Physica D 60 (1992) 84–93
North-Holland

PHYSICA D

A 3D adaptive mesh refinement algorithm for multimaterial gas dynamics

E.G. Puckett [a] and J.S. Saltzman [b]

[a] *Department of Mathematics, and Institute of Theoretical Dynamics, University of California, Davis, CA 95616, USA*
[b] *Computer Research Group (C-3), Los Alamos National Laboratory, Los Alamos, NM 87545, USA*

Adaptive mesh refinement (AMR) in conjunction with high order upwind finite difference methods has been used effectively on a variety of problems. In this paper we discuss an implementation of an AMR finite difference method that solves the equations of gas dynamics with two material species in three dimensions. An equation for the evolution of volume fractions augments the gas dynamics system. The material interface is preserved and tracked from the volume fractions using a piecewise linear reconstruction technique.

1. Introduction

Adaptive techniques offer the advantage of resolving important phenomena while minimizing the use of machine resources. In this paper we describe an efficient adaptive finite difference algorithm for a 3D, compressible, multimaterial, inviscid fluid. We develop ideas based on Adaptive Mesh Refinement algorithms (AMR) pioneered by Berger and Oliger [4] and fluid interface reconstruction techniques, based on volume fractions, first used by DeBar [6].

AMR is used in conjunction with finite difference schemes to selectively resolve regions of the solution domain. The savings in only resolving the necessary parts of the domain can be significant. As an example, if the 2D multifluid calculations in Henderson et al. [7] were performed on a uniform mesh, the computations would have taken several hours instead of approximately ten minutes. The volume fraction description of a fluid carries an additional quantity along with density, momentum, and energy of each species. This quantity is the volume fraction describing how much of a given volume (a computational cell volume for finite volume methods) is occupied by a given material. The advantage of using volume fractions is their robustness. The interfaces are locally constructed from the fractions on a cell by cell basis eliminating the need for complex data structures and the complex logic associated with some tracking methods.

The outline of the paper follows. After this introduction we write the equations modeling a multispecies gas. The interface reconstruction techniques studied and implemented are described in the next section. A surprising difficulty found when testing various reconstruction techniques is the problems encountered by the SLIC reconstruction algorithm when applied to rotational flows. The section on 3D AMR follows next. Here we briefly outline the major components of the AMR algorithm. Finally we outline and solve a test problem. We simulate a flow that develops the Richtmyer–Meshkov instability. The test problem demonstrates the effectiveness of the error estimation and adaptivity on even a small three-dimensional problem.

2. Multimaterial gas dynamics

In this section we describe a multimaterial gas dynamics model used within the AMR algo-

rithm. The model was developed by Colella and Glaz [5]. Our presentation will be limited to the physical model and will not outline the details of a practical implementation. Our derivation is equivalent to [5] but arrives at the same results along a different path. We start with two basic assumptions about the fluid that covers a variety of flow regimes. The following two assumptions plus the usual thermodynamic relations and conservation relations leads to the model:

(1) A single velocity field describes the movement of materials.

(2) Material interfaces are and remain in pressure equilibrium.

By having a single velocity field (assumption 1) the gas dynamics equations for a single material can be used to find the velocity field everywhere if an appropriate equation of state can be found. We then begin, *away from material interfaces*, for a species α, with the well known equations of gas dynamics in Lagrangian form

$$\frac{\mathrm{d}\rho_\alpha}{\mathrm{d}t} + \rho_\alpha \nabla \cdot \boldsymbol{u} = 0,$$

$$\rho_\alpha \frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t} + \nabla p_\alpha = 0,$$

$$\rho_\alpha \frac{\mathrm{d}E_\alpha}{\mathrm{d}t} + \nabla \cdot (p_\alpha \boldsymbol{u}) = 0,$$

where

$$E_\alpha = e_\alpha + \tfrac{1}{2}u^2, \quad p_\alpha = p_\alpha(e_\alpha, \rho_\alpha).$$

