U}_j(t_n)] + \frac{\partial \mathbf{F}_i}{\partial \mathbf{X}_j}(t_n)[\mathbf{X}_j(t_{n+1}) - \mathbf{X}_j(t_n)], \tag{21}$$

which is valid for a small time increment Δt_{n+1}, since the changes in the cylinder velocities $\mathbf{U}_j$ and in the cylinder positions $\mathbf{X}_j$ are small, and the forces $\mathbf{F}_i$ are linearly dependent on the accelerations $\mathbf{a}_j$ (see Batchelor, 1967, p. 407).

The expansion (21) can be simplified in many ways. First we assume that the variation of the forces on one cylinder caused by the change of its own motion is much larger than the variation caused by the change of the motion of the other cylinders. This assumption breaks down in extreme cases like kissing cylinders, but we use the results derived here as an initial guess for another implicit scheme. Thus in the expansion (21) the summation over j can be replaced with index i. Then we argue that the forces acting on the cylinders only weakly depend on the positions of the cylinders. The terms of the matrix $\partial \mathbf{F}_i/\partial \mathbf{X}_j$ can be neglected. Furthermore the coupling of the forces due to the accelerations in different directions is weak, e.g., the force in the x direction is slightly affected by the change of the acceleration in the y direction. Under these simplifications (21) reduces to the following component form:

$$\begin{aligned} F_{ix}(t_{n+1}) = F_{ix}(t_n) &+ \frac{\partial F_{ix}}{\partial a_{ix}}(t_n)\left[\frac{dU_i}{dt}(t_{n+1}) - \frac{dU_i}{dt}(t_n)\right] + \frac{\partial F_{ix}}{\partial U_i}(t_n)[U_i(t_{n+1}) - U_i(t_n)] \\ &+ \frac{\partial F_{ix}}{\partial V_i}(t_n)[V_i(t_{n+1}) - V_i(t_n)] + \frac{\partial F_{ix}}{\partial \Omega_i}(t_n)[\Omega_i(t_{n+1}) - \Omega_i(t_n)], \end{aligned} \tag{22}$$

$$\begin{aligned} F_{iy}(t_{n+1}) = F_{iy}(t_n) &+ \frac{\partial F_{iy}}{\partial a_{iy}}(t_n)\left[\frac{dV_i}{dt}(t_{n+1}) - \frac{dV_i}{dt}(t_n)\right] + \frac{\partial F_{iy}}{\partial U_i}(t_n)[U_i(t_{n+1}) - U_i(t_n)] \\ &+ \frac{\partial F_{iy}}{\partial V_i}(t_n)[V_i(t_{n+1}) - V_i(t_n)] + \frac{\partial F_{iy}}{\partial \Omega_i}(t_n)[\Omega_i(t_{n+1}) - \Omega_i(t_n)], \end{aligned} \tag{23}$$

$$F_{im}(t_{n+1}) = F_{im}(t_n) + \frac{\partial F_{iz}}{\partial \Omega_i}(t_n)[\Omega_i(t_{n+1}) - \Omega_i(t_n)], \tag{24}$$

where the repeated indices over i does not mean summation over i. We want to evaluate the derivatives on the right-hand side of (22), (23), and (24).

We know that in potential theory the forces acting on a single cylinder moving in an infinite domain with velocities $(U(t), V(t), \Omega(t))$ are given as (see Milne–Thomson, 1960)

$$F_x = -m_v \frac{dU}{dt} - 2m_v \Omega V$$

and

$$F_y = -m_v \frac{dV}{dt} + 2m_v \Omega U,$$

where $2m_v \Omega V$ and $2m_v \Omega U$ are the "lift" forces. We then generalize the above expressions in viscous fluid by writing

$$F_{ix} = -m_v \frac{dU_i}{dt} - 2m_v \Omega_i V_i - \frac{d}{2} \rho_f U_i^2 C_D(U_i) + \tilde{F}_{ix}, \tag{25}$$

$$F_{iy} = -m_v \frac{dV_i}{dt} + 2m_v \Omega_i U_i - \frac{d}{2} \rho_f V_i^2 C_D(V_i) + \tilde{F}_{iy}, \tag{26}$$

and

$$F_{im} = -C_m \Omega_i + \tilde{F}_{im}, \tag{27}$$

where $(d/2)\rho_f U_i^2 C_D(U_i)$ and $(d/2)\rho_f V_i^2 C_D(V_i)$ are the viscous drags in the x and y directions, respectively, and $C_m \Omega_i$ is the viscous torque. C_D is the drag coefficient defined as usual and C_m is the coefficient for the torque. m_v is the "virtual" mass in real fluid, which is different from the one in an ideal fluid. Residual forces are called $\tilde{F}_{ix}$, $\tilde{F}_{iy}$, and $\tilde{F}_{im}$. They do not depend strongly on the direct accelerations and the velocities of the associated cylinder and they do not enter into the evaluation of the derivatives.

Using the force expressions (25)–(27) the derivatives in the expansions (22)–(24) can be evaluated:

$$\frac{\partial F_{ix}}{\partial a_{ix}} = \frac{\partial F_{iy}}{\partial a_{iy}} = -m_v, \tag{28}$$

$$\frac{\partial F_{ix}}{\partial V_i} = -2m_v \Omega_i, \qquad \frac{\partial F_{ix}}{\partial \Omega_i} = -2m_v V_i, \tag{29}$$

$$\frac{\partial F_{iy}}{\partial U_i} = 2m_v \Omega_i, \qquad \frac{\partial F_{iy}}{\partial \Omega_i} = 2m_v U_i, \tag{30}$$

$$\frac{\partial F_{ix}}{\partial U_i} \overset{\text{def}}{=} -C_{ix} = -d\rho_f U_i C_D(U_i) - \frac{d}{2} \rho_f U_i^2 C_D'(U_i), \tag{31}$$

$$\frac{\partial F_{iy}}{\partial V_i} \overset{\text{def}}{=} -C_{iy} = -d\rho_f V_i C_D(V_i) - \frac{d}{2} \rho_f V_i^2 C_D'(V_i), \tag{32}$$

$$\frac{\partial F_{im}}{\partial \Omega_i} = -C_{im}. \tag{33}$$

In these expressions the virtual mass m_v for a viscous fluid can be estimated by numerical experiments. The value of the drag coefficient C_D can be obtained from experimental curves or empirical formulations. C_m can be estimated by considering a cylinder rotating in viscous fluid in an infinite domain, which gives $C_m = 2\mu_f$. Substituting all the expressions back into (19) we have three equations for updating the velocity vector $\mathbf{U}_i(t_{n+1}) = (U_i(t_{n+1}), V_i(t_{n-1}), \Omega_i(t_{n+1}))$ for each cylinder:

$$\left(\frac{m + m_v}{\Delta t_{n+1}} + C_{ix}\right)[U_i(t_{n+1}) - U_i(t_n)] + 2m_v \Omega_i(t_n)[V_i(t_{n-1}) - V_i(t_n)] + 2m_v V_i(t_n)[\Omega_i(t_{n+1}) - \Omega_i(t_n)]$$
$$= F_{ix}(t_n) + m_v \frac{U_i(t_n) - U_i(t_{n-1})}{\Delta t_n}, \tag{34}$$

$$\left(\frac{m+m_v}{\Delta t_{n+1}}+C_{iy}\right)[V_i(t_{n+1})-V_i(t_n)]-2m_v\Omega_i(t_n)[U_i(t_{n+1})-U_i(t_n)]-2m_vU_i(t_n)[\Omega_i(t_{n+1})-\Omega_i(t_n)]$$

$$=F_{iy}(t_n)+m_v\frac{V_i(t_n)-V_i(t_{n-1})}{\Delta t_n}, \tag{35}$$

$$\left(\frac{I}{\Delta t_{n+1}}+C_{im}\right)[\Omega_i(t_{n+1})-\Omega_i(t_n)]=F_{im}(t_n). \tag{36}$$

From these equations, $U_i(t_{n+1})$, $V_i(t_{n+1})$, and $\Omega_i(t_{n+1})$ can easily be found. Having found $\mathbf{U}_i(t_{n+1})$, the cylinder positions can be updated

$$\mathbf{X}_i(t_{n+1})=\mathbf{X}_i(t_n)+\Delta t_{n+1}\mathbf{U}_i(t_{n+1}). \tag{37}$$

By replacing (14) and (15) in the Fully Explicit Scheme with (34)–(37) we have the Improved Explicit Scheme. This scheme is stable and works well in most situations, especially in the case of only one cylinder. However, in some cases when the cylinders are close to each other or the cylinders are close to the channel walls, this scheme breaks down, as may be expected from the assumptions made in the derivation of the scheme.

