Journal of Computational Physics **175,** 200–224 (2002)
doi:10.1006/jcph.2001.6935, available online at http://www.idealibrary.com on IDEAL®

Coupling an Eulerian Fluid Calculation to a Lagrangian Solid Calculation with the Ghost Fluid Method

Ronald P. Fedkiw[1,2]

Computer Science Department, Stanford University, Stanford, California 94305

Received April 6, 2000; revised June 25, 2001

We propose a numerical method for modeling multimaterial flows where the domain is decomposed into separate Eulerian and Lagrangian subdomains. That is, the equations are written in Eulerian form in one subdomain and in Lagrangian form in the other subdomain. This is of interest, for example, when considering the effect of underwater explosions on the hull of a ship or the impact of a low speed projectile on a soft explosive target. On the one hand, high-speed fluid flows are traditionally modeled by applying shock-capturing schemes to the compressible Euler equations to avoid problems associated with tangling of a Lagrangian mesh. On the other hand, solid dynamics calculations are traditionally carried out using Lagrangian numerical methods to avoid problems associated with numerical smearing in Eulerian calculations. We use the ghost fluid method to create accurate discretizations across the Eulerian/Lagrangian interface. The numerical method is presented in both one and two spatial dimensions; three-dimensional extensions (to the interface coupling method) are straightforward.

1. INTRODUCTION

Solid/fluid interaction problems are still rather difficult to solve with modern computational methods. In general, there are three classical approaches to such problems: treat both the solid and the fluid with Eulerian numerical methods, treat the fluid with an Eulerian numerical method and the solid with a Lagrangian numerical method, or treat both the solid and the fluid with Lagrangian numerical methods.

[1] Research supported in part by the California Institute of Technology Center for Simulation of Dynamic Response of Materials, which is sponsored by the U.S. DOE as part of the Academic Strategic Alliances Program of the Accelerated Strategic Computing Initiative (Subcontract B341492 of DOE Contract W-7405-ENG-48).

[2] Research supported in part by ONR N00014-97-1-0027.

0021-9991/02 $35.00

Certain fluids, e.g., high-speed gas flows with strong shocks and large deformations, are very difficult to solve for with Lagrangian numerical methods. Lagrangian numerical methods use artificial viscosity to smear out numerical shock profiles over a number of zones to reduce post-shock oscillations or ringing. Thus one has to choose the form for the artificial viscosity, which can be both problem and material dependent. In addition, Lagrangian numerical methods have difficulties treating flows with large deformations, since this causes large deformations of the mesh and subsequent large numerical errors that can only be removed with complicated remeshing and/or mesh generation procedures that tend to be low-order accurate. In particular, flows with vorticity cause the mesh to tangle and sometimes invert, in which case the calculation needs to be stopped. Eulerian numerical methods intrinsically avoid these mesh associated problems, since they use a stationary mesh. Furthermore, Eulerian shock-capturing schemes capture shocks in a straightforward way using conservation and robust limiters, eliminating the need for problem-dependent artificial viscosity formulations altogether. This allows shocks to be modeled with as few as one grid cell (without oscillations), whereas Lagrangian numerical methods usually suffer from some amount of post-shock oscillations until the shock is spread out over about six grid cells; see, e.g., [3, 4].

While Eulerian numerical methods are superior for high-speed gas flows, they may perform poorly for many solid dynamics calculations. Since Eulerian numerical methods *capture* the properties of a material, they are not very accurate or robust when tracking material properties that are important for modeling time history variables. Because of this, Eulerian numerical methods do not give accurate results when dealing with material response to loading and damage. In contrast, Lagrangian numerical methods are extremely accurate and well tested in this area.

At this point, it is obvious that it is preferable to use Eulerian numerical methods on the fluid and Lagrangian numerical methods on the solid. It remains to address exactly how these two calculations can be coupled together to model solid/fluid interactions. There are essentially two techniques for treating this solid/fluid interface. One technique is to smear out the nature of the numerical approximations using a Lagrangian numerical method in the solid and an Eulerian numerical method in the gas with some "mushy" region in between where the grid moves with an intermediate speed in between that of the Lagrangian mesh and the stationary Eulerian mesh. That is, the grid speed is smoothly varying in between these two regions. This is essentially the idea behind the arbitrary Lagrangian–Eulerian (ALE) numerical algorithm; see, e.g., [3]. The problem here is that these variable-speed meshes are not well studied and the numerical algorithms employed here tend to be low-order accurate and sometimes suspect. The other technique for treating the solid/fluid interface is to keep the mesh representation sharp so that the Eulerian and Lagrangian meshes are in direct contact. The problem with this method is that the Lagrangian mesh moves, causing Eulerian mesh points to appear and disappear. In addition, the Eulerian cells tend to have irregular shapes, sometimes referred to as cut cells. These cut cells can lead to numerical errors and stiff time-step restrictions; see, e.g., [3] and the references therein, specifically [13], [19], and [18] which discuss the PISCES, CEL, and PELE codes respectively.

We prefer the second approach for treatment of the solid/fluid interface. That is, we keep the Lagrangian and Eulerian meshes in direct contact to avoid special methods for an arbitrary speed mesh. In our approach, problems with cut cells are avoided by the use of ghost cells for the Eulerian mesh. These ghost cells are covered (or partially covered) by the Lagrangian mesh, but they are used in the Eulerian finite difference scheme to circumvent

small time step restrictions. These ghost cells are defined in a way consistent with the ghost fluid method [11] so that the interface boundary conditions or jump conditions are properly captured. This method also avoids the blending problems associated with covering and uncovering of grid points since covered real grid nodes are treated as ghost nodes and uncovered ghost nodes are treated as real grid nodes. Our approach is novel in that the numerical treatment of the solid/gas interface does not compromise the solution techniques for the solid or the fluid at the interface. That is, once the ghost cell values are specified, a standard Eulerian code can be used to advance the fluid (and its ghost nodes) in time. Likewise, once the Lagrangian boundary conditions are specified, a standard Lagrangian code can be used to advance the Lagrangian mesh in time.

In this paper, our focus is on the interaction between materials with a compressible nature. If one is interested, for example, in treating the solid as a rigid body or the fluid as incompressible, then nontrivial modifications may be needed for both efficiency and accuracy. Moreover, in these and other cases, one should also consult the available literature. For example, [24] couples an elastic solid to a Stokes fluid (see [16] as well).

Sections 2 and 3 discuss the Euler and Lagrange equations respectively, and the novel method for coupling them together is discussed in Section 4. Section 5 shows how this new method can be used on stiff fluid/fluid interfaces to significantly improve the results obtained with the ghost fluid method in [11]. Finally, Section 6 presents the results obtained from the numerical experiments in one and two spatial dimensions.

2. EULER EQUATIONS

The Euler equations for two-dimensional compressible flow are

$$\begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E \end{pmatrix}_t + \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ (E+p)u \end{pmatrix}_x + \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ (E+p)v \end{pmatrix}_y = 0, \tag{1}$$

where t is time, x and y are the spatial dimensions, ρ is the density, u and v are the velocities, E is the total energy per unit volume, and p is the pressure. The total energy is the sum of the internal energy and the kinetic energy,

$$E = \rho e + \frac{\rho(u^2+v^2)}{2}, \tag{2}$$

where e is the internal energy per unit mass. The pressure can be written as a function of density and internal energy, $p = p(\rho, e)$. For the sake of simplicity only a gamma law gas, $p = (\gamma - 1)\rho e$, is considered in this paper. The one-dimensional Euler equations are obtained by setting $v = 0$. These equations are discretized using third-order-accurate ENO methods. See [12, 23] for more details.

On the Eulerian mesh, the convective time step restriction is given by

$$\Delta t \left(\frac{|u| + c}{\Delta x} \right) \leq 1 \tag{3}$$

in one spatial dimension and by

$$\Delta t\left(\frac{|u|+c}{\Delta x}+\frac{|v|+c}{\Delta y}\right)\leq 1 \tag{4}$$

in two spatial dimensions where $c=\sqrt{\gamma p/\rho}$ is the speed of sound. Note that Δt is chosen so that Eq. (3) (or (4)) is valid at every grid node of the Eulerian mesh.

3. LAGRANGE EQUATIONS

The Lagrange equations are written in nonconservative form with position, velocity, and internal energy as the independent variables. Since the equations are quite different in one, two, and three spatial dimensions we address the one- and two-dimensional equations separately throughout this paper. Our two-dimensional numerical method is essentially the method proposed in [6, 8, 9] with only minor modifications. In [6], the authors constructed a Lagrangian numerical method that can treat arbitrary forces in a simple straightforward fashion. In [8], the authors showed how subzonal pressures can be used to avoid mesh tangling. And in [9], the authors presented an edge-centered artificial viscosity. Our one-dimensional numerical method is quite standard and can be seen as a straightforward simplification of the two-dimensional numerical method from [6, 8, 9] to a single spatial dimension. Three-dimensional Lagrangian methods are not discussed this paper, but the interested reader is referred to [7], which is a three-dimensional extension of the method proposed in [6, 8, 9]. The interested reader is also referred to [17] and [10].

3.1. *One Spatial Dimension*

In one spatial dimension the independent variables are x, u, and e. Both x and u are defined at the grid nodes while e is defined at the cell centers located midway between the grid nodes. Each cell or zone is split into two subzones based on the midpoint of each cell. To initialize the calculation, the mass of each zone, M^z, is determined, and then the subzonal masses, m^z, are defined as half the zonal mass. The nodal mass, M^p, is defined as the sum of the neighboring subzonal masses. The nodal, zonal, and subzonal masses all remain fixed throughout the calculation.

Each time step, the location of each grid node is updated according to

$$\frac{x^{n+1}-x^n}{\Delta t}=u^n, \tag{5}$$

where Δt is the size of the time step. The velocity at each node is updated using

$$\frac{u^{n+1}-u^n}{\Delta t}=\frac{F^n}{M^p}, \tag{6}$$

where F^n is the net force on the grid node. And the internal energy in each zone is updated with

$$\frac{e^{n+1}-e^n}{\Delta t}=\frac{H^n}{M^z}, \tag{7}$$

where H^n is the heating rate of the zone. One can apply either force or velocity boundary conditions to the grid nodes on the boundary. Velocity boundary conditions are enforced by simply setting the velocity of a boundary node to the desired boundary velocity instead of solving Eq. (6). Force boundary conditions are applied by adding the boundary force to the net nodal force F^n in Eq. (6).

The density of each subzone is determined by dividing the subzonal mass by the subzonal length. Then the subzonal pressure is defined based on the subzonal density and the zonal internal energy. For example, in the case of a gamma law gas, $p = (\gamma - 1)\rho e$. In one spatial dimension, each zone has two subzonal pressures that are identical to each other, implying that only a zonal pressure need be defined. However, in multiple dimensions, the subzonal pressures are not equal and help to avoid mesh tangling [8]. For this reason, we use subzonal pressures in the one-dimensional exposition. Each subzonal pressure provides a contribution to the net force on the adjacent grid node as well as a contribution to the heating rate of the zone that contains it. If the adjacent node is to the left of the subzone, the contribution to the nodal force is $-p$ and the contribution to the zone heating rate is pu, where u is the velocity of the grid node. Otherwise, if the adjacent node is to the right of the subzone, the contribution to the nodal force is p and the contribution to the zone heating rate is $-pu$.

