results are everywhere in the Computational Region in qualitative agreement with the corresponding experimental results. This includes agreement on the times and locations at which the various separation and feature-interaction events occur. Unfortunately, however, the point values in the experiment of [61] can only be very roughly estimated from the published Schlieren Photographs, so no attempt is made to perform a comparison of point values between the computational solution and the corresponding experimental results. The computational results obtained here are also in close agreement with those obtained in [291] and in [417].

Three different maximum refinement levels were attempted for this test case, but only the results from the highest refinement level; namely, 11, are shown here. While the Figures 8.66 and 8.68 show that this maximum refinement level adequately resolves all the flow features that develop over the initial stages of the interaction, Figure 8.70 shows that this maximum refinement level is barely able to fully resolve all the details of the interactions between the different flow features at later stages, especially the interactions between the vortices and the shock and shears waves. For example, Figure 8.70 shows that the main roll-up vortex from the first set of triple points is not fully resolved, and also that the vortex sheet from the first set of triple points, which weakens relative to the other vortex sheets present in the computation, escapes from the highest refinement level, and therefore rapidly dissipates. It is clear from this figure that full resolution of this computation would require the maximum refinement level to be increased by 1, and perhaps even 2 if the computation is to be carried further in time with full resolution. One reason why the maximum refinement level could not be increased for this test case by one, within the given hardware-imposed limitations, was that this test case requires adequate resolution of a relatively large fractional area of the Computational Region, compared to, for ex-

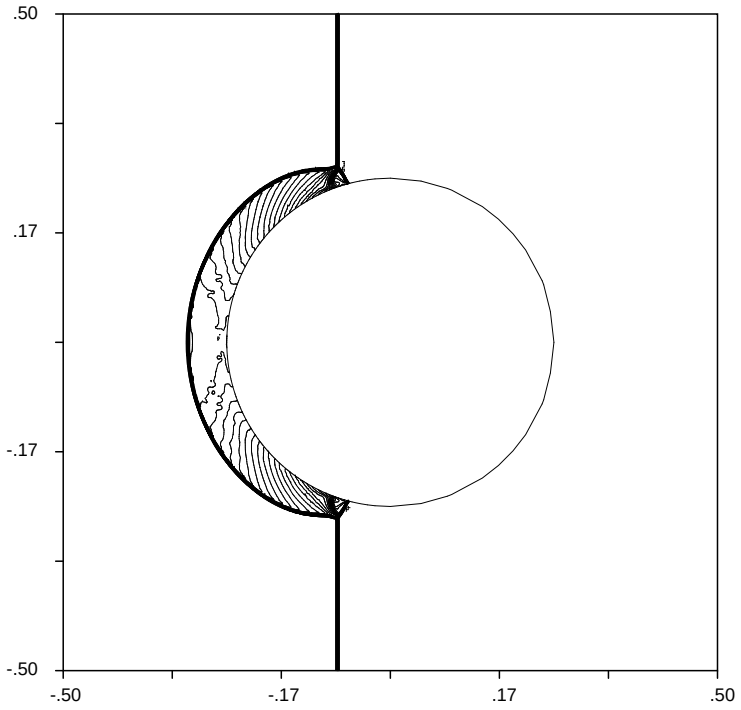

Figure 8.66: Contour-line plot of the normalized density for a shock-cylinder problem, shortly after the transformation of the Transitional Mach Reflection into a Single Mach Reflection on the surface of the cylinder.

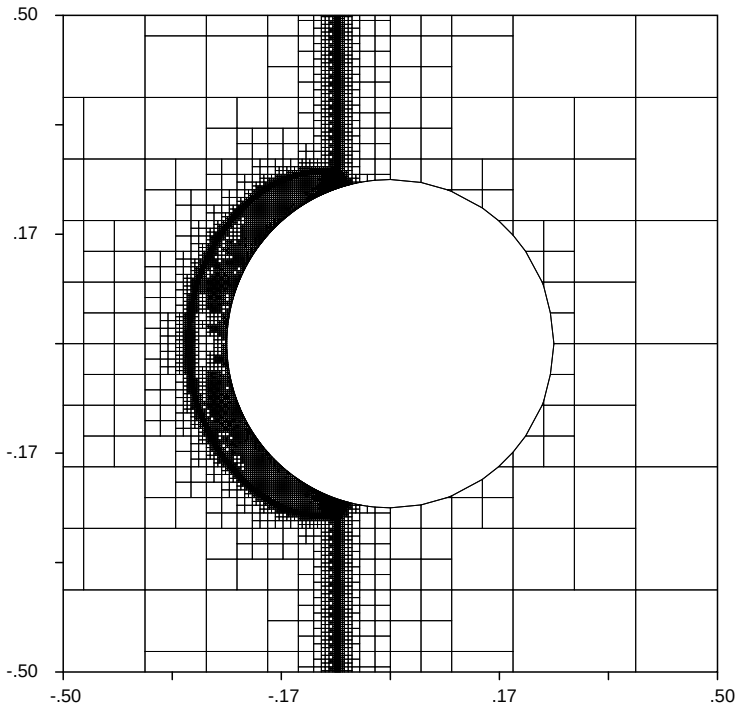

Figure 8.67: The Grid plot corresponding to the contour-line plot of Figure 8.66.

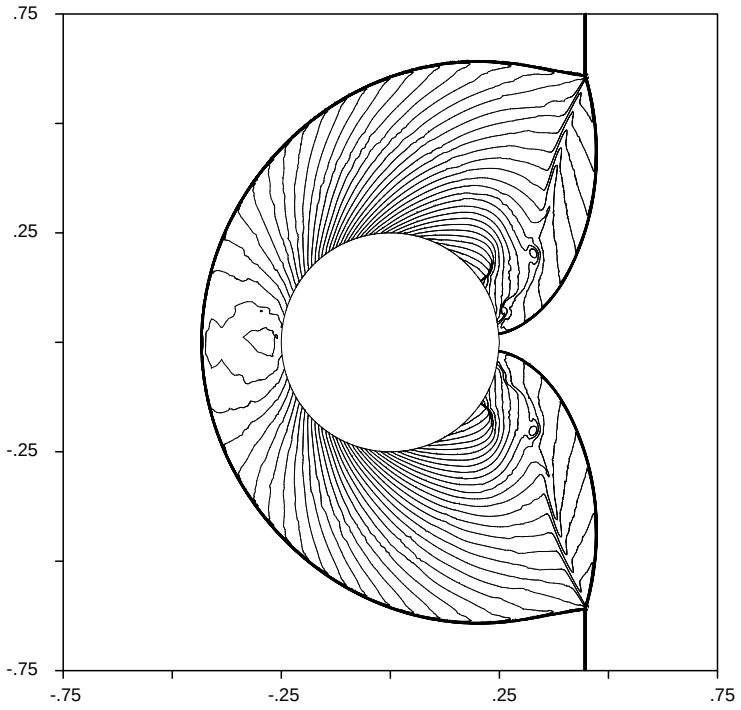

Figure 8.68: Contour-line plot of the normalized density for a shock-cylinder problem, shortly before the first Mach shocks collide at the trailing-edge, showing the flow separation and the establishment of free vortices.

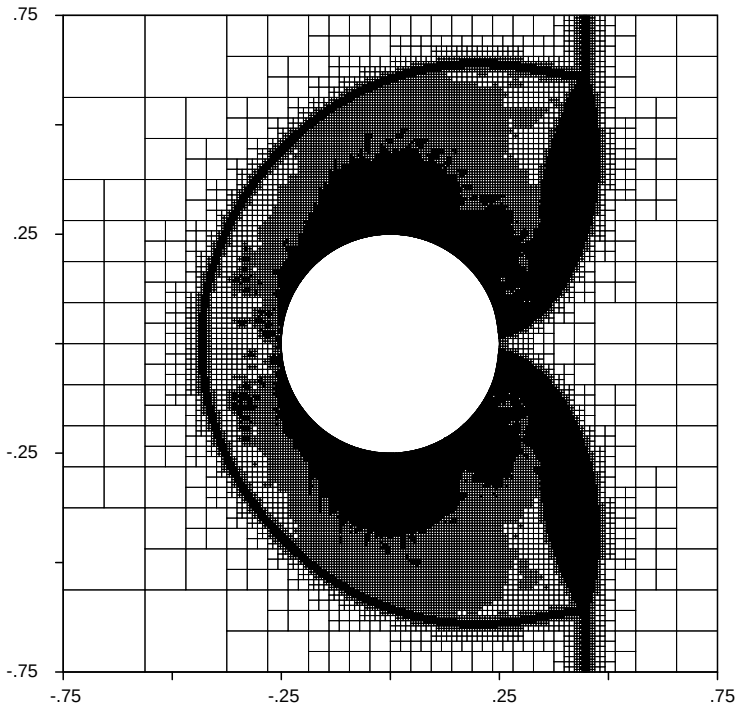

Figure 8.69: The Grid plot corresponding to the contour-line plot of Figure 8.68.

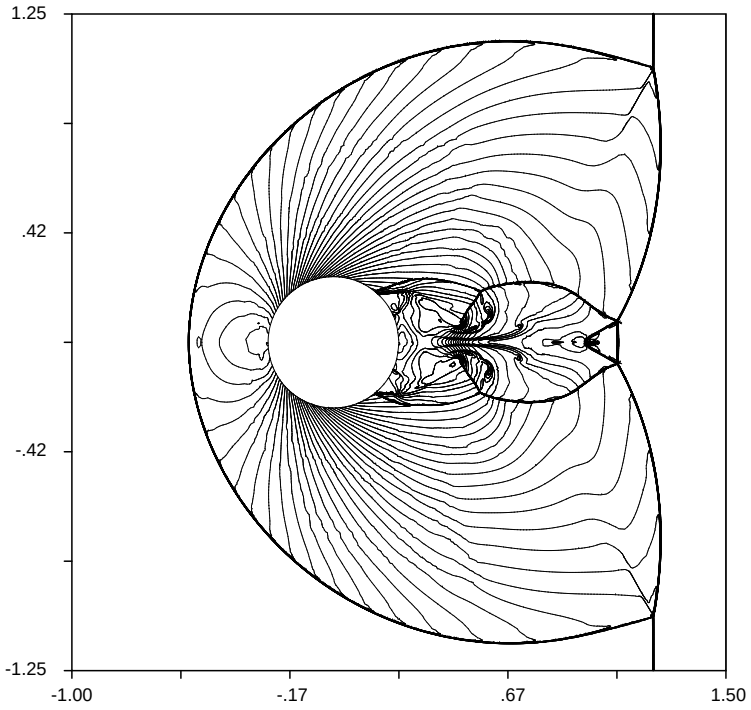

Figure 8.70: Contour-line plot of the normalized density for a shock-cylinder problem, well after the formation of the second set of Mach shocks and triple points, showing the large-scale flow separation behind the cylinder.

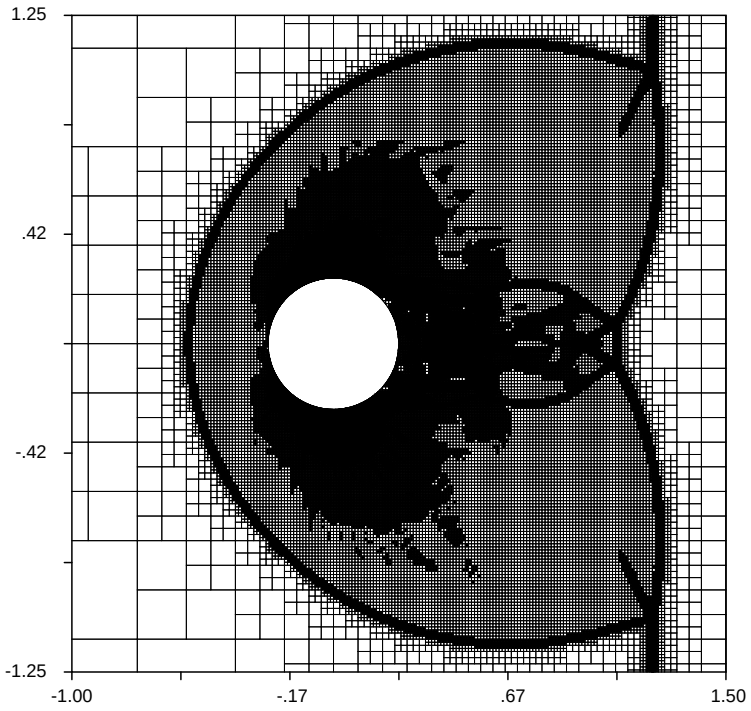

Figure 8.71: The Grid plot corresponding to the contour-line plot of Figure 8.70.

ample, the test cases of the preceding sub-sections, in which the main hydrodynamic variations are confined to a relatively smaller proportional area of the Computational Region.

The fluxes for the test case of this sub-section were computed with the Exact Riemann Solver in combination with the HLLE numerical flux function, as described in detail in Chapter III. The reasons for this choice are the same as they are for the preceding test case. The selection made for the gradient reconstruction, the limiting, and the time-stepping procedures were also all identical to those made for the test case of the preceding sub-section.

8.5 Unsteady, Moving-Boundary Computations

This section presents computations for problems in which the geometry of boundaries undergoes change, or motion relative to the subregion set, including problems in which the topology of boundaries undergoes change. Most of these problems are devoted to demonstrating the capabilities of the computational methodology developed in this work, but basic verification and validation test cases are also included.

8.5.1 Prescribed Motion

This sub-section presents test cases in which the motions of boundaries are analytically pre-specified, and not determined as part of the computational solution. The motions in these test cases are specified by prescribing either the displacement or the velocity of the boundaries, typically as a function of time, using the formulations and procedures explained in Chapter V.

8.5.1.1 Satisfaction of the Geometric Conservation Laws

The importance and the role in computational schemes for moving-boundary problems of the Geometric Conservation Laws in general, and the Discrete Geometric Conservation Laws (hereafter abbreviated to the DGCLs) in particular is described in detail in Section 3.4. This sub-sub-section presents computational results from two different test cases, which demonstrate that the methodology developed in this work satisfies the DGCLs. As would be expected from the definition of the DGCLs, both test cases involve boundaries that move at a constant, uniform velocity, in a flow-field having a constant uniform velocity equal to that of the boundaries.