To avoid the problems of the improved fully explicit scheme to which we have alluded, we developed the Explicit–Implicit Scheme which uses an explicit method to update the positions of the cylinders and an implicit method to update the velocities of the cylinders. Therefore at each time step the velocities of the cylinders are determined iteratively, coupled with the solution of the flow. We found that the iteration for the cylinder velocities can be buried inside the nonlinear Newton iteration of the Navier–Stokes solver, allowing for an increased efficiency such that the increased stability of the scheme can be achieved at a cost not much greater than in the fully explicit scheme.

Explicit–Implicit Scheme

(1) Initialization:

$$t_0=0, \qquad n=0 \qquad \text{(index for time step)},$$

$$\mathbf{u}(\mathbf{x}(t_0),0)=0, \qquad p(\mathbf{x}(t_0),0)=\rho_f g x(t_0),$$

and

$$\mathbf{X}_i(t_0)=\mathbf{X}_{i0}, \qquad \mathbf{U}_i(t_0)=0, \qquad \mathbf{f}_i(t_0)=((\rho_s-\rho_f)/\rho_s,0,0), \qquad \text{for } i=1,\ldots,N.$$

(2) Updating 1: select an appropriate time step Δt_{n+1},

$$t_{n+1}=t_n+\Delta t_{n+1},$$

$$\mathbf{X}_i(t_{n+1})=\mathbf{X}_i(t_n)+\Delta t_{n+1}\mathbf{U}_i(t_n) \qquad \text{for} \quad i=1,\ldots,N. \tag{38}$$

$k=0$ (index for iteration), initial updating for the cylinder velocities

$$\mathbf{U}_i^{(0)}(t_{n+1})=(U_i(t_{n+1}),V_i(t_{n+1}),\Omega_i(t_{n+1})) \qquad \text{from (34), (35), and (36).} \tag{39}$$

(3) Remeshing and projection: based on the positions of the cylinders $\mathbf{X}_i(t_{n+1})$, a new mesh $\mathbf{x}(t_{n+1})$ is generated. The velocity $\mathbf{u}(\mathbf{x}(t_n),t_n)$ on the old mesh is projected onto the new mesh to get $\mathbf{u}(\mathbf{x}(t_{n+1}),t_n)$.

(4) One step Navier–Stokes iteration: on the new mesh $\mathbf{x}(t_{n+1})$, with $\mathbf{u}(\mathbf{x}(t_{n+1}),t_n)$ as the initial values and $\mathbf{U}_i^{(k)}(t_{n+1})$ prescribed on the cylinder surfaces, perform *one* Newton iteration in the Navier–Stokes solver to get a flow field $\mathbf{u}^{(k)}(\mathbf{x}(t_{n+1}),t_{n+1})$, $p^{(k)}(\mathbf{x}(t_{n+1}),t_{n+1})$, and calculate the force vector $\mathbf{F}_i^{(k)}(t_{n+1})$, or $\mathbf{f}_i^{(k)}(t_{n+1})$.

(5) Updating 2: $k:=k+1$,

$$\Delta\mathbf{U}_i^{(k)}(t_{n+1})=-\mathbf{U}_i^{(k)}(t_{n+1})+\mathbf{U}_i(t_n)+\Delta t_{n+1}\mathbf{f}_i^{(k-1)}(t_{n+1}), \tag{40}$$

$$\mathbf{U}_i^{(k)}(t_{n+1})=\mathbf{U}_i^{(k-1)}(t_{n+1})+\alpha\Delta\mathbf{U}_i^{(k)}(t_{n+1}) \qquad \text{for } i=1,\ldots,N, \tag{41}$$

where α is a underrelaxation parameter.

(6) Convergence test:

$$\varepsilon_1 = \|\Delta \mathbf{U}_i^{(k)}(t_{n+1})\| \quad \text{and} \quad \varepsilon_2 = \|\mathbf{u}^{(k)}(\mathbf{x}(t_{n+1}), t_{n+1}) - \mathbf{u}^{(k-1)}(\mathbf{x}(t_{n+1})t_{n+1})\|,$$

if ε_1 and ε_2 are greater than a prescribed value ε go to (4), otherwise,

$$\mathbf{U}_i(t_{n+1}) = \mathbf{U}_i^{(k)}(t_{n+1}), \qquad \mathbf{f}_i(t_{n+1}) = \mathbf{f}_i^{(k)}(t_{n+1}),$$

and

$$\mathbf{u}(\mathbf{x}(t_{n+1}), t_{n+1}) = \mathbf{u}^{(k)}(\mathbf{x}(t_{n+1}), t_{n+1}), \qquad p(\mathbf{x}(t_{n+1}), t_{n+1}) = p^{(k)}(\mathbf{x}(t_{n+1}), t_{n+1}).$$

(7) If the time t_{n+1} is less than T, then $n := n + 1$ and go to (2); otherwise stop.

The time accuracy of the scheme is Δt, where Δt is the time step. The choice of the time step Δt_{n+1} in the scheme depends on many factors. We choose the time step to restrict the maximum distance each cylinder can travel in that time step, to restrict the maximum change in the cylinder velocities, to avoid collisions between the cylinders and between the cylinder and channel walls. We also restrict the time step so that it is small enough to capture the vortex shedding in the flow, the time step is always less than one-tenth of the period of vortex shedding. The Jacobian matrix for the linearized equations in the nonlinear Newton iteration of the Navier–Stokes solver is calculated and inverted once, and is recalculated and inverted only when the rate of convergence is too slow. In this scheme, for each time step, the updating equations (34)–(36) are used only to provide an initial guess for the cylinder velocities. Usually this guess is a very good starting point for the iteration. The use of the fully explicit updating scheme for this initial guess of the cylinder velocities usually causes oscillation in the iteration, requires more iterations to converge, and sometimes even blows up. We find that it is helpful not to update the cylinder velocities for the first one or two Newton iterations since the flow is not converged yet. In updating the velocities of the cylinders with (40) and (41) it is necessary to introduce underrelaxation. We found that a good value for the relaxation parameter α is about 0.4 for most cases. This can be explained more or less by the virtual mass argument described before. A smaller value of α should be used when the cylinders are very close to each other or when the cylinders are very close to the channel walls, otherwise the iteration may not converge.