For species α, ρ_α is the density, p_α is the pressure, e_α is the internal energy, E_α is the total energy, and $\boldsymbol{u}$ is the velocity. The differential operator $\mathrm{d}/\mathrm{d}t$ is the Lagrangian derivative. Also for each species α a dimensionless scalar is defined as

$$\Gamma_\alpha = \frac{\rho_\alpha c_\alpha^2}{p_\alpha},$$

where c_α is the sound speed for the species α. For each material

$$\frac{1}{p_\alpha}\frac{\mathrm{d}p_\alpha}{\mathrm{d}t} = -\frac{\Gamma_\alpha}{V_\alpha}\frac{\mathrm{d}V_\alpha}{\mathrm{d}t},$$

where Γ_α describes the compressibility of the fluid α and V_α its volume. For the treatment given below one should think of the volumes V_α as the Jacobian transformations of initial configurations to final configurations. The larger a material's Γ_α, the harder it is to compress the material. Also away from material interfaces

$$\nabla \cdot \boldsymbol{u} = \frac{1}{V_\alpha}\frac{\mathrm{d}V_\alpha}{\mathrm{d}t} = \frac{1}{V}\frac{\mathrm{d}V}{\mathrm{d}t},$$

where V is the total volume. We now introduce a new quantity called the volume fraction as there is insufficient information given from individual species densities to completely describe a fluid's structure. Let $\chi_\alpha(\boldsymbol{x}, t)$ be the characteristic function of a species α at time t and point $\boldsymbol{x}$. The volume fraction of a material species α is defined as

$$f_\alpha(\boldsymbol{x}, t) = \frac{V_\alpha}{V} = \lim_{\mathrm{Vol}(\Omega)\to 0} \frac{\int_\Omega \chi_\alpha(\boldsymbol{x}, t)\,\mathrm{d}\Omega}{\int_\Omega \mathrm{d}\Omega}$$

where the region Ω shrinks around the point $\boldsymbol{x}$ at a given time t. The definition is analogous to density in the sense that the limiting process is finite so that a large number of molecules is encompassed by the volume yet is is small enough to contain a nearly constant distribution. From the definition it is clear that

$$1 = \sum_\alpha f_\alpha(\boldsymbol{x}, t), \quad V = \sum_\alpha V_\alpha.$$

Given individual volume fractions the total density and energy of a mixed fluid is

$$\rho = \sum_\alpha f_\alpha \rho_\alpha, \quad E = \sum_\alpha f_\alpha \rho_\alpha E_\alpha.$$

By assumption 2 the pressure of the mixed fluid is the pressure of any of the components. The *preservation* of pressure equilibrium leads to the following relations:

$$\frac{1}{V}\frac{\mathrm{d}V}{\mathrm{d}t} = \sum_\alpha \frac{1}{V}\frac{\mathrm{d}V_\alpha}{\mathrm{d}t} = -\sum_\alpha \frac{V_\alpha}{V p_\alpha \Gamma_\alpha}\frac{\mathrm{d}p_\alpha}{\mathrm{d}t}$$

$$= -\frac{1}{p}\frac{\mathrm{d}p}{\mathrm{d}t}\sum_\alpha \frac{f_\alpha}{\Gamma_\alpha}. \qquad (1)$$

Using the first and last part of the previous relation gives an effective Γ for the composite fluid as

$$\Gamma = \left(\sum_\alpha \frac{f_\alpha}{\Gamma_\alpha}\right)^{-1}.$$

The effective Γ along with the pressure equilibrium assumption leads to an effective sound speed in terms of the constituent volume fractions, densities, and sound speeds. For two fluids this relation agrees with an effective sound speed derived by Woodward in [11].

Using pressure equilibrium again, an equation for the evolution of the volume fractions can be found by computing the Lagrangian derivative of f_α as

$$\begin{aligned}\frac{\mathrm{d}f_\alpha}{\mathrm{d}t} &= \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{V_\alpha}{V}\right) = \frac{1}{V}\frac{\mathrm{d}V_\alpha}{\mathrm{d}t} - f_\alpha \frac{1}{V}\frac{\mathrm{d}V}{\mathrm{d}t} \\ &= -\frac{f_\alpha}{\Gamma_\alpha p_\alpha}\frac{\mathrm{d}p_\alpha}{\mathrm{d}t} - f_\alpha \nabla \cdot \boldsymbol{u} \\ &= -\frac{f_\alpha}{\Gamma_\alpha p}\frac{\mathrm{d}p}{\mathrm{d}t} - f_\alpha \nabla \cdot \boldsymbol{u} \\ &= -\frac{f_\alpha \Gamma}{\Gamma_\alpha}\nabla \cdot \boldsymbol{u} - f_\alpha \nabla \cdot \boldsymbol{u}. \end{aligned} \tag{2}$$

