JOURNAL OF COMPUTATIONAL PHYSICS 17, 132–159 (1975)

Calculating Three-Dimensional Fluid Flows at All Speeds with an Eulerian-Lagrangian Computing Mesh

WILLIAM E. PRACHT*

University of California, Los Alamos Scientific Laboratory, Los Alamos, New Mexico 87544

Received August 5, 1974

A computing technique is presented for the calculation of three-dimensional, time-dependent fluid dynamics problems. The full nonlinear Navier–Stokes equations are solved with a finite-difference scheme based upon an Arbitrary Lagrangian–Eulerian (ALE) computing mesh with vertices that may move with the fluid (Lagrangian), remain fixed (Eulerian), or move in any prescribed manner. The method is applicable to three-dimensional flows at all speeds, employing an implicit formulation similar to the Implicit Continuous-Fluid Eulerian (ICE) technique. Marker particles are used that may follow exactly the motion of the fluid to aid in flow visualization, or they can represent particulate matter whose behavior is affected by inertia, drag, gravity, and molecular and turbulent diffusion. Calculational examples are shown in the form of a variety of perspective view plots.

I. INTRODUCTION

A new numerical method using a mixed Lagrangian–Eulerian finite-difference mesh has recently been developed by Hirt *et al.* [1]. The method is described in [1] for the solution of a wide variety of problems involving two-dimensional or axisymmetric fluid flow at all speeds. It has proven to be advantageous over other finite-difference methods for solving the full nonlinear, time-dependent Navier–Stokes equations primarily as a result of the Arbitrary Lagrangian–Eulerian computational mesh (the ALE feature) together with the implicit treatment of the density and velocity similar to the Implicit Continuous-Fluid Eulerian (ICE) method [2].

In this paper, an extended version of the ICED–ALE computing technique is presented for calculating the fully three-dimensional dynamics of flow at any speed contained within arbitrarily shaped boundaries. This three-dimensional version is based upon a network of six-surfaced, deformable volume elements,

* This work was performed under the joint auspices of the US Atomic Energy Commission and the Defense Nuclear Agency (DNA Subtask No. HC-061, DNA Work Unit No. 15- Calculations at Low Altitude).

each defined by eight vertices, which may (1) move with the fluid (Lagrangian mode), (2) remain fixed with the fluid moving through cells (Eulerian mode), or (3) move in a prescribed manner in relation to the moving fluid (Arbitrary Lagrangian–Eulerian).

Various schemes for calculating three-dimensional flows have previously been reported. Many of these, however, are restrictive in range of applicability, applying only to a very specific set of problems. Examples are procedures for blunt body flow [3], steady, chemically reacting nozzle flow [4], flow between two concentric cylinders [5], steady flow in ducts [6], and steady flows with recirculation [7]. There have also been reported several numerical methods that apply to a wider range of three-dimensional flow problems. Eulerian finite-difference techniques have been developed for the three-dimensional, transient dynamics of an incompressible fluid [8], as well as for a compressible fluid [9], and are most useful for problems involving large fluid distortions. Their main disadvantages are related to difficulties in accurately resolving free surfaces or material interfaces and treating curved or moving boundaries. Lagrangian methods overcome some of these disadvantages, but are not applicable to flows undergoing large fluid distortions. One such method for three-dimensional compressible flow problems has been briefly reported by Wilkins [10].

The advantages of the ICED–ALE method over previous methods becomes evident in considering the complex nature of transient three-dimensional fluid flows, especially in view of the restrictions imposed by present-day computer capacity and cost. The principal advantages are derived from the following features.

(1) The solution technique for the fully three-dimensional, nonlinear, time-dependent Navier–Stokes equations provides for a wide scope of applicability.

(2) Implicit features allow for efficient solution of flows of all speeds, from low speed (incompressible) to high speed (compressible).

(3) Arbitrary Lagrangian–Eulerian zoning permits optimal use of computational zones, allows for the calculation of flows involving curved or moving boundaries, and makes possible calculations with minimum computational diffusion without excessive grid distortions.

Sample flow calculations shown in Section VIII were run on a CDC 7600 computer with a small core memory (SCM) capacity of 58,000 words and large core memory (LCM) capacity of 400,000 words. Maximum allowable mesh size and calculation time per cell per cycle depend on a number of related factors. Calculation time per cell can be significantly reduced by storing various geometric factors used frequently. Unfortunately, this increase in storage requirements

may then too severely restrict allowable calculational mesh size. The number of marker particles, as well as the manner in which these move through the mesh, also affect maximum mesh size and calculation time. These factors clearly impose restrictions on three-dimensional flow calculations; however, the examples demonstrate that we nevertheless can perform numerous meaningful and interesting calculations even with limited resolution. Running the double burst calculation shown in Section VIII, for example, with our unoptimized developmental code, required approximately 100,000 LCM words, 40,000 SCM words, and 3.5–4.0 msec per cell per cycle calculation time.

II. Differential Equations

The differential equations to be solved can be expressed rather compactly with the aid of Cartesian tensor notation as:

$$(\partial\rho/\partial t) + (\partial\rho u_j/\partial x_j) = 0, \tag{1}$$

$$(\partial\rho u_i/\partial t) + (\partial/\partial x_j)(\rho u_i u_j - p_{ij}) = g_i\rho, \tag{2}$$

$$(\partial\rho E/\partial t) + (\partial/\partial x_j)(\rho u_j E - p_{ij}u_i - \mu B(\partial I/\partial x_j)) = \rho u_j g_j\,. \tag{3}$$

The total specific energy E is defined by $E = \frac{1}{2}u_i^2 + I$. The stress tensor p_{ij} includes both the scalar pressure and the viscous stress and is written

$$p_{ij} = -p\delta_{ij} + \tfrac{1}{2}\lambda e_{kk}\delta_{ij} + \mu e_{ij}\,, \tag{4}$$

where p, the scalar pressure, is a prescribed function of the density ρ and of the specific internal energy I; δ_{ij} is the Kronecker delta; and

$$e_{ij} = (\partial u_i/\partial x_j) + (\partial u_j/\partial x_i).$$

The viscosity coefficients, λ and μ, have, for simplicity, been chosen as constants and yield terms that, for compressible flows, demonstrate similarity to the "artificial" viscosity terms ordinarily used in numerical fluid dynamics calculations. The heat conduction term, often written in terms of T, the temperature, has been altered by introducing the coefficient B. In Eqs. (2) and (3), g_i represents a body acceleration term, which in most cases is gravity.

Equations (1)–(3) express conservation of mass, momentum, and energy, and are of the basic conservation law form, which can be written in differential form as

$$(\partial A/\partial t) + (\partial C_j/\partial x_j) = F, \tag{5}$$

where A represents a quantity per unit volume, C_j its flux in the j direction, and F denotes the density of an external volume source. Equation (5) is fundamentally an expression of the rate of change of quantity A, constrained in a selected volume element of fluid bound by surface s, as a consequence of both external influences and fluxes at the volume element boundaries. This can be more easily seen by integrating Eq. (5) over a volume element fixed in space.

$$\frac{\partial}{\partial t}\int_V A\,dV + \int_s C_j n_j\,ds = \int_V F\,dV. \tag{6}$$

As we see in the next section, this integral form is easily related to our computational grid, where the volume integrals are taken over a finite volume V, specified by our six-surfaced computational cells. The surface integral is over the nonplanar surfaces of these finite volume elements, with n_j denoting the outward normal to the surface. A typical computational cell is shown in Fig. 1. More detail is given in the next two sections on the exact form of the finite-difference approximations as obtained from the integral form of Eqs. (1)–(3).

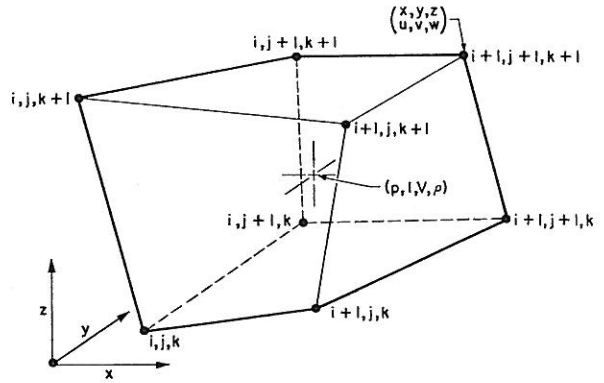

FIG. 1. Typical cell layout with i, j, k indices and cell variable placement.

III. Method of Solution

The finite-difference approximations of Eqs. (1)–(3) are solved in a sequence of steps or calculational cycles, each advancing the solution a finite interval δt in time. The results of each cycle are calculated from data remaining from the previous cycle, or supplied as initial conditions, and are constrained by prescribed boundary conditions. The calculational sequence for each cycle is as follows.

(1) New velocities and energies are calculated from pressures and densities left over from the previous cycle. Although mesh vertices are not yet

actually moved, this stage of the calculation is performed on the basis of a pure Lagrangian approach, i.e., no fluid is fluxed across calculational cell boundaries.

(2) An iteration involving velocities, pressures, and densities is performed to obtain advanced-time pressure forces. This procedure allows sound signals to travel throughout the entire mesh in a time δt, rather than being restricted to one cell width by the usual Courant-like numerical stability condition.

(3) Calculational cell vertex positions can be moved, and the appropriate convective fluxes calculated in any of three options:

a. with the fluid velocity (pure Lagrangian);

b. back to their respective beginning-of-cycle positions (pure Eulerian); or

c. any other prescribed manner (mixed).