An artificial viscosity, Q, is computed in each zone and is used to augment the net nodal force of the two adjacent grid nodes as well as the heating rate of the zone. The contribution to the net nodal force of the node on the left is $-Q$, while the contribution to the net nodal force of the node on the right is Q. The contribution to the zone heating rate is $Qu_i - Qu_{i+1}$, where u_i and u_{i+1} are the velocities of the left and right nodes respectively. Artificial viscosity is only applied when the zone is under compression, so Q is set identically to zero if $\Delta u = u_{i+1} - u_i \geq 0$. Otherwise.

$$Q = (1 - \psi)(c_1 \rho c |\Delta u| + c_2 \rho (\Delta u)^2), \tag{8}$$

where ρ is the density of the zone determined by dividing the zonal mass by the length of the zone, c is the zonal sound speed determined using the density and internal energy of the zone, c_1 and c_2 are constants used to set the magnitude of the linear and quadratic components of the artificial viscosity respectively, and ψ is used to smoothly switch the artificial viscosity on and off. ψ is defined as

$$\psi = \max\left(0, \min\left(\frac{r^- + r^+}{2}, 2r^-, 2r^+, 1\right)\right), \tag{9}$$

where $r^- = u_x^-/u_x$ and $r^+ = u_x^+/u_x$. The spatial derivative of the velocity, u_x, in each zone is defined as Δu over the length of the zone. u_x^- and u_x^+ are the derivatives in the zone to the left and to the right of the current zone respectively. If either the zone to the left or the zone to the right is undefined because the current zone lies on a boundary, then the corresponding value of r is set identically equal to 1.

Material strength is applied by putting springs in each zone to connect the neighboring nodes. Each zone is assigned a rest length, L_o, and restoring forces occur when a zone's length, L, is not equal to its rest length. The restoring force is simply $S = -k(\frac{L}{L_o} - 1)$ where $k \geq 0$ is some measure of the stiffness of the material. This restoring force makes contributions of $-S$ to the net nodal force of the node on the left, S to the net nodal force of the node on the right, and $Su_i - Su_{i+1}$ to the zone heating rate.

For the one-dimensional Lagrangian mesh,

$$\Delta t \left(\frac{\hat{c}}{L} \right) \leq 1 \tag{10}$$

is enforced in every zone where $\hat{c} = \sqrt{\gamma \hat{p}/\rho}$ is the effective speed of sound using the effective pressure $\hat{p} = p + Q + |S|$, which includes the effects of artificial viscosity and material strength. Note that the magnitude of S is used as S can be negative, unlike Q, which is always positive.

3.2. *Two Spatial Dimensions*

In two spatial dimensions the independent variables are $\mathbf{X} = \langle x, y \rangle$, $\mathbf{V} = \langle u, v \rangle$, and e. Both $\mathbf{X}$ and $\mathbf{V}$ are defined at the grid nodes which are connected in same fashion as an Eulerian grid producing quadrilateral zones. Each quadrilateral zone is split into four subzones by connecting the midpoints of opposite edges of the zone. e is defined at the zone center, which is located at the intersection of the four subzones and can be computed by averaging the four nodes that make up the quadrilateral zone. To initialize the calculation, the mass of each zone, M^z, is determined, and then the subzonal masses, m^z, are defined as one-fourth the zonal mass. The nodal mass, M^p, is defined as the sum of the (at most four) neighboring subzonal masses. Once again, the nodal, zonal, and subzonal masses all remain fixed throughout the calculation. The independent variables are updated with Eqs. (5), (6), and (7) replacing x, u, and F^n with $\mathbf{X}$, $\mathbf{V}$, and $\mathbf{F}$ respectively. Either force or velocity boundary conditions are applied to the grid nodes on the boundary.

The density of each subzone is determined by dividing the subzonal mass by the subzonal area. Then the subzonal pressure is defined based on the subzonal density and the zonal internal energy. Exactly one of the four corners of each subzone corresponds to a grid node. The two subzonal edges connected to this grid node are used to calculate the contribution of the subzonal pressure to the net nodal force at this node. For each of these two edges, this is done by multiplying the subzonal pressure by the length of the edge and by the unit vector perpendicular to the edge. The dot product of this force with the velocity of the grid node gives the contribution of the subzonal pressure to the heating rate of the zone.

In each zone, four separate artificial viscosities are computed, i.e., one along each edge. For a given edge with nodes designated by subscripts i and $i+1$, $\Delta\mathbf{V} = \mathbf{V}_{i+1} - \mathbf{V}_i$ and $\mathbf{N}_V = \Delta\mathbf{V}/|\Delta\mathbf{V}|$ define the velocity jump and the unit vector in the direction of the velocity jump respectively. Let L_s designate the length of the subzonal edge that connects the midpoint of the zone edge to the center of the zone. Let $\mathbf{N}_s$ designate the unit normal to this subzonal edge that points from the node designated by i to the node designated by $i+1$. The scalar artificial viscosity, Q, is multiplied by $L_s(\mathbf{N}_s \cdot \mathbf{N}_V)\mathbf{N}_V$ to obtain the vector form of the artificial viscosity $\mathbf{Q}$. Artificial viscosity is only applied when the zone is under compression, so $\mathbf{Q}$ is set identically to zero if $\mathbf{N}_s \cdot \mathbf{N}_V \geq 0$. Otherwise, $-\mathbf{Q}$ is the contribution to the net nodal force at the node designated by i and $\mathbf{Q}$ is the contribution to the net nodal force at the node designated by $i+1$. The contribution to the zone heating rate is $\mathbf{Q} \cdot \mathbf{V}_i - \mathbf{Q} \cdot \mathbf{V}_{i+1}$.

The scalar artificial viscosity, Q, is computed using Eq. (8), replacing Δu with $\Delta\mathbf{V}$ and defining $(\Delta\mathbf{V})^2$ as $\Delta\mathbf{V} \cdot \Delta\mathbf{V}$. The density in Eq. (8) is defined as $(2\rho_i\rho_{i+1})/(\rho_i + \rho_{i+1})$, where the nodal densities, ρ_i and ρ_{i+1}, are calculated by dividing the nodal mass by the nodal area, defining the nodal area as the sum of the areas of all the adjacent subzones.

The sound speed in Eq. (8) is defined as min (c_i, c_{i+1}), where the nodal sound speeds, c_i and c_{i+1}, are computed using the nodal density and the nodal internal energy. The nodal internal energy is calculated by dividing the total internal energy (not per unit mass) of the node by the nodal mass. The total internal energy of a node is calculated by summing the total internal energy of all the neighboring subzones, i.e., summing em_z from each subzone.

ψ is defined using Eq. (9), where $r^- = D^-\mathbf{V}/D\mathbf{V}$ and $r^+ = D^+\mathbf{V}/D\mathbf{V}$. $D\mathbf{V}$ is defined as $\Delta\mathbf{V} \cdot \mathbf{N}_V/\Delta\mathbf{X} \cdot \mathbf{N}_X = |\Delta\mathbf{V}|/|\Delta\mathbf{X}|$, where $\Delta\mathbf{X} = \mathbf{X}_{i+1} - \mathbf{X}_i$ and $\mathbf{N}_X = \Delta\mathbf{X}/|\Delta\mathbf{X}| \cdot D^-\mathbf{V}$ and $D^+\mathbf{V}$ are defined as $\Delta^-\mathbf{V} \cdot \mathbf{N}_V/\Delta^-\mathbf{X} \cdot \mathbf{N}_X$ and $\Delta^+\mathbf{V} \cdot \mathbf{N}_V/\Delta^+\mathbf{X} \cdot \mathbf{N}_X$, respectively, with $\Delta^-\mathbf{V} = \mathbf{V}_i - \mathbf{V}_{i-1}$, $\Delta^+\mathbf{V} = \mathbf{V}_{i+2} - \mathbf{V}_{i+1}$, $\Delta^-\mathbf{X} = \mathbf{X}_i - \mathbf{X}_{i-1}$, and $\Delta^+\mathbf{X} = \mathbf{X}_{i+2} - \mathbf{X}_{i+1}$. Once again, if information is missing, then the corresponding value of r is set identically equal to 1.

Material strength is applied using forces based on springs. While this trivial material strength model lacks several desirable features (e.g., it is anisotropic), it is more than adequate for our purposes. That is, our goal here will be to demonstrate that the interface coupling method can correctly deal with a discontinuous pressure profile across the interface as shown in Fig. 11 of example 2. Every edge of every zone is assigned a rest length, L_o, and every zone is assigned a measure of stiffness, $k \geq 0$. A zone admits a restoring force of $\mathbf{S} = -k(\frac{L}{L_o} - 1)\mathbf{N}_X$ when the current edge length, L, is not equal to its rest length. For each zone adjacent to a particular edge, the restoring force makes contributions of $-\mathbf{S}$ to the net nodal force of the node at $\mathbf{X}_i$, $\mathbf{S}$ to the net nodal force of the node at $\mathbf{X}_{i+1}$, and $\mathbf{S} \cdot \mathbf{V}_i - \mathbf{S} \cdot \mathbf{V}_{i+1}$ to the heating rate of the zone. The straightforward adaption of this material strength model into the numerical algorithm illustrates the power of the general force algorithm proposed in [6], which allows one to construct a compatible numerical algorithm for any set of forces even if they originate only in discrete form.

Equation (10) is used to estimate the time step restriction in two spatial dimensions as well, although some of the terms are defined differently than they are in one spatial dimension. The length of a zone, L, is defined as the minimum of all the edge lengths and the two line segments that connect the midpoints of the opposite edges of the zone. When defining $\hat{c}$, the density ρ is defined as the minimum of the four subzonal densities, and the pressure p is defined as the maximum of the four subzonal pressures. A scalar artificial viscosity is defined on each zone edge using the two-dimensional equivalent of Eq. (8), and then Q is defined as the maximum of the four scalar artificial viscosities in the zone. Finally, a scalar material strength is determined by dividing the magnitude of the material strength on each zone edge by the length of the line segment that connects the midpoint of that edge to the cell center. Then $|S|$ is defined as the maximum of the four scalar material strengths in a zone.

4. TREATING THE INTERFACE

Boundary conditions need to be imposed on both the Eulerian and Lagrangian grids. Standard boundary conditions can be applied everywhere except at the internal boundary where the Lagrangian grid partially overlaps the Eulerian grid. These internal boundary conditions are the main focus of this paper. First the interface itself needs to be defined, and since the Lagrangian grid nodes move at the local material velocity, these nodes can be used to determine the position of the interface. In one spatial dimension, the interface is simply defined as the single Lagrangian boundary node. In two spatial dimensions, a

piecewise linear interface is defined by the Lagrangian mesh lines that connect the nodes on the boundary. This interface divides the Eulerian mesh into separate regions, i.e., a region populated by real grid nodes and a region populated by ghost nodes. Interface boundary conditions for the Eulerian mesh are imposed by defining the conserved variables, i.e., mass, momentum, and energy, in the ghost nodes. Interface boundary conditions for the Lagrangian mesh are imposed by either specifying the velocity of the grid nodes on the Lagrangian boundary or specifying the force applied to that boundary.