The motion of the boundaries in both of the test cases of this sub-sub-section is specified by identifying each of the boundaries present in each of the test cases as a rigid boundary with a constant, uniform velocity, as described in Chapter V. The trajectory of each control point of each boundary can therefore be expressed by the equation $\vec{x}(t) = \vec{x}(t_0) + \vec{v}_{fs}t$, where t is the integration time since the start of the computation (which occurs at $t = t_0$), $\vec{x}(t)$ is the displacement of the control point at time t, and $\vec{v}_{fs}$ is free-stream velocity vector.

The time-invariant, spatially-uniform free-stream in both of the test cases of this sub-sub-section is obtained by setting the boundary conditions on all four edges of the Root Square of the Computational Region to the far-field boundary condition corresponding to the free-stream condition, by setting the Initial Conditions throughout the Computational Region to the free-stream values, and by setting the boundary conditions on all boundaries of "internal" objects to the standard impermeability condition, as explained in Section 3.5. The computations are performed with second-order accuracy in time and space, using the Least Squares procedure for gradient reconstruction, and using no gradient limiting. The numerical flux functions

are computed with Roe's Flux Difference Splitting. The details of these techniques as they are implemented in this work are described in Chapter III.

As described in Section 3.4, a computational scheme which satisfies the DGCLs will maintain the chosen uniform flow with no disturbance around the boundaries (or anywhere else in the Computational Region) as the boundaries and the flow move "together" across the Computational Region.

The first test case presented in this sub-sub-section features a rigid rectangle with axis-aligned edges moving along a trajectory identically parallel to the x-axis, from left to right across Computational Region. The rectangle has dimensions $\Delta x = 2$ and $\Delta y = 4$, has its centroid located at the point $\vec{x} = (-2.5, 0.0)$ at time $t = t_0 = 0$, and moves till it passes the mirror-image in the y-axis of its initial location (that is, until its centroid passes the point $\vec{x} = (2.5, 0.0)$. The time-invariant, uniform velocity chosen for this test case is $\vec{v}_{fs} = (100.0, 0.0)$.

Figures 8.72 and 8.73 respectively show a contour-line plot of the total velocity, and the corresponding grid plot for the moving-rectangle case after 10 time-steps in the computation, corresponding to time $t = 0.00106291s$. Figures 8.74 and 8.75 respectively show a contour-line plot of the total velocity, and the corresponding grid plot after 470 time-steps in the computation, corresponding to time $t = 0.049957s$. The total displacement of the rectangle during the period between the two time-plots shown in Figures 8.72 and 8.74 and their accompanying grid plots is therefore approximately $4.99569m$.

As the legends in Figures 8.72 and 8.74 show, the total velocity is maintained at its free-stream value throughout the motion of the rectangle, to the arithmetic precision of the computation. This is accomplished despite the motion of the boundary of the rectangle on the stationary grid, and despite the resulting repeated changes in the

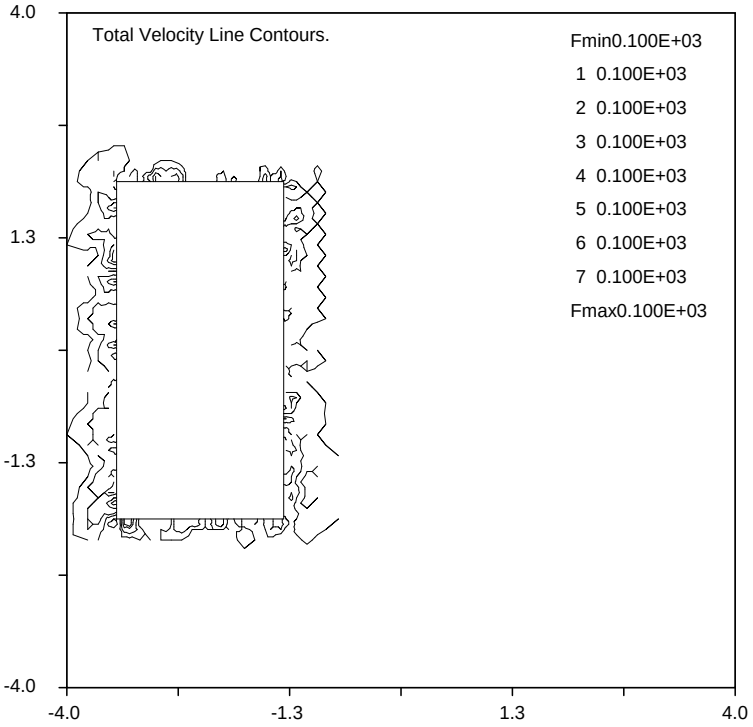

Figure 8.72: A contour-line plot of the total-velocity for a rectangular object moving at the free-stream velocity of the surrounding gas, after 10 time-steps.

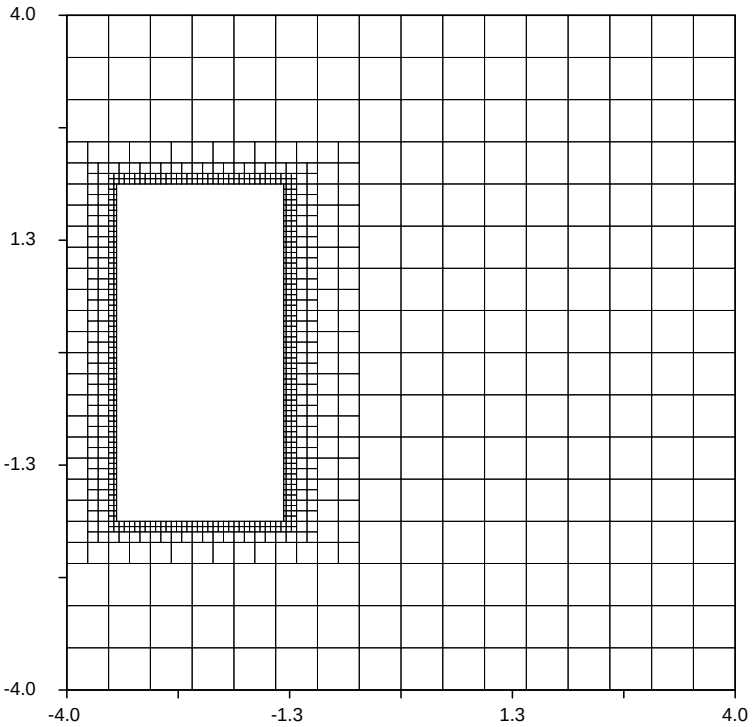

Figure 8.73: The grid plot corresponding to the contour-line plot of Figure 8.72.

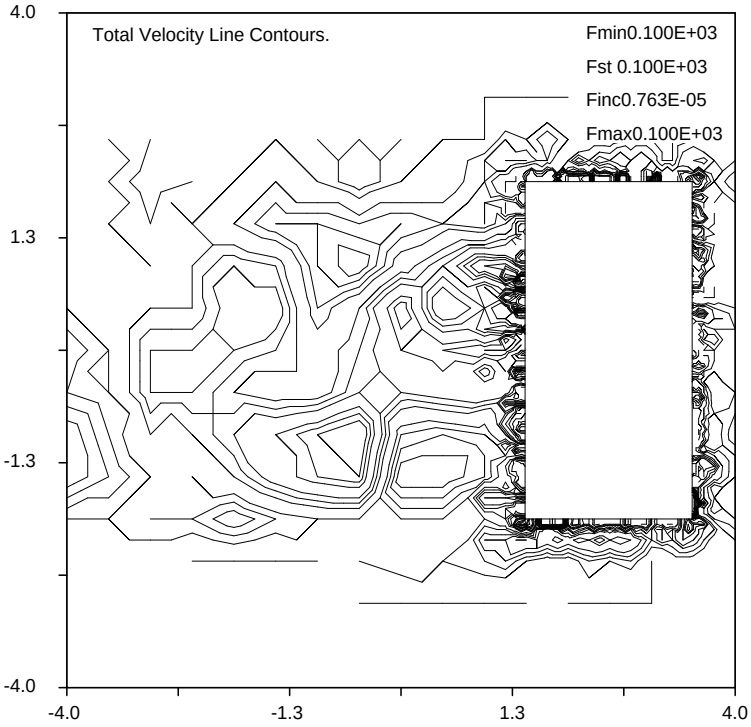

Figure 8.74: A contour-line plot of the total-velocity for a rectangular object moving at the free-stream velocity of the surrounding gas, after 470 time-steps.

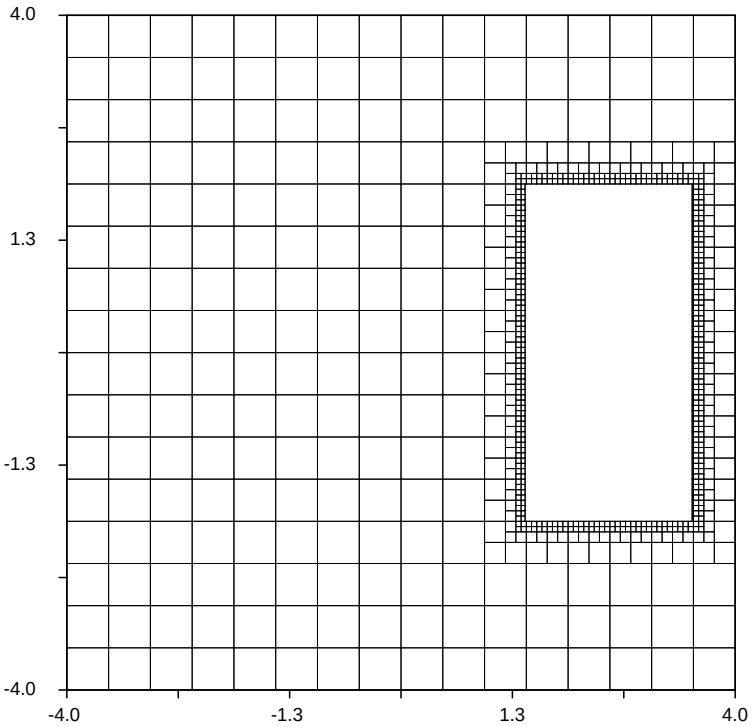

Figure 8.75: The grid plot corresponding to the contour-line plot of Figure 8.74.

geometry of all composite cells along the width of the path swept by the moving rectangle. Because the gasdynamic properties remain uniform, the time-step also remains uniform throughout the computation, at approximately $\Delta t = 0.000106291s$.

The maintenance of the uniformity of the flow-field as the rectangle moves across the grid, despite the repeated changes in the geometry of the affected composite cells, to the arithmetic precision of the calculations at all times during the computation, demonstrates that the computational scheme satisfies the DGCLs for this problem, as explained in Section 3.4. The amplitude of the perturbations in the solution from the free-stream value is determined partly by the arithmetic precision used for the computations. In particular, in order to increase the amplitude of the perturbations in the solution to the level where they would become detectable with the plotting package used here, it is necessary to use single-precision arithmetic instead of double-precision arithmetic for both the geometric and the gasdynamic computations for this test case.

The second test case presented in sub-sub-section features a multi-body, composite-region geometry comprising seven different rigid bodies, isolating six different simply-connected regions, with some of the regions multiply embedded within others, as shown in the grid plot of Figure 8.77 or of Figure 8.79. As these two figures show, three of the bodies consist of infinitesimally-thin boundaries separating two different fluid regions. As explained in Chapter VII, such "two-sided" boundaries require two-sided cell merging. Each of the six, isolated regions in the test case may have its own distinct fluid properties (which here are described fully by the values of R and γ associated with that isolated region), and these regions will retain their original "identities", and the invariance of their properties as the rigid boundaries move together across the grid and the Computational Region.

All boundaries in this test case are again given a prescribed, uniform, time-invariant velocity equal to that of the free-stream, which in this case is chosen to be $\vec{v}_{fs} = (100.0, 100.0)$. The total bulk displacement of the composite object in this test case is approximately $\Delta\vec{x} = (2.60, 2.60)m$ between time-steps numbers 1 and 6500, corresponding to a total integration time interval of approximately $0.026s$.

Unlike the simpler moving-rectangle test case presented above, this test case contains all the elements of the most general possible geometric configuration and motion pattern that can be used to test the satisfaction of the DGCLs by the methodology developed in this work.

The preservation of the free-stream is clearly evident in Figures 8.76 and 8.78 and their legends, even for the infinitesimally-thin boundaries that separate two fluid regions, that is, for the situation in which two-sided merging is applied. The reason for choosing to plot entropy contours in Figure 8.76 and the Mach Number contours in Figure 8.78 is that because the computation is performed with double-precision arithmetic for both the geometric and the gasdynamic variables, the Mach Number at time-step number 1 is uniform to the arithmetic precision of the plotting package, and could therefore not be used to draw contour-line plots.

In another variant of the test case just described, the computation is repeated but with each of the six different isolated regions having mutually differing values of γ and R. The pressures are set to be the same in all the regions, and therefore, the densities and temperatures are different. As expected, the fluid in each of these regions moved with a uniform velocity, equal to that of the free-stream, but with different Mach Numbers. Again, the uniformity and constancy of the gasdynamic state in each of these regions is found to be preserved to the arithmetic precision of the computation.

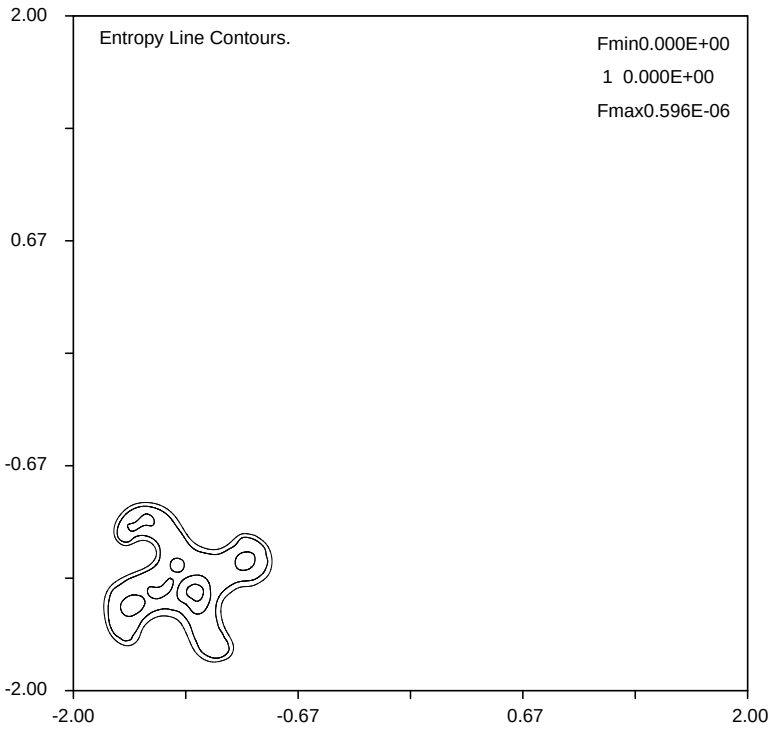

Figure 8.76: A contour-line plot of the entropy around a multi-body object moving at the free-stream velocity of the surrounding gas, after 1 time-step.