(4) Final end-of-cycle values of energies, volumes, etc., are computed for use in the cycle that follows and a variety of output plots and prints may at this time be generated.

Since the calculational mesh is made up of six-surfaced volume elements whose eight vertices may be allowed to move with the fluid, the starting point for the derivation of the finite-difference equations is the integral form (Eq. 6) of Eqs. (1)–(3).

$$\int_V \frac{\partial \rho}{\partial t}\, dV + \int_s \rho(u_j - U_j)\, n_j\, ds = 0, \tag{7}$$

$$\int_V \frac{\rho u_i}{\partial t}\, dV' + \int_s [\rho u_i(u_j - U_j) - p_{ij}]\, n_j\, ds' = \int_V g_i \rho\, dV', \tag{8}$$

$$\int_V \frac{\partial \rho E}{\partial t}\, dV + \int_s \left[\rho E(u_j - U_j) - p_{ij}u_i - \mu B \frac{\partial I}{\partial x_j}\right] n_j\, ds = \int_V \rho u_j g_j\, dV. \tag{9}$$

Here, U_j is the velocity of the bounding surface s. The integration control volume specified in Eqs. (7) and (9) is the computational cell itself. Figure 1 shows one such cell defined by eight vertices labeled by integer triplets (i, j, k) counting in the x, y, z direction, respectively. Cell centers are denoted by $(i + \frac{1}{2}, j + \frac{1}{2}, k + \frac{1}{2})$. Also shown are the locations in the cells at which the variables are assigned.

Because of the staggered locations of the variables, Eq. (8) requires a momentum integration control volume (denoted by prime quantities ds' and dV') that is shifted one-half cell height down, one-half cell width to the left, and one-half cell depth forward from the volume used in the other two equations. In this way, the integration volume around a cell vertex is defined by the geometric centers of volumes, faces, and edges of the eight cells surrounding the vertex as shown in Figs. 3 and 5.

Basic to a conservative set of finite-difference equations for an irregular, deformable calculational mesh is an accurate and unique means of defining, in terms of computing cells, surface area elements, ds and ds', as well as the volume elements, dV and dV', appearing in integrals (7)–(9). The four vertices of a face of a computing cell define a ruled surface rather than a planar surface. Its area can be calculated by subdividing it into four triangular planes with a common vertex at the geometric center, as illustrated in Fig. 2. With surface area elements ds_1,

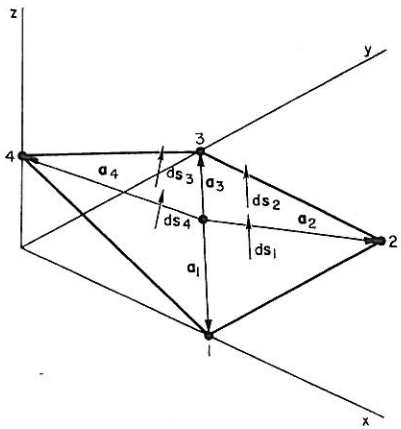

FIG. 2. Subdividing a computational cell face into four triangular planes.

ds_2, ds_3, and ds_4, we can associate the vectors $d\mathbf{s}_1$, $d\mathbf{s}_2$, $d\mathbf{s}_3$, and $d\mathbf{s}_4$, which have the direction of the normal chosen on their respective elements and have lengths equal to their areas. By introducing the vectors $\mathbf{a}_1$, $\mathbf{a}_2$, $\mathbf{a}_3$, and $\mathbf{a}_4$ along the four concurrent triangle sides, we can evaluate these surface area element vectors as

$$\pm d\mathbf{s}_l = \tfrac{1}{2}(\mathbf{a}_l \times \mathbf{a}_{l+1}), \tag{10}$$

where $l = 1, 2, 3$, and 4 (when $l = 4$, $l + 1 = 1$), and the sign is chosen to represent the outward normal direction. Surface integrals in Eqs. (7) and (9) can be approximated by expressions of the form,

$$\int_s C_j n_j\, ds \approx \sum_{m=1}^{6} \sum_{l=1}^{4} (\mathbf{C}_l \cdot d\mathbf{s}_l)_m\,, \tag{11}$$

where l denotes cell vertices and m denotes cell faces. We can illustrate this operator by evaluating, for one of the six calculational cell faces, relation (11) for $\mathbf{C} = \mathbf{u}$.

$$\sum_{l=1}^{4} (\mathbf{u}_l \cdot d\mathbf{s}_l) = \sum_{l=1}^{4} (\bar{u}_l\, ds\, x_l + \bar{v}_l\, ds\, y_l + \bar{w}_l\, ds\, z_l)$$
$$= \bar{u}_1\, ds\, x_1 + \bar{v}_1\, ds\, y_1 + \bar{w}_1\, ds\, z_1$$
$$+ \bar{u}_2\, ds\, x_2 + \bar{v}_2\, ds\, y_2 + \bar{w}_2\, ds\, z_2$$
$$+ \bar{u}_3\, ds\, x_3 + \bar{v}_3\, ds\, y_3 + \bar{w}_3\, ds\, z_3$$
$$+ \bar{u}_4\, ds\, x_4 + \bar{v}_4\, ds\, y_4 + \bar{w}_4\, ds\, z_4\,, \tag{12}$$

where, for example,

$$ds\, x_1 = \tfrac{1}{2}[(y_1 - y_c)(z_2 - z_c) - (z_1 - z_c)(y_2 - y_c)],$$
$$ds\, y_1 = \tfrac{1}{2}[(x_2 - x_c)(z_1 - z_c) - (x_1 - x_c)(z_2 - z_c)], \tag{13}$$
$$ds\, z_1 = \tfrac{1}{2}[(x_1 - x_c)(y_2 - y_c) - (y_1 - y_c)(x_2 - x_c)],$$

are the x, y, and z components, respectively, of surface element

$$d\mathbf{s}_1 = \tfrac{1}{2}(\mathbf{a}_1 \times \mathbf{a}_2), \tag{14}$$

and denote the areas of the projection of the triangle on the yz plane, xz plane, and xy plane.

In these expressions x_1, y_1, z_1 denote the values of x, y, and z at vertex number 1; and $x_c = \frac{1}{4}(x_1 + x_2 + x_3 + x_4)$, $y_c = \frac{1}{4}(y_1 + y_2 + y_3 + y_4)$, and $z_c = \frac{1}{4}(z_1 + z_2 + z_3 + z_4)$ denote the values at the geometric center of the face. The bar quantities represent velocity components associated with their respective surface elements. For example, the x, y, and z components of velocity associated with surface element ds_1 are, respectively,

$$\bar{u}_1 = \tfrac{1}{3}(u_1 + u_2 + u_c),$$
$$\bar{v}_1 = \tfrac{1}{3}(v_1 + v_2 + v_c), \tag{15}$$
$$\bar{w}_1 = \tfrac{1}{3}(w_1 + w_2 + w_c),$$

where

$$u_c = \tfrac{1}{4}(u_1 + u_2 + u_3 + u_4),$$
$$v_c = \tfrac{1}{4}(v_1 + v_2 + v_3 + v_4), \tag{16}$$

and

$$w_c = \tfrac{1}{4}(w_1 + w_2 + w_3 + w_4).$$

A similar procedure is followed in defining surface areas bounding the offset momentum control volumes. In this case, however, the area of each face is obtained by dividing it into eight triangular planes, with vertices defined at geometric centers of computational cell volumes, faces, and edges, as illustrated in Fig. 3. For ease of illustration we have shown here only one of the six faces of the momentum control volume enclosing the dotted vertex. Let ds' represent a planar triangular surface area element. With ds' can be associated the area element vector $d\mathbf{s}'$, which is evaluated in terms of appropriate vector products. It is convenient to combine pairs of triangular planes lying in the same computational cell and work with the resulting quadrilateral volume elements as illustrated by the shaded areas in Fig. 3. For each computational cell there will be three such quadrilateral elements associated with each vertex designated as shown in Fig. 5.

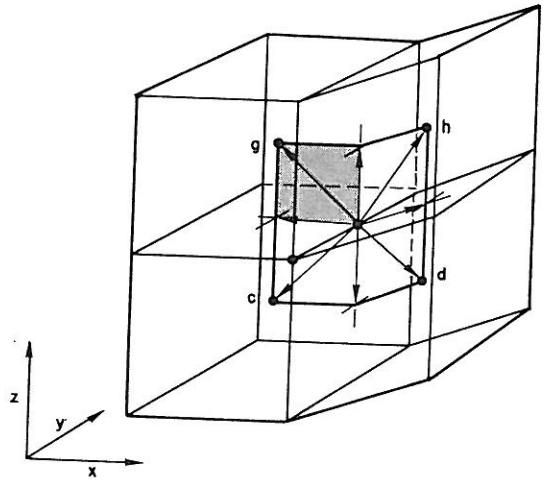

FIG. 3. Subdividing a momentum control volume face into triangular planes.

Volumes of computational cells are calculated by uniquely subdividing each into a set of tetrahedrons. The triangular faces of these tetrahedrons are defined by the geometric centers of the calculational cell itself, together with the triangular-shaped differential surface elements previously defined. As an illustration, Fig. 4 shows one of the four tetrahedrons whose bases are the bottom face of a calculational cell. The vertices are the geometric center of the cell (c_m); the geometric center of ruled surface 1–2–3–4 (c_B); and cell vertices (1) and (2). The volume of the tetrahedron shown in this figure is

$$V_{\text{TET}} = \tfrac{1}{6} \mid d\mathbf{s}_1 \cdot \mathbf{c} \mid, \tag{17}$$

where $\mathbf{c} = \mathbf{c}_B\mathbf{c}_m$.