Since the interface moves with the local material velocity, it can be treated as a contact discontinuity for the Eulerian calculation. Then the Rankine–Hugoniot jump conditions imply that both the pressure and the normal velocity, $V_N = \mathbf{V} \cdot \mathbf{N}$, are continuous across the interface while both the entropy and the tangential velocities are completely uncoupled across the interface [11]. The interface values of the uncoupled variables can be captured by extrapolating these variables across the interface into the ghost cells. The continuous or coupled variables can be determined using the values from both the Eulerian and the Lagrangian mesh.

The interface normal velocity can be determined by applying any number of interpolation techniques to the Eulerian and Lagrangian mesh values. However, one should be careful to define the interface normal velocity in a way that is consistent with the material in the Lagrangian mesh. That is, perturbations to the velocity of the Lagrangian grid nodes can provide enormous stress due to resistive forces such as material strength. For this reason, to determine an accurate (and Lagrangian mesh consistent) value of the normal velocity at the interface, only the Lagrangian mesh is used to determine the interface velocity, similar to [13, 18, 19]. However, both calculations use this interface normal velocity so that $[V_N] = 0$ is enforced. The Lagrangian mesh simply uses the computed velocities of its boundary nodes, while the Eulerian calculation captures this interface normal velocity by assigning each ghost node the interface normal velocity of the nearest point on the interface.

Since the interface normal velocity is defined as the velocity of the nodes on the Lagrangian mesh boundary with no contribution from the Eulerian mesh, velocity boundary conditions cannot be enforced on the Lagrangian mesh at the interface. Instead, force boundary conditions are applied by interpolating the Eulerian grid pressure to this Lagrangian interface. In this way, the interface pressure is determined using only the Eulerian grid values, ignoring contributions from the Lagrangian mesh as in [13, 18, 19]. However, both calculations use this interface pressure so that $[p] = 0$ is enforced. The interface pressure is captured by the Eulerian calculation by extrapolating the pressure across the interface into the ghost cells in a manner similar to the treatment of entropy and tangential velocity. Then the interface pressure can be interpolated from the Eulerian grid to apply force boundary conditions to the Lagrangian calculation.

Note that [19] suggests that it might be better to use some average of the Lagrangian and Eulerian grid values when determining the pressure at the interface. Actually, for Lagrangian calculations with artificial viscosity and material strength, the jump condition implies that the net stress in the normal direction, not just the pressure, is continuous. Therefore, this averaging procedure should take place between the pressure in the gas and the normal component of the net stress in the normal direction in the solid. However, this can be dangerous, for example, when the Lagrangian material is in tension since near zero or negative stress might be calculated at the interface. While Lagrangian methods can be quite robust under tension, Eulerian methods can suffer a number of problems when treating near zero or negative pressures associated with rarefied or cavitated fluids.

4.1. *One Spatial Dimension*

The one-dimensional interface is defined by the location of the Lagrangian boundary nodes that are adjacent to grid nodes of the Eulerian mesh. This interface location is used to construct a signed distance function in order to apply level set methods [20] near the interface. The level set function is defined at every Eulerian grid node with $\phi \leq 0$ for real grid nodes and $\phi > 0$ for ghost nodes. For each Lagrangian interface node, ϕ is defined analytically as $\phi = \pm(x - x_o)$, where x_o is the location of the node. The "+" sign is used if the Lagrangian mesh lies to the right and the "−" sign is used if the Lagrangian mesh lies to the left. Since this is done for every Lagrangian interface node, ϕ is multiply defined. At each Eulerian grid node, a single value of ϕ is chosen from the possible candidates by choosing the candidate with the minimum magnitude.

Before defining values in the Eulerian ghost nodes, a check is performed to see if enough ghost nodes are present. That is, since the Lagrangian mesh is moving, one needs to ensure that there is adequate overlap between the two meshes. This is done by examining the values of ϕ on the computational boundaries of the Eulerian mesh. If the computational boundary is an Eulerian ghost node, then the value of ϕ gives the distance to the interface and can be used to estimate the number of ghost nodes that exist between the interface and the computational boundary. Then the size of the Eulerian mesh can be increased if there are not enough ghost nodes to successfully apply the numerical method.

The Eulerian ghost nodes are defined by first extrapolating S and p using the fast extension procedure in [2] (based on the fast marching method; see, e.g., [21]), which extends variables to be constant in the normal direction to first-order accuracy. Then u at each ghost node is assigned the value of u at the nearest Lagrangian boundary node that lies on the interface between the Eulerian and Lagrangian grids. Once S, p, and u are determined at each ghost node, the conserved variables are reassembled.

Force boundary conditions are applied to the Lagrangian interface using the pressure from the Eulerian grid. First, the pressure at the interface is determined using linear interpolation form the Eulerian mesh. Note that this linear interpolation requires valid pressure values in both the real and the ghost nodes. Therefore, the pressure extension step (above) needs to be carried out before this linear interpolation step. This Eulerian interface pressure makes a contribution of $\pm p$ to the net force on the Lagrangian boundary node depending on whether the Lagrangian mesh lies to the right or to the left of the interface respectively.

With boundary conditions specified on both the Eulerian and Lagrangian mesh, both can be advanced one Euler step in time. Note that both the Eulerian real grid nodes and a band of Eulerian ghost nodes are advanced in time. These ghost nodes are advanced in time so that they have valid values of the conserved variables in case they are uncovered by the Lagrangian mesh, i.e., in case they become real grid nodes at the end of the time step.

4.2. *Two Spatial Dimensions*

The two-dimensional interface is defined by the line segments of the Lagrangian mesh boundary that are adjacent to grid nodes of the Eulerian mesh. This interface is used to construct a signed distance function defined at every Eulerian grid node with $\phi \leq 0$ for real grid nodes and $\phi > 0$ for ghost nodes. For each Eulerian grid node, the distance to each line segment on the Lagrangian mesh boundary is calculated, and the minimum of these distances is designated as the magnitude of ϕ.

The sign of ϕ is calculated using the closed polygon defined by connecting the Lagrangian interface to the computational boundary of the Eulerian mesh adjacent to the Eulerian ghost nodes. The line segments of this polygon are swept through clockwise (or counterclockwise) to calculate the angle made between the line segment connecting the Eulerian grid node to the first polygonal endpoint of the line segment and the line segment connecting the Eulerian grid node to the second polygonal endpoint of the line segment. If these angles sum to 2π (or -2π) then the Eulerian grid node is inside the polygon; otherwise these angles sum to zero and the Eulerian grid node is outside the polygon. The nodes outside the polygon are real grid nodes with $\phi \leq 0$, while the nodes inside the polygon are ghost nodes with $\phi > 0$. If there is more than one polygon, then this procedure can be used to determine whether a grid node lies within any of the polygons, in which case it is a ghost node with $\phi > 0$. Points that do not lie in any of the polygons are the real grid nodes with $\phi \leq 0$.

This method for constructing ϕ can be optimized by identifying the grid points near the interface and defining ϕ only at those points. Then the $O(N \log N)$ fast marching method [21] can be used to define ϕ elsewhere. Points near the interface can be identified by finding the Cartesian grid intersections of each line segment of the polygonal boundary of the Lagrangian mesh. The complexity of finding these Cartesian grid intersections scales with the number of grid points on the boundary of the Lagrangian mesh and is independent of the size of the Eulerian mesh since that mesh is Cartesian.

Once again, ϕ is examined on the computational boundaries of the Eulerian mesh to ensure that enough ghost nodes are present, and the size of the Eulerian mesh is increased when necessary.

The Eulerian ghost nodes are defined by first extrapolating S, p, and $\mathbf{V}$ using the fast extension procedure in [2]. Then the closest point on the Lagrangian interface is determined by looping through all the line segments that make up this interface. If the closest point happens to be on the end of a linear segment, i.e., a Lagrangian grid node, then that velocity can be designated the closest interface velocity. Otherwise, the closest point is on an edge connecting two Lagrangian grid nodes, and the closest interface velocity is determined using linear interpolation between those two nodes. Designating the closest interface velocity by $\mathbf{V}_I$ and the extrapolated velocity by $\mathbf{V}_{ext}$, the basis free projection method from [11] is used to combine the normal component of the interface velocity with the tangential component of the extrapolated velocity, resulting in a ghost cell velocity of

$$\mathbf{V} = (\mathbf{V}_I \cdot \mathbf{N})\,\mathbf{N} + \mathbf{N}_{ext} - (\mathbf{V}_{ext} \cdot \mathbf{N})\,\mathbf{N}, \tag{11}$$

where the unit normal vector is defined locally at the ghost node as

$$\mathbf{N} = \frac{\nabla\phi}{|\nabla\phi|} \tag{12}$$

and the derivatives are computed with central differencing. In the rare case that the denominator of Eq. (12) is identically zero, the derivatives are computed with one-sided differencing instead of central differencing to obtain a nonzero denominator. Once S, p, and $\mathbf{V}$ have been determined at each ghost node, the conserved variables are reassembled.

Once the Eulerian ghost nodes have valid values for the extrapolated pressure, force boundary conditions can be determined at the Lagrangian interface. The midpoint of each linear interface segment is defined as a control point, and bilinear interpolation is used to determine the Eulerian mesh pressure at each of these control points. Then this pressure

is multiplied by both the length and the inward pointing normal of the line segment to determine the magnitude and direction of the Eulerian pressure force on this segment. Finally, half of this Eulerian pressure force is added to each of the two nodes that make up this segment.

With boundary conditions specified on both the Eulerian and Lagrangian mesh, both can be advanced one Euler step in time. Once again, note that both the Eulerian real grid nodes and a band of Eulerian ghost nodes are advanced in time in case some ghost nodes are uncovered by the Lagrangian mesh.

5. MODIFYING THE ORIGINAL GHOST FLUID METHOD

At this point we pause to consider the implications of the last section, in relation to [11]. Consider a contact discontinuity in two-phase compressible flow where the pressure and normal velocity are continuous, while the entropy and tangential velocities are discontinuous. At the contact discontinuity, the discontinuous variables are multivalued and [11] recommends using one-sided extrapolation into ghost cells to capture the interface values on each side. In [11], the continuous variables are captured using the values already defined at each node; i.e., pressure and normal velocity are copied from the real fluid into the ghost fluid in a node by node fashion. This is in contrast to the method of the last section where one side of the interface (the Eulerian side) determines the interface pressure while the other side of the interface (the Lagrangian side) determines the interface normal velocity.

The interface values of pressure and normal velocity need to be determined using some sort of interpolation technique and noting that these variables are continuous but may posses kinks due to differing equations of state across the interface. Copying these variables into the ghost cells node by node, as proposed in [11], corresponds to one choice of interpolation. Using the fluid on one side of the interface to determine the interface pressure and the fluid on the other side of the interface to determine the interface velocity, as discussed in the last section, corresponds to another choice. Different interpolation techniques lead to $O(\triangle x)$ differences in the interface values of pressure and normal velocity, which vanish as the mesh is refined, guaranteeing convergence as dictated by the Rankine–Hugoniot jump conditions.

At this point, it is not clear exactly which interpolation technique should be used, and the answer is most likely problem related. For smooth well-behaved problems with commensurate equations of state, the method proposed in [11] is probably superior, while the method proposed in the last section is most likely superior when one fluid is very stiff compared to the other.