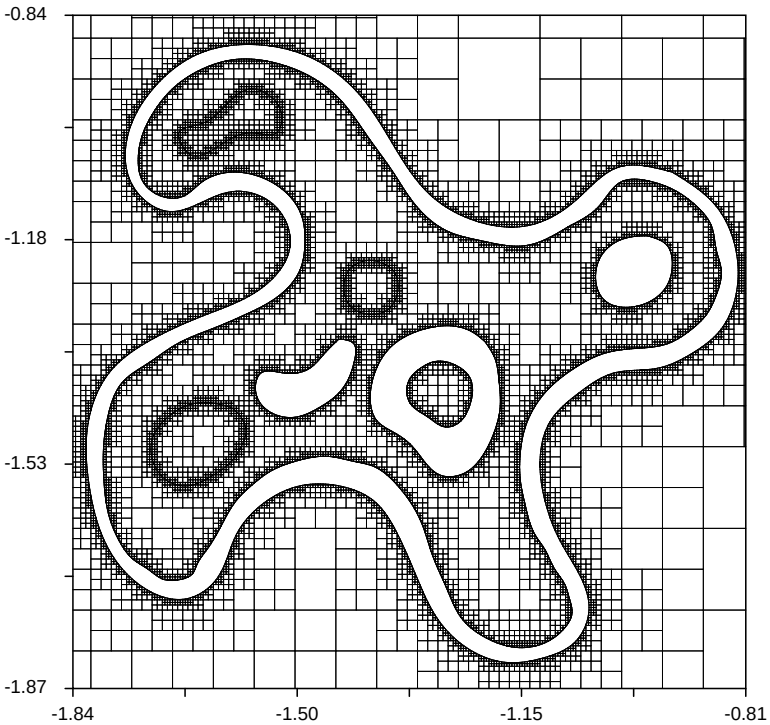

Figure 8.77: Zoomed grid plot corresponding to the contour-line plot of Figure 8.76.

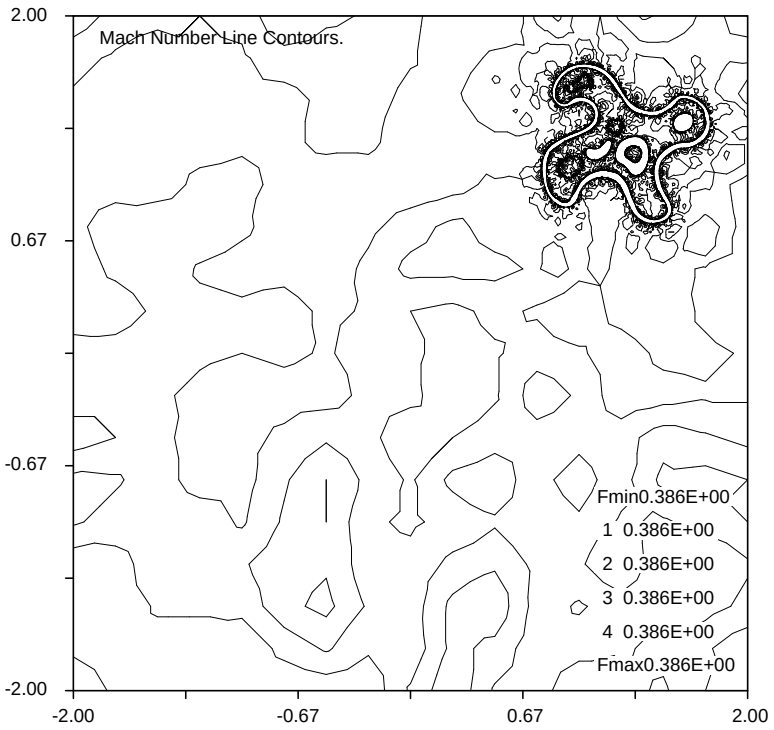

Figure 8.78: A contour-line plot of the Mach Number for a multi-body object moving at the free-stream velocity of the surrounding gas, after 6500 time-steps.

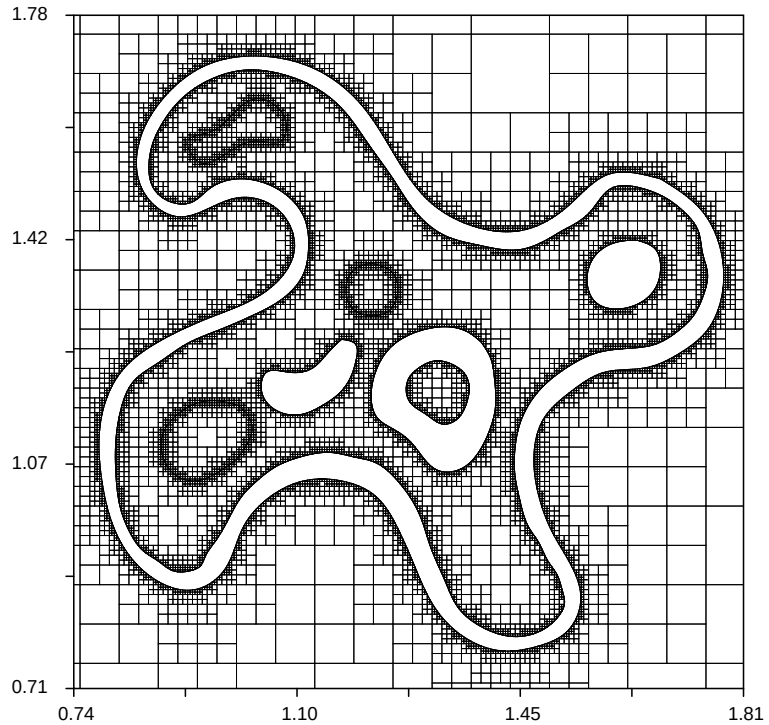

Figure 8.79: Zoomed grid plot corresponding to the contour-line plot of Figure 8.78.

In yet a third variant of the second test case presented in this sub-sub-section, the seven bodies are identified as massless, flexible bodies with zero stiffness, and with their motion determined in accordance with the "fluid-coupled", which allows the control points of the boundaries to move with the local flow field, as described in more detail in Chapter V. Again, the free-stream conditions are found to be preserved to the arithmetic precision of the computation.

The two test cases presented above (and the two variants of the second test case described above) serve to verify and validate the satisfaction of the DGCLs by the computational solution-procedure adopted and developed in this work. In both of the two test cases presented above, the boundaries are discretized using a large number of straight segments, in order to ensure satisfaction of the DGCLs to the arithmetic precision of the computation, as explained in detail in Section 3.4. If instead, the boundaries are represented using piece-wise curved segments, then the non-uniformity of the truncation of these boundaries for the second test case will give small errors in the DGCLs, here on the order of about 0.1% in the free-stream speed. For the first test case, however, using curved boundary segments will have little or no effect on the solution, since the boundary is inherently composed of straight edges, and the truncation in the very slight curvature will almost be of the order of the arithmetic precision of the computations. The errors introduced in the solution from failure to enforce the DGCLs could be as large as 50% of the free-stream velocity in the two particular test cases presented above.

The test cases presented in this sub-sub-section are confined to boundaries whose geometry remains fully-resolved and correctly and fully represented on the grid. As explained in more detail in Chapter IX, however, the DGCLs cannot be enforced nor satisfied for any object which is not fully resolved on the computational grid.

Examples of such objects are thin spikes, and boundaries with inadequately resolved high curvatures, as shown in Chapter IX. The additional work required to be able to represent and correctly treat such bodies with sub-cell resolution is briefly described in Chapter IX.

8.5.1.2 Flow in a Variable-Geometry Supersonic Inlet

In the test case presented in this sub-sub-section, a geometry resembling the inlet of an aerospace engine is immersed into a stream of Mach Number 2.540. After the solution residuals drop by about one order of magnitude from their initial values, the geometry of the inlet starts to deform from its initial shape towards a final shape which is pre-specified at the start of the computation. The solution is continued after the geometry attains its final shape, until the residuals drop by about 3 orders of magnitude from their values at the end of the deformation excursion. The interior and exterior flowfields respond appreciably to the change in geometry during (and, to a lesser extent, after the end of) the deformation excursion, most visibly through the large-scale motion of their major features, such as the shock waves and rarefaction fans.

The deformation of the engine geometry is linear over the deformation period, with the motion of each control point in the geometry being specified in terms of a prescribed-displacement formulation, as described in more detail in Chapter V, of the form $\vec{x}(t) = (1 - \lambda)\vec{x}_0 + \lambda\vec{x}_f$, where $\lambda = \frac{t-t_0}{t_f-t_0}$, where t is the time, where t_0 and t_f are respectively the time at the start and the end of the deformation interval, and where $\vec{x}_0$ and $\vec{x}_f$ are the position vectors of the control point at the initial and final geometries of the inlet. The choices made for the initial and final geometries and for the parameters t_0 and t_f are such that the magnitude of the velocity of all points in

the boundary at all times during the deformation excursion does not exceed about a tenth of the magnitude of the free-stream velocity. The test case may therefore be considered a "slow deformation" problem.

The Boundary Conditions on all four sides of the Root Square of the Computational Region are set to represent the appropriate free-stream conditions for this supersonic flow, as described in more detail in Section 3.5. The initial conditions are also set to the free-stream condition. The Numerical Flux Function is computed using Roe's Flux Difference Splitting, the gradients are computed using the Least Squares Formulation, and the gradient limiting is performed using the standard Barth Limiter, as described in Chapter III.

Figures 8.80, 8.82, and 8.84 show the contour-line plots of the Mach Number at three different times during the deformation excursion executed by the geometry. The corresponding grid plots for these contour-line plots, which are respectively shown in Figures 8.81, 8.83, and 8.85, show how the grids adapt to the solution, and how the refinement and coarsening effectively track the motion of boundaries, as well as the motion and evolution of the key flow features. The number of contour lines in each of the contour-line roughly ranges between 30 and 60, as can be determined exactly from the legends of the plots, and each of the contour-line plots shows the Mach Number at some of the most salient locations. The number of cells in the calculation varied from about 47K at the start of the deformation, to about 83K by the end of the calculation, well after the end of the deformation excursion.

More specifically, Figure 8.80 shows the inlet shortly after the start of the deformation from the initial position. The figure clearly shows how the wide inlet lips, the wide spike, and the narrow area between the spike and the inner inlet walls strongly impede the flow through the inlet, to the extent of causing a bow shock to form

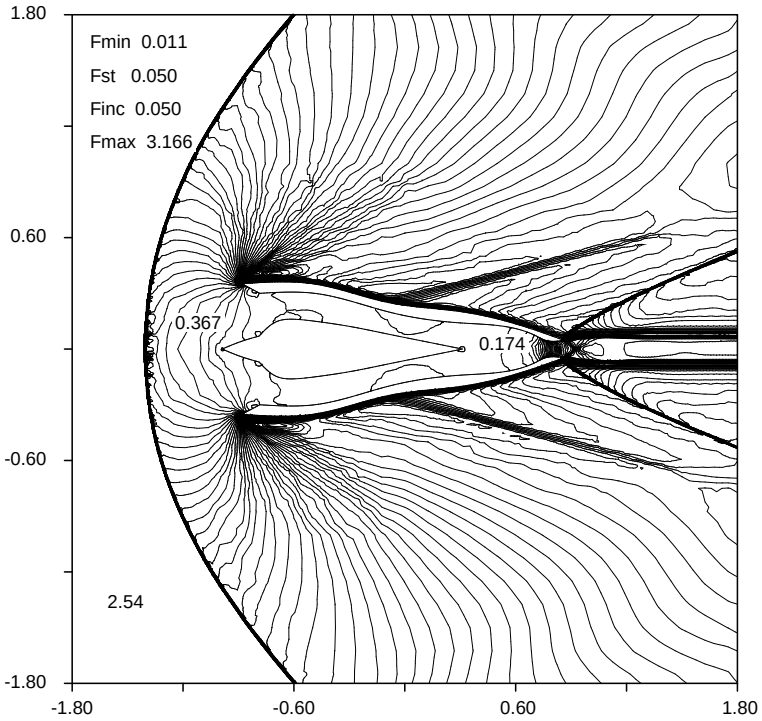

Figure 8.80: A contour-line plot of the Mach Number for a variable-geometry inlet, shortly after the inlet geometry starts to deform from its initial shape.

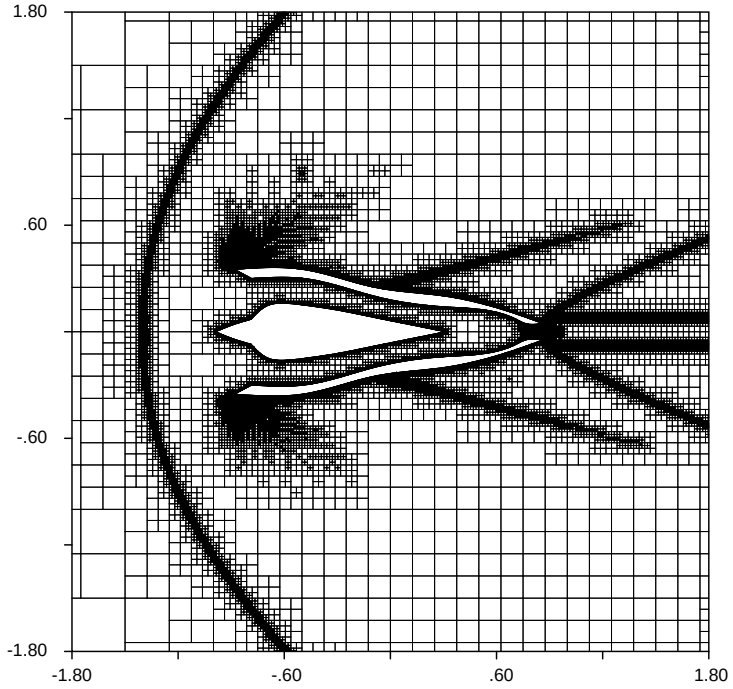

Figure 8.81: The grid plot corresponding to Figure 8.80, showing the adaptation to the solution and to the deformation of the inlet geometry.