The volume of the calculational cell can then be expressed as

$$V = \tfrac{1}{6} \sum_{m=1}^{6} \sum_{l=1}^{4} |(d\mathbf{s}_l \cdot \mathbf{c})_m|, \tag{18}$$

where l denotes cell vertices (mod 4) and m denotes cell faces.

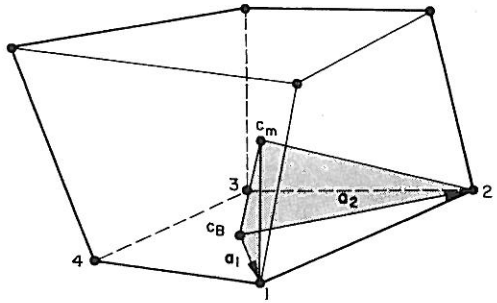

FIG. 4. One of the 24 tetrahedrons contained within a computational cell.

IV. FINITE-DIFFERENCE EQUATIONS

Because of the long and tedious nature of the full set of finite-difference equations used in the developmental program of this extended version of the ICED–ALE method, only selected portions of these equations are given here. These, together with a basic outline of the solution procedure should adequately serve to illustrate the technique. For a more complete description of the equations, flow charts, and available options, the reader is referred to [11].

The specific set of finite-difference equations chosen can take on a variety of forms, depending on such things as the nature of the specific application, the computer storage capacity, and the central processor speed. Those that follow are essentially direct extensions of the original ICED–ALE method described in [1].

Depending on available computer memory in relation to computational mesh size, increased efficiency may be obtained by calculating and storing various quantities at the beginning of each calculational cycle. These include: cell volumes V; cell face surface elements, $ds\,x$, $ds\,y$, $ds\,z$; cell total energy E; and masses assigned to cell vertices M_v. Sample finite-difference equations are given in the same sequence as the solution procedure steps described in the preceding section.

A. *Explicit Lagrangian Phase*

As noted earlier, this step calculates Lagrangian velocities resulting from the previous cycle's pressure gradients, body forces, and viscous shear forces. The three components of velocity for a particular vertex are adjusted in this phase of the calculation according to the pressure and viscous forces acting on the six surfaces of the momentum control volume surrounding that vertex. This explicit advancement of velocities is divided into a series of steps. For example, the velocity

components associated with the dotted vertex in Fig. 5 and denoted by the subscript 0, are first advanced by the body force or gravity term,

$$
\begin{aligned}
{}_0\tilde{u} &= {}^n_0u + \delta t\, g_x\,,\\
{}_0\tilde{v} &= {}^n_0v + \delta t\, g_y\,,\\
{}_0\tilde{w} &= {}^n_0w + \delta t\, g_z\,.
\end{aligned}
\tag{19}
$$

Here the tildes over the quantities on the left signify temporary new values, and the superscript n denotes last cycle values for the velocity component. Next, the

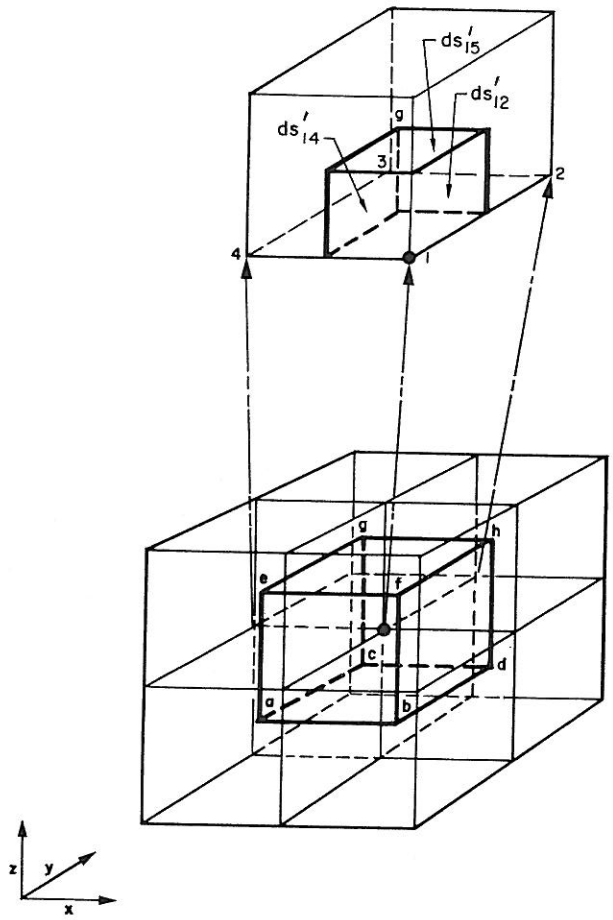

FIG. 5. The heavy lines enclose the momentum integration control volume surrounding the dotted vertex.

velocity components are adjusted in response to the appropriate component of surface force across that portion of momentum control volume surface contained within each of the eight cells surrounding the vertex. In this way each vertex of a particular cell is in various stages of its eight-step incremental calculation process. For example, the dotted vertex in Fig. 5 undergoes a sequence of adjustments starting first in cell a, then cell b, and so on. To further illustrate, consider the advancement of velocity components at vertex 1, in response to pressure and viscous effects in cell g. The expressions for this adjustment are:

$$\begin{aligned}
\tilde{u}_1 &= \tilde{u}_1 + (\delta t/M_1)[\pi_{xx}\, DS'\, x_1 + \pi_{xy}\, DS'\, y_1 + \pi_{xz}\, DS'\, z_1], \\
\tilde{v}_1 &= \tilde{v}_1 + (\delta t/M_1)[\pi_{xy}\, DS'\, x_1 + \pi_{yy}\, DS'\, y_1 + \pi_{yz}\, DS'\, z_1], \\
\tilde{w}_1 &= \tilde{w}_1 + (\delta t/M_1)[\pi_{xz}\, DS'\, x_1 + \pi_{yz}\, DS'\, y_1 + \pi_{zz}\, DS'\, z_1],
\end{aligned} \tag{20}$$

where M_1 denotes the mass associated with vertex 1, and $DS'\, x_1$, $DS'\, y_1$, $DS'\, z_1$ are the areas of the projections of that portion of the momentum control volume around vertex 1 that is contained within cell g, on the yz plane, xz plane, and the xy plane.

The π terms here represent the six stress tensor terms previously expressed in Eq. (4).

$$\begin{aligned}
\pi_{xx} &= -p + \lambda D + \mu(u_x), \\
\pi_{xy} &= \mu(u_y + v_x), \\
\pi_{xz} &= \mu(u_z + w_x), \\
\pi_{yy} &= -p + \lambda D + \mu(v_y), \\
\pi_{yz} &= \mu(v_z + w_y), \\
\pi_{zz} &= -p + \lambda D + \mu(w_z).
\end{aligned} \tag{21}$$

Here p is the scalar fluid pressure and $D = u_x + v_y + w_z$. The x, y, and z subscripts denote approximations to the various spatial velocity derivatives. Examples are

$$\begin{aligned}
\frac{\partial u}{\partial x} &\approx u_x = \frac{1}{V}\sum_{m=1}^{6}\sum_{l=1}^{4}(\bar{u}_l\, ds\, x_l)_m, \\
\frac{\partial v}{\partial y} &\approx v_y = \frac{1}{V}\sum_{m=1}^{6}\sum_{l=1}^{4}(\bar{v}_l\, ds\, y_l)_m, \\
\frac{\partial w}{\partial z} &\approx w_z = \frac{1}{V}\sum_{m=1}^{6}\sum_{l=1}^{4}(\bar{w}_l\, ds\, z_l)_m, \\
\frac{\partial u}{\partial y} &\approx u_y = \frac{1}{V}\sum_{m=1}^{6}\sum_{l=1}^{4}(\bar{u}_l\, ds\, y_l)_m.
\end{aligned} \tag{22}$$

The exact choice of solution form for the energy equation is somewhat arbitrary and may, to some extent, depend upon the specific application. Regardless of the specific form chosen, however, the principle of conservation of total energy provides appropriate guidance for the finite-difference equation derivation. Reference [12] contains a detailed description of a procedure for solving the internal energy equation directly while exactly conserving total energy. In summary, the idea is the following. During any calculational cycle, the change in energy of the fluid must balance the work done on the fluid by all the stresses. This is most conveniently thought of in terms of work done by a particular cell on the eight vertices of that cell. This must produce a corresponding change in the internal energy (a cell-centered quantity). Work done is easily calculated as the product of the force exerted and the resultant displacement. Total energy conservation is achieved by meeting two conditions. First, the same force terms must be used in calculating the change in internal energy as was used in the calculation producing the respective changes of velocities (i.e., the changes in kinetic energy). Second, the displacement that the fluid undergoes as a result of the acting forces must be time-centered.