For example, consider interactions between water and air as discussed in [11] where the air is treated as a gamma law gas and the water is treated with a stiff Tait equation of state. Since the technique in [11] gives equal weighting to the values of the pressure and normal velocity on both sides of the interface, any kinks in these values will be smeared out to some extent, causing small erros in the captured interface values of these variables. Small errors in the normal velocity of the water create small density errors when updating the equation for conservation of mass (the first line in Eq. (1)). In turn, these small density errors can lead to large spurious pressure oscillations in the water, since the Tait equation of state is stiff. While small errors in the velocity of the air cause the same small density errors, these have little effect on the gas, since the gamma law gas equation of state is rather robust (i.e., not stiff). Again, since the Tait equation of state is rather stiff, one can expect large variations in

the pressure of the water near the interface, which in turn can lead to poor predictions of the interface pressure. While these errors in the interface pressure have a relatively small effect on the denser water, they can have a rather large effect on the less dense gas. Conversely, since the gamma law gas equation of state is rather robust, the gas pressure tends to be smooth near the interface and is therefore a good candidate for the interface pressure.

The aforementioned difficulties can be removed in large part by using the water to determine the interface velocity and the air to determine the interface pressure to produce a more robust version of the original ghost fluid method proposed in [11]. When updating the stiffer fluid (in this case the Tait equation of state water), pressure is still copied over node by node in the ghost region while the total velocity and the entropy are extrapolated into the ghost cells. When updating the fluid with the more robust equation of state (in this case the gamma-law gas air), the normal velocity is still copied over node by node in the ghost region while the pressure and the entropy are extrapolated into the ghost cells. This procedure was first used in [5] on an interface separating incompressible and compressible flow. There, the compressible normal velocity is a poor choice for the interface velocity, and the errors this induces into the incompressible velocity field can cause large jumps in the incompressible pressure as this pressure forces the velocity field to be divergence free.

Numerical results have shown that this new method behaves in a fashion similar to the original method in [11], except for the increased interface dissipation, which leads to greater stability. To illustrate this, examples 5 and 6 from [11] are reexamined here, noting that the stiffness in example 6 required some tampering of the high-order numerical method to increase stability (see [11]). Figures 1 and 3 show the results obtained with this new robust method using third-order ENO-LLF and third-order TVD Runge–Kutta [23] without the scheme tampering required with the original scheme.

In example 5 of [11], an interface separates gas on the left from water on the right. Solid wall boundary conditions are enforced on both sides of the domain. Initially, a right-going

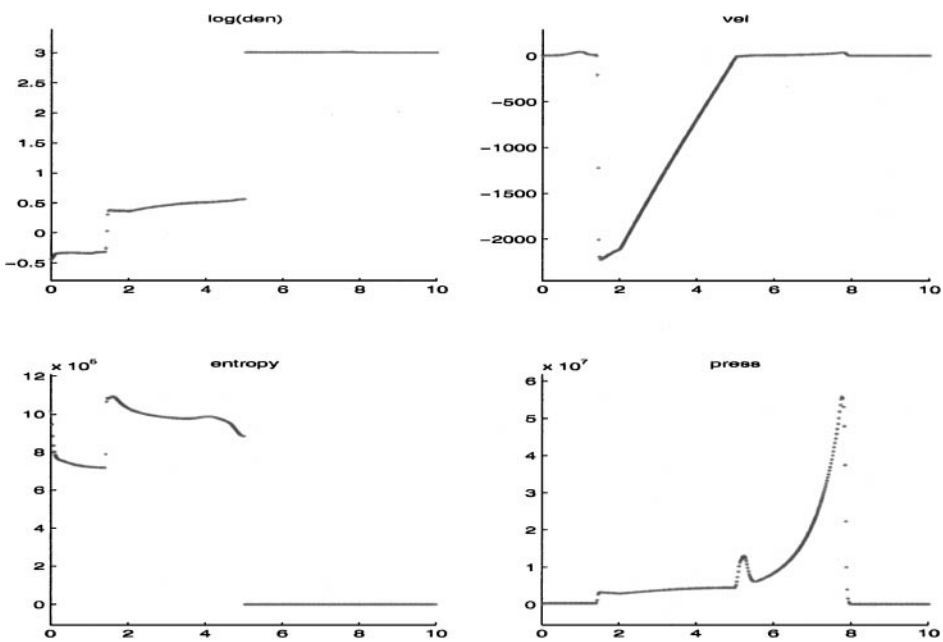

FIG. 1. Gamma law gas ($x < 5$) and water ($x > 5$), with 500 grid cells.

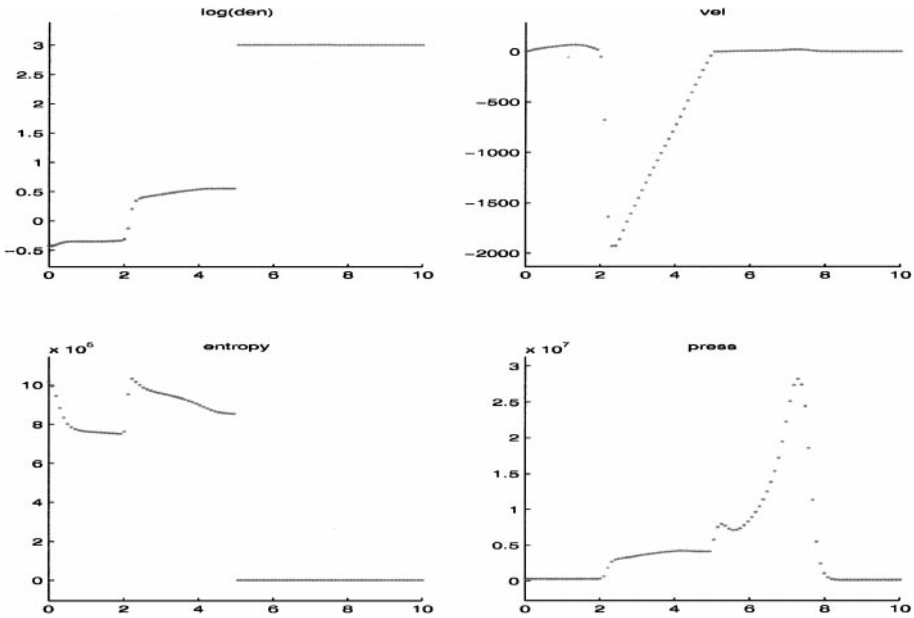

FIG. 2. Gamma law gas ($x < 5$) and water ($x > 5$), with 100 grid cells.

shock wave is located in the gas and a left-going shock wave is located in the water. These shock waves propagate toward the interface, producing a complex wave interaction. In Fig. 1, one can see reflected shock waves traveling outward near $x = 1$ and $x = 8$. Figure 2 shows the same calculation with 100 grid cells as opposed to the 500 grid cells used in Fig. 1. This illustrates the robustness of this new scheme on a coarse grid, especially considering that the scheme in [11] produces spurious cavitation, which leads to failure of the numerical method on this same coarse grid.

In example 6 of [11], interfaces separate gas on the outside of the domain from water on the inside of the domain. A solid wall boundary is enforced on the left and an outflow boundary condition is enforced on the right. Initially, all the fluids are moving to the right at 500 m/s, causing a rarefaction wave to start at the solid wall on the left. This rarefaction wave propagates to the right, slowing down the fluids. Note that it is much easier to slow down the less dense gas as opposed to the denser water. Figure 3 shows the steep pressure profile that forms in the water in an attempt to slow it down. One of the difficulties in [11] was a nonphysical pressure overshoot in the water near the interface on the left. Figure 1 shows that this new numerical method removes the overshoot and produces a monotonic pressure profile near the interface. Figure 4 shows the same calculation with 100 grid cells as opposed to the 400 grid cells used in Fig. 3. This again illustrates the robustness of this new scheme on a coarse grid, especially considering that the scheme in [11] produces spurious cavitation, which leads to failure of the numerical method on this same coarse grid.

6. EXAMPLES

Since both second- and third-order TVD Runge–Kutta schemes [22] can be written as a convex combination of simple Euler steps (see [14, 22]), it is straightforward to generalize

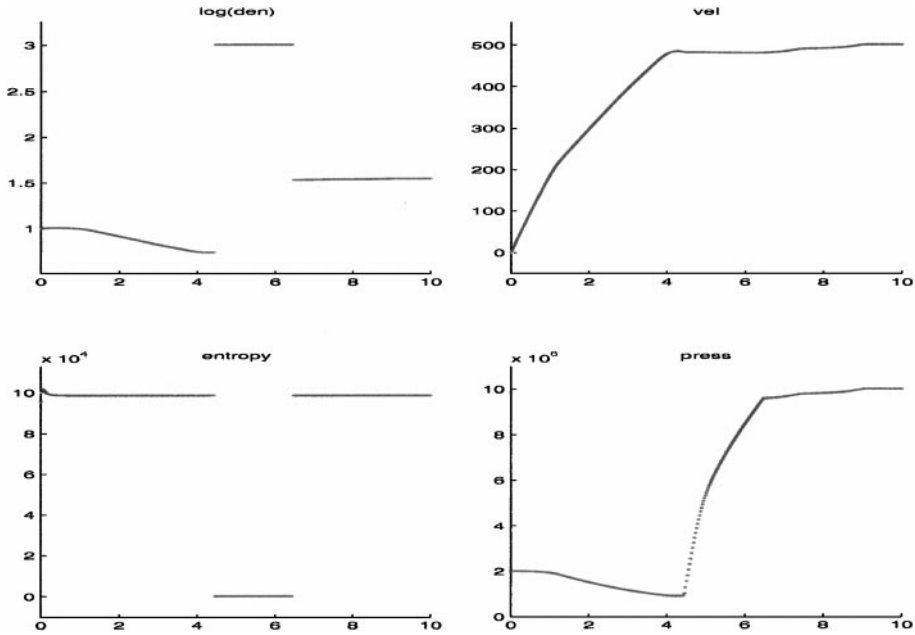

FIG. 3. Gamma law gas ($x < 4.5$; $x > 6.5$) and water ($4.5 < x < 6.5$), with 400 grid cells.

the first-order time discretization discussed so far in this paper to third-order TVD Runge–Kutta, which is the scheme used in the numerical examples. Mass, momentum, and energy are averaged on the Eulerian mesh using ghost cell values where necessary, while position, velocity, and internal energy are averaged on the Lagrangian mesh.