The equations of conservation of mass for each individual species are derived from the Lagrangian invariant m_α/m where $m_\alpha = V_\alpha \rho_\alpha$ and $m = V\rho$. Then

$$0 = \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{m_\alpha}{m}\right) = \frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\rho_\alpha f_\alpha}{\rho}\right). \tag{3}$$

The equation for the evolution of momentum is needed only to compute a single velocity update and thus remains unchanged. But, it is also used to derive a total energy equation. To do so the momentum is partitioned by multiplying the momentum equation by the density fraction $\mu_\alpha = \rho_\alpha/\rho$ so that

$$\rho_\alpha \frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t} + \mu_\alpha \nabla p = 0. \tag{4}$$

Meanwhile, the internal energy or $P - \mathrm{d}V$ work equation for a single material species can be written in terms of the divergence of a velocity field and the equilibrium pressure using the expression for the composite Γ as

$$0 = \rho_\alpha \frac{\mathrm{d}e_\alpha}{\mathrm{d}t} + \frac{p_\alpha}{V_\alpha}\frac{\mathrm{d}V_\alpha}{\mathrm{d}t} = \rho_\alpha \frac{\mathrm{d}e_\alpha}{\mathrm{d}t} + \frac{p\Gamma}{\Gamma_\alpha}\nabla \cdot \boldsymbol{u}.$$

Finally, dotting $\boldsymbol{u}$ into the partitioned momentum equation and adding it to the new form of the $P - \mathrm{d}V$ equation results in an equation for the evolution of total energy for each material species

$$\rho_\alpha \frac{\mathrm{d}E_\alpha}{\mathrm{d}t} + \mu_\alpha \boldsymbol{u} \cdot \nabla p + p\frac{\Gamma}{\Gamma_\alpha}\nabla \cdot \boldsymbol{u} = 0. \tag{5}$$

Equations (2), (3), (4), and (5) describe the evolution of a set of materials both around and away from interfaces. The relations have several important properties:

(1) The equations of evolution are hyperbolic.

(2) Volume fractions individually remain between zero and one and sum up to one if initially they sum to one.

(3) The equations reduce to the proper single fluid equations away from interfaces.

Proofs of these properties can be found in [5].

3. Interface reconstruction

A volume fraction representation of a fluid interface has been used for several decades (see, for example, Debar [6], and Hirt and Nichols [8]). The motivation for the development of the mixed fluid equations above comes primarily from the ease of use and flexibility that volume fractions provide. To keep interfaces sharp, volume fractions are updated in a special way and a fluid interface is reconstructed from the volume fractions. The interface is tracked rather than the fraction advected. Fig. 1 shows the difference between tracking and advection. The distinguishing characteristic of advection is that the

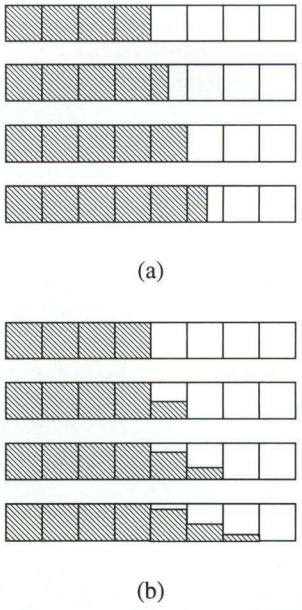

Fig. 1. The numerical evolution of front when it is tracked (a) or advected (b).

material propagates a distance of one cell in one time step. A higher order advection scheme diminishes the amount of material advected but not the speed of propagation.

One of the simpler interface reconstruction schemes is the simple line interface construction (SLIC) algorithm of Noh and Woodward [9]. Its virtue is that it can be implemented in one-dimensional sweeps. That is, a front is reconstructed from the data in a zone and its left and right neighbors. More sophisticated interface reconstruction schemes using all the cell neighbors have also been developed by Youngs in two dimensions in [14] and in three dimensions in [12].