We have chosen to advance the specific internal energy in parallel with velocity advancement. This simplifies the task of using the same force type terms in both calculations. Again with the tilde sign signifying a temporary quantity, the expression to explicitly advance the internal energy is

$$\begin{aligned}
\tilde{I} = {}^nI + (\delta t\, V/M)[&p(\bar{u}_x + \bar{v}_y + \bar{w}_z) + (\pi_{xx} - p)\,\bar{u}_x + (\pi_{yy} - p)\,\bar{v}_y \\
&+ (\pi_{zz} - p)\,\bar{w}_z + \pi_{xy}(\bar{u}_y + \bar{v}_x) + \pi_{xz}(\bar{u}_z + \bar{w}_x) + \pi_{yz}(\bar{v}_z + \bar{w}_y)].
\end{aligned} \tag{23}$$

Here the bar signifies a time-centered quantity such as $\bar{u}_x = \frac{1}{2}(u_x + \tilde{u}_x)$, and M denotes the cell-centered mass.

B. *Implicit Lagrangian Phase*

The equations are solved in this implicit Lagrangian step by an extension of the method described in [1]. The basic task of this implicit Lagrangian phase is to permit the propagation of sound signals throughout the computational mesh each time step or calculational cycle when the sound speed is large compared to the ratio of cell size to time step. In summary the procedure is as follows.

(1) The velocities, pressures, and densities from the explicit Lagrangian phase serve as initial guesses for the iteration.

(2) A change in pressure δp is calculated and added to the field on the basis of one step in a Newton-type iteration scheme, as $\delta p = -[Q(p)/(\partial Q/\partial p)]$, where $Q(p)$ relates changes in densities and velocities to pressure.

(3) Velocities are changed to be consistent with the advanced pressure field.

velocity components are adjusted in response to the appropriate component of surface force across that portion of momentum control volume surface contained within each of the eight cells surrounding the vertex. In this way each vertex of a particular cell is in various stages of its eight-step incremental calculation process. For example, the dotted vertex in Fig. 5 undergoes a sequence of adjustments starting first in cell a, then cell b, and so on. To further illustrate, consider the advancement of velocity components at vertex 1, in response to pressure and viscous effects in cell g. The expressions for this adjustment are:

$$\begin{aligned}
\tilde{u}_1 &= \tilde{u}_1 + (\delta t/M_1)[\pi_{xx}\, DS'\, x_1 + \pi_{xy}\, DS'\, y_1 + \pi_{xz}\, DS'\, z_1], \\
\tilde{v}_1 &= \tilde{v}_1 + (\delta t/M_1)[\pi_{xy}\, DS'\, x_1 + \pi_{yy}\, DS'\, y_1 + \pi_{yz}\, DS'\, z_1], \\
\tilde{w}_1 &= \tilde{w}_1 + (\delta t/M_1)[\pi_{xz}\, DS'\, x_1 + \pi_{yz}\, DS'\, y_1 + \pi_{zz}\, DS'\, z_1],
\end{aligned} \tag{20}$$

where M_1 denotes the mass associated with vertex 1, and $DS'\, x_1$, $DS'\, y_1$, $DS'\, z_1$ are the areas of the projections of that portion of the momentum control volume around vertex 1 that is contained within cell g, on the yz plane, xz plane, and the xy plane.

The π terms here represent the six stress tensor terms previously expressed in Eq. (4).

$$\begin{aligned}
\pi_{xx} &= -p + \lambda D + \mu(u_x), \\
\pi_{xy} &= \mu(u_y + v_x), \\
\pi_{xz} &= \mu(u_z + w_x), \\
\pi_{yy} &= -p + \lambda D + \mu(v_y), \\
\pi_{yz} &= \mu(v_z + w_y), \\
\pi_{zz} &= -p + \lambda D + \mu(w_z).
\end{aligned} \tag{21}$$

Here p is the scalar fluid pressure and $D = u_x + v_y + w_z$. The x, y, and z subscripts denote approximations to the various spatial velocity derivatives. Examples are

$$\begin{aligned}
\frac{\partial u}{\partial x} &\approx u_x = \frac{1}{V}\sum_{m=1}^{6}\sum_{l=1}^{4}(\bar{u}_l\, ds\, x_l)_m, \\
\frac{\partial v}{\partial y} &\approx v_y = \frac{1}{V}\sum_{m=1}^{6}\sum_{l=1}^{4}(\bar{v}_l\, ds\, y_l)_m, \\
\frac{\partial w}{\partial z} &\approx w_z = \frac{1}{V}\sum_{m=1}^{6}\sum_{l=1}^{4}(\bar{w}_l\, ds\, z_l)_m, \\
\frac{\partial u}{\partial y} &\approx u_y = \frac{1}{V}\sum_{m=1}^{6}\sum_{l=1}^{4}(\bar{u}_l\, ds\, y_l)_m.
\end{aligned} \tag{22}$$

The exact choice of solution form for the energy equation is somewhat arbitrary and may, to some extent, depend upon the specific application. Regardless of the specific form chosen, however, the principle of conservation of total energy provides appropriate guidance for the finite-difference equation derivation. Reference [12] contains a detailed description of a procedure for solving the internal energy equation directly while exactly conserving total energy. In summary, the idea is the following. During any calculational cycle, the change in energy of the fluid must balance the work done on the fluid by all the stresses. This is most conveniently thought of in terms of work done by a particular cell on the eight vertices of that cell. This must produce a corresponding change in the internal energy (a cell-centered quantity). Work done is easily calculated as the product of the force exerted and the resultant displacement. Total energy conservation is achieved by meeting two conditions. First, the same force terms must be used in calculating the change in internal energy as was used in the calculation producing the respective changes of velocities (i.e., the changes in kinetic energy). Second, the displacement that the fluid undergoes as a result of the acting forces must be time-centered.

We have chosen to advance the specific internal energy in parallel with velocity advancement. This simplifies the task of using the same force type terms in both calculations. Again with the tilde sign signifying a temporary quantity, the expression to explicitly advance the internal energy is

$$\begin{aligned}
\tilde{I} = {}^{n}I + (\delta t\, V/M)[&p(\bar{u}_x + \bar{v}_y + \bar{w}_z) + (\pi_{xx} - p)\,\bar{u}_x + (\pi_{yy} - p)\,\bar{v}_y \\
&+ (\pi_{zz} - p)\,\bar{w}_z + \pi_{xy}(\bar{u}_y + \bar{v}_x) + \pi_{xz}(\bar{u}_z + \bar{w}_x) + \pi_{yz}(\bar{v}_z + \bar{w}_y)].
\end{aligned} \tag{23}$$

Here the bar signifies a time-centered quantity such as $\bar{u}_x = \frac{1}{2}(u_x + \tilde{u}_x)$, and M denotes the cell-centered mass.

B. *Implicit Lagrangian Phase*

The equations are solved in this implicit Lagrangian step by an extension of the method described in [1]. The basic task of this implicit Lagrangian phase is to permit the propagation of sound signals throughout the computational mesh each time step or calculational cycle when the sound speed is large compared to the ratio of cell size to time step. In summary the procedure is as follows.

(1) The velocities, pressures, and densities from the explicit Lagrangian phase serve as initial guesses for the iteration.

(2) A change in pressure δp is calculated and added to the field on the basis of one step in a Newton-type iteration scheme, as $\delta p = -[Q(p)/(\partial Q/\partial p)]$, where $Q(p)$ relates changes in densities and velocities to pressure.

(3) Velocities are changed to be consistent with the advanced pressure field.

(4) The resulting spatial variations in density are accounted for and convergence tests are performed.

(5) The iteration is considered sufficiently converged when some criteria such as, $| \delta p | < (\epsilon) | p_{MAX} |$ is satisfied, where typically $\epsilon \approx 10^{-4}$.

One sweep through the computational mesh is made for each iteration. During this sweep, steps (2)–(4) are performed for each calculation cell. The expression first solved in step (2) is

$$\delta p = -\omega Q(p, \rho, \mathbf{u})/Q_p , \tag{24}$$

where

$$Q(p, \rho, \mathbf{u}) = (1/\delta t)({}^L\rho - {}^n\rho) + {}^L\rho {}^L D, \tag{25}$$

and Q_p is some appropriate approximation to $\partial Q/\partial p$, which serves as a relaxation factor for the iteration procedure. This will be defined later. The superscript L denotes a current Lagrangian iteration level value. The current guess for the cell pressure is updated according to

$${}^L p^{\text{new}} = {}^L p^{\text{old}} + \delta p. \tag{26}$$

Velocity adjustments of step (3) are made for each of the eight vertices, each sweep through the mesh. The velocity changes for vertex 1, for example, are

$$\begin{aligned} {}^L u_1^{\text{new}} &= {}^L u_1^{\text{old}} + (\delta t/M_1)(\delta p\, DS' x_1), \\ {}^L v_1^{\text{new}} &= {}^L v_1^{\text{old}} + (\delta t/M_1)(\delta p\, DS' y_1), \\ {}^L w_1^{\text{new}} &= {}^L w_1^{\text{old}} + (\delta t/M_1)(\delta p\, DS' z_1). \end{aligned} \tag{27}$$

Finally, variations in density are accounted for by the expression

$${}^L \rho^{\text{new}} = {}^L \rho^{\text{old}} + R({}^L p, I), \tag{28}$$

where R represents some appropriate equation of state function of pressure and energy.