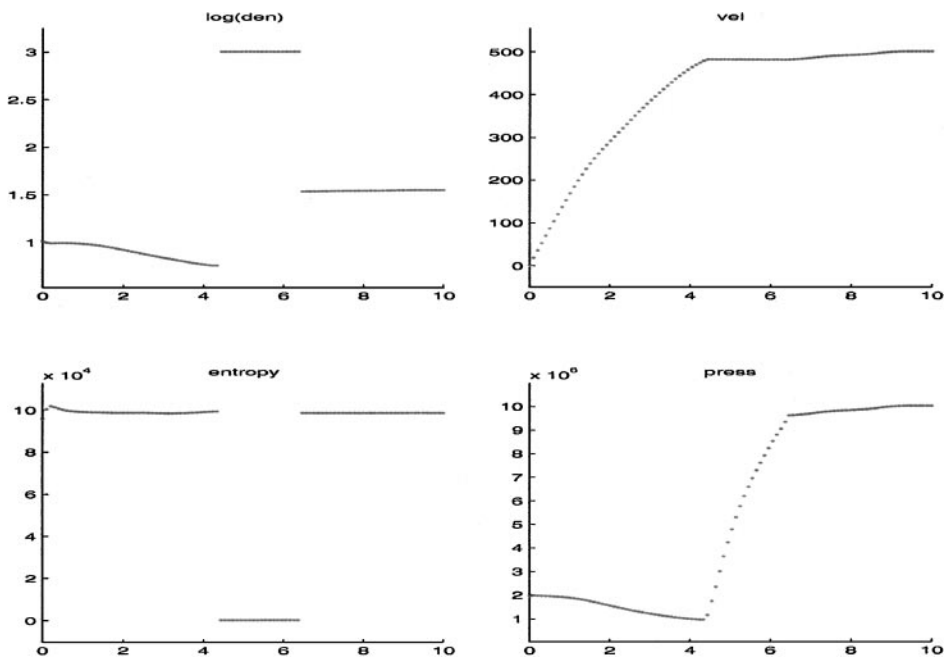

FIG. 4. Gamma law gas ($x < 4.5$; $x > 6.5$) and water ($4.5 < x < 6.5$), with 100 grid cells.

Adaptive time stepping is used in which the overall time step is the minimum of the Eulerian mesh and Lagrangian mesh time steps; i.e.,

$$\Delta t = 0.5 \min(\Delta t^E, \Delta t^L), \tag{13}$$

where we have chosen a CFL restriction of 0.5.

The Lagrangian artificial viscosity was applied with $c_1 = 1$ and $c_2 = 1$ as recommended in [9].

Note that high-resolution methods (e.g., third-order-accurate ENO for the Euler equations and third-order-accurate Runge–Kutta for time discretization) are used in the numerical examples. Even though these schemes degrade to first-order accuracy near discontinuiuties such as shock waves, the lower numerical truncation error gives better results in smooth regions of the flow. In general, the numerical results tend to be first-order accurate in quantities such as the location of the Eulerian/Lagrangian interface.

6.1. *Example 1*

In this example we compute solutions to Test B, Test C, and the two cases of Test D that were first proposed and solved in [15] and later solved in [11] using the fully Eulerian version of the ghost fluid method for two-phase flows. In Test B a shock wave impinges upon an interface producing a transmitted shock wave and a reflected rarefaction wave, while in Test C the same shock wave produces both transmitted and reflected shock waves. The two cases of Test D are similar to Test B and Test C except for having a stronger initial shock wave.

All tests are computed on $\alpha[0\text{ m}, 1\text{ m}]$ domain with the interface located in the center of the domain at $x = 0.5$ m. A fixed Eulerian mesh initially containing 200 grid points is used on the left-hand side of the interface, while a moving Lagrangian mesh containing 200 grid points is used on the right-hand side of the interface. Note that the exact solutions for density, velocity, and pressure are displayed by a solid line in the figures for the sake of comparison.

Three fluids are used in the study and each initially starts with $u = 0$ m/s and $p = 1 \times 10^5$ Pa. Fluid 1 has $\gamma = 1.4$ and $\rho = 1\text{ kg/m}^3$, fluid 2 has $\gamma = 1.67$ and $\rho = 0.1379\text{ kg/m}^3$, and fluid 3 has $\gamma = 1.249$ and $\rho = 3.1538\text{ kg/m}^3$.

6.1.1. Test B. In Test B, fluid 1 is on the left and fluid 2 is on the right. A right-going shock wave is originally located at $x = 0.05$ m with a post-shock state of $\rho = 1.3333\text{ kg/m}^3$, $\rho = 1.5 \times 10^5$ Pa, and $u = 0.3535\sqrt{10^5}$ m/s. Figure 5 shows the computed solution at a final time of 0.0012 s. There is a small (barely noticeable) glitch in density near $x = 0.2$ due to start-up errors that are generated when the exact initial shock profile is resolved by the shock-capturing scheme.

Figure 6 shows the results with fluid 2 on the left and fluid 1 on the right with a left-going shock wave initially located at $x = 0.95$ m (of course the post-shock velocity is then $u = -0.3535\sqrt{10^5}$ m/s. Note that the start-up errors in density near $x = 0.8$ are significantly worse for the Lagrangian scheme. Also note that there are some low-amplitude pressure and velocity waves near the interface. These low-amplitude waves seem to be related to start-up errors and are caused by the changes in the numerical shock profile as the shock wave moves from one grid to the other, especially since different numerical schemes are used on the different grids. In general, these low-amplitude waves seem to be significantly worse for shocks crossing from the Lagrangian grid to the Eulerian grid than they are for shocks crossing from the Eulerian grid to the Lagrangian grid. This is fortuitous since strong

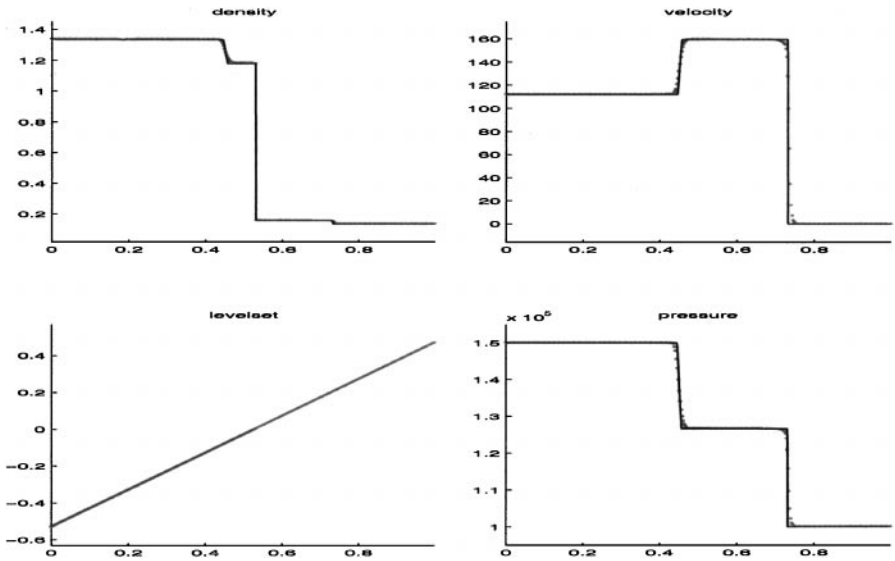

FIG. 5. Test B: A shock wave propagates from the Eulerian grid (left) towards the Lagrangian grid (right) producing both reflected and transmitted waves when it hits the multimaterial interface in between the grids.

shocks usually form in highly deformable reactive materials that are best modeled with an Eulerian scheme, and one is interested in the effect these shocks have on inert materials with strengths that are best modeled with a Lagrangian scheme; i.e., in the physical problems of interest the strongest shocks tend to travel from the Eulerian grid to the Lagrangian grid.

6.1.2. Test D, case 1. The first case of Test D is similar to Test B except that the shock strength is increased using a post-shock state of $\rho = 4.3333\ \mathrm{kg/m^3}$, $p = 1.5 \times 10^6$ Pa, and $u = 3.2817\sqrt{10^5}$ m/s. The results for the right-going shock are plotted in Fig. 7 at a final

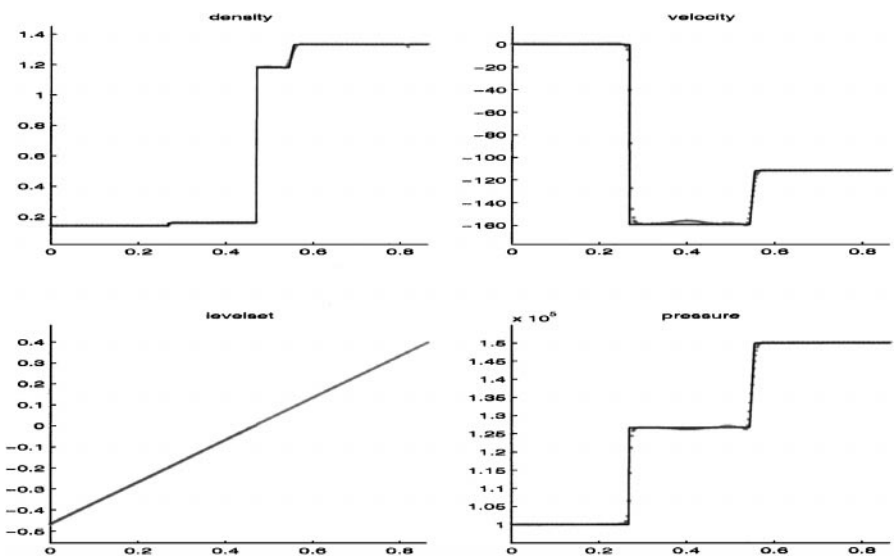

FIG. 6. Test B: A shock wave propagates from the Lagrangian grid (right) toward the Eulerian grid (left) producing both reflected and transmitted waves when it hits the multimaterial interface.

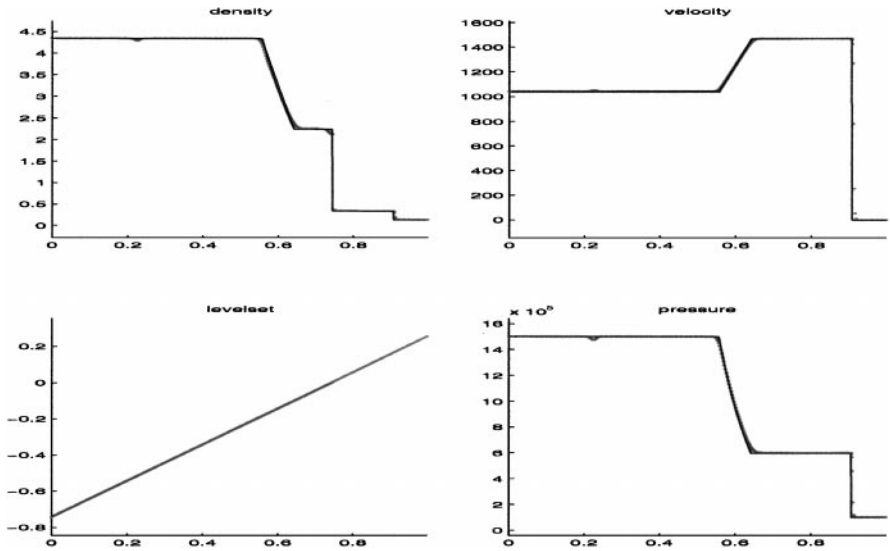

FIG. 7. Test D, case 1: similar to test B, except a stronger shock wave propagates from the Eulerian grid (left) toward the Lagrangian grid (right) impinging upon the interface.

time of 0.0005 s. Note that the errors in all variables near $x = 0.2$ are start-up errors. Note too that there are small "overheating" errors on the left-hand side of the interface.

6.1.3. Test C. Test C is similar to Test B except that fluid 2 is replaced with fluid 3. The results for the right-going shock are plotted in Fig. 8 at a final time of 0.0017 s. While the start-up errors are negligible, the numerical method seems to have some difficulty with the reflected shock wave. Although the reflected shock wave is in the correct spatial location, there are low-amplitude waves in pressure and velocity to the right of this wave.