In the development process of the 3D multimaterial adaptive algorithm we tested several techniques. These include a finite difference method of Youngs, a center of mass calculation we developed, and the SLIC algorithm. All the algorithms with the exception of the SLIC method return a normal that along with a volume fraction determines a planar surface dividing the two fluids. The SLIC algorithm may

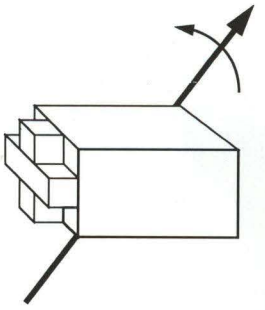

Fig. 2. The notched block is rotated at a rate of 2π radians per unit time.

have more than one interface in a cell but the orientation(s) of the interface(s) will always be normal to the sweep direction. The center of mass calculation generates a normal in a simple manner. The center of mass of the volume fraction distribution is computed for a region including the central cell and its twenty-six nearest neighbors. The vector from the center of mass to the center of the central cell is normalized and used as a normal. All the methods described above do not reconstruct planar fronts exactly. However, the test below indicates that unsplit reconstruction techniques behave significantly better than the split SLIC technique.

The problem chosen to test the reconstruction schemes is the advection of a cube with some notches cut out within a rotating velocity field. This problem is of some relevance as the Taylor–Meshkov problem outlined below will have significant vortical flows. Fig. 2 shows a schematic of the initial conditions and the axis of rotation. The size of the computation region is $100\times100\times100$. Fig. 3 shows an isosurface plot of the material interface at the initial time and after one revolution for each of the methods. What is most apparent is the oscillations the SLIC algorithm produces on the interface. After one revolution the notched cube is not easily recognized.