The relaxation factor Q_p is basically a measure of the rate of change of Q with changes in p and, as such, serves to automatically control the iteration procedure. An efficient but stable progression toward the solution of the implicit relations (24)–(28) requires a reasonably good approximation of this term. A direct differentiation of Q would, especially for distorted meshes, yield an expression that would be very complicated and difficult to evaluate. A better way is to introduce into the first pass of the iteration a small pressure change, $\delta p_0 \sim (\epsilon) | p_{MAX} |$ for all cells and allow the iteration to proceed as usual. New iteration-level velocities

and densities are then calculated, resulting in a new iteration-level value for the function $Q(p, \rho, \mathbf{u})$. The relaxation factor for that cell for all remaining iterations is then given by,

$$Q_p = (Q^{\text{new}} - Q^{\text{old}})/\delta p_0 . \tag{29}$$

Following the convergence of the iteration a final adjustment of internal energy is required to again insure the exact balance between the change in kinetic energy and the work done, in this case by the scalar pressure effects.

$${}^L I = {}^L \tilde{I} - (\delta t\, V/M)[({}^L p - {}^n p)(\bar{u}_x + \bar{v}_y + \bar{w}_z)], \tag{30}$$

where the barred quantities are the "time-centered" velocities gradients of the form,

$$\bar{u}_x = \tfrac{1}{2}(\tilde{u}_x + {}^L u_x).$$

In addition, in preparation for the next step of the calculation we must calculate and store for each cell a quantity representing the total energy.

$$\begin{aligned} \tilde{E} = {}^L I + (1/16M)[&M_1(u_1^2 + v_1^2 + w_1^2) + M_2(u_2^2 + v_2^2 + w_2^2) \\ &+ M_3(u_3^2 + v_3^2 + w_3^2) + M_4(u_4^2 + v_4^2 + w_4^2) + M_5(u_5^2 + v_5^2 + w_5^2) \\ &+ M_6(u_6^2 + v_6^2 + w_6^2) + M_7(u_7^2 + v_7^2 + w_7^2) + M_8(u_8^2 + v_8^2 + w_8^2)], \end{aligned} \tag{31}$$

where the subscripts denote the eight different vertices for each calculational cell. All velocity terms are taken at the advanced-time Lagrangian level L.

C. *Rezone-Convective-Flux Phase*

If at this point in the calculation, all the vertices were moved with advanced-time Lagrangian velocities, allowing the calculational grid to exactly follow the fluid motion, the result would be a pure Lagrangian calculation, and no convective flux calculations would be required. As is well known for highly distorted flows, however, allowing the calculational mesh to exactly follow the flow quickly results in a disaster. Cells can turn inside out, producing negative volumes; they get stretched and distorted to the point where the finite-difference approximations no longer resemble the original equations. To prevent this, the ICED–ALE method employs a Rezone-Convective-Flux Phase designed to move the mesh relative to the fluid so as to maintain a reasonably undistorted mesh. This phase is divided into two parts: the calculation of the grid motion, and the fluxing of mass, momentum, and energy through the faces of the calculational cells in response to the relative fluid motion.

There are numerous factors that affect the ideal mesh-moving prescription. Shearing flow, for example, would require a different prescription from that of

compressional flow. The importance of minimizing artificial numerical diffusion of the flow variables is another factor affecting the choice of zone prescriptions. Leaving the mesh fixed (pure Eulerian), for example, may introduce excessive smearing and thereby be invalid for calculations of many classes of flow problems. In some cases, local rectilinearity obtained by a prescription such as the one described in [12] provides the guidelines for the best mesh. In others, a simple averaging scheme is best.

Since computational diffusion is proportional to the relative motion of the mesh with respect to the fluid, any reduction in this motion will result in less computational diffusion than for a conventional Eulerian calculation. Also in many cases additional advantages of an intermediate Eulerian–Lagrangian mesh are derived from the capability for allowing the zones to move in such an intermediate fashion as to produce finer zoning in areas of steeper gradients, as well as to produce zone orientation in a direction normal to the principal gradients. A further discussion of this balance in mesh-moving prescriptions between a reduction of relative grid motion and a reduction of grid distortion is given in [12].

In the rezone-type calculation shown in Section VIII, a simple centroid averaging scheme is used. In this the velocity of each vertex is chosen to reduce the displacement of the vertex from the average of the positions of the six closest vertex neighbors. Denoting this grid velocity by the superscript G, the expression to be solved for a typical nonboundary vertex (i, j, k) is given by

$$ {}^{G}\mathbf{u}_{i,j,k} = {}^{L}\mathbf{u}_{i,j,k} + (\psi/\delta t)(\bar{\mathbf{r}} - \mathbf{r}_{i,j,k}), \tag{32} $$

where

$$ \bar{\mathbf{r}} = \tfrac{1}{6}(\mathbf{r}_{i-1,j,k} + \mathbf{r}_{i+1,j,k} + \mathbf{r}_{i,j-1,k} + \mathbf{r}_{i,j+1,k} + \mathbf{r}_{i,j,k-1} + \mathbf{r}_{i,j,k+1}), $$

δt is the time step, ψ is the proportion of grid averaging to be performed each calculational cycle, and $\mathbf{r}$ denotes the vertex positions. The new vertex positions are determined from the expression,

$$ \mathbf{r}^{n+1}_{i,j,k} = \mathbf{r}^{n}_{i,j,k} + \delta t({}^{G}\mathbf{u}_{i,j,k}). \tag{33} $$

Following the determination of new vertex positions, the convective fluxes are calculated. Provided that $\psi \neq 0$, this results in a certain volume of fluid being exchanged between calculational cells. Together with the exchange of fluid is also an exchange of mass, momentum, and energy, all quantities that are to be conserved.

To illustrate, consider two calculational cells, one on top of the other as shown in Fig. 6. If the relative grid motion is upward or downward, some incremental

volume of fluid will be exchanged between these two cells, through common face 1–2–3–4, denoted by the subscript m. It is given by

$$ \delta V_m = \delta t \sum_{l=1}^{4} [{}^{R}\bar{u}_l \, ds \, x_l + {}^{R}\bar{v}_l \, ds \, y_l + {}^{R}\bar{w}_l \, ds \, z_l]_m , \tag{34} $$

where the superscript R denotes the grid velocity relative to the fluid velocity, determined by

$$ {}^{R}\mathbf{u} = {}^{L}\mathbf{u} - {}^{G}\mathbf{u}, $$

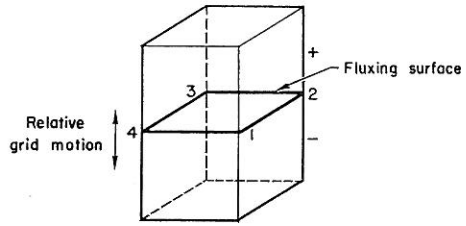

FIG. 6. Fluid volume exchange between cells through face 1–2–3–4.

and the bar denotes average values determined as Eqs. (15). The accompanying exchanges of mass M and total energy E through this face are given by

$$ \delta M_m = (\delta V_m/2)[(1 - \alpha_m) \, {}^{L}\rho_+ + (1 + \alpha_m) \, {}^{L}\rho_-], \tag{35} $$

and

$$ \delta(ME_m) = (\delta V_m/2)[(1 - \alpha_m)({}^{n}\rho\tilde{E})_+ + (1 + \alpha_m)({}^{n}\rho\tilde{E})_-]. \tag{36} $$

Here the $+$ and $-$ subscripts refer to the cell centers above (positive z direction) and below (negative z direction), respectively. The donor cell weighting factor α is given by

$$ \alpha_m = \alpha_0(\delta V_m/|\,\delta V_m\,|) + (2\beta_0 \, \delta V_m/(V_+ + V_-)), \tag{37} $$

where α_0 and β_0 are constants determining the type of fluxing desired. The choices are given in Table I.

TABLE I

Differencing Choice	α_0	β_0
Centered	0	0
Donor Cell	1	0
Interpolated Donor Cell	0	1
Mixed	otherwise specified	

Also accompanying a change in vertex position is an exchange of momentum between neighboring vertices. This convective flux of momentum is accounted for by computing the flow of momentum through the faces of the offset momentum integration control volumes described earlier. Just as in calculating the changes in cell-centered quantities, the fluid volume swept through the faces of these offset momentum control volumes must be calculated.

Within the calculation itself, it is convenient to sweep through the mesh cell by cell, adjusting the velocity of each vertex of a particular cell in response to the convective flux of momentum through the faces of the momentum control volume contained within that particular cell. In this way all interior vertices reach their final value in a sequence of eight steps, with each step accounting for the momentum exchanged between pairs of vertices.