6. Results and Discussion

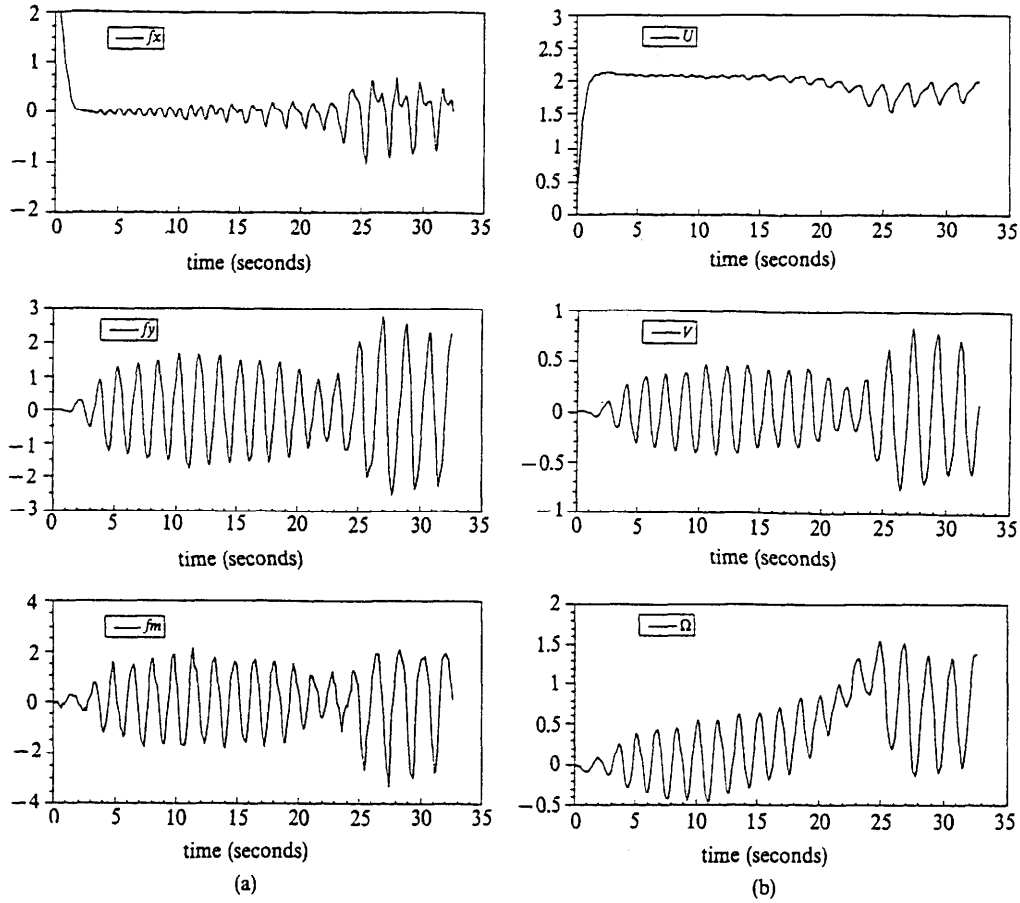
In our computation, the "inflow" boundary Γ_3 of the computational domain is placed 10 diameters ahead of the first cylinder and the "outflow" boundary Γ_4 is placed 20 diameters behind the last cylinder. The surface of each cylinder is divided into 20 equal segments, a polygon of 20 sides.

All our computation was done on a Cray 2. For each time step, computed results of the flow field (velocities and pressure) and the motion of the cylinders (force, velocity, and position vectors) are saved. This data is used for postprocessing on a personal computer. We developed a program that can interactivity display all the information about the simulation and can animate the motion of the cylinders. The program can plot the boundary of the domain, the mesh, stream lines, isovorticity lines, isopressure lines, and velocity vectors.

The fluid in our simulation is water, $\rho_f = 1.0$ g/cm^3 and $\mu_f = 0.01$ poise. The density of the solid cylinders is $\rho_s = 1.01$ g/cm^3 so that the Reynolds numbers in our calculations are not too large. All the numerical results are obtained with the Explicit–Implicit Scheme. Our results are displayed graphically in the figures starting with Figure 7. All quantities in the graphs are expressed in CGS units.

In the computation the total number of nodes (nnode), the total number of elements (nelem), and the total unknowns (nvar) vary depending on the number and positions of the cylinders and also on the width of the channel. Typically nnode is about 2000–4000, nelem is about 1000–2000, and nvar is around 4500–9000. A run of 200 time steps takes 2–10 h on Cray 2 depending on the values of parameters.

First we simulate the motion of one sedimenting cylinder. The width of the channel is $4d$ where d is the diameter of the cylinder. In the first run $d = 0.5$ cm. The cylinder starts at the center of the channel. The time history of the motion of the cylinder is shown in Figure 7. The dominant feature in the figure is the periodic oscillation. This oscillation is associated with vortex shedding behind the



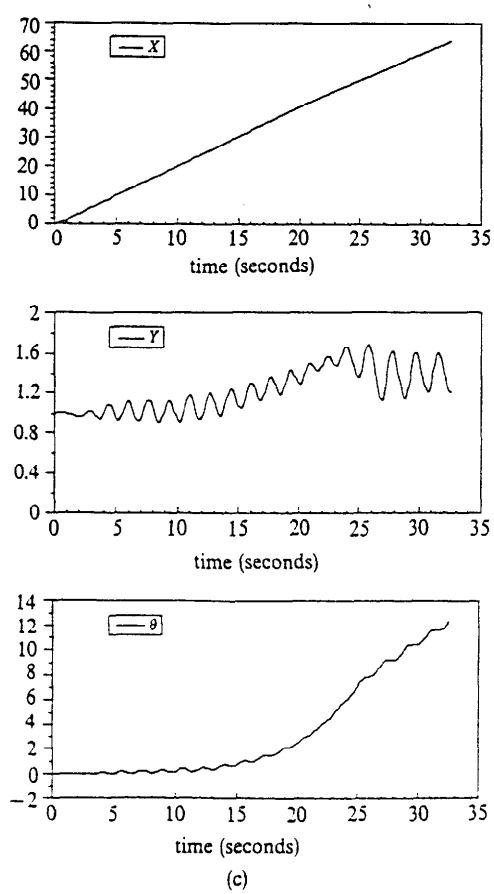


Figure 7. A sedimenting cylinder in a channel. The diameter of the cylinder is $d = 0.5$ cm, the width of the channel is $W = 4d$. The cylinder starts at the center of the channel. Time histories of the acceleration vector, the velocity vector, and the position vector are shown in (a), (b), and (c).

cylinder. The figure shows that after 25 s the period of the oscillation is about 1.85 s (or the frequency $f = 0.541$ Hz) and the average velocity of the falling cylinder is about 1.8 cm/s (the Reynolds number based on the cylinder diameter is about $Re = 90$). These values give a dimensionless Strouhal number $S = fd/U = 1.50$. In the case of uniform flow past a stationary cylinder, Swanson (1970) gives the following empirical formula for the Strouhal number:

$$S = 0.212(1 - 21.2/Re) \qquad \text{for} \quad Re < 150, \tag{42}$$

which yields $S = 1.62$ when $Re = 90$. This value is quite close to the value measured in our numerical simulation. The slight difference may be caused by the fact that in our simulation the cylinder is moving in response to the vortex shedding.

Some features of the dynamics of vortex shedding behind the sedimenting cylinder are presented in Figure 8. From these plots we can clear see the growth, separation, and decay of the vortices. We see the wake in the stream-line plot and the high pressure zone in front of the cylinder in the isopressure plot. As the cylinder falls, it zigzags in the channel in response to the alternate shedding of vortex. In our postprocessing software, the stream function is directly integrated from the values of the velocities instead of solving the corresponding partial differential equation. Sometimes the stream lines are not so smooth, especially in the region behind the cylinder where the mesh is coarse and value of the stream function is small.