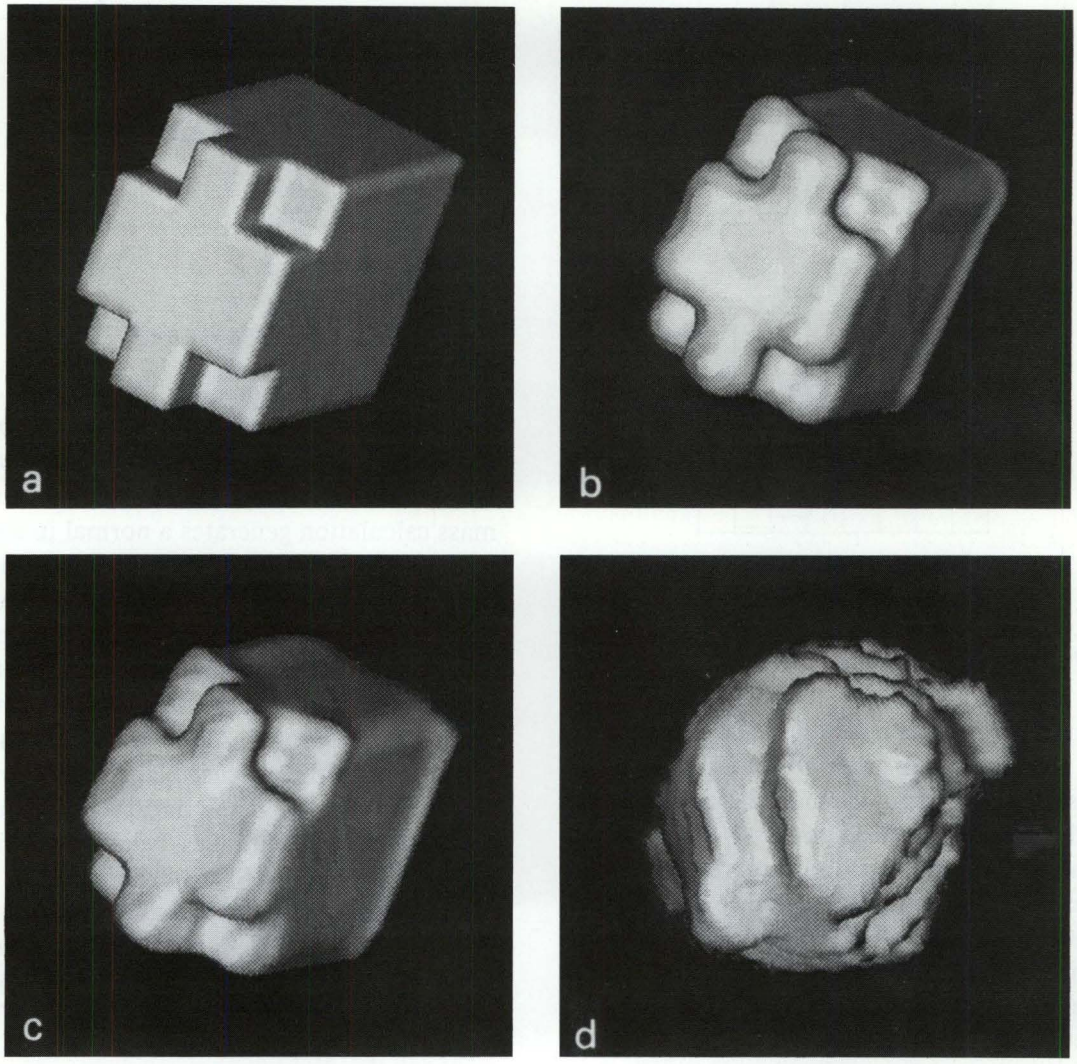

Fig. 3. Isosurface renderings of (a) initial configuration, (b) finite difference method of Youngs, (c) center of mass normal construction, and (d) SLIC. (b), (c), and (d) are after one revolution.

4. 3D adaptive mesh algorithm

The multimaterial algorithm was developed using many components from a single fluid AMR algorithm. A more detailed description of the single fluid algorithm can be found in Bell, Berger, Saltzman, and Welcome [2]. This 3D algorithm was in turn based on a 2D algorithm of Berger and Colella [3]. A 2D operator split multimaterial algorithm using the same fluid models and adaptive mesh techniques is described in Henderson, Colella, and Puckett [7].

The AMR algorithm solves the flow equations using a nested sequence of rectangular grids. The grids are properly nested and aligned. Properly nested grids have the property that a grid at some level n is always found embedded in some subset of the level $n-1$ grids. Properly aligned grids have the characteristic that a grid at some level n always has its boundary cells aligned with cell

edges of the level $n-1$ grids.

The AMR algorithm has five major functions that together generate an adaptive mesh hierarchy and advance the data on this structure. These functions are error estimation, grid generation, interpolation in tandem with data integration, and flux correction. There is a natural division between the integration routines and the rest of the AMR algorithm. The error estimation, grid generation, interpolation, and flux correction routines have been implemented as a shell for a general set of conservation laws in [2]. The calls to the integration routines and problem dependent routines are accomplished through predefined interfaces. The components of the AMR algorithm are outlined in the following paragraphs.

The error estimation process makes a crude estimate of the error in the computation (leading order terms in the truncation error) and then flags cells on the computation mesh that require additional resolution. The technique used to estimate the error is Richardson extrapolation. Given a level n grid set, the grids are coarsened by a factor of two in each coordinate direction. The level n grid set is advanced two steps in time while the coarsened grid is advanced a single step. Using the same CFL condition the two grids coincide in time at the end of their respective integrations. The finer level n grid set at the new time is coarsened and compared with the advanced coarsened grid set at the same time. The differences in the grid values is used to compute the leading order terms in the truncation error of the finite difference scheme. Where the truncation errors are large enough a finer level $n+1$ grid is placed.

The grid generation algorithm's purpose is to create meshes that contain, as a subset, those points flagged by the error estimation process. The grid generation algorithm tries to cover the flagged points in such a manner as to create as few rectangular mesh patches as possible. At the same time the covering should be efficient in the sense that the ratio of the flagged cells to the total number of cells covered by the new mesh patches approaches unity. We also specify that mixed zones should always be refined. This avoids having special interpolation and fluxing routines for mixed zones because away from them the evolution equations described above become the standard conservation laws for a single fluid.

The error estimation and grid generation algorithm may not be called every major cycle. If the grid generation creates a sufficiently large fine grid buffer area around the high error cells and because the PDEs being approximated are hyperbolic then the algorithm can go several steps without the important phenomena leaving a resolved region.

Interpolation and flux correction are important parts of the AMR algorithm as they provide the necessary communication between meshes at a given level of refinement and between meshes at adjacent levels. The data on the mesh hierarchy is advanced in a recursive fashion. The coarsest level is first advanced one time step and its cell fluxes are stored. Even at the coarsest level, the computation region may be made up of a number of patches. When updating each patch, several other patches must be used to provide data for boundary conditions. Once the coarsest level is updated. The next coarsest level is advanced using interpolated boundary conditions supplied by the coarsest level. The fluxes of the coarsest mesh are corrected using the fluxes from the finer level. If there is still a finer level, it is advanced using the interpolated data from the next coarsest level because of proper nesting. As before, the next coarsest level has its fluxes corrected to agree with those of the finest level.

The constraint of insuring that mixed zones are all contained in the finest level mesh patches has allowed us to use the AMR shell in [2] without changes to its data structures. We only increased the number of field variables managed by the AMR shell and the rest of the work involved writing a new integrator to handle a multiple species fluid.