For example, momentum convecting through the portions of the surface shown in Fig. 7 exchanges momentum between vertices 1 and 4, 2 and 3, 5 and 8, and 6

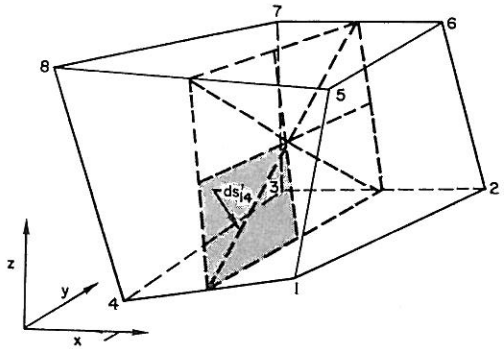

FIG. 7. Momentum exchange between vertices 1 and 4, 2 and 3, 5 and 8, and 6 and 7 through momentum control volume face denoted by the heavy dotted lines.

and 7. The u-momentum exchange between vertices 1 and 4 through the shaded portion of the surface can be written:

$$uF_{14} = {}^{L}\rho(\delta V'_{14}/2)[{}^{L}u_4(1 + \alpha_{14}) + {}^{L}u_1(1 - \alpha_{14})], \tag{38}$$

where

$$\delta V'_{14} = \delta t({}^{R}\bar{u}_{14}\, ds'x_{14} + {}^{R}\bar{v}_{14}\, ds'y_{14} + {}^{R}\bar{w}_{14}\, ds'z_{14}) \tag{39}$$

is the fluid volume swept through momentum control volume surface element ds'_{14}, whose areas of projection on the yz plane, xz plane, and xy plane are given by $ds'x_{14}$, $ds'y_{14}$, and $ds'z_{14}$. The velocity of surface element ds'_{14} relative to the fluid is calculated as

$${}^{R}\bar{\mathbf{u}}_{14} = \tfrac{1}{4}({}^{R}\mathbf{u}_1 + {}^{R}\mathbf{u}_4 + 2{}^{R}\mathbf{u}_c),$$

where

$${}^{R}u_c = \tfrac{1}{8}({}^{R}\mathbf{u}_1 + {}^{R}\mathbf{u}_2 + {}^{R}\mathbf{u}_3 + {}^{R}\mathbf{u}_4 + {}^{R}\mathbf{u}_5 + {}^{R}\mathbf{u}_6 + {}^{R}\mathbf{u}_7 + {}^{R}\mathbf{u}_8).$$

Here, again the α factor is related to the type of differencing chosen and is given by

$$\alpha_{14} = \alpha_0(\delta V'_{14}/|\,\delta V'_{14}\,|) + (\beta_0\,\delta V'_{14}/V). \tag{40}$$

The velocity at vertex 1 will also be affected by exchanges through a portion of the other two control volume faces with vertex 5 and vertex 2. To illustrate, the stage of velocity adjustments resulting from these momentum exchanges for vertex 1 is given by

$${}^{n+1}u_1 = {}^{n+1}u_1 + (1/M_1)[uF_{14} - uF_{12} - uF_{15}]. \tag{41}$$

Final specific internal energy is now calculated to complete the basic cycle. It is given by

$$\begin{aligned} {}^{n+1}I = E &- (1/16M)[M_1(u_1^2 + v_1^2 + w_1^2) + M_2(u_2^2 + v_2^2 + w_2^2) \\ &+ M_3(u_3^2 + v_3^2 + w_3^2) + M_4(u_4^2 + v_4^2 + w_4^2) + M_5(u_5^2 + v_5^2 + w_5^2) \\ &+ M_6(u_6^2 + v_6^2 + w_6^2) + M_7(u_7^2 + v_7^2 + w_7^2) + M_8(u_8^2 + v_8^2 + w_8^2)], \end{aligned} \tag{42}$$

where all the terms on the right-hand side are taken at the advanced time $(n + 1)$ level.

V. Initial and Boundary Conditions

The initial state is described by arrays of arbitrarily prescribed distributions of velocities, densities, and internal energies. These arrays are generated by a set-up program and represent the starting configuration of the fluid. To complete the initialization procedures requires, for many problems, the specification and generation of the computing mesh. Here, as in most finite-difference techniques, physical boundaries correspond with cell boundaries. However, the ALE features enable us to calculate flows confined within arbitrarily shaped boundaries. Shown in Section VIII are several examples of the variety presently available. The curvilinear meshes are generated by a three-dimensional extension of the mesh relaxation process described in [13].

Many kinds of boundaries are possible. We are concerned here with only rigid free-slip, rigid no-slip, prescribed inflow, and prescribed outflow. Generally, these conditions are specified in terms of the required fluxes of mass, momentum, and

energy and are accomplished by making adjustments to the velocities of the boundary vertices. These adjustments are made at each of the three phases previously described. At a rigid wall, for example, the vanishing of all convective fluxes implies the vanishing of the normal velocity component. In the case of a no-slip condition the tangential velocity also vanishes. The general case of the free-slip rigid wall is the more difficult condition to handle because only the normal velocity is set to zero. If the boundary is nonplanar or does not correspond to one of the x, y, or z coordinate directions, the orientation of the boundary must first be determined; then all the three velocity components are adjusted so as to leave the two tangential velocity components unchanged while replacing the normal velocity with zero. In the example shown in Section VIII, only the simpler case of a rigid-free-slip reflective plane and the rigid-no-slip conditions are used. A procedure that provides a more general free-slip boundary capability for irregularly shaped and movable boundaries confining or obstructing the flow is described in [14].

Prescribed inflow and outflow conditions, as in the examples shown in Fig. (12) are easily determined by the fluxes that are required.

VI. Numerical Stability

Calculations of three-dimensional fluid flows using the ICED–ALE method are restricted by the same form of numerical stability and accuracy conditions as for the two-dimensional flows [1]. Again, because of the generalized and varying relation between the computational grid and the fluid itself, as represented by the field variables, only general guidelines can be suggested. These are primarily based on analysis of simpler, fixed-cell finite-difference schemes and analogous systems of equations. These conditions pose two basic restrictions on the time-step size, one related to convection-type terms and one to diffusion and diffusion-like terms.

The restrictions resulting from the convective flux of fluid through computational cell boundaries is clearly also an accuracy consideration. It is expressed as

$$\delta t < | V/({}^{R}\mathbf{u} \cdot d\mathbf{s})|_{\text{MIN}}, \tag{43}$$

where V is the average volume surrounding the fluxed quantity and ${}^{R}\mathbf{u} \cdot d\mathbf{s}$ gives the volume of fluid swept through the flux boundary. We can note here that for pure Lagrangian runs the relative velocity ${}^{R}\mathbf{u} \equiv 0$, and this restriction does not arise.

Another restriction has as its basis the convective flux phase of the calculation but is related to computational diffusion. From the convective flux approximations,

there arise certain truncation errors that behave as negative diffusion terms. In order that the calculation run stably, these must be eliminated as far as possible, by counterbalancing them with positive diffusion terms. Choosing the form of convective flux approximations, as discussed in the previous section, provides one means of control. A greater proportion of donor cell fluxing provides larger effective computational diffusion, which automatically occurs only where needed. Adjusting the actual diffusion terms in the momentum provides a more predictable means of control, especially if the coefficients are allowed to vary from cell to cell as needed. In practice these can be used to provide a means for regulating the time step for the calculations, and one can include an option for an automatic determination of the viscosity coefficients λ and μ.

Finally, perhaps the most restrictive of the various numerical stability restrictions effectively limits the distance over which momentum can diffuse in one time step. Once again we are able to only estimate this, but experience has shown the validity of the following expression.

$$\delta t < \left[\frac{2(2\mu + \lambda)}{\rho}\left(\frac{1}{\delta x^2} + \frac{1}{\delta y^2} + \frac{1}{\delta z^2}\right)\right]^{-1}. \tag{44}$$

The effective cell size, expressed here as δx, δy, and δz, is defined as the average component distances of the four computational cell diagonals.

A third, but somewhat different type of numerical stability-like disturbance may arise. This undesirable feature is manifested in a saw-tooth appearance of the Lagrangian grid, or an uneven appearance of the marker particles. It arises from alternating vertex drifting because of a lack of coupling between adjacent vertices in the Lagrangian phase. An effective means for controlling this form of distortion is to explicitly couple adjacent vertices by adding to each a small acceleration that reduces the relative velocities between it and its nearest neighbors. The prescription suggested in [1], when extended to three-dimensional flows, couples each vertex to its six nearest neighbors. Brackbill has suggested an alternative prescription [15] coupling each to its nearest 18 neighbors. The second of these offers the advantage of adequately coupling adjacent vertices without adding excessive damping.

VII. Particulate Transport

Marker particles are used in numerical fluid dynamics calculations for a variety of purposes. One common usage for Lagrangian marker particles that follow exactly the motion of the fluid is to aid in visualizing complicated, highly distorted flows. One example of this is shown on the right-hand side of Fig. 10. In this

calculation the particles allow us to follow the motion of two hot spheres of material rising in an ambient atmosphere. To avoid the numerical difficulties of a Lagrangian computational mesh, the run shown has been continuously rezoned and translated upward. While this allows us to follow the collapse and distortion of the bubbles, it also means that material is being fluxed through cell boundaries so the calculational mesh motion no longer represents the actual fluid configuration. Other, somewhat different examples of applications for marker particles are illustrated in [16] for incompressible flow through curved vessels. The purpose for the particles in this is to aid in the examination of secondary flow effects for several configurations similar to the example shown in Fig. (12).