Now let us took at the motion of the cylinder in Figure 7 more carefully. The cylinder first accelerates smoothly to a velocity of about 2 cm/s (or $Re = 100$) without rotation or motion in the y direction. Meantime the vortices behind the cylinder grow and are shed from the cylinder in a

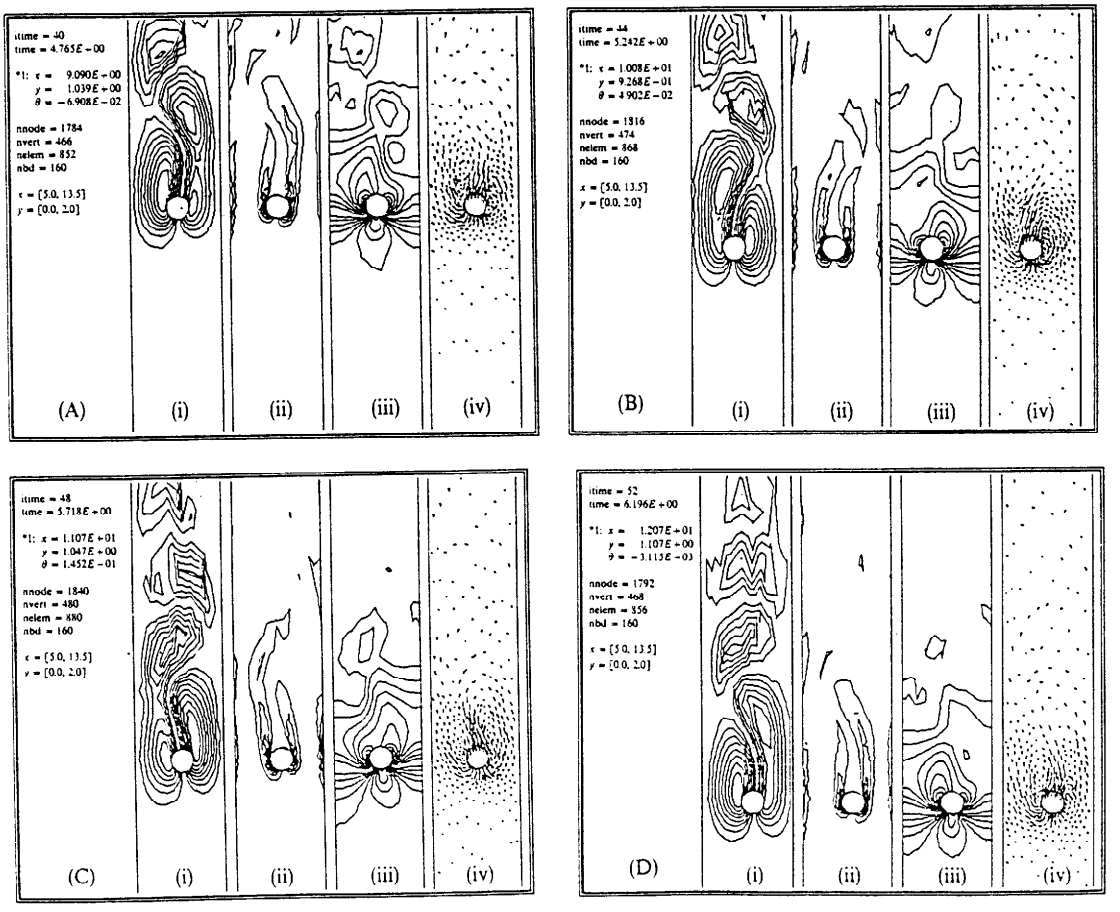


Figure 8. Fluid dynamics of vortex shedding behind a sedimenting cylinder in a channel. The diameter of the cylinder $d = 0.5$ cm, the width of the channel is $W = 4d$. The resulting average Reynolds number is about 90. In each window: (i) stream lines, (ii) isovorticity lines, (iii) isopressure lines, and (iv) velocity vector plot.

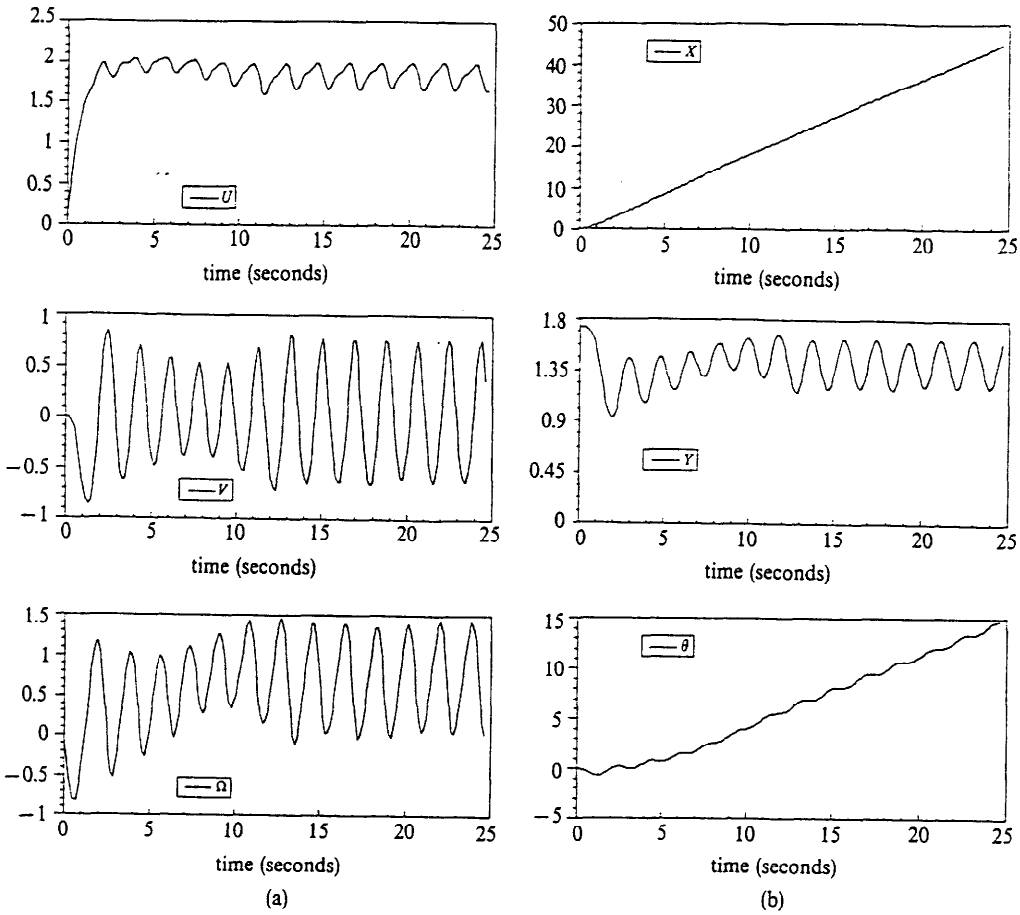


Figure 9. A sedimenting cylinder in a channel. The same condition as in the Figure 7 except that the cylinder starts at a position $Y = 1.725$ cm, very close to the channel wall $y = 2$ cm. (a) and (b) are the time histories of the velocity vector and the position vector.

staggered sequence. This vortex shedding creates an oscillating force and torque on the cylinder, and induces a motion of the cylinder. As the oscillation develops, the cylinder does not stay at the center of the channel, it drifts to one of the channel walls. Due to uneven shear on the two sides of the cylinder, the cylinder rotates more in one direction as it drifts. As the cylinder comes closer to the wall, the squeezing motion creates a lubrication force that pushes the cylinder away from the wall. After a few zigzags the cylinder enters into a periodic oscillation centered around an equilibrium position at $y = 0.7W = 1.4$ cm. As the cylinder falls it oscillates around this position with a constant amplitude.

Figure 9 shows another run with the same cylinder, but the cylinder starts at a position $y = 1.725$ cm, very close to the channel wall $y = 2$ cm. As soon as the cylinder is set free, it drifts and rotates to the center of the channel as it falls down, but the cylinder does not stay at the center, it reaches a periodic oscillating orbit much faster than in the previous run. The vortex-shedding frequency is almost the same as in the previous case. We noted that the equilibrium oscillation of the cylinder is centered at the same position $y = 0.7W = 1.4$ cm and with the same amplitude as in the previous case. Thus, for a cylinder sedimenting at large Reynolds numbers, the vortex shedding causes the cylinder to drift off the center of the channel and keeps the cylinder oscillating around an equilibrium position $y = 0.7W$ with a certain amplitude. We do not yet know how this equilibrium position depends on other parameters, like the Reynolds number, the channel width, etc.