5. Richtmyer–Meshkov problem

An example of a computation that can be carried out by the multifluid AMR algorithm is a simulation of a Richtmyer–Meshkov instability. The instability is a phenomenon where the interface between two compressible fluids becomes unstable following the traversal of a shock through the interface. This instability is studied as a mechanism for understanding the early phases of the turbulent mixing of compressible fluids driven by shocks. Much like the Rayleigh–Taylor instability, the dynamics are driven by oppositely directed gradients of the pressure and density. The linear analysis of a shock driven instability was first studied by Richtmyer [10]. The first experiment and consequent analysis was performed by Andronov et al. [1]. Pioneering numerical simulations by Youngs were carried out in two dimensions in [13].

The computation is performed in a box of size $300 \times 50 \times 50$ mm in the x, y, and z directions respectively. Within the box are two fluids initially at rest and in equilibrium at standard temperature and pressure. The fluid occupying the box from $x = 0$ to $x = 125$ mm is air. The air's initial pressure and density is .10133 $\mathrm{gm/(mm - ms^2)}$ and 1.293×10^{-6} $\mathrm{gm/mm^3}$ respectively. The air's equation of state is modeled by an ideal gas relation with a γ of 1.4. The remaining part of the box is occupied by helium. The helium has pressure and density of .10133 $\mathrm{gm/(mm - ms^2)}$ and $.179 \times 10^{-6}$ $\mathrm{gm/mm^3}$ respectively. The equation of state for helium is also modeled by an ideal gas relation with a γ of 5/3. At $x = 0$, a Mach 1.2 inflow of air is maintained. At $x = 300$ mm a reflecting boundary condition is imposed. Periodic boundary conditions are set at the four other faces of the box. A perturbation of the material interface is superimposed with an amplitude and geometry given by the relation

$$x = 125 - [1 - \cos(2\pi y/50)] \times [1 - \cos(2\pi z/50)]. \quad (6)$$

The length of the simulation is 1.25 milliseconds which will allow the perturbation to grow into a large structure.

The base coarse grid chosen is of size $72 \times 12 \times 12$ cells in the respective x, y, and z directions. Two additional levels of grids will be used. The two additional grid levels of refinement are a factor of two in each coordinate direction. The code was set to regrid every other major time step and the appropriate number of buffer zones were added to insure waves and interfaces requiring high resolution will not leave the fine mesh regions before regridding. No refinement of shocks will be allowed for $x < 100$ mm. This restriction will act like an absorbing boundary condition to catch waves moving towards the inflow region.

The computation evolves in the following fashion. The initial shock moves in the positive x direction and passes through the material interface. The perturbed interface inverts but only drifts without growth in the positive x direction as the right passing shock does not cause an unstable situation. The shock reflects off the wall and intensifies the pressure within the helium to the point where opposing gradients are formed when the shock passes through the interface traveling from right to left. At this point the perturbation grows and a single bubble can be observed growing in the center of the computation region. At late times it is observed that the bubble starts to roll up and form a "mushroom cap" driven by the vortex ring surrounding the bubble.

Fig. 4 shows volume renderings of the adaptive mesh, density field and material interface just after the shock first reflects off the boundary at $x = 300$ (0.16 ms). In the volume rendering of the adaptive mesh, the red regions are level 3 meshes, the green regions are level 2 meshes and the blue regions are level 1 (coarsest) meshes.