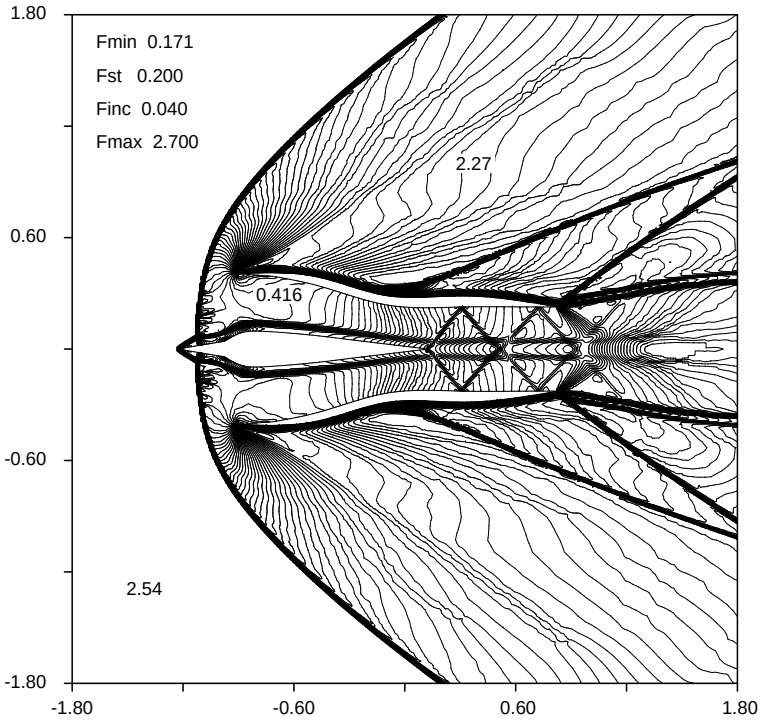

Figure 8.82: A contour-line plot of the Mach Number for a variable-geometry inlet, close to the intermediate inlet shape during the deformation excursion.

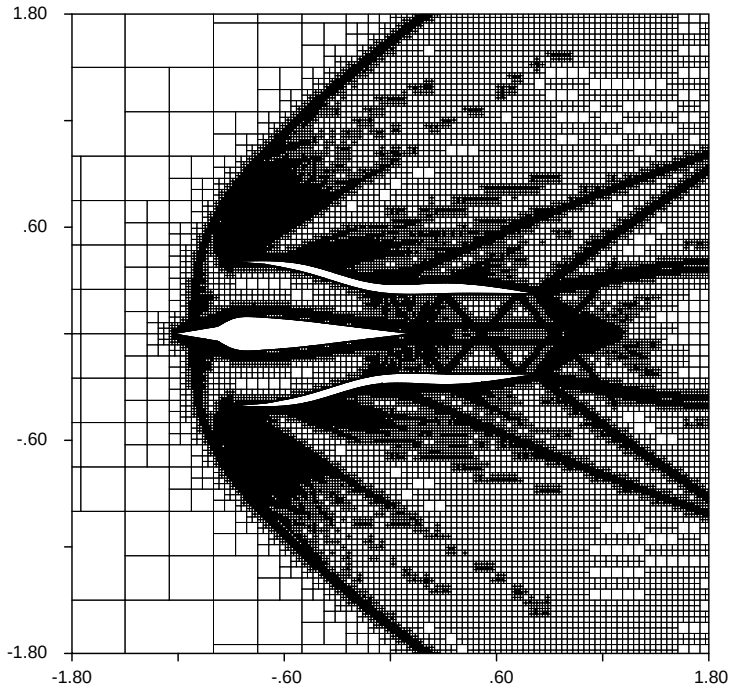

Figure 8.83: The grid plot corresponding to Figure 8.82, showing the adaptation to the solution and to the deformation of the inlet geometry.

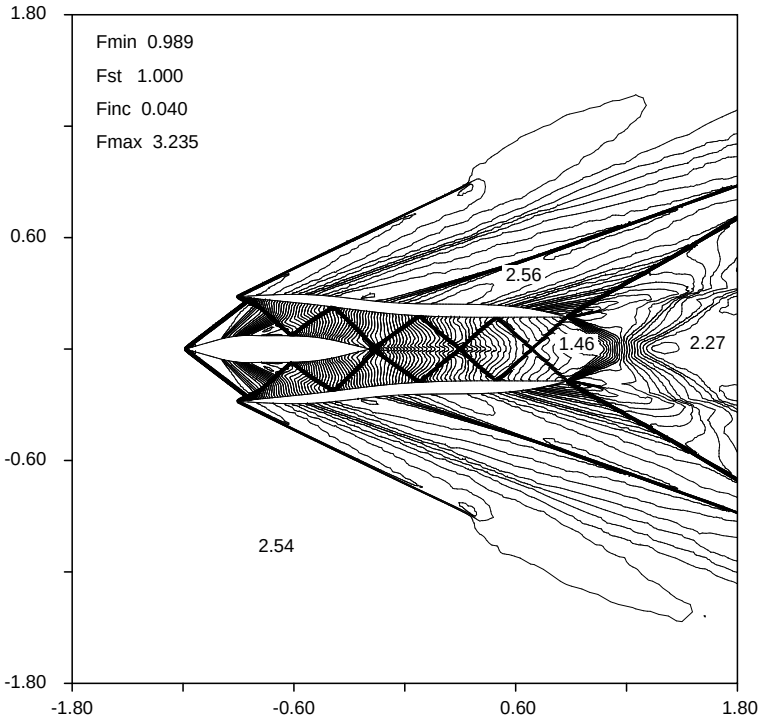

Figure 8.84: A contour-line plot of the Mach Number for a variable-geometry inlet, shortly after the geometry of the inlet attains its final shape.

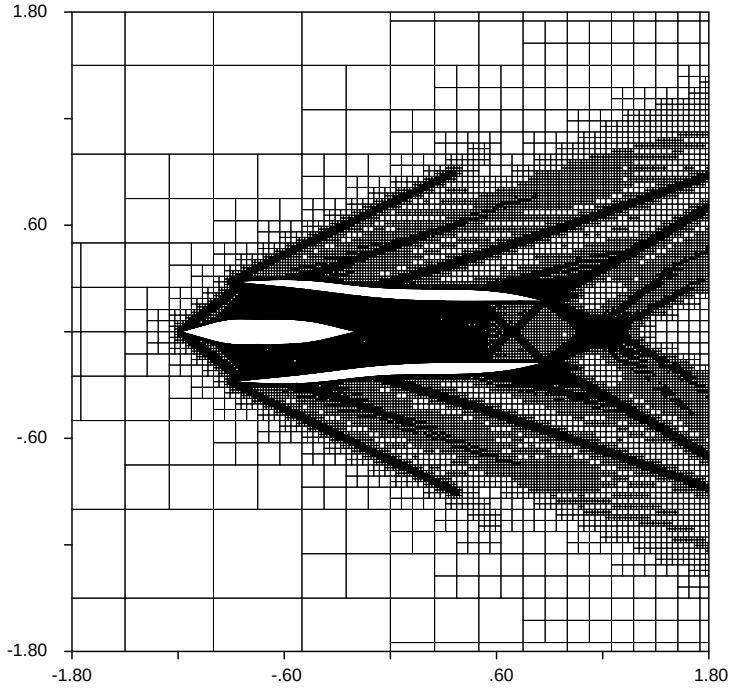

Figure 8.85: The grid plot corresponding to Figure 8.84, showing the adaptation to the solution and to the deformation of the inlet geometry.

ahead of and around the inlet. Figure 8.82 shows the shape of the inlet close to the half-way point between the initial and final geometries. The figures shows how, in response to the narrowing of the inlet lips and the spike, and to the widening of the area between the spike and the inner walls of the inlet, the bow shock system formed before the start of the deformation begins to move closer to the inlet, to the extent of forming an attached shock on the spike. The figure clearly shows how a reflected oblique shock system forms within the inlet. Figure 8.84 shows the inlet and the flowfield around it after the inlet attains its final shape. The figure clearly shows how the continued narrowing of the inlet lips and the spike, and the continued widening of the area between the spike and the inner walls of the inlet eventually allows all the inflow reaching the face of the inlet to pass through the inlet without any "spill-over". The figure clearly shows how a complex system of oblique reflected shocks is formed and enclosed within the inlet at the final shape, and how the Mach Number at the exit of the inlet is close to 1.4, the value typically considered acceptable for the imposition of normal shock to decelerate the flow to subsonic conditions before entry the entry of the flow into the engine.

The entropy layers around the boundary that can be seen in Figures 8.80 and 8.82 are due to the fact that the gradient limiting is applied using the standard Barth Limiter which effectively reduces the formal order of accuracy near a moving boundary to 1. Once the boundary motion ceased, the formal order of accuracy increased back to 2, eliminating these effects, as can be seen in Figure 8.84.

The variable-geometry-inlet test case presented above demonstrates the effective handling of a problem involving the prescribed motion of a boundary, and the adaptation of the grid to the motion of the geometry and to the evolution of the computational solution.

8.5.1.3 Diffraction of a Shock on a Moving Cylinder

The invariance of solutions of The System of Euler Equations to Galilean Transformations is mentioned and discussed in Section 8.3. More specifically, the satisfaction of the DGCLs by a computational scheme, as described above in this sub-section, and as described and discussed in Section 3.4, can be viewed as a special instance of this invariance at the computational level. Thus, the test cases presented in the first sub-sub-section of this sub-section can be seen as validating this invariance for the special case of uniform, time-invariant flow. The specific test case discussed in this sub-sub-section describes a more general instance of this invariance, using the diffraction of a shock wave on a moving cylinder.

The test case described in this sub-sub-section is derived from the shock-cylinder diffraction test case presented in detail in Section 8.4, with the differences between the derived and original test cases being reducible to the following: instead of having a shock wave propagate from left to right with Mach Number 2.82 and collide with a stationary cylinder as in the original test case, the derived test case has the shock Mach Number reduced to 1.82 (from 2.82), while the cylinder and the surrounding pre-shock gas are made to move to the left with Mach Number 1.0 (instead of being stationary). Thus, the relative speed between the shock and the cylinder are kept unchanged. Since the boundary conditions and the initial conditions are always specified in the frame of reference of the stationary grid, they must be correspondingly modified for the moving-cylinder variant of the test case. All other solution parameters, including the adaptation indicators, and the selections for the reconstruction and limiting algorithms are identical for the two variants of the test case.

The computational results obtained from the moving-cylinder calculation are found to agree very closely with those of the stationary-cylinder variant. In par-

ticular, the main features of the corresponding contour and surface line plots at corresponding integration times are very similar, and the corresponding point values of the primitive variables differ by no more than a few percent everywhere in the Computational Region. Some differences, such as those due to the temporary, local isolated "islands" of refined cells, as seen in the grid plots of Figures 8.67, 8.69, and 8.71 are markedly different, as would be expected, but their effect on the solution is negligible. The biggest differences (of about 3-4%) are actually recorded in the surface plots, and are due to the effects of cell-merging. In particular, the cell merging process causes the grid in the two cases to be different, and also causes the grid in the moving-cylinder case to continuously change as the cylinder moves across the stationary grid. Such differences, however, are expected to diminish with increasing grid refinement, to the extent of vanishing for corresponding "grid-converged" solutions.

The close agreement between the results of the two variants of this test case not only validate the invariance of the solution to Galilean Transformations, but also validate and verify that the perturbation to the solution introduced by repeated cell merging and un-merging around a moving body have a negligible local and global effect on the computational solution.

8.5.2 Motion Induced by Aerodynamic and Body Forces

This sub-section presents the computational results from two test cases in which the motions of boundaries are determined as part of the solution procedure, from the aerodynamic forces that are established between the boundaries and the surrounding gas, and, in one of the test cases, also from the gravitational force applied to a body.

8.5.2.1 Separation of a Store from an Airfoil

The test case of this sub-sub-section simulates the time-accurate separation of an object resembling a store from another resembling an airfoil. The airfoil remains stationary throughout the computation, but the store is allowed to move on the grid under the combined influence of the aerodynamic and the (here much less significant) gravity forces applied to it, as described in more detail below. The translational and rotational trajectories of the store are computed fully from these applied forces, and are therefore computed fully as part of the solution.

Throughout the computation, constant far-field boundary conditions with a free-stream Mach Number of 0.88 are imposed, giving a transonic solution around the airfoil and the store, with peak Mach Numbers approaching about 1.80. The specific Boundary Conditions applied to each of the four sides of the Root Square of the Computational Region are chosen to represent the appropriate free-stream inflow and outflow conditions for the subsonic flow involved, as explained in more detail in Section 3.5. The Initial Conditions are set to the free-stream state.

The solution is first evolved time-accurately, from the impulsive free-stream initial condition cited above, and with the store and the airfoil stationary, until the residuals drop by about three orders of magnitude relative to their values at the first time-step. Then, the store is allowed to move, starting from rest, under the influence of the aerodynamic and gravitational forces that are applied to it. The computation is continued until the store moves away from the airfoil by a distance measuring several chord lengths of the airfoil.

Figures 8.86, 8.88, and 8.90 show respectively the Mach Number contour-line plots at the instant of start of the motion, after the store has moved away from the airfoil by a fraction of the chord length of the airfoil, and after the store has

moved away from the airfoil by a few chord lengths. The figures show how the solution evolves around the stationary airfoil in response to the change in the global geometry, and how the local solution evolves around the store in response to the motion of the store and to the change in its orientation to the free-stream. Figure 8.86 shows the high-speed flow around the store before its release, and reveals that the average pressure below the store is less than that above it then. Figures 8.88 and 8.90 show how the velocity and pressure differentials across the store increase as it rotates and moves away from the airfoil. The vortex shedding pattern behind the store, as exhibited in Figures 8.88 and 8.90 is at least partly determined by the numerical dissipation in the computational solution, as explained in Chapter II, and therefore cannot be assumed to be computed with high accuracy.

Figures 8.87, 8.89, and 8.91 show the grid plots corresponding respectively to the contour-line plots of Figures 8.86, 8.88, and 8.90. These three figures clearly show the evolution of the grid refinement pattern in response to the motion of the store, and in response to the evolution of the solution, and show that all the critical flow features in the solution remain captured and adequately resolved on the adapted grid throughout the computation.