We call the reader's attention to an asymmetry in the oscillation response to vortex shedding. This asymmetry is present say in the f_x and U plot in Figure 7(a) and (b). Since the cylinder is closer to one side of the channel, the vortex developed on that side of the cylinder is smaller and weaker than

the vortex on the other side of the cylinder. The large negative peaks on the curve of f_x correspond to the separations of the the large and strong vortices. The high positive peaks correspond to the separations of the small and weak vortices on the side of the cylinder close to the wall. The large vortex takes longer to grow and separate.

Since the time accuracy of the scheme is Δt, we rerun the case corresponding to Figure 9 with the time step reduced by half. There is no noticeable differences between the results at the first few time steps. As the time steps increase, the result starts to deviate due to the accumulated error, but the characteristics of the motion of the cylinder are the same. The differences in the frequency and amplitude of the oscillation and in the equilibrium position are within 10%.

Next we try the case of a smaller cylinder, $d = 0.25$ cm, and the channel width is still $4d$. The cylinder starts from a position close to one side wall of the channel. As we see from Figure 10, the cylinder first drifts to the center of the channel, oscillates back and forth a few times, then stabilizes around a position $y = 0.53$ cm (the center of the channel is at $y = 0.5$ cm). The motion of the cylinder is not fully stabilized yet, its rotation is still slowly decaying. The cylinder rotates slowly while falling straight down. This small rotation keeps the cylinder slightly off the center of the channel. For this case the final velocity for the cylinder is about 1.13 cm/s, corresponding to $Re = 28.3$. Thus we see that, for a cylinder sedimenting at small Reynolds numbers without vortex shedding, the cylinder will stabilize close to the center of the channel.

Now let us look at the interaction of two cylinders. We simulated drafting, kissing, and tumbling which is a local rearrangement mechanism in a two-phase flow of solids and liquids in beds of particles confined to move in two dimensions. Figure 11 shows what happens when two cylinders of

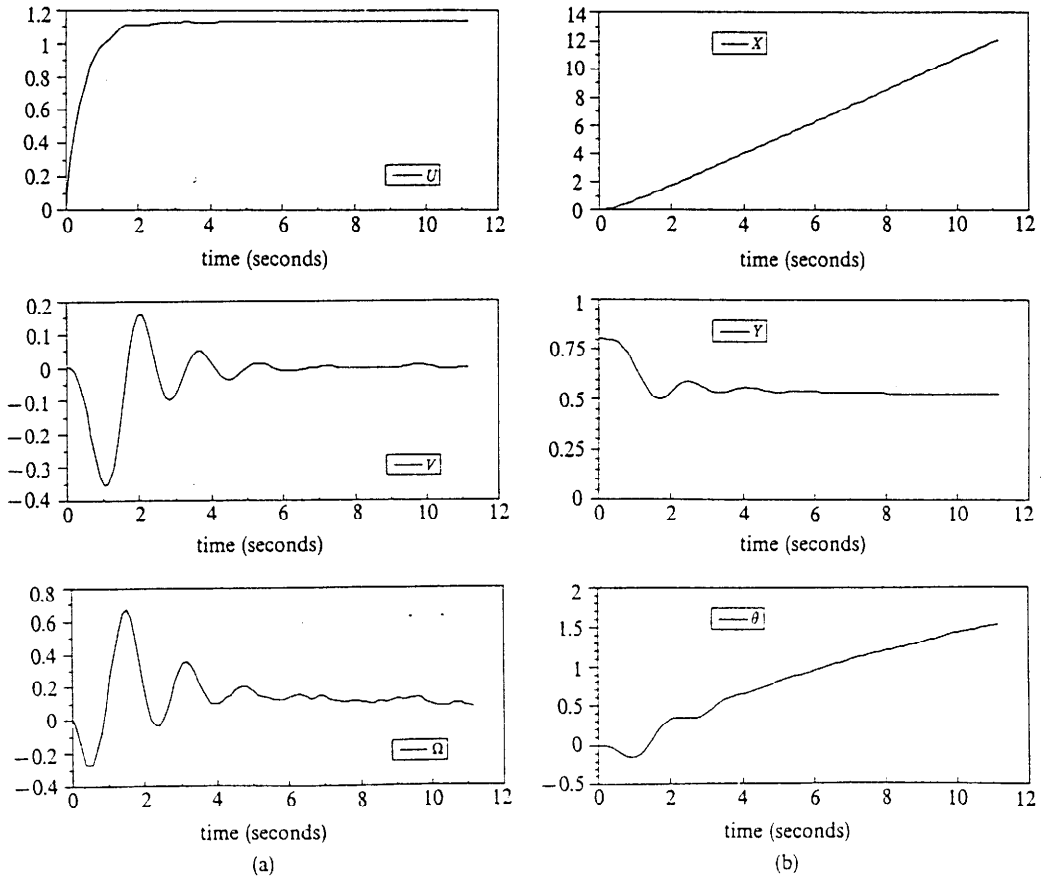


Figure 10. A sedimenting cylinder in a channel. The diameter of the cylinder is $d = 0.25$ cm, the width of the channel is $W = 4d$. The cylinder start at a position $Y = 0.8$ cm, close to the channel wall $y = 1$ cm. (a) and (b) are the time histories of the the velocity vector and the position vector.