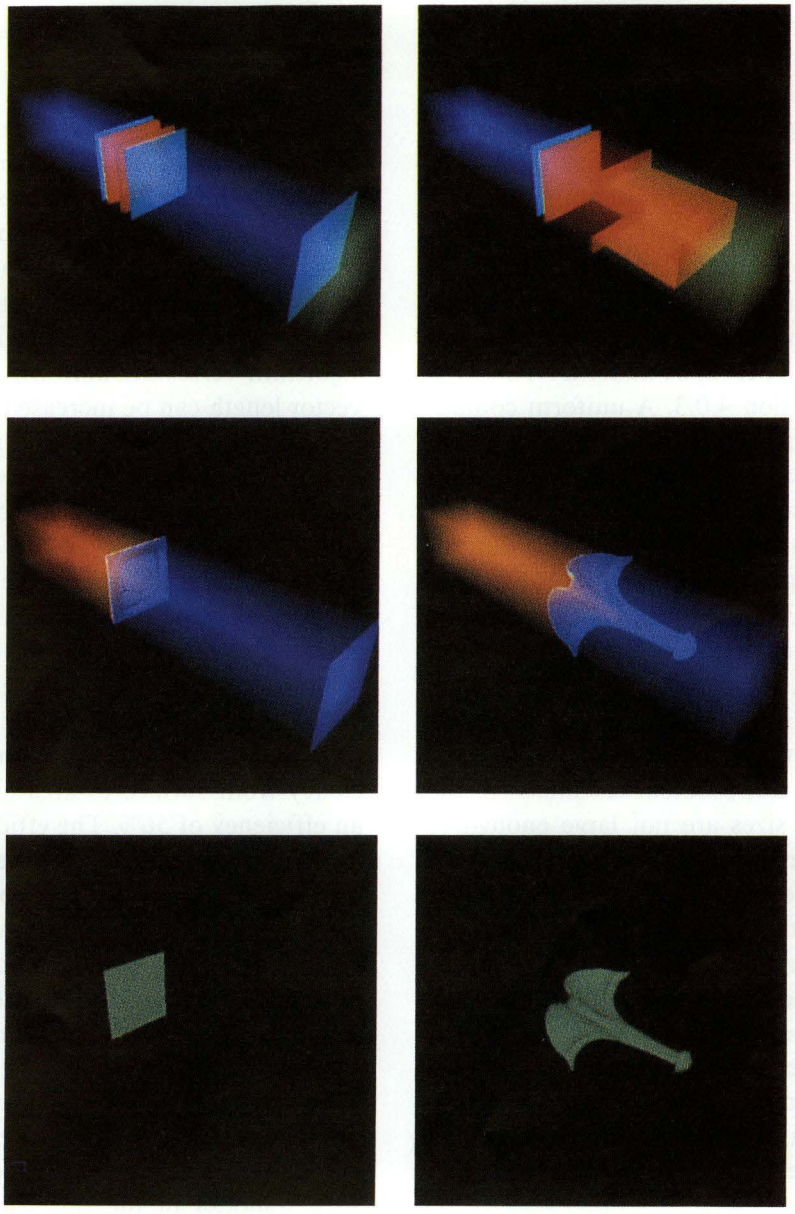

Fig. 4. The left top, middle, and bottom renderings are of the adaptive mesh, density field, and material interface at an early time in the calculation (0.16 ms). The right figures correspond to the left in type and are all from a late time (1.25 ms) in the computations.

By comparing the renderings it is easy to see how the adaptive mesh is tracking the shocks and material interfaces in the region. Fig. 4 also shows volume renderings late (1.25 ms) in the calculation. Even at late times the adaptive mesh conforms only to the region surrounding the material interface.

The adaptive computation took 30 616 seconds with an average time of 140 microseconds per cell. The computations were done on a single processor of a YMP-2/64 under UNICOS 6.0 using CFT77 version 4.0.3. A uniform computation with a resolution equivalent to the adaptive calculation has an average cell computation time of 50 microseconds per cell. The equivalent amount of time necessary for the multifluid calculation assuming a comparable number of time steps would be 63 466 seconds. The ratio of the computation times of the adaptive to uniform calculation is 2.1. Had the problem only been run to 0.125 milliseconds the ratio of computation times is greater (3.4).

The particular test problem is "small" in the sense that patch sizes are not large enough to compensate for redundant computations done on adjacent meshes. This is reflected in the factor of three in grind times between the uniform and adaptive computations. The overhead of the error estimation and grid generation is quite small relative to overlap problems (on the order of 15%). Where the AMR algorithm was able to recover was the number of points it needed to advance. The uniform computation integrates a factor of six times more points over the length of the problem.

6. Conclusions

We have described the components of a three-dimensional adaptive mesh algorithm for multimaterial gas dynamics. From the "small" computation it is seen that the AMR algorithm will become more efficient as the adaptive mesh patches become larger. Perhaps the most compelling evidence for larger gains with larger problems is reflected in the test problem in [2]. Here a factor of at least twenty in overall speedup was achieved as larger mesh refinement ratios were used. We can examine several areas to understand how to achieve similar gains. The efficiency of the integrator computations is proportional to the surface to volume ratio of the patches, the vector length of the computations, and the number of points integrated by the AMR algorithm verses the uniform case. The average vector length can be increased by concatenating sweep lines of the operator split integrator. This is because in two of the three dimensions the maximum problem size cannot exceed 48. The vector length of a Cray Y-MP computer is 64.