Lagrangian marker particles are moved with the local fluid velocity each time step. The velocity of each particle is obtained from a trilinear interpolation among the particle's eight nearest vertex velocities. The usual interpolation scheme cannot be directly applied because of the arbitrarily shaped and distortable computational cells. The generalized trilinear interpolation scheme is based on the concept of a one-to-one mapping of the x, y, z coordinates of the eight vertices of each cell onto a ξ, η, ζ-coordinate system, where ξ, η, and ζ take on values between 0 and 1 in each cell. A schematic is shown in Fig. 8. The transformation is given implicitly in vector form as

$$\begin{aligned}\mathbf{r} = {} & \xi(1-\eta)(1-\zeta)\,\mathbf{r}_1 + \xi\eta(1-\zeta)\,\mathbf{r}_2 + (1-\xi)\,\eta(1-\zeta)\,\mathbf{r}_3 \\ & + (1-\xi)(1-\eta)(1-\zeta)\,\mathbf{r}_4 + \xi(1-\eta)\,\zeta\mathbf{r}_5 + \xi\eta\zeta\mathbf{r}_6 \\ & + (1-\xi)\,\eta\zeta\mathbf{r}_7 + (1-\xi)(1-\eta)\,\zeta\mathbf{r}_8\,, \end{aligned} \tag{45}$$

where $\mathbf{r}_1$ through $\mathbf{r}_8$ denote the x, y, z coordinates of the eight cell vertices. The particle coordinates in ξ, η, ζ space can be determined by inverting transformation (45) with the aid of a Newton–Raphson iteration. In practice this converges very

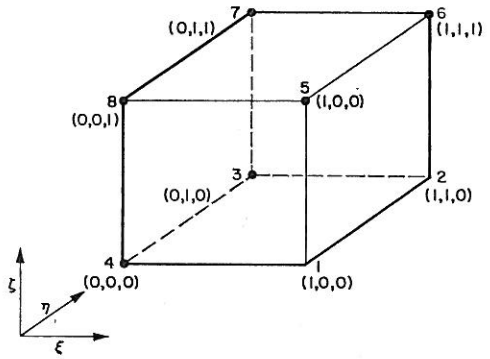

FIG. 8. Computational cell vertex positions in terms of a ξ, η, ζ-coordinate system.

rapidly even for cases in which the cells are badly distorted if, for each particle, the initial guess is the ξ, η, ζ from the preceding cycle. The transformed coordinates also provide a simple and efficient means for tracking the marker particles as they move through the computational cells. More specifically, we assume, when determining the ξ, η, ζ for a particular particle, that the particle lies in the same calculational cell as last cycle, and for the vast majority of cases, this will be the case. If it is not, the newly computed values of ξ, η, and ζ indicate the direction to move to locate the proper cell. Finally, an expression of the form of Eq. (45) in terms of velocities yields an appropriately interpolated velocity with which to move the particle.

Another purpose of particles in flow studies is to represent particulate matter whose behavior is affected by inertia, drag exerted on the particulate by the fluid motion, gravity, and molecular and turbulent diffusion [17]. The capability for calculating time-varying, three-dimensional particulate motions in highly distorted fluid flows can be applied to solving a wide scope of complicated problems. One example is the problem of pollution dispersed in the atmosphere. The flow into which these pollutants may be emitted is likely to be distorted by odd-shaped buildings, hills, valleys and canyons, thus requiring for solution a fully three-dimensional capability. Another related example is an examination of particulate paths and deposition patterns in the trachea and bronchial passages of inhaled parcels of pollutants. A variety of flow examples closely related to the second of these two problems is given in [16]. The numerical prescription for determining the motion of these inertial particles is accomplished by assigning an equation of motion to each particle. An example of an equation which includes the additional effects of inertia, drag, and diffusion is given by

$$d\mathbf{u}_p/dt = ((\mathbf{u}_f - \mathbf{u}_p)/\tau) + (\mathbf{u}_{\text{diff}}/\tau) + \mathbf{g}, \tag{46}$$

where τ can be thought of as the relaxation time for the particle velocity $\mathbf{u}_p$ to become zero if the fluid velocity $\mathbf{u}_f$ were zero. Alternatively τ could be thought of as the reciprocal of the coefficient of Stokes drag divided by the mass of the particle. The average speed with which particles diffuse relative to the mean flow is indicated here by

$$\mathbf{u}_{\text{diff}} = ((4\delta t\,\lambda)^{1/2}/\delta t)\,\text{erf}^{-1}(y),$$

where y are random numbers on the interval $0 < y < 1$, and λ is the diffusion coefficient.

VIII. Sample Applications

To demonstrate the applicability of this newly extended version of the ICED–ALE computing method, we illustrate calculations for both high-speed and low-

speed flow examples. These include the interaction of two large heated spheres rising side-by-side in an ambient atmosphere and the flow of an incompressible fluid through a curved tube. These are presented as primarily qualitative examples to demonstrate a range of capabilities. Although we do not here attempt any detailed comparison with experimental or analytical data, proof testing and comparisons during the design and construction of the experimental program BAAL give considerable confidence in the validity of the calculations. For example, a variety of comparisons were made with a two-dimensional, axisymmetric version [1] of the ICED–ALE method with results that were almost identical. In addition, the examination of numerous examples of one-dimensional shock-tube problems yielded very favorable comparisons.

One of the most satisfactory ways of presenting the enormous amount of data produced in calculations of three-dimensional flow fields is in the form of perspective view plots. Hirt *et al.* [18] have developed conceptually simple and efficient techniques for the presentation of data in this form for fixed, regularly spaced computational cells. A number of the features of these techniques can be readily adapted for use with the arbitrarily shaped cells in the work described here. These include perspective views of selected planes of contour lines, as well as views of velocity and vorticity vectors, grid plots, particle distributions, and contour surfaces. Other features such as hidden line removal and construction of contour plots in any arbitrary cross section through the calculational mesh are somewhat more difficult to extend.

A. *Compressible Flow*

In the first example, the initial configuration shows two hot, light (50:1 density ratio) spheres of radius 0.2 km embedded at an altitude of 3.0 km in a stratified atmosphere under pressure equilibrium. Perspective views of density contour lines and a density contour surface are shown, respectively, on the left and right sides of Fig. 9. An eye point of approximately 30° right of center was chosen for the contour surface plot to better demonstrate the three dimensionality of the interaction. These surfaces of constant, 60% ambient density, show the front half of the two side-by-side, hot, light spheres as they rise and interact. The contour lines, viewed from the center position, show the density distribution in the vertical plane of symmetry that passes through the bursts' centers. Representing the sphere initially rather coarsely in terms of a "stack" of rectangular solids results in some roughness or unevenness of the contour surface. However, a variety of truly three-dimensional aspects of the flow are evidenced in this sequence of frames. A comparison of the plots in the "back" vertical plane of the calculational grid and particles initially in the spheres shown in Fig. 10 shows the necessity for mesh rezoning. Clearly if the mesh itself were allowed to undergo the same amount of distortion shown by the marker particles, the calculation would have long

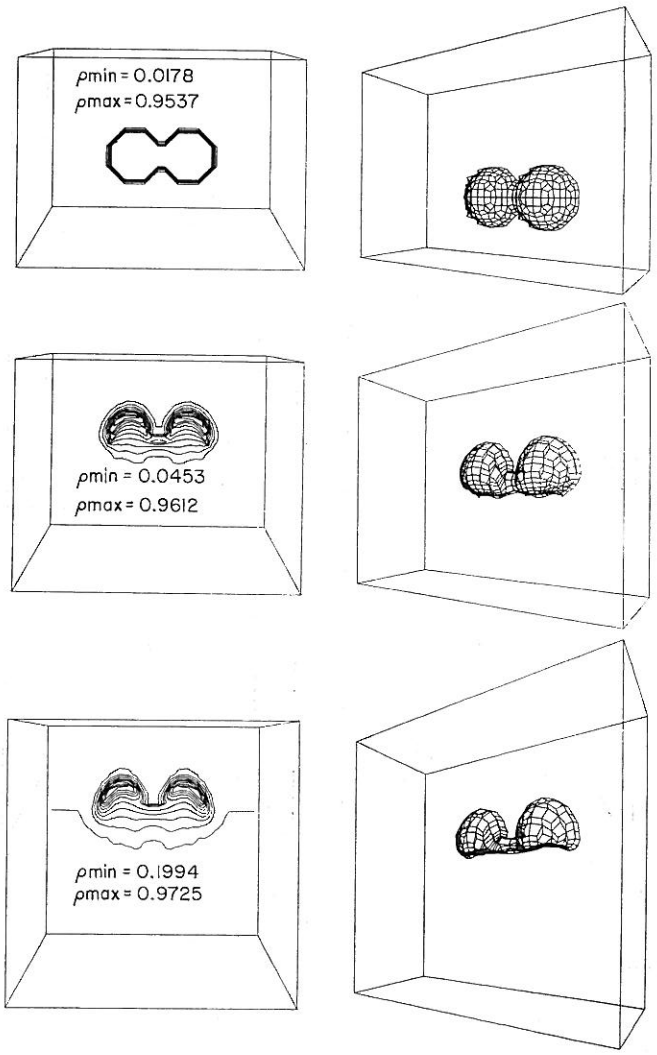

FIG. 9. Calculation of two, simultaneous, side-by-side explosions in the atmosphere. Plots shown at 0.3, 10.0, and 20.0 sec are: on the left, lines of constant density in the *yz* plane of symmetry, and on the right, surfaces of constant, 60% ambient density.

since been a disaster. At the same time, the contour lines plotted in Fig. 9 show a minimum of diffusion, demonstrating the advantage of the mixed Eulerian–Lagrangian capability. Additional effects of the two spheres rising side-by-side can be seen in the flow patterns plotted in Fig. 11. Shown in the top frame is a