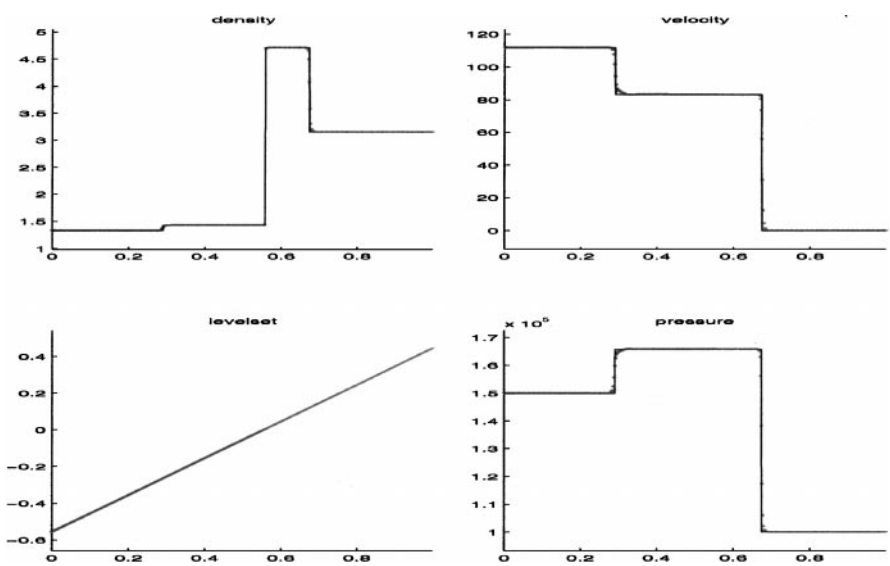

FIG. 8. Test C: similar to test B, except a different material is used on the Lagrangian grid to the right of the interface. The initial shock wave is again moving from left to right.

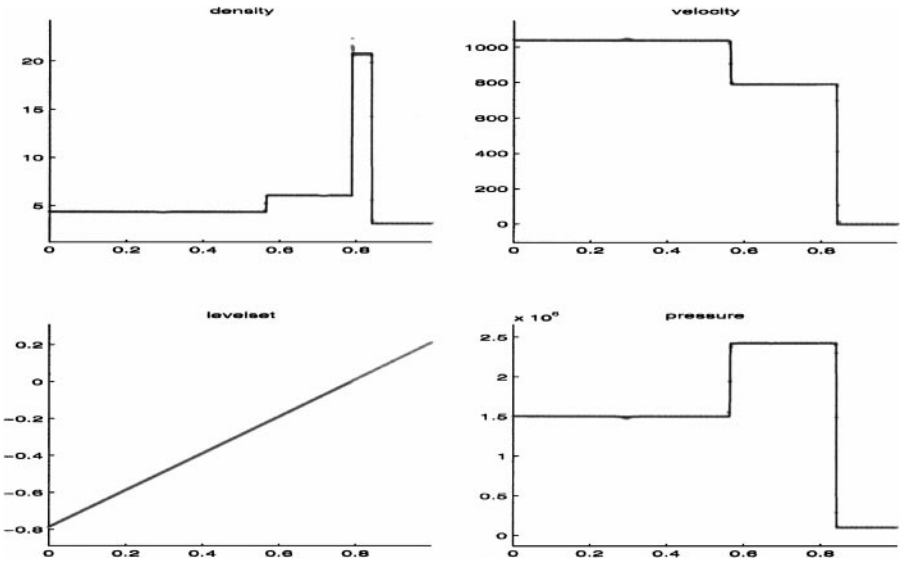

FIG. 9. Test D, case 2: similar to test C, except a stronger shock wave propagates from the Eulerian grid (left) toward the Lagrangian grid (right) impinging upon the interface.

6.1.4. Test D, case 2. The second case of Test D is similar to the first case except fluid 2 is replaced with fluid 3. The results for the right-going shock are plotted in Fig. 9 at a final time of 0.0007 s. The errors near $x = 0.3$ and $x = 0.7$ are start-up errors, while the errors in density to the right of the interface are overheating errors. To illustrate the behavior of the scheme under grid refinement, Fig. 10 shows the computed results under one level of grid refinement. Note that the overheating errors improve in the L^2 norm but not in the L^∞ norm.

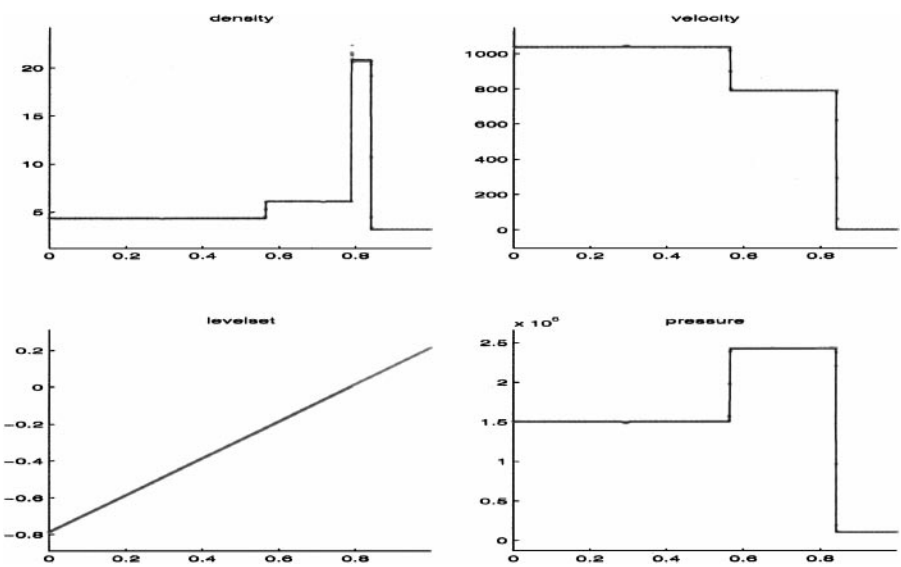

FIG. 10. Test D, case 2: after one level of grid refinement.

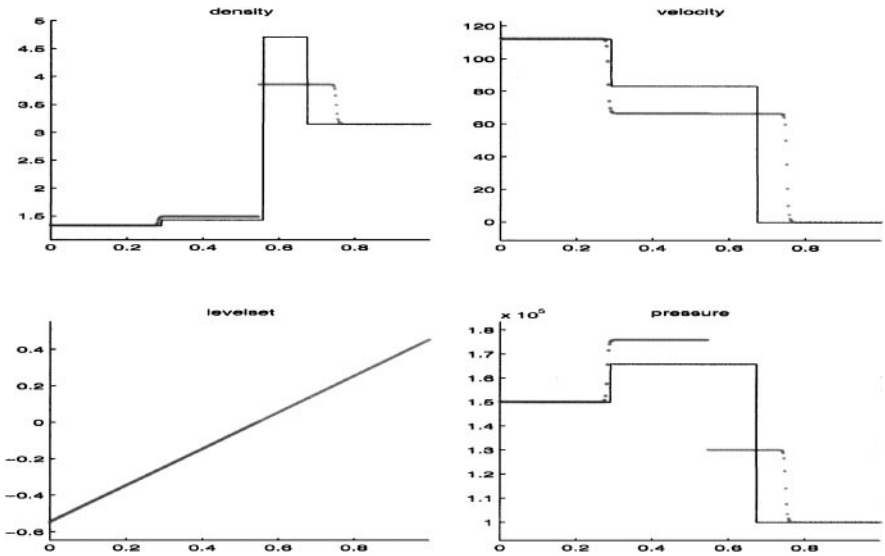

FIG. 11. Test C with material strength in the Lagrangian fluid (right). The material strength is balanced at the interface by the jump in pressure producing a continuous normal stress. The solid line shows the exact solution to this problem without material strength.

6.2. *Example 2*

In this example, we repeat the right-going shock case from Test C of example 1, except that material strength is added to the material on the right using $k = 2.5 \times 10^5$ N/m. The computed results are shown in Fig. 11 at a final time of 0.0017 s. For comparison, the exact solution for Test C *without* material strength is also shown in the figure. Note that there is a jump in pressure at the interface. The higher pressure in the fluid on the left is needed to balance the material strength expansion force of the compressed material on the right and produce a continuous normal stress across the interface. To show the behavior of the computed solution under grid refinement, Fig. 12 shows the computed results under one level of grid refinement.

6.3. *Example 3*

Here we consider Test C in two spatial dimensions. Consider a rectangular domain of size $[0,\ 1] \times [0,\ 0.25]$ with initial conditions for Test C specified in the x direction and constant initial data in the y direction. The interface is at $x = 0.5$ m and the initial 100×50 grid point Eulerian mesh is to the left of the interface while the 100×50 grid point Lagrangian mesh is to the right of the interface. While the left and right boundaries of the computational domain are unaffected, the top and bottom boundaries need to have boundary conditions specified. Constant extrapolation of all variables is used to fill fictitious ghost cells on the top and bottom of the Eulerian mesh, whereas the top and bottom of the Lagrangian mesh are treated with a fixed velocity boundary condition that forces the edge velocity to be equal to the velocity of the closest nonedge node. For example, the velocity at (i, n) on the top of the Lagrangian mesh is set equal to the velocity at $(i, n-1)$, while the velocity at $(i, 1)$ on the bottom of the Lagrangian mesh is set equal to the velocity at $(i, 2)$.

Figure 13 shows the pressure at a final time of 0.0017 s. One can see that the solution stays one dimensional as it should. This is an important test, since many Lagrangian calculations

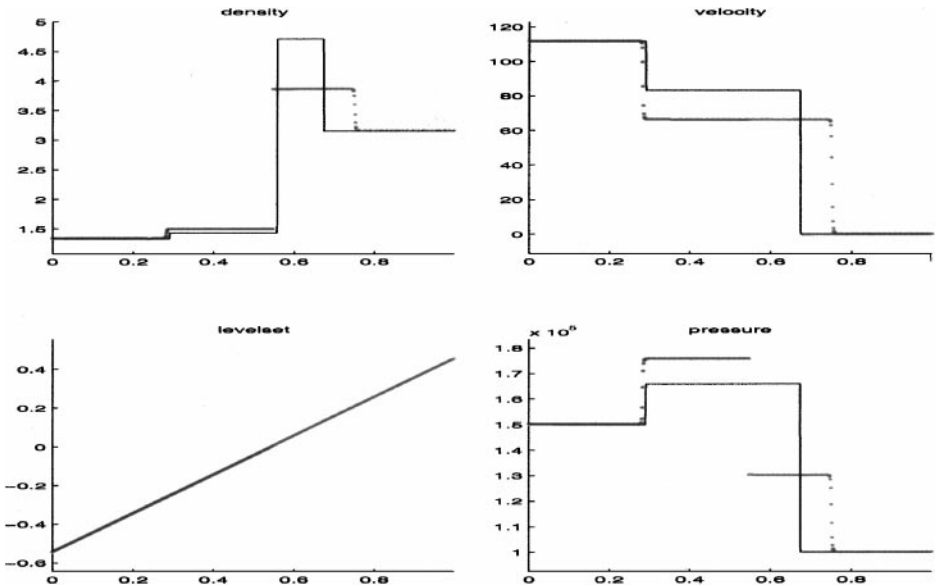

FIG. 12. Test C with material strength in the Lagrangian fluid (right) after one level of grid refinement.

break down and become multidimensional (although Eulerian calculations tend to stay one dimensional). Furthermore, this test shows that our interface treatment allows the calculation to stay one dimensional as well. Figure 14 shows a side view of the same calculation. For the most part, the data in the y direction are uniform and one can only see the edge of the grid in this side view. Note that the exact solutions for density, velocity, and pressure are displayed by a solid line in the figures for the sake of comparison.