In order to increase the drag on the store to cause it to move backwards at a higher speed, the trailing edge of the store is made to deform gradually to a prescribed, "blunted" shape during the very initial phases of its motion. This change in geometry can be seen by closely examining and comparing the shape of the rear end of the store in Figures 8.86 and 8.88, for example. Other than for this deformation, the airfoil and the store are treated as rigid bodies.

The gravitational force on the store remains constant throughout its motion, and is computed in the usual manner, from the equation $\vec{F}_g = m\vec{g}$, where $\vec{F}_g$ is the

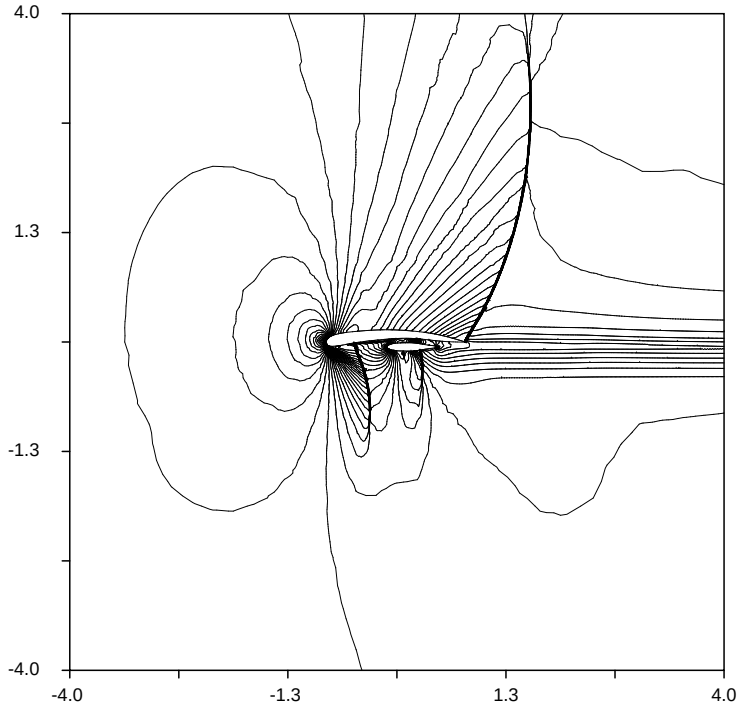

Figure 8.86: A contour-line plot of the Mach Number for a store-separation problem, immediately before the store starts moving away from the airfoil.

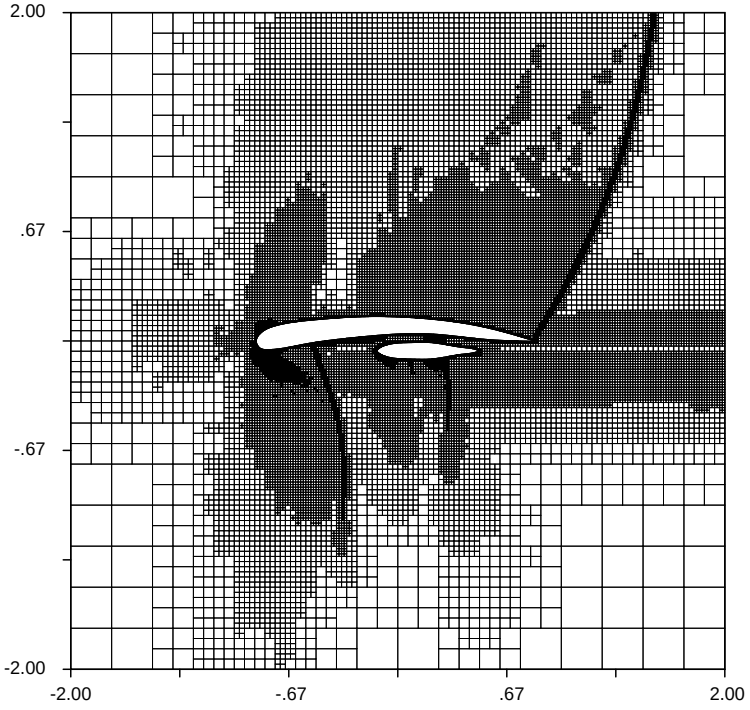

Figure 8.87: The grid plot corresponding to the contour-line plot of Figure 8.86.

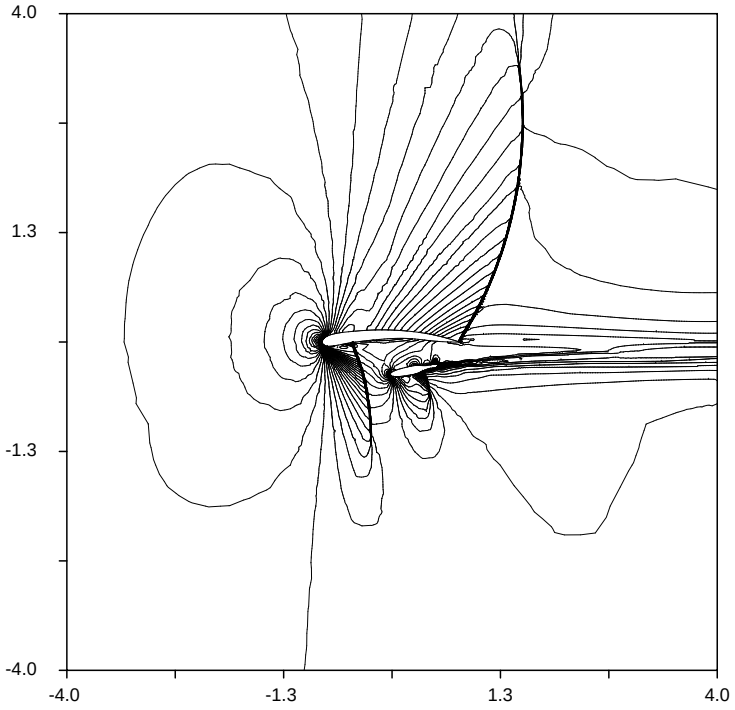

Figure 8.88: A contour-line plot of the Mach Number for a store-separation problem, after the store has moved away a small distance from the airfoil.

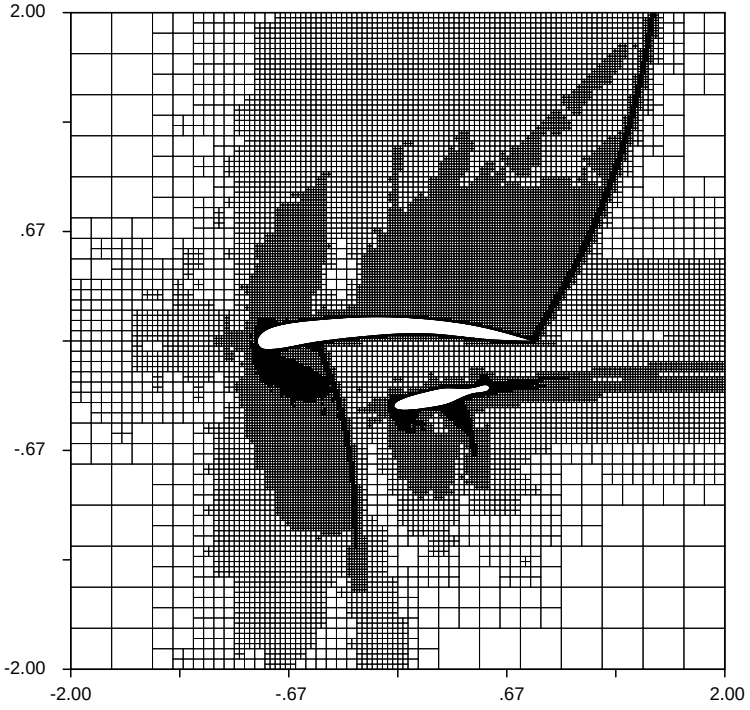

Figure 8.89: The grid plot corresponding to the contour-line plot of Figure 8.88.

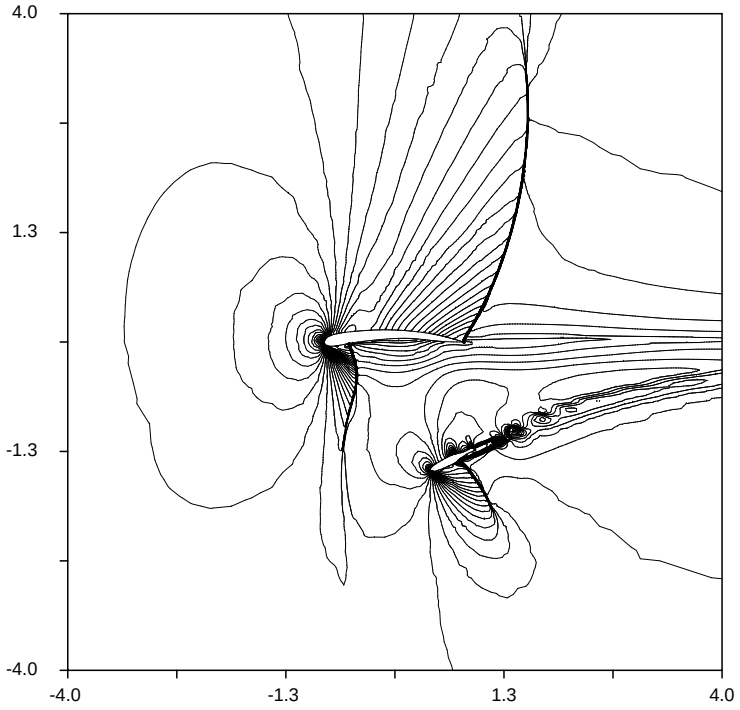

Figure 8.90: A contour-line plot of the Mach Number for a store-separation problem, after the store has moved well away from the airfoil.

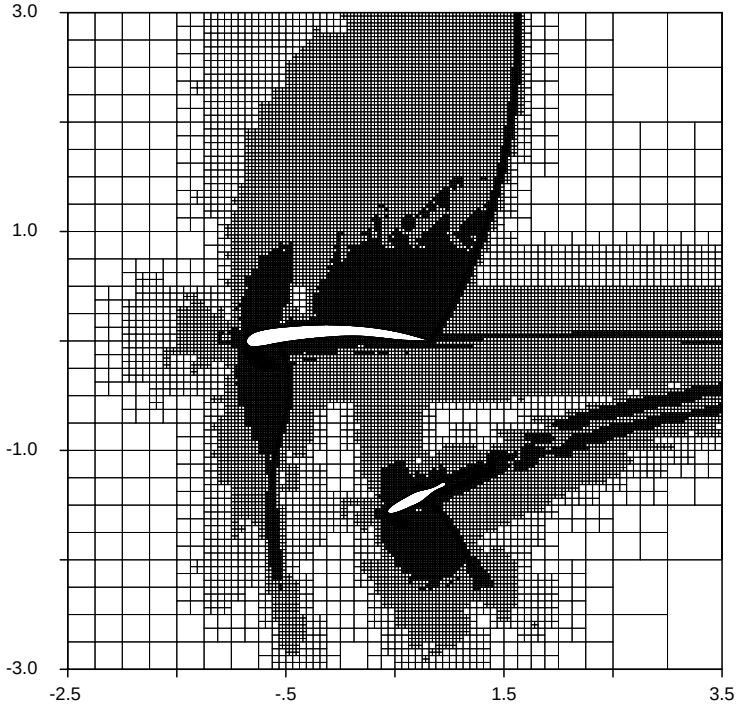

Figure 8.91: The grid plot corresponding to the contour-line plot of Figure 8.86.

gravitational force vector, m is the mass of the store, and $\vec{g}$ is the gravitational-acceleration constant, here taken to be $9.81\hat{k}m/s^2$. The mass of the store is set to the artificially low value of $0.1kg$ in order to cause it to accelerate more rapidly, so as to reduce the total integration time, and hence to reduce the overall computational time and effort. The gravity force causes no angular acceleration of the store.

The aerodynamic forces on the store are computed by summing the pressure forces on each of the faces of the discretized geometry of the boundary of the store, effectively through the equation

$$\vec{F}_a = \sum_{f=1}^{nF} -\tilde{p}_f A_f \vec{n}_f$$

as explained in more detail in Section 5.4, where $\vec{F}_a$ is the total aerodynamic force vector, f denotes the the number of the generic face in the list of faces in the discretized geometry of the boundary of the store, nF is the total number of faces in the discretized geometry of the boundary of the store, $\tilde{p}_f$ is the extrapolated pressure at the centroid of face f, A_f is the n-area of the face f (which here reduces to the length of face f), and $\vec{n}_f$ is the outward-pointing normal of face f. Section 5.4 also explains how the specific formula given above is a special case of the general formula used for computing the forces on deformable boundaries with fluid-coupled motion. As also explained in Section 5.4, in a solution of The System of Euler Equations, the only forces that a fluid and a boundary may exert on each other arise from the pressure.

The moment of the aerodynamic forces applied to the store is computed using a similar discretization procedure to the one given above for the aerodynamic forces, as explained in more detail in Section 5.4. The specific formula effectively used in

this case is given by

$$\vec{M}_a = \sum_{f=1}^{nF} -p_f A_f \vec{r}_f \times \vec{n}_f$$

where $\vec{M}_a$ is the total moment of the aerodynamic forces, $\vec{r}_f$ is the vector connecting the point about which the moment is being computed (here chosen to be the geometric centroid of the control points defining the boundary of the store) to the centroid of the face f, and all other symbols are as denoted above.

With the forces and moments applied to the store being computed as described above, the trajectory of the store is evolved by solving the equations of motion of the store; namely, the equations of rigid-body dynamics, as and in the manner described in detail in Section 5.4.

The Numerical Flux Function for this test case is computed using the Exact Riemann Solver. The gradients are reconstructed using the Least Squares algorithm, and the gradients are limited using the modified form of the Barth Limiter. These methods and techniques are described in detail in Chapter III.

8.5.2.2 Propulsive Separation

The test case of this sub-sub-section simulates the time-accurate separation of two objects; namely, a relatively light cylinder and a relatively heavier "c-shaped" vessel, under the influence of aerodynamic forces, which are generated by the release of an initial charge of high-pressure and high-temperature gas in the vessel. The trajectories of the two bodies in this test case are determined fully from the aerodynamic forces applied to them, and are therefore determined fully as part of the computational solution.

Figures 8.92, 8.94, and 8.96, and the respectively corresponding grid plots of Figures 8.93, 8.95, and 8.97 show the solution to the flowfield and the geometry of

the bodies and their locations at three different times during the computation.