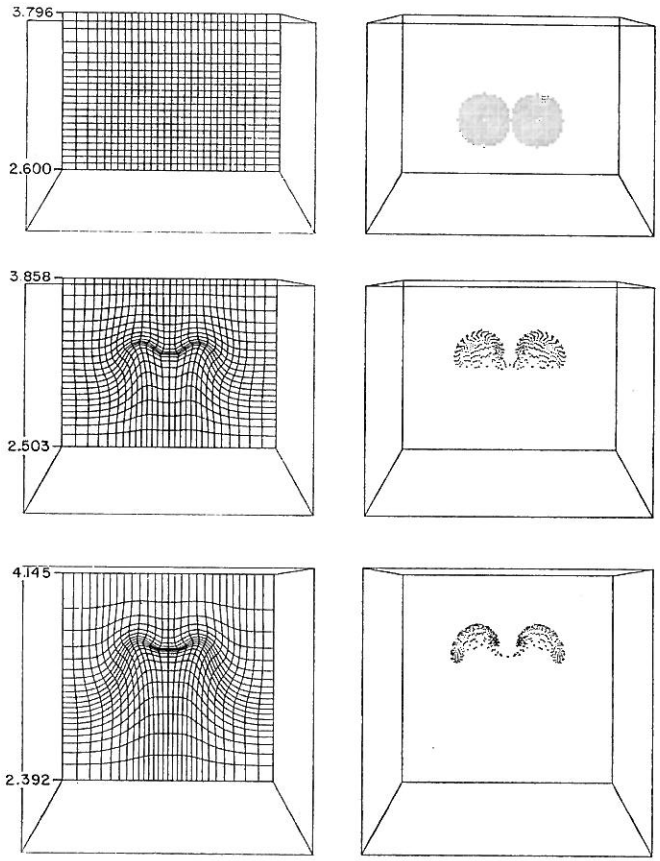

FIG. 10. A portion of the computing grid and the marker particle configuration, both in the *yz* plane of symmetry are shown at the same times as in Fig. 9. The expansion of the mesh is indicated here and in Fig. 9 by a changing postion of the grid boundaries relative to a fixed viewing position.

perspective view of vectors depicting the velocity in the vertical plane through the two spheres. The bottom frame is a perspective view, looking down on the mesh, of vorticity vectors showing the pair of distorted vortex rings. Evidence of the vortex ring distortion, present in both the velocity and vorticity field plots is seen primarily as a thinning and raising of the vortex rings in the area between the spheres.

B. *Incompressible Flow*

To illustrate an entirely different capability of the BAAL program, we have examined the steady flow of an incompressible fluid around a 90° bend in a tube.

For calculations such as this in the incompressible limit (here the sound speed is set to 10^{15}), it is best to bypass the calculation of pressure for phase one as an equation of state function of density and internal energy. It is better to use instead, the pressure left from the preceding phase-two iteration as a first guess for the next cycle. The reason is that small fluctuations in density from phase-three fluxing may result in large pressure fluctuations difficult for the phase-two iteration to overcome. As a rule of thumb if the calculation has a sound speed significantly larger than $u/\epsilon^{1/2}$, where u is a typical fluid speed and ϵ is the phase-two convergence criterion, the calculation should be considered incompressible and this procedure

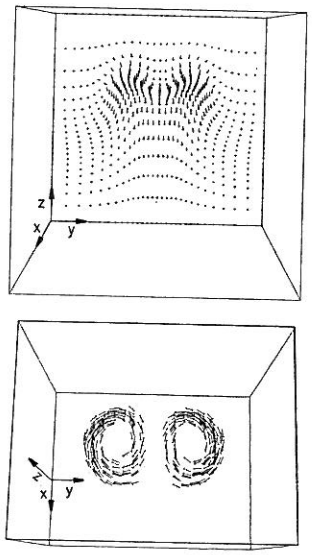

FIG. 11. Late time velocity distribution in the *yz* plane of symmetry (top half) and vortex rings as viewed by looking down on the entire double burst configuration. The vortex rings are formed by plotting vorticity vectors.

should be followed. For fully incompressible flow, it is the task of the iteration process to adjust the velocity field, by calculating changes in pressure, so that the incompressibility condition, $\nabla \cdot \mathbf{u} = 0$, is everywhere satisfied. In principle this could be accomplished by starting from any arbitrary pressure field, but the better the first guess, the more efficient will be the iteration. However, the iteration process will always be terminated with some error, which if left unchecked could accumulate after many cycles. This accumulation of error can be avoided by using a self-correcting procedure [19] in which a cycle-to-cycle adjustment is automatically made by having the error from one cycle, expressed for each computa-

tional cell as $D = u_x + v_y + w_z$, serve as a source term for the iteration for the next cycle. This effectively limits the accumulation of incompressibility error, even with a relatively coarse pressure iteration.

Velocity profile, tube radius, and viscosity were chosen for the calculation shown in Fig. 12 to represent the flow of blood in an artery. In this example, the calculational mesh, shown in Fig. 12a, is held fixed with respect to the fluid, but having the walls move in an arbitrarily prescribed manner or in response to the fluid motion is well within the capability of the method.

Velocity vectors and contours of constant pressure are also shown in the figure. We have, in this calculation, taken the horizontal plane through the tube center as a plane of symmetry and calculated only the bottom half. Figure 12c is a

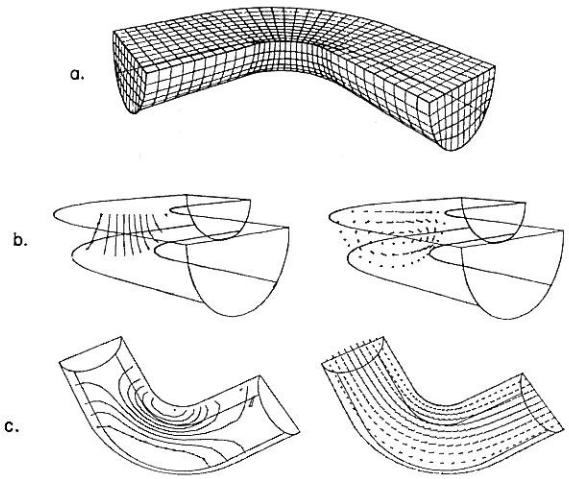

FIG. 12. Calculation of incompressible flow around 90° bend. Shown are perspective view plots of the computing grid (a) together with velocity vectors and pressure contours (b) and (c).

perspective view plot from above the tube and shows the velocity and pressure distribution in this plane of symmetry. Figure 12b plots these variables in a vertical cross section through the center of radius of curvature. The velocity vectors here have been enlarged to better illustrate the secondary flow pattern. The pressure gradient across the pipe to balance the centrifugal force is evident in both Figs. 12b and 12c.

ACKNOWLEDGMENTS

The author wishes to thank J. U. Brackbill, B. J. Daly, F. H. Harlow, and C. W. Hirt for their interest, encouragement, and many helpful discussions.

REFERENCES

1. C. W. HIRT, A. A. AMSDEN, AND J. L. COOK, *J. Computational Phys.* **14** (1974), 227; A. A. AMSDEN, AND C. W. HIRT, "YAQUI: An Arbitrary Lagrangian–Eulerian Computer Program for Fluid Flow at All Speeds," Los Alamos Scientific Laboratory Report LA-5100 (1973).
2. F. H. HARLOW AND A. A. AMSDEN, *J. Computational Phys.* **8** (1971), 197.
3. A. W. RIZZI AND M. INOUYE, *AIAA J.* **11** (1973), 1478.
4. M. C. CLINE AND J. D. HOFFMAN, *J. Computational Phys.* **12** (1973), 1.
5. G. P. WILLIAMS, *J. Fluid Mech.* **37** (1969), 727.
6. W. R. BRILEY, *J. Computational Phys.* **14** (1974), 8–28.
7. L. S. CARETTO, A. D. GOSMAN, S. W. PATANKAR AND D. B. SPALDING, "Two Calculation Procedures for Steady, Three-Dimensional Flows with Recirculation," Proceedings of Third International Conference on Numerical Method in Fluid Mechanics, Universities of Paris VI and XI (July, 1972).
8. C. W. HIRT AND J. L. COOK, *J. Computational Phys.* **10** (1972), 324–340.
9. W. E. JOHNSON, "TRIOIL (A Three-Dimensional Version of the OIL Code)," Gulf General Atomic, San Diego, Report GAMD-7310 (June 1, 1967).
10. M. L. WILKINS, S. J. FRENCH, AND M. SOREM, "Finite-Difference Schemes for Calculating Problems in Three Space Dimensions and Time," Proceedings of the Second International Conference on Numerical Methods in Fluid Dynamics, University of California (September, 1970).
11. W. E. PRACHT AND J. U. BRACKBILL, "BAAL: A Code for Calculating Three-Dimensional Fluid Flows at All Speeds with an Eulerian–Lagrangian Computing Mesh," Los Alamos Scientific Laboratory Report, to be published (1974).
12. J. U. BRACKBILL AND W. E. PRACHT, *J. Computational Phys.* **13** (1973), 455–482.
13. A. A. AMSDEN AND C. W. HIRT, *J. Computational Phys.* **11** (1972), 348–359.
14. R. A. GENTRY, L. R. STEIN AND C. W. HIRT, "Blast Loading on Three-Dimensional Structures," manuscript in preparation.
15. J. U. BRACKBILL, Los Alamos Scientific Laboratory, private communication (1974).
16. W. E. PRACHT, "Particulate Transport in Three-Dimensional Fluid Flows Through Curved and Bifurcating Vessels," manuscript in preparation.
17. R. S. HOTCHKISS AND C. W. HIRT, "Particulate Transport in Highly Distorted Three-Dimensional Flow Fields," Proceedings of the Computer Simulation Conference, San Diego, CA (1972); also Los Alamos Scientific Laboratory Report LA-DC-72-364.
18. C. W. HIRT AND J. L. COOK, "Perspective Displays for Three-Dimensional Finite Difference Calculations," *J. Computers Fluids*, accepted for publication (1974).
19. C. W. HIRT AND F. H. HARLOW, *J. Computational Phys.* **2** (1967), 114–119.