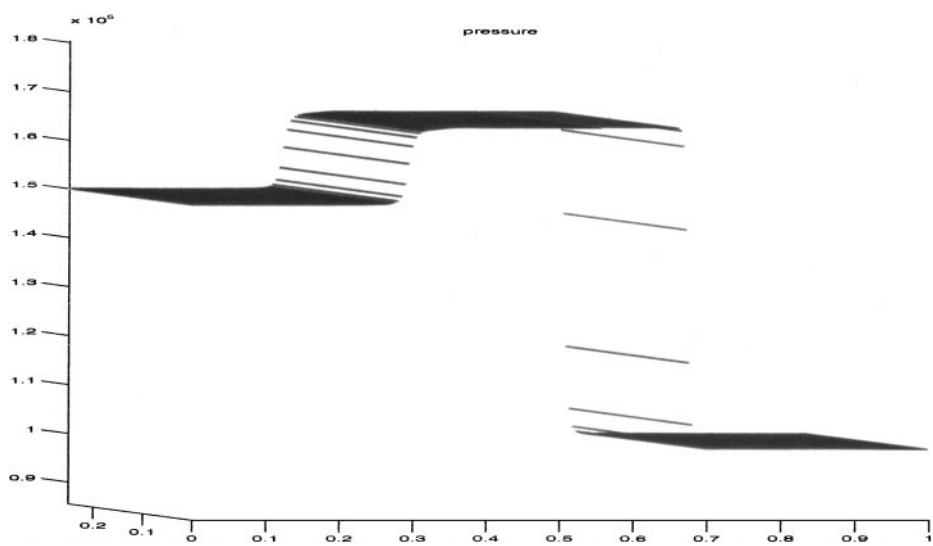

FIG. 13. Test C in two spatial dimensions: A shock wave propagates from the Eulerian grid (left) toward the Lagrangian grid (right) producing both reflected and transmitted waves when it hits the multimaterial interface in between the grids.

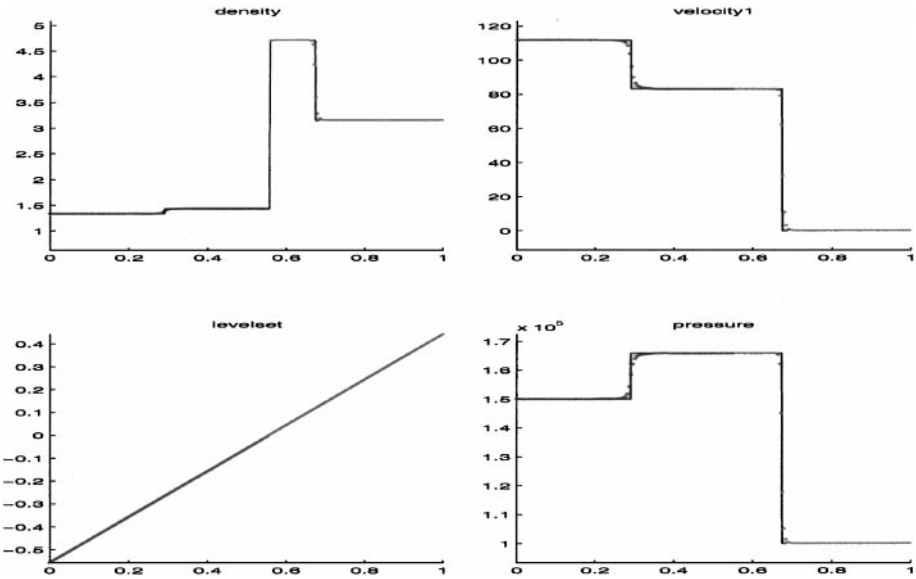

FIG. 14. Test C in two spatial dimensions (side view).

6.4. *Example 4*

Consider a rectangular domain of size $[0, 1] \times [0, 0.75]$ divided into three regions by the two lines $y = 0.25$ and $y = 0.5$. The regions with $y \leq 0.25$ and $y \geq 0.5$ are modeled with separate Lagrangian meshes and filled with fluid 3 with added material strength set by $k = 2.5 \times 10^5$ N/m. The region in between the two Lagrangian meshes is modeled with an Eulerian mesh and filled with fluid 1. Similar to Test D, fluid 1 contains a right-going shock wave initially located at $x = 0.05$ m with a post-shock state of $\rho = 4.3333$ kg/m^3, $p = 1.5 \times 10^6$ Pa, and $u = 3.2817\sqrt{10^5}$ m/s. The Lagrangian mesh initially located in the

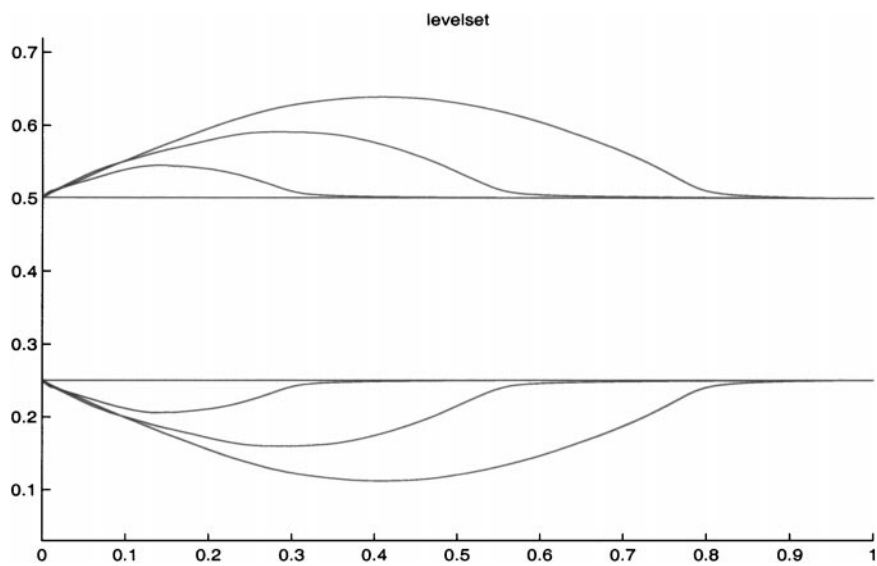

FIG. 15. Interface location at $t = 0$, 0.0002, 0.0004, and 0.0006 s.

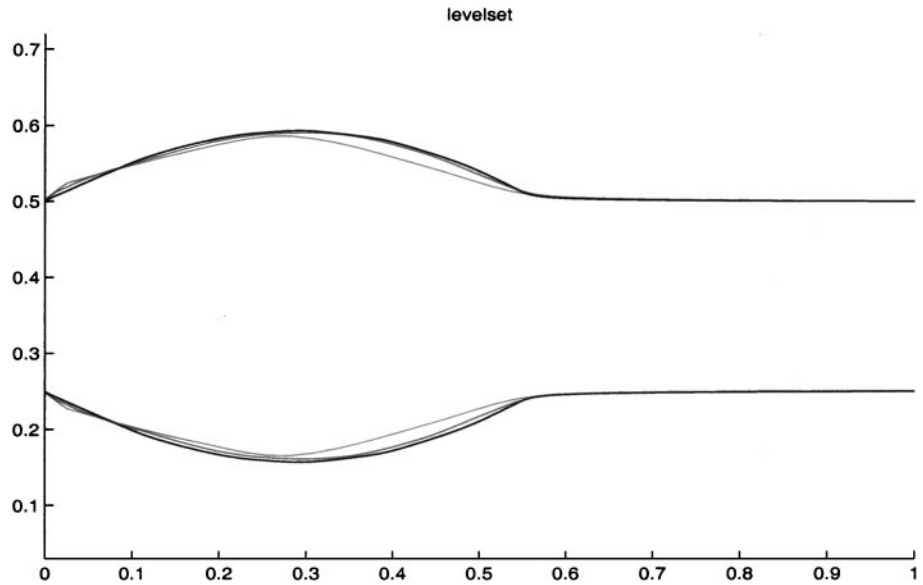

FIG. 16. Interface location at $t = 0.0004$ s for three different grids.

region defined by $y \leq 0.25$ has a fixed zero-velocity boundary condition applied to the left, right, and bottom edges while the other Lagrangian mesh has the same boundary condition applied to the left, right, and top edges. The post-shock state is used to apply a fixed inflow boundary condition to the left-hand side of the Eulerian mesh while constant extrapolation is applied to all variables on the right-hand side of the Eulerian mesh.

The calculation is carried out using an initial Eulerian grid of 100×25 grid points and Lagrangian grids of 100×25 grid points each. Figure 15 shows the location of the

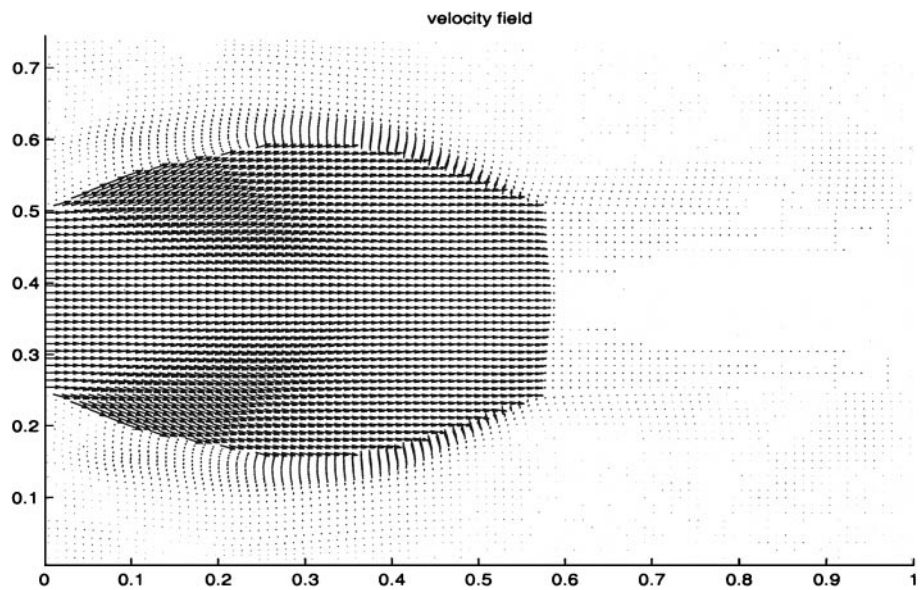

FIG. 17. Velocity field at $t = 0.0004$ s.