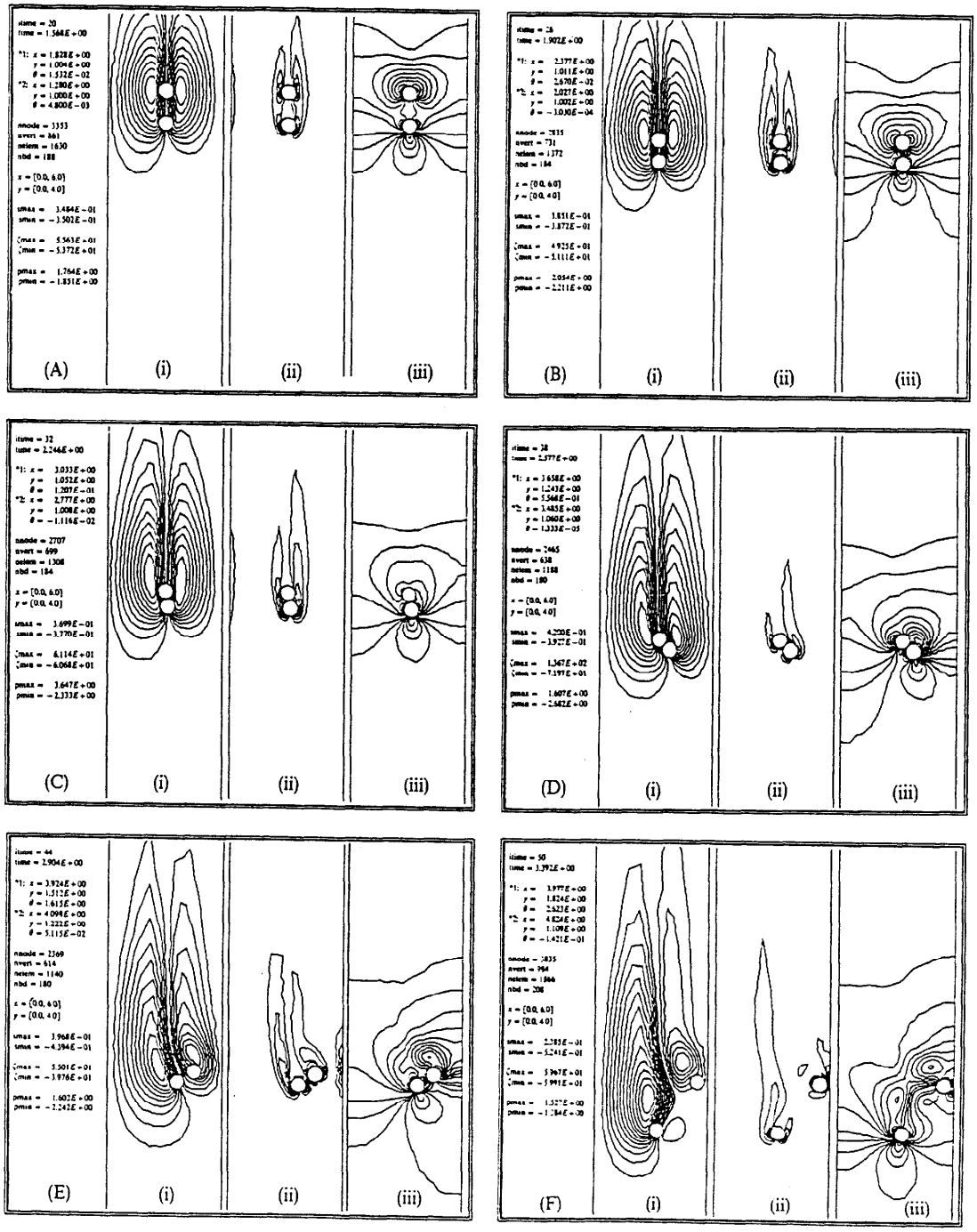


Figure 11. Drafting, kissing, and tumbling—interactions of two cylinders sedimenting in a channel. The diameter of the cylinders is $d = 0.25$ cm. The width of the channel is $W = 8d$. In each window smax and smin, ζ min, and pmax and pmin are the maximum and minimum values of the stream function, vorticity, and pressure, respectively; (i) stream lines, (ii) isovorticity lines, and (iii) isopressure lines.

diameter $d = 0.25$ cm fall in a channel of width $W = 8d$. The Reynolds number corresponding to the terminal velocity of the cylinders is about 30 in this case. Thus the wakes behind the cylinders are stable and there is no vortex shedding. As the cylinders fall in the channel, one of them is caught in the wake of the other as shown in Figure 11(A). The cylinder behind experiences less drag and accelerates to a higher velocity following the front cylinder, as in a drafting strategy in cycling racing (B). Finally, the cylinders "kiss" (C). In the present simulation the cylinders do not actually collide, the relative squeezing motion between them creates a lubrication force that keeps them apart. The streamwise alignment of the two cylinders is very unstable, a slight misalignment will push the front cylinder aside, the cylinder behind then takes the lead. The cylinders tumble (D), (E). The tumbling of these two cylinders is very robust. The cylinder originally in front stops falling, even acquires a negative velocity and moves upwards slightly. It is being ejected with a quite large velocity, comparable with the terminal velocity of the cylinder, toward the side wall of the channel. After the tumbling the cylinders separate (F). Sometime later this scenario repeats.

We ran some tests for different combinations of cylinder diameter and channel width. We reproduced the drafting, kissing, and tumbling scenario in most cases except for the case in which our simulation broke down. We suspect that in these few cases the actual collision happens between cylinders or between cylinder and channel wall, which is still not incorporated into our program. Figures 12–14 shows the time history for four cases tested. Two characteristic features of the drafting, kissing, and tumbling shown in the time-history curves are the sudden decrease of velocity of one cylinder shown in the plots of the velocity U versus time and the crossing of trajectories which are shown in the plots of X versus time. We say that the interacting cylinders tumble if they exchange position in the plot of the position X.

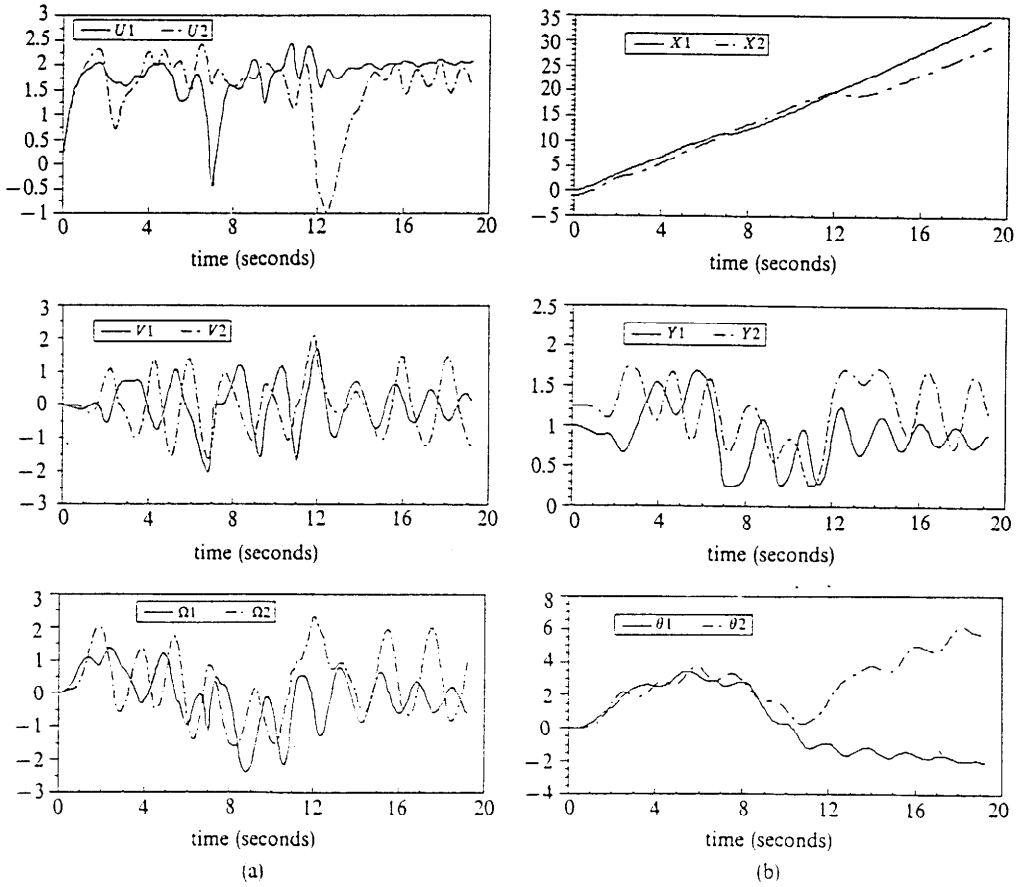


Figure 12. Two sedimenting cylinders in a channel. The diameter of the cylinders is $d = 0.5$ cm, the width of the channel is $W = 4d$. The starting positions for the cylinders are $(X, Y) = (0, 1.0)$ and $(-1, 1.25)$. (a) and (b) are the time histories of the velocity vectors and position vectors.

In Figure 12 the cylinder diameter is $d = 0.5$ cm and the width of the channel is $W = 4d$. The front cylinder starts at the center of the channel and the other one starts about two diameters behind and slightly off the center. There are two crossings in the plot of the position X, corresponding to two tumbling interactions. There is another tumbling attempt at approximately 2.5 s, but it does not complete because the vortex shedding on the front cylinder sweeps aside the cylinder behind. The Reynolds number for this case is quite large, about 100, the periodic oscillation in the curves is due to the vortex shedding. In this simulation the two tumblings occur when one of the cylinder is close to the channel wall, as seen in the plot of the position Y. To be sure that the tumbling we see is not totally caused by the channel wall, we ran a test with the same cylinders in a wider channel, $W = 8d$. One cylinder is placed directly behind the other. The result is shown in Figure 13. The near crossing in the plots of position X shown in Figure 12 actually crosses in Figure 13. This tumbling occurs at the center of the channel, it is not due to the walls.

We then reduce the size of the cylinders to determine effects due to the Reynolds number. Figure 14 shows the result for two cylinders with $d = 0.25$ cm in a channel of $W = 8d$ corresponding to the case displayed in Figure 11. The drafting, kissing, and tumbling scenario is very much like the one shown in Figure 13. However, the Reynolds number in Figure 14 is much smaller (around 30) and there is no vortex shedding.