Another inefficiency is the patch size. The number of ghost cells used in the computation is six and the maximum patch dimension in two of the three dimensions is, as encountered in the vector length comments, 48. Even assuming the third dimension is infinite, the maximum efficiency from surface to area arguments implies an efficiency of 56%. The efficiency is probably much less than this as can be seen in the discrepancy in the average cell timings. By decreasing the number of ghost cells and increasing the patch size, gains should be achieved for larger problems. For example, a patch size of 64 and using four ghost cells (the number of ghost cells can be reduced to 4 since the flows of interest have relatively low Mach numbers. The extra dissipation and corresponding larger stencil are not needed since there are no near stationary strong shocks in the problem) instead of six leads to an efficiency of 75%. Finally, larger refinement ratios will increase the general size of the patches and boost the integrator efficiency.

Acknowledgements

We would like to thank John Bell, Marsha Berger, Phil Colella,and Mike Welcome for their support of this project. Work by E. Puckett was

performed under the auspices of the US Department of Energy at the Lawrence Livermore National Laboratory under contract number W-7405-Eng-48 and partially supported by the Applied Mathematical Sciences Program of the Office of Energy Research under contract number W-7405-Eng-48 and by the Defense Nuclear Agency under contract number IARCO 90-824. Work by J. Saltzman was supported by the U.S. Department of Energy through Los Alamos National Laboratory under contract No. W-7405-ENG-36 and additional support came from the Center for Research in Parallel Computation.

References

[1] V.A. Andronov, S.M. Bakhrakh, E.E. Meshkov, V.N. Mokhov, V.V. Nikiforov, A.V. Pevnitskii and A.I. Tolshmyakov, Turbulent mixing at contact surface accelerated by shock waves, Sov. Phys. JETP 44 (1976) 424.

[2] J. Bell, M. Berger, J. Saltzman and M. Welcome, Three dimensional adaptive mesh refinement for hyperbolic conservation laws, preprint UCRL-JC-108794, Lawrence Livermore National Laboratory, LLNL, Livermore, CA 94550 (December 1991).

[3] M.J. Berger and P. Colella, Local adaptive mesh refinement for shock hydrodynamics, J. Comput. Phys. 82 (1989) 64.

[4] M.J. Berger and J. Oliger, Adaptive mesh refinement for hyperbolic partial differential equations, J. Comput. Phys. 53 (1984) 482.

[5] P. Colella, R. Ferguson and H. Glaz, Multimaterial hydrodynamics, in preparation.

[6] R. DeBar, A method in two-D Eulerian hydrodynamics, informal report UCID-19683, Lawrence Livermore National Laboratory (March 1974).

[7] L.F. Henderson, P. Colella and E.G. Puckett, On the refraction of shock waves at a slow-fast gas interface, J. Fluid Mech. 224 (1991) 1.

[8] C.W. Hirt and B.D. Nichols, Volume of fluid (vof) method for the dynamics of free boundaries, J. Comput. Phys. 39 (1981) 201.

[9] W.F. Noh and P.R. Woodward, SLIC (simple line interface calculation), in: A.I. van der Vooren and P.J. Zandbergen, eds., Lecture Notes in Physics 59 (Springer, 1976) p. 330.

[10] R.D. Richtmyer, Commun. Pure Appl. Math. 13 (1960) 297.

[11] P.R. Woodward, Piecewise-parabolic methods for astrophysical fluid dynamics, in: Astrophysical Radiation Hydrodynamics, eds. K. Winkler and M. Norman (Reidel, 1986).

[12] D.L. Youngs, an interface tracking method for a 3D Eulerian hydrodynamics code, UK Unclassified AWRE/44/92/35, Atomic Weapons Research Establishment, AWRE, MOD(PE), Aldermasten, Berks., April 1987.

[13] D.L. Youngs, Numerical simulation of turbulent mixing by Rayleigh–Taylor instability, in: Fronts, Interfaces and Patterns, eds. A.R. Bishop, L.J. Campbell and P.J. Channell, Fronts (North-Holland, 1984) p. 32.

[14] D.L. Youngs, Time-dependent multi-material flow with large fluid distortion, in: K.W. Morton and M.J. Baines, eds., Numerical Methods for Fluid Dynamics, Institute of Mathematics and Its Applications (Academic Press, 1982).