The Initial Conditions in the test case can be described as follows: the two bodies in the test case are stationary, and the velocity of the gas everywhere is zero. The pressure and temperature of the gas are respectively everywhere $1.0\text{e}{+}05 N/m^2$ and $300K$, except in the cavity surrounded by the vessel: there, the pressure and temperature are respectively increased from their background values just cited by factors of 100 and 10 respectively. The high-pressure cavity region includes all of the circular region surrounded by the vessel, and part of the "nozzle" of the vessel. More precisely, it extends exactly one quarter of the way into the nozzle towards the exterior. The cylinder is initially located so that its center falls also one quarter of the way along the nozzle towards the exterior of the cavity, on the center-line of the nozzle. Thus, at time $t = 0$, the left-hand-side of the cylinder is exposed to the high-pressure, high-temperature gas, while its right-hand-side half is exposed to the "ambient", background conditions.

Upon "release" of the compressed gas at time $t = 0$, the gas expands through the nozzle of the vessel, propelling the cylindrical body along with it (in the positive x-direction), and propelling the vessel in the opposite direction. Both bodies are treated as rigid bodies, and the aerodynamic forces and moments applied to them are computed and used to solve the equations of motion of the bodies in the manner described for the preceding test case, and, in more detail, in Section 5.4. Because the masses of the bodies are chosen to be small, both bodies quickly attain very high speeds, as described in more detail below. The gas speeds that are generated in the blast zone in this test case are extremely high, exceeding $8000 m/s$ at certain times during the computation, while the Mach Numbers exceed 4.5 at certain times.

The Boundary Conditions on all four sides of the Root Square of the Compu-

tational Region are imposed through the combined "inflow-outflow" boundary procedure described in Section 3.5. As described in the latter section, this boundary procedure determines the boundary condition to be applied at a boundary cell-face depending on whether the flow is locally entering or leaving the Computational Region through that face, and whether the flow is locally subsonic or supersonic. This local determination of the most appropriate boundary condition to be applied locally, on a cell-face by cell-face basis, is essential in this test case because some of the flow features arriving at the sides of the Root Square will not only contain combinations of inflow and outflow, but also combinations of subsonic and supersonic flow. Under such circumstances, it would be a grievous error to apply a boundary condition which is neither allowed to vary along the length of a side of the Root Square, nor with time.

Figures 8.92 and 8.93 show the flowfield and the locations of the two bodies after 200 time-steps of the computation. The figure clearly shows that the cylinder has already acquired a relatively small velocity to the right-hand side, while the vessel has already acquired a relatively small velocity to the left-hand side. As Figure 8.92 shows, a complex flow has already developed in the cavity and in the nozzle. In particular, expansion fans and shock waves can be seen to have formed in the cavity, while in the nozzle, a strong expansion fan is seen to have formed around the downstream sides of the cylinder, with the flow separating from the cylinder walls further along those downstream sides. Further downstream along the nozzle, close to its exit, two strong Mach Stems are seen to have formed. As would be expected, the two Mach Stems form mirror images of each other in the center-line of the nozzle.

Figures 8.94 and 8.95 show the flowfield and the locations of the bodies after 1400 time-steps. The most prominent flow features visible in Figure 8.94 include the

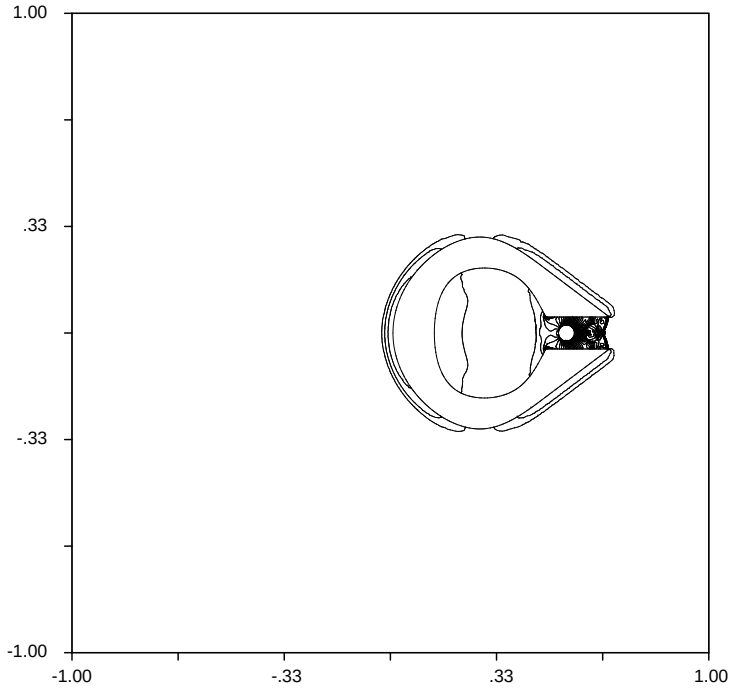

Figure 8.92: A contour-line plot of the Mach Number in a propulsive-separation problem, shortly after release of the pressurized initial charge.

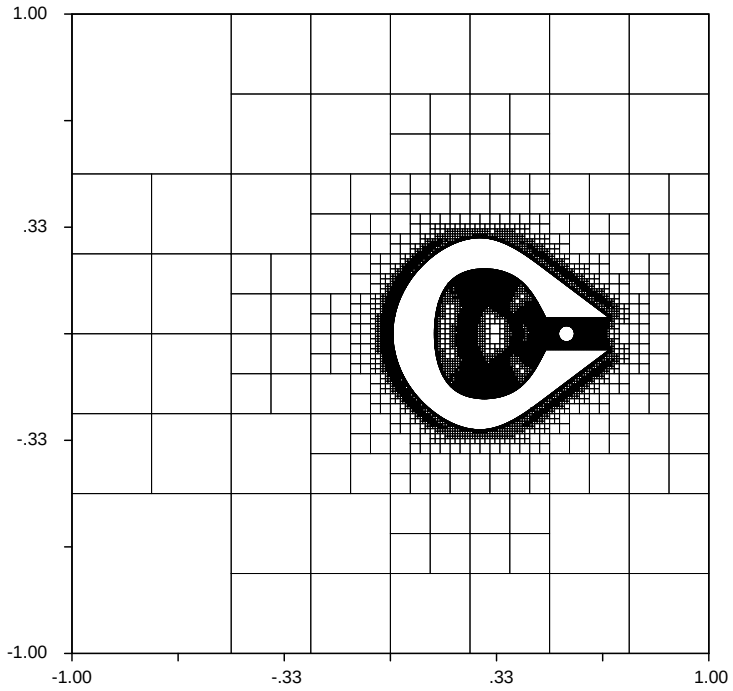

Figure 8.93: The grid plot corresponding to the contour-line plot of Figure 8.92.

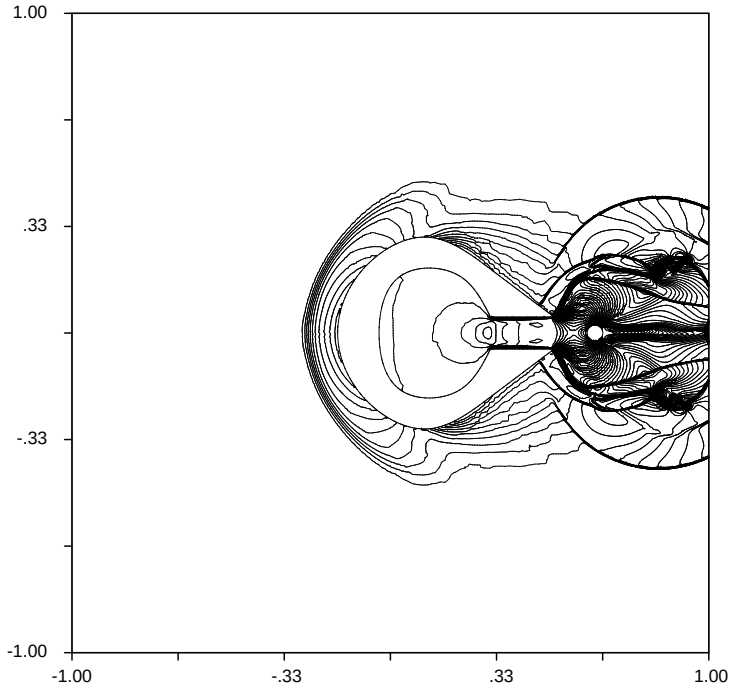

Figure 8.94: A contour-line plot of the Mach Number in a propulsive-separation problem, shortly after expulsion of the cylinder from the vessel.

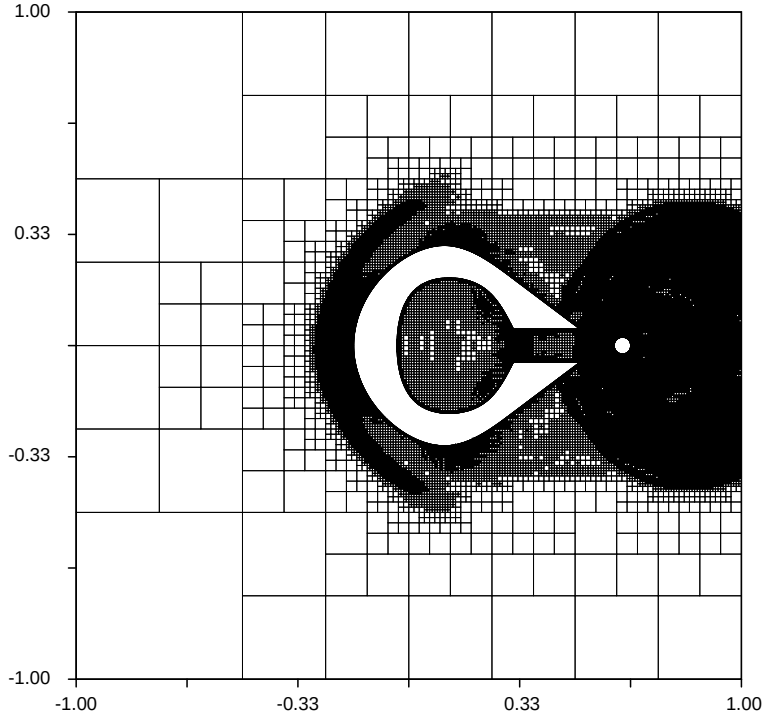

Figure 8.95: The grid plot corresponding to the contour-line plot of Figure 8.94.

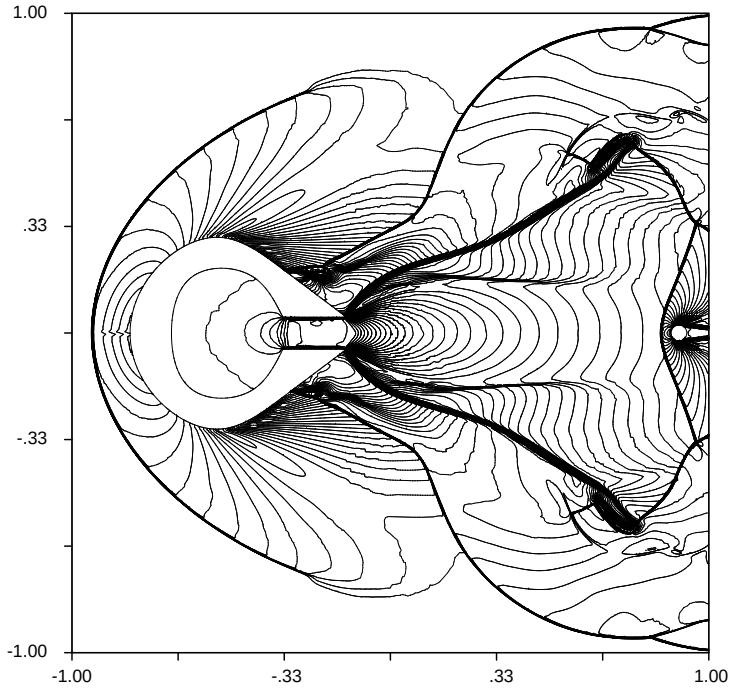

Figure 8.96: A contour-line plot of the Mach Number in a propulsive-separation problem, after the attainment of supersonic speed by the vessel.

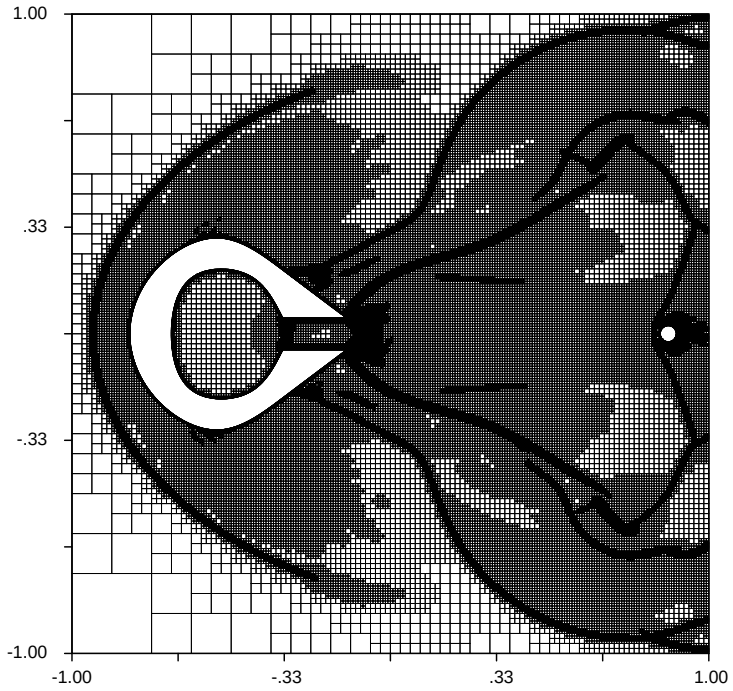

Figure 8.97: The grid plot corresponding to the contour-line plot of Figure 8.96 (after the elimination of all cells at the penultimate refinement level).

pressure wave created ahead of the moving vessel as this wave is gradually developing into a bow shock, the strong primary blast wave propagating almost radially behind the exit plane of the nozzle, secondary blast waves propagating forward and backward within the primary blast wave, and several expansion fans and shear waves interacting with each other and with the secondary blast waves, in a highly complex pattern.