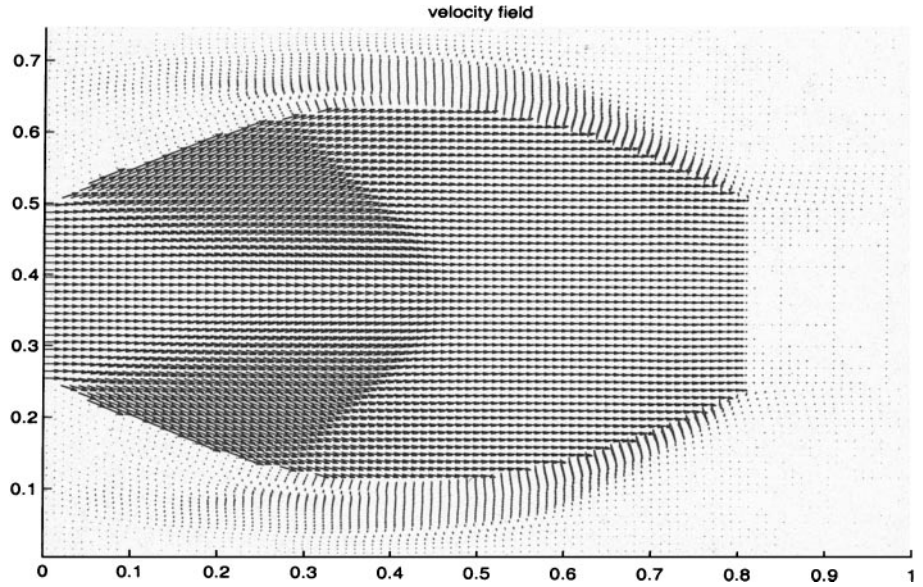

FIG. 18. Velocity field at $t = 0.0006$ s.

Eulerian/Lagrangian interface at times $t = 0, 0.0002,\ 0.0004$, and 0.0006 s. Figure 16 shows the interface location at $t = 0.0004$ s on Eulerian grids of 40×10, 80×20, and 160×40 grid points with corresponding Lagrangian grids of the same size. Figures 17 and 18 show the velocity field at $t = 0.0004$ and $t = 0.0006$ s respectively. To illustrate the effect of material strength, Figs. 19 and 20 show the interface location and the velocity field at $t = 0.0004$ s for the same calculation without material strength, i.e., with $k = 0$.

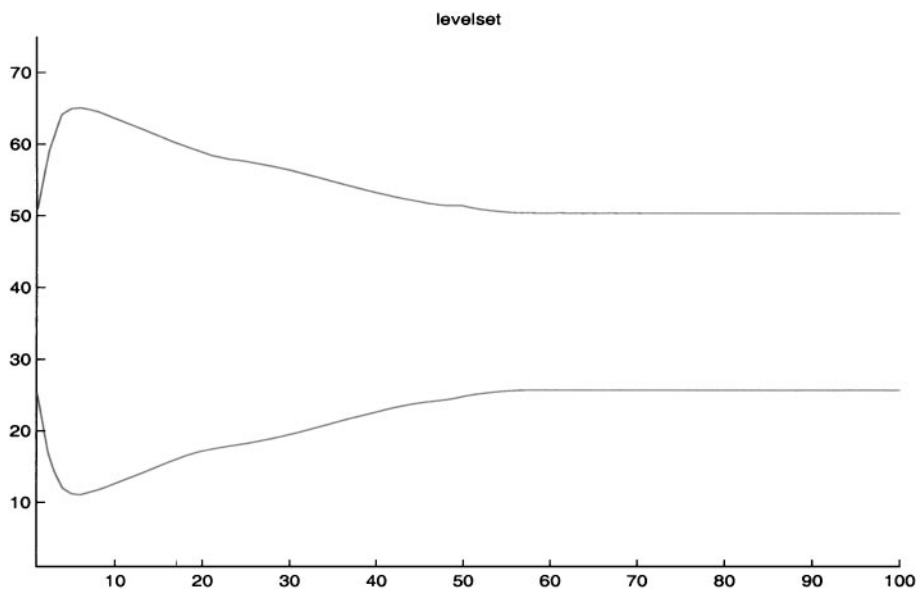

FIG. 19. Interface location at $t = 0.0004$ s.

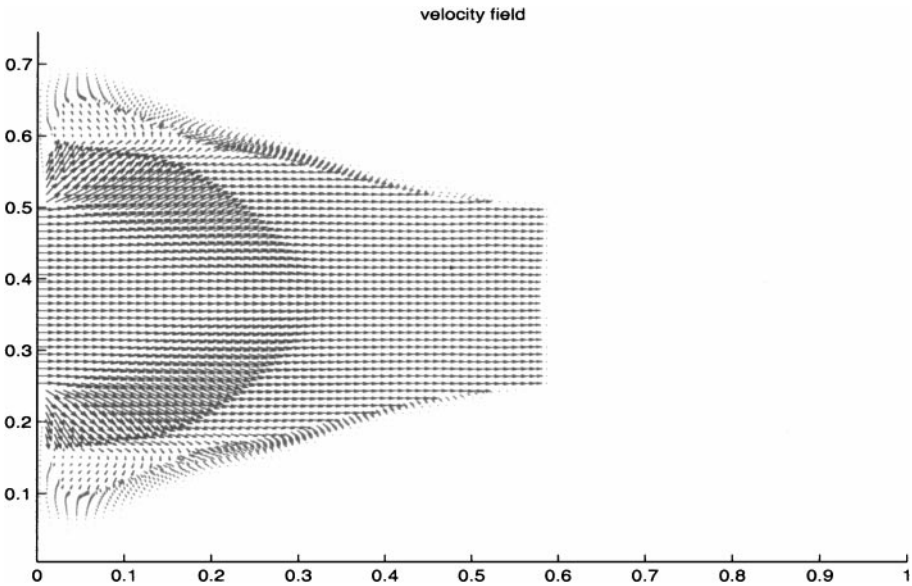

FIG. 20. Velocity field at $t = 0.0004$ s.

7. CONCLUSION

We have developed a novel method for coupling an Eulerian fluid model to a Lagrangian solid model and shown that it performs well on a number of test cases. While this paper models the solid material using the Lagrangian formulations from [6, 8, 9], this is not a limitation of the interface coupling method proposed in this paper. For example, Ref. [1] successfully uses the method first proposed in this paper with a highly sophisticated Lagrangian material model. In general, any Lagrangian formulation could be employed without significant modification of the interface coupling method. This is important since both the literature and the commercial world of Lagrangian codes contain a wide variety of Lagrangian models (elastic, plastic, etc.) for materials that undergo all manner of deformation and failure.

The interface coupling techniques developed here allowed us to more accurately treat coupling with stiff materials. Two rather difficult examples from [11] were redone here with superior results. Moreover, they were computed on a 4–5 times coarser mesh without jepordizing stability.

ACKNOWLEDGMENTS

This work was carried out in part while the author participated in the California Institute of Technology Center for Simulation of Dynamic Response of Materials, and he benefited from extensive interactions with the faculty, students, and staff of the center. In particular, Dan Meiron, Joe Shepherd, Michael Ortiz, Michael Aivazis, Eric Morano, Patrick Hung, Sean Mauch, and Marco Arienti worked with the author in exploring the application of the ghost fluid method to coupling disparate solution methods used in fluid and solid mechanics. A summary of the center's "Virtual Test Facility" can be found in [1].

REFERENCES

1. M. Aivazis, W. Goddard, D. Meiron, M. Ortiz, J. Pool, and J. Shepherd, A virtual test facility for simulating the dynamic response of materials, *Comput. Sci. Eng.* **2**(2), 42 (2000).

2. D. Adalsteinsson and J. A. Sethian, The fast construction of extension velocities in level set methods, *J. Comput. Phys.* **148**, 2 (1999).
3. D. Benson, Computational methods in Lagrangian and Eulerian hydrocodes, *Comput. Methods Appl. Mech. Eng.* **99**, 235 (1992).
4. D. Benson, A new two-dimensional flux-limited shock viscosity for impact calculations, *Comput. Methods Appl. Mech. Eng.* **93**, 39 (1991).
5. R. Caiden, R. Fedkiw, and C. Anderson, A numerical method for two phase flow consisting of separate compressible and incompressible regions, *J. Comput. Phys.* **166**, 1 (2001).
6. E. J. Caramana, D. E. Burton, M. J. Shashkov, and D. E. Burton, The construction of compatible hydrodynamics algorithms utilizing conservation of total energy, *J. Comput. Phys.* **146**, 227 (1998).
7. E. J. Caramana, C. L. Rousculp, and D. E. Burton, A compatible, energy and symmetry preserving Lagrangian hydrodynamics algorithm in three-dimensional Cartesian geometry, *J. Comput. Phys.* **157**, 89 (2000).
8. E. J. Caramana and M. J. Shashkov, Elimination of artificial grid distortion and hourglass-type motions by means of Lagrangian subzonal masses and pressures, *J. Comput. Phys.* **142**, 521 (1998).
9. E. J. Caramana, M. J. Shashkov, and P. P. Whalen, Formulations of artificial viscosity for multi-dimensional shock wave computations, *J. Comput. Phys.* **144**, 70 (1998).
10. E. J. Caramana and P. P. Whalen, Numerical preservation of symmetry properties of continuum problems, *J. Comput. Phys.* **141**, 174 (1998).
11. R. Fedkiw, T. Aslam, B. Merriman, and S. Osher, A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method), *J. Comput. Phys.* **152**, 457 (1999).
12. R. Fedkiw, B. Merriman, R. Donat, and S. Osher, The penultimate scheme for systems of conservation laws: Finite difference ENO with Marquina's flux splitting, in *Progress in Numerical Solutions of Partial Differential Equations*, edited by M. Hafez (Arachon, France, 1998).
13. S. Hancock, PISCES 2DELK Theoretical Manual (Physics Int., 1985).
14. M. Kang, R. Fedkiw, and X.-D. Liu, A boundary condition capturing method for multiphase incompressible flow, *J. Sci. Comput.* **15**, 323 (2000).
15. S. Karni, Hybrid multifluid algorithms, *SIAM J. Sci. Comput.* **17**(5), 1019 (1996).
16. G. Lapenta and J. U. Brackbill, Immersed boundary method for plasma simulation in complex geometries, *IEEE Trans. Plasma Sci.* **24**, 105 (1996).
17. L. Margolin and M. Shashkov, Using a curvilinear grid to construct symmetry-preserving discretizations for Lagrangian gas dynamics, *J. Comput. Phys.* **149**, 389 (1999).
18. McMaster, W. H., Computer codes for fluid-structure interactions, *Proc. 1984 Pressure Vessel and Piping Conference, San Antonio, TX*, LLNL UCRL-89724, 1984.
19. W. F. Noh, CEL: A Time-Dependent, Two Space Dimensional, Coupled Eulerian–Lagrange Code, Methods in Computational Physics, Vol. 3, Fundamental Methods in Hydrodynamics (Academic Press, New York, 1964), pp. 117–179.
20. S. Osher and J. A. Sethian, Fronts propagating with curvature dependent speed: Algorithms based on Hamilton–Jacobi formulations, *J. Comput. Phys.* **79**, 12 (1988).
21. J. A. Sethian, Fast marching methods, *SIAM Rev.* **41**, 199 (1999).
22. C. W. Shu and S. Osher, Efficient implementation of essentially non-oscillatory shock capturing schemes, *J. Comput. Phys.* **77**, 439 (1988).
23. C. W. Shu and S. Osher, Efficient implementation of essentially non-oscillatory shock capturing schemes II, *J. Comput. Phys.* **83**, 32 (1989).
24. D. Sulsky and J. U. Brackbill, A numerical method for suspension flow, *J. Comput. Phys.* **96**, 339 (1991).