In Figure 15 the diameter of the cylinder is reduced to 0.1 cm, and the width of the channel is fixed at $W = 8d$. The Reynolds number for this simulation is about 0.8. In this case the computation breaks down, since the iteration between the implicit cylinder velocity equations and the Navier–Stokes equations does not converge when two cylinders are very close together. In this figure we can still see drafting, kissing, and the start of tumbling. In a real situation, at small Reynolds numbers, we suspect that the cylinders actually kiss, momentarily glue together, and then tumble.

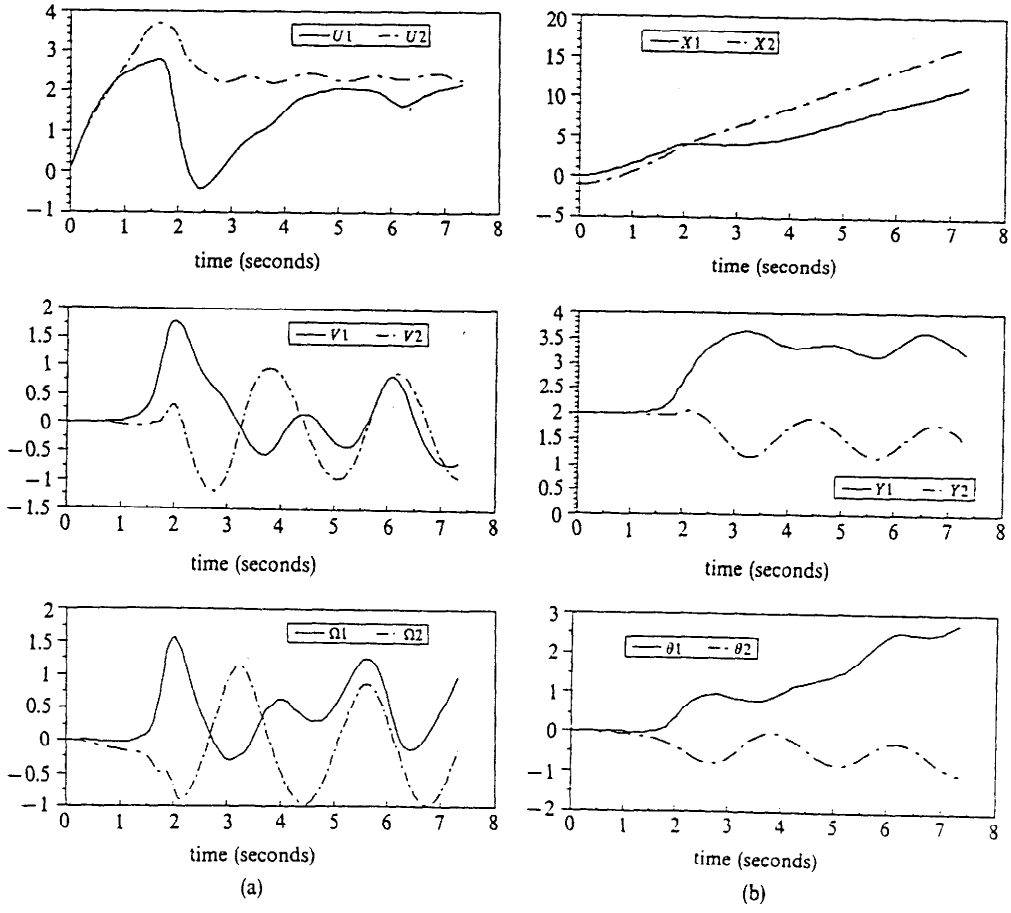


Figure 13. Same as Figure 12 except that the width of the channel is $W = 8d$ and the cylinders start at the center of the channel with one cylinder two diameters behind the other.

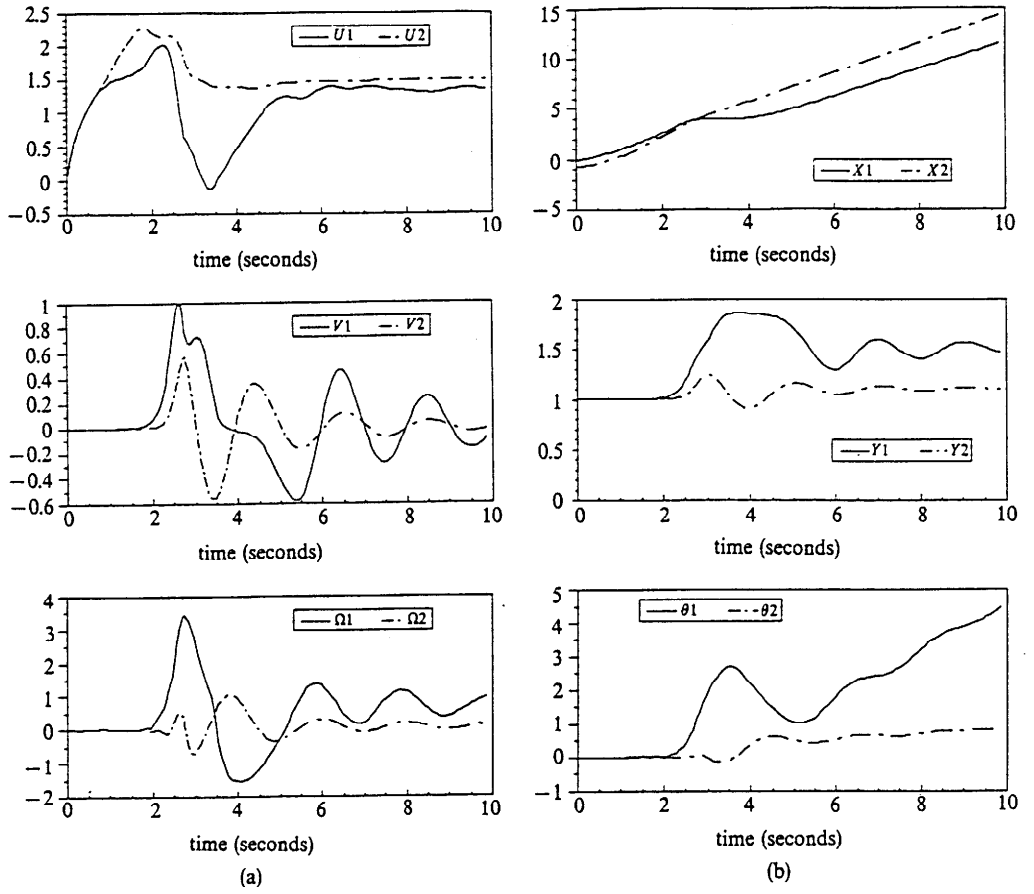


Figure 14. Two sedimenting cylinders in a channel. The diameter of the cylinders is $d = 0.25$ cm, the width of the channel is $W = 8d$. The cylinders start at the center of the channel with one cylinder three diameters behind the other.

We did flow-tracing for our program. A typical 50 s of CPU time is required to advance one time step in a simulation of one cylinder sedimenting in a channel of width $4d$. About 99% of the time used is consumed in the Navier–Stokes solver. The remeshing part takes about 0.46%, the projection part takes about 0.28%, and writing a formatted result takes about 0.2%. For the cases of two cylinders in a wider channel, the remeshing and the projection takes an even smaller percentage of the total time. For each time step, the Navier–Stokes solver usually needs about 10 iterations, and the Jacobian matrix for the iteration usually needs to be calculated and inverted twice. We think that the problem of slow convergence is due to the poor initial guess for the flow field in nonlinear iteration. The projected initial value of velocity at each time step usually is not a good initial guess to start the nonlinear iteration. Some more work can be done in this part to save CPU time. In general, however, the inversion of the Jacobian is the most time-consuming part in this kind of computation.

7. Conclusions

1. We developed a package that simulates unsteady two-dimensional solid–liquid two-phase flow using the Navier–Stokes equations for the liquid and Newton's equations of motion for the solid particles. This package consists of four major parts: a remeshing routine, a projection routine, an updating routine for the positions and velocities of the particles, and a Navier–Stokes solver. The package can be and is being improved in many ways, like a more accurate scheme (of Δt^2). More applications are going to be carried out in the near future.