Figures 8.96 and 8.97 show the flowfield and the locations of the bodies after 3200 time-steps. Figure 8.96 shows how the flow features shown in Figure 8.94 have evolved, and how they have interacted with each other and with newly formed features. For example, the figure shows how the primary blast wave interacts with the expansion fan which forms on the backward-facing outer walls of the vessel, and how the primary blast wave is bent as a result of this interaction. The figure shows that the flowfield within the blast zone at this time has become highly complex, incorporating roll-up vortices close to the outer walls of the vessel, and several expansion fans, vortex sheets, and shock waves, many of which are interacting with each other. The strong bow shock in-front of the vessel in Figure 8.96 clearly indicates that the vessel has attained supersonic speed by this time. However, one of the strongest and most unexpected shocks at the time shown in Figure 8.96 is the one that forms, with a Mach Number of about 3.0, in-front of the cylinder. This shock is created as a result of the strong decompression and large increase in the velocity experienced by the gas within the blast wave zone upstream of the backward-moving cylinder. This low density, high-speed flow can only be decelerated through a bow shock ahead of the slower-moving cylinder. As Figure 8.96 shows, after passing through this shock, the flow accelerates rapidly is it passes and expands around the cylinder, again reaching a Mach Number greater than 3, and then decelerates rapidly through two shocks attached to the downstream sides of the cylinder.

Figures 8.94 and 8.96 clearly show the effectiveness of the chosen "inflow-outflow" boundary procedure at allowing geometrically and physically complex flow features to leave the Computational Region without reflection, and without the creation of instabilities at the boundaries.

Figures 8.93, 8.95, and 8.97 clearly show that the various features in the computational solution are all captured, resolved, and tracked effectively as they form and evolve throughout the computation. The figures show clearly that the solution-adaptation algorithm achieves a high degree of success in capturing and resolving critical features, even when these features multiply and interact with each to form highly complex patterns. In order to show how the critical flow features are resolved and captured in the adaptive grid of Figure 8.97 more clearly, and without having to revert to multiple zoom plots, all cells at the highest refinement level but one in that figures have been removed by a use of a special post-processing step.

The Numerical Flux Function for this test case is computed using Roe's Flux Difference Splitting. The gradients are reconstructed using the Least Squares algorithm, and the gradients are limited using the modified form of the Barth Limiter, as described in Chapter III. The choice of Numerical Flux Function in this problem is probably the source [293] of the "carbuncle" instability [293] seen to form around the bow shock ahead of the vessel. The number of cells in this computation increased from about 15K shortly after it starts, to about 250K after about 3600 time-steps.

8.5.3 Motion with Structural Deformation and Contact Forces

This sub-section presents the computational results from two test cases in which the motions of boundaries are determined as part of the solution procedure, from the aerodynamic forces involved, and for some of the boundaries, also from the internal

structural-dynamic forces, and the external contact forces that arise as a result of the collision of boundaries with each other.

8.5.3.1 Projectile Flight and Impact, I

In the test case presented in this sub-sub-section, a two-stage projectile is accelerated from rest under an aerodynamic thrust force imparted as a result of the discharge of pressurized gas contained in a cavity in the projectile's first stage (which may also be called the projectile's rear part). Once the projectile attains a pre-selected speed corresponding to a Mach Number of roughly 2.5 during its acceleration, the pressure and temperature of the gas in a cavity in the projectile's second stage (which may also be called the projectile's front part) are programmed to instantaneously rise to pre-set values. The resulting additional discharge of pressurized gas from the cavity of the front part of the projectile gives rise to an additional thrust force on the front part of the projectile, and an additional drag force on the rear part of the projectile, causing the front part to accelerate ahead of the rear part. The front part of the projectile subsequently collides with, deforms the walls of, and enters the enclosure of a target at which the projectile is originally aimed. The breaching of the walls of the target by the front part of the projectile opens an entry path into the enclosure of the target for the rear part of the projectile, which continues to move towards the target after its separation from the front part.

The Figures 8.98, 8.100, 8.102, 8.104, and 8.106 show the contour-line plots, with about 50 contour lines in each, of the density distribution in the Computational Region at five different times during the flight of the projectile and its interaction with its target, while the Figures 8.99, 8.101, 8.103, 8.105, and 8.107 show the grid plots corresponding respectively to the five contour-line plots just cited. These five

pairs of corresponding figures also show the geometry and the dimensions of the projectile and the target, and show the overall flight trajectory of the projectile and the impact process described above. Only the first two pairs of corresponding figures show the whole Computational Region; the last three pairs show zoom plots which focus on the regions of greatest solution activity and boundary motion.

In order to eliminate from this test case the possibility of any coalescence or disintegration of boundaries, or any other topologic transformations, appropriate artifacts are incorporated into the geometry of the boundaries, and the contact forces (which physically arise to prevent boundaries from inter-penetrating each other when they come into direct geometrical contact, as during an impact process) are applied at appropriately-small standoff distances from boundaries.

The geometric artifacts include the introduction of a slit in the front wall of the target in the region of impact, dividing the front wall into a lower and an upper part, as can be seen in all the accompanying contour-line and grid plots. This slit allows the target to be breached by the projectile without the need for the walls of the target to undergo a topologic transformation (which would here have amounted to a simple disintegration).

The spatial offsetting of the contact forces to prevent direct geometrical contact between boundaries, and hence to prevent topological transformations in the boundaries, is accomplished by representing the contact force as an elastic force that, when active, obeys a force-deflection relation analogous to that of the standard, linear elastic spring. This relation can be expressed in the form $\vec{F}_s = -k(\vec{x} - \vec{x}_0)$, where $\vec{F}_s$ is the force in the spring, k is the stiffness of the spring, and $\vec{x}$ and $\vec{x}_0$ are respectively the loaded and unloaded deflections of the spring. Unlike the situation with the spring model, however, the contact force in this work is brought into effect only

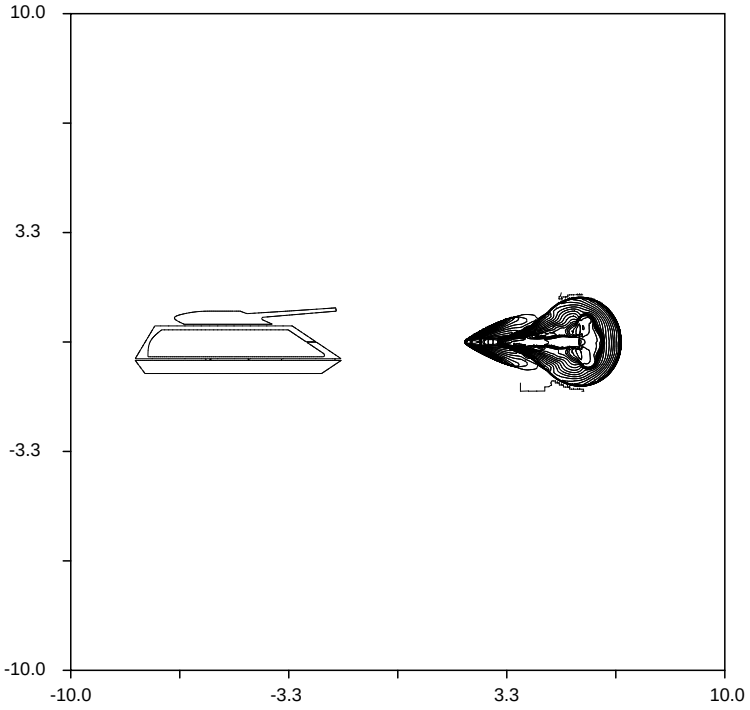

Figure 8.98: A contour-line plot of the density distribution shortly after the release of a two-stage projectile from a state of rest towards its target.

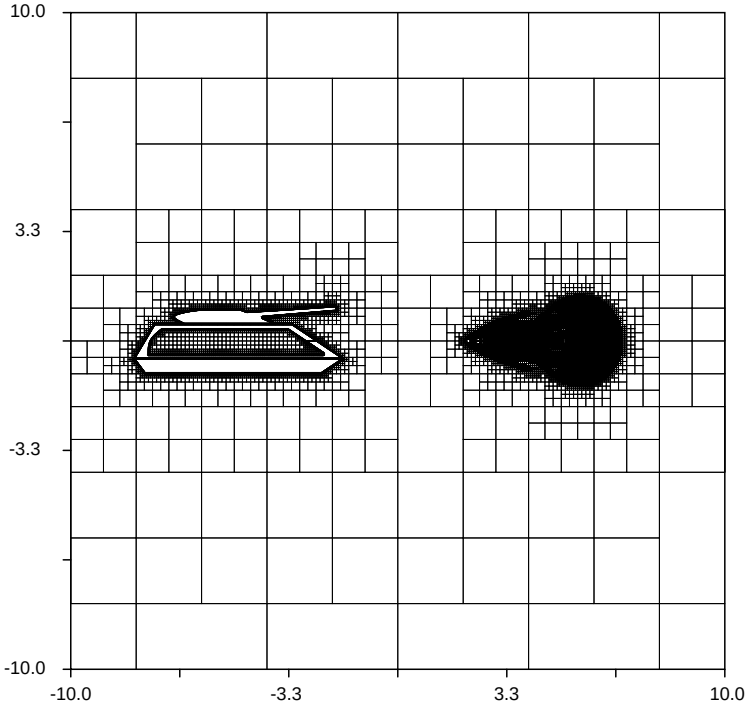

Figure 8.99: The grid plot corresponding to the contour-line plot of Figure 8.98.

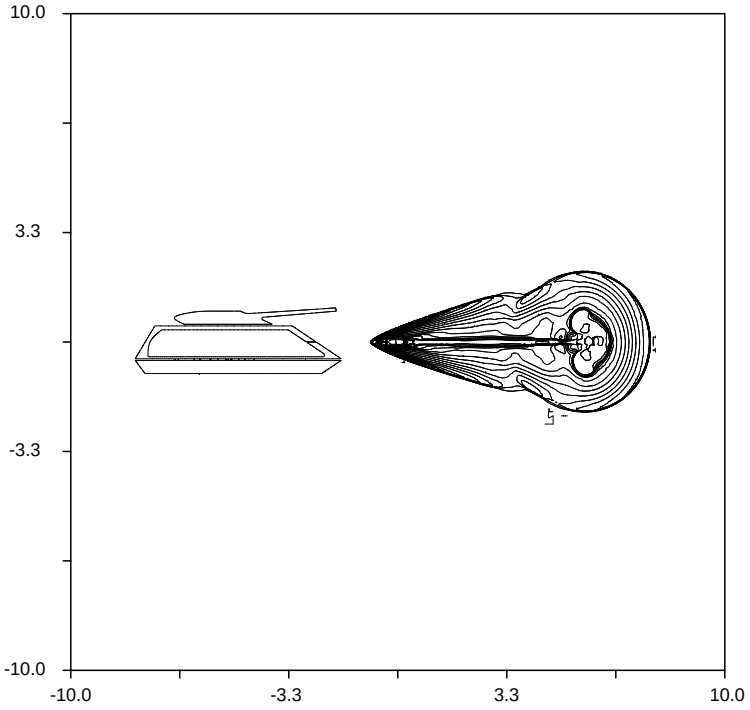

Figure 8.100: A contour-line plot of the density distribution shortly after the front and rear parts of a two-stage projectile separate on its way to its target.

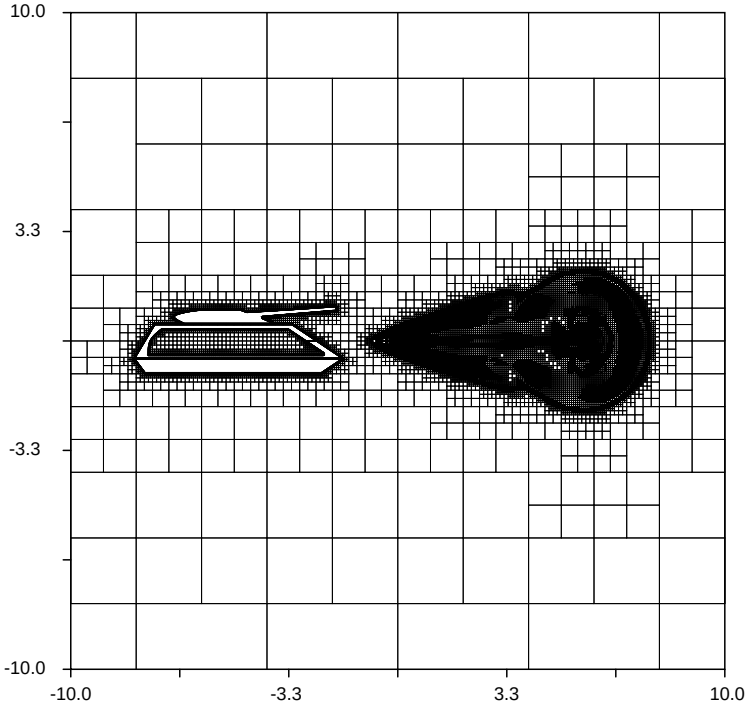

Figure 8.101: The grid plot corresponding to the contour-line plot of Figure 8.100.

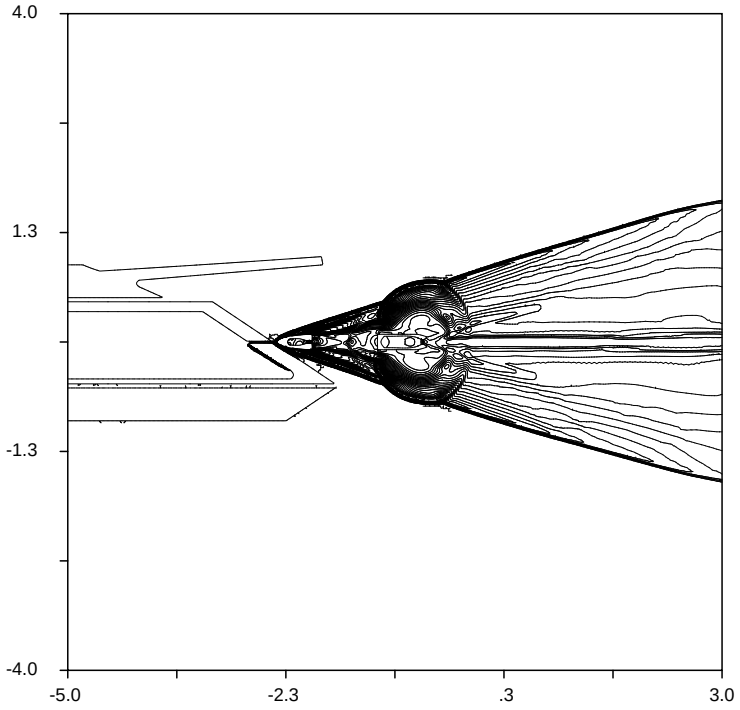

Figure 8.102: A contour-line plot of the density distribution shortly after the "cushioned collision" of the front part of a two-stage projectile with its target.