2. We assemble a remeshing routine which automatically generates a triangular mesh with six

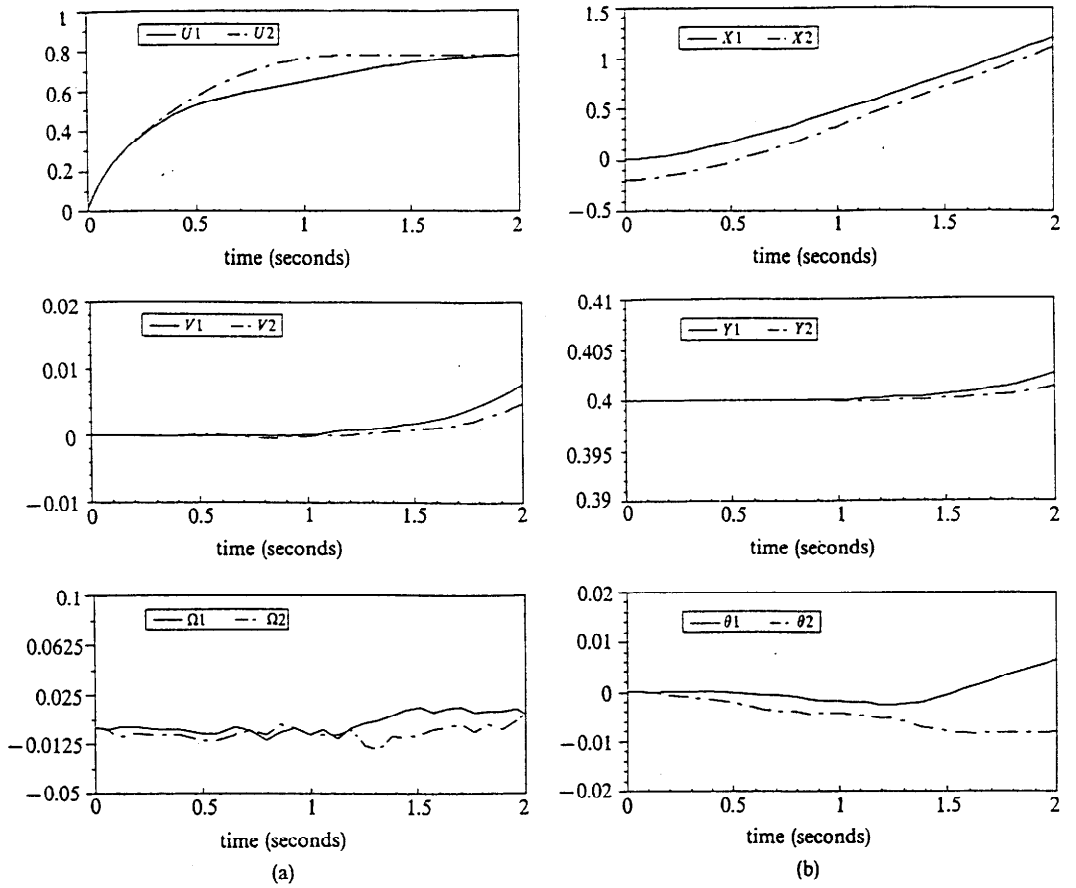


Figure 15. Two sedimenting cylinders in a channel. The diameter of the cylinders is $d = 0.1$ cm, the width of the channel is $W = 8d$. The cylinders start at the center of the channel with one cylinder two diameters behind the other.

nodes according to the positions of the cylinders. We introduce geometric stretching to reduce the number of elements used in the computation. We generate all the midnodes in the mesh and reorder the elements to minimize the frontwidth using a new mesh data structure based on segments.

3. We develop an efficient searching routine based on segments to locate the element where a given node lies. The local coordinates for this node in the element are calculated numerically. The values of the velocity components are then interpolated directly.

4. We show that the simplest fully explicit scheme to update the motion of the cylinder is unstable due to the acceleration of the mass of fluid around the cylinder. To correct this instability we generalize the expression for the drag force acting on a cylinder due to virtual mass in a viscous fluid and derive a stable explicit scheme. We propose and implement an Explicit–Implicit Scheme which explicitly updates the positions of the cylinders and implicitly updates the velocities of the cylinders. We found that the iteration for the cylinder velocities can be buried inside the nonlinear Newton iteration of the Navier–Stokes solver allowing for an increased efficiency such that the increased stability of the scheme can be achieved at a cost not much greater than in a fully explicit scheme.

5. Our numerical simulation shows that a single cylinder sedimenting in a channel at small Reynolds numbers will drift to the center of the channel. The channel center appears to be an equilibrium position at Reynolds numbers below the critical one for vortex shedding. At higher Reynolds numbers the cylinder drifts off the center of the channel. It rotates due to uneven shear and oscillates due to vortex shedding.

6. The numerical simulation for two sedimenting cylinders gave rise to the kind of drafting, kissing, and tumbling which is observed in the sedimentation (and in the fluidization, see Figure 1) of two spheres. However, unlike the spheres the cylinders do not actually touch and appear to repel each

other at a distance at large Reynolds numbers due apparently to lubrication forces or to a failure of our numerical scheme for touching cylinders at small Reynolds numbers.

Acknowledgments

Numerical results were obtained under grant from Minnesota Supercomputer Institute and AHPCRC. We wish to thank professor O. Pironnueau for making the subroutine MSHPTG available for us. Part of the work of H. Hu was done while visiting at UCL, Belgium, in September 1990. He wishes to thank Th. Avalosse and A. Couniot for sharing their work and experience in the computation.

References

Avalosse, Th., Couniot, A., and Crochet, M.J. (1990), Finite element simulation of compounding in domains of arbitrary shape, Private communication.

Batchelor, G.K. (1967), *An Introduction to Fluid Dynamics*, Cambridge University Press, Cambridge.

Couniot, A., Dheur, L., Hansen, O., and Dupret, F. (1989), A finite element method for simulating injection molding of thermoplastics, *Proc. NUMIFORM '89 Conf.*, Fort Collins, 26–30 June 1989, pp. 235–241, Balkema, Rotterdam.

Crochet, M.J., Debaut, B., Keunings, R., and Marchal, J.M. (1991), A finite element program for modelling processes, in *CAE for Polymer Processing: Application in Extrusion and Other Continuous Processes*, K.T. O'Brian, ed., Hanser Verlag, Munchen.

Cuthill, E., and McKee, J.M. (1969), Reducing the bandwidth of sparse symmetric matrices, *Proc. 24th Nat. Conf. of the Association for Computing Machinery*, p. 69

Fortes, A.F., Joseph, D.D., and Lundgren, T.S. (1987), Nonlinear mechanics of fluidization of beds of spherical particles, *J. Fluid Mech.*, 177, 467–483.

Hecht, F., and Saltel, E. (1989), Emc^2 un logiciel d'édition de maillages et de contours bidimensionels, Rapport, INRIA.

Milne-Thomson, L.M. (1960), *Theoretical Hydrodynamics*, Macmillan, New York.

Swanson, W.M. (1970), *Fluid Mechanics*, Holt, Rinehart and Winston, New York, Section 11.5.

Tezduyar, T.E., Liou, J., and Behr, M. (1990a), A new strategy for finite element computations involving moving boundaries and interfaces—the DSD/ST procedure: I. The concept and the preliminary numerical tests. Supercomputer Institute Research Report UMSI 90/169, University of Minnesota, to appear in *Comput. Methods Appl. Mech. Engrg.*

Tezduyar, T.E., Liou, J., Behr, M., and Mittal, S. (1990b), A new strategy for finite element computations involving moving boundaries and interfaces—the DSD/ST procedure: II. Computation of free-surface flows, two-liquid flows, and flows with drafting cylinders. Supercomputer Institute Research Report UMSI 90/170, University of Minnesota, to appear in *Comput. Methods Appl. Mech. Engrg.*