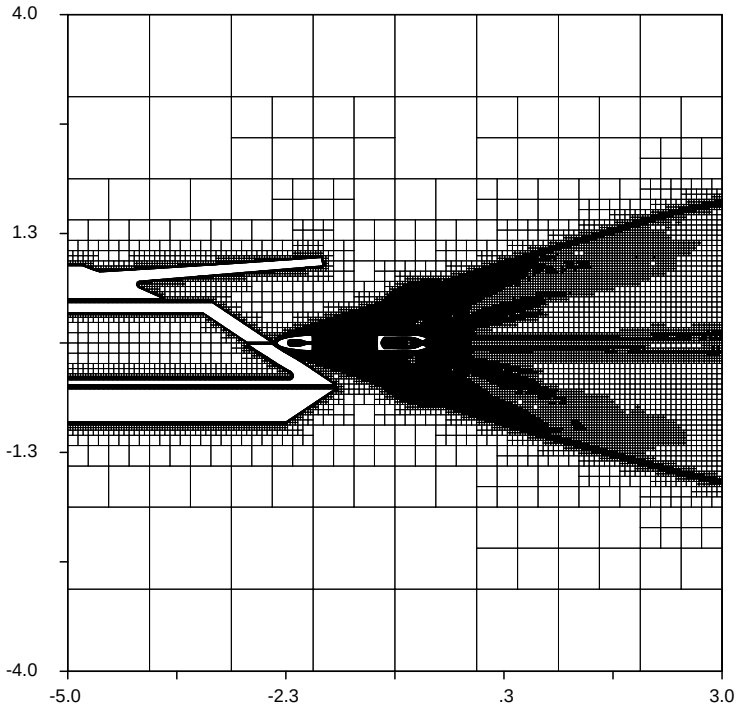

Figure 8.103: The grid plot corresponding to the contour-line plot of Figure 8.102.

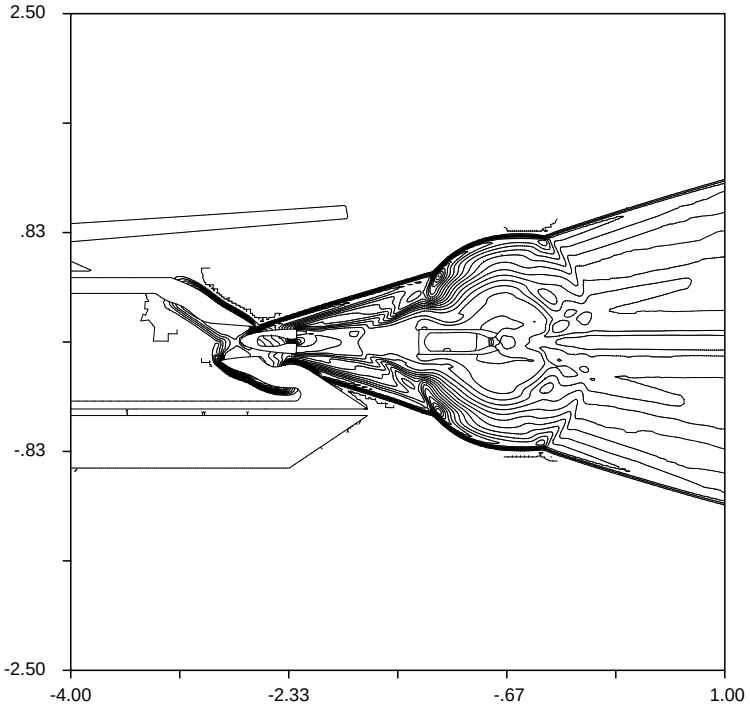

Figure 8.104: A contour-line plot of the density distribution shortly after the front part of a two-stage projectile starts penetrating its target.

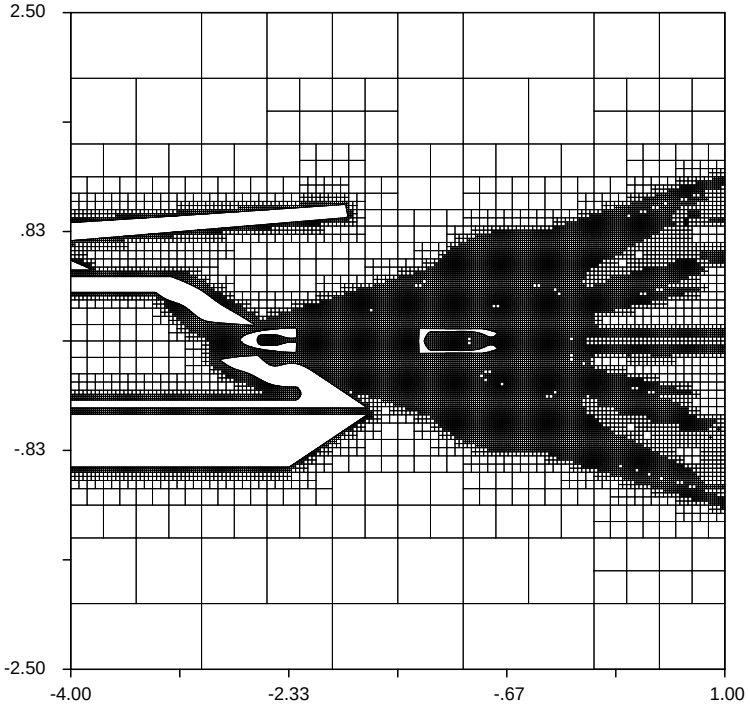

Figure 8.105: The grid plot corresponding to the contour-line plot of Figure 8.104.

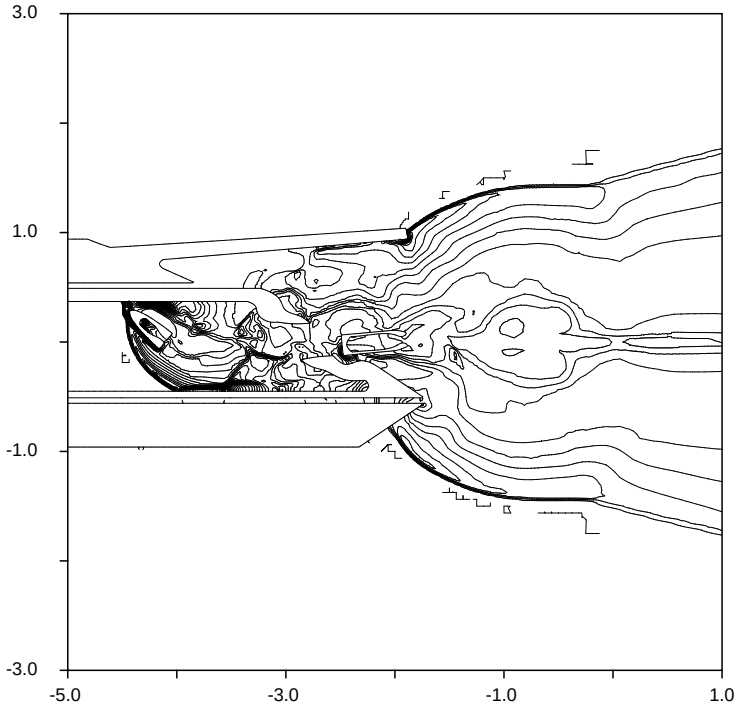

Figure 8.106: A contour-line plot of the density distribution showing the front part of a two-stage projectile rotating as it traverses the interior of its target.

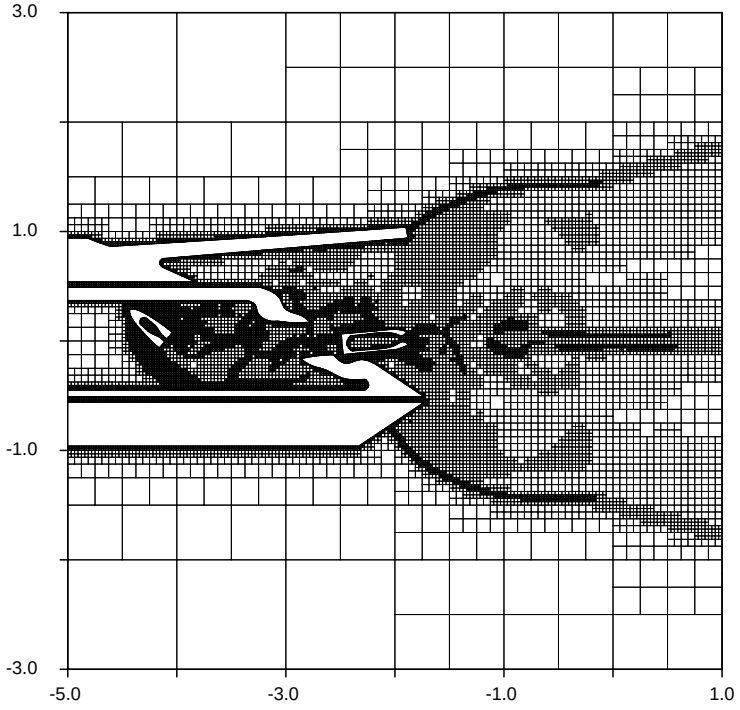

Figure 8.107: The grid plot corresponding to the contour-line plot of Figure 8.106.

if the minimum distance between any two boundaries falls below a certain threshold. The specific formulation for the contact force used in this work is given by the equation

$$\vec{F}_c = \begin{cases} \vec{0}, & \vec{x} \geq \vec{x}_0; \\ -k(\vec{x} - \vec{x}_0), & \vec{x} < \vec{x}_0, \end{cases}$$

where $\vec{F}_c$ is the "remotely-applied" elastic contact force, k is the representative stiffness of the contact resistance, $\vec{x}$ is the actual "minimum" displacement vector between the two boundaries that are coming into contact, and $\vec{x}_0$ is a reference displacement vector between these two boundaries, corresponding to the unloaded state of the "remotely-applied" contact resistance. The formulation expressed in the above equation clearly ensures that the contact force is applied only in a compressive "mode", to prevent boundaries from coming together, and only if the distance between the two boundaries involved falls below a threshold value, also clearly given by $\vec{x}_0$. It should be noted that use of the standard arithmetic inequalities to relate vectors as done above has unambiguous and definite meanings because the vectors involved are parallel.

The value of $\vec{x}$ in the contact resistance equation may be determined by finding the minimum distance between every spline in one of the boundaries and every spline in the other boundary. However, doing this exhaustively is computationally inefficient, and is therefore not recommended here since these minimum distances must be re-evaluated after each time step. Instead, a more efficient technique that relies on evaluating and then installing the bounding boxes of the splines in tree data-structures is used in this work. The values of $\vec{x}_0$ and k in the above equation may be pre-specified independently for every pair of boundaries in the computation.

The "remotely-applied elastic contact force model" described above is utilized for two purposes in this test case: (i) keeping the front and rear parts of the projectile

separated by a certain minimum distance during the combined acceleration period; and, (ii) keeping the front part of the projectile separated by a small distance from the upper and lower parts of the front wall of the target during the impact process. Both of these forced "separations" can be seen in all the accompanying contour-line and grid plots, but with greater clarity in the zoom plots.

The simplest guide to choosing the value of $\vec{x}_0$ to reliably prevent geometrical contact between different boundaries in the generic problem is to relate its magnitude to the dimension of the largest computational cells allowed on boundaries. In this test case, for example, $\vec{x}_0$ is chosen to lie along the outward-pointing normal of the boundary, and is given a magnitude that equals the width of about three cells at the "intersection refinement level" (which is the coarsest refinement level allowed for cells intersected by a boundary, as explained in Chapter VI).

The simplest guide to choosing the value of k for a pair of boundaries is that it must be large enough to keep the pair of boundaries apart. In this test case, for example, the value of k required to separate the front part of the projectile from the walls of the target is much larger than the value of k required to keep the two parts of the projectile apart during the acceleration of the projectile. It should also be noted that in principle, the use of a separating force based on the formulation adopted here in conjunction with an explicit time-integration scheme, as also used here, cannot guarantee that boundaries will not geometrically collide if $\vec{x}_0$ is allowed to go below the width of a local cell separating two boundaries.

The two parts of the projectile are treated as rigid bodies throughout the computation. The target, which consists of three parts, as can be seen in the accompanying figures, is treated as rigid in some places, and as deformable in others. In particular, the lower and upper parts of the target are treated as stationary, rigid objects whose

shapes and locations are not affected by the impact, while the middle part of the target is treated as a rigid solid in some zones, and as a deformable solid in others. More specifically, the rear (vertical) wall and the top and bottom (horizontal) walls of the middle part are all treated as rigid, stationary objects, while the upper and lower front walls (that is, the front wall portions lying immediately above and below the slit) are treated as elasto-plastic, deformable solids.

The model used in this work for predicting the internal forces and the dynamic response of deformable solids can be described as an "elasto-plastic dynamic-deformation force model". The model represents a solid body or structural member as a set of point masses connected by elasto-plastic springs. The mass-spring system occupies the same spatial region as the solid or structural member it represents, with the point masses placed at the boundaries of the solid and within its interior, and with the springs forming a network of triangles whose sides lie wholly on the boundaries of the solid or within its interior. In this specific test case, the masses of the deformable upper and lower parts of the front wall of the target are manually discretized into a set of about 120 point masses each, distributed in three rows: one each along the inner and outer boundaries of the wall part, and one along the center-line of the wall part. The discrete masses are given the same total mass as the wall part, taken to be composed of steel, and having the dimensions discernible most clearly in the grid plots of this test case. The spring connections between the point masses are also manually established, and the elasto-plastic springs are given individual stiffnesses and yield strengths so that composite system of springs has the same stiffness and yield strength as the wall (taken to be composed of steel), in pure extension (and compression). The resistance of each elasto-plastic spring in the

model is expressed by the conditional equation

$$\vec{F}_s = \begin{cases} \vec{F}_{s-}, & \vec{x} \geq \vec{x}_+; \\ -k(\vec{x} - \vec{x}_0), & \vec{x}_- < \vec{x} < \vec{x}_+; \\ \vec{F}_{s+}, & \vec{x} \leq \vec{x}_-. \end{cases}$$

where $\vec{F}_{s-}$ and $\vec{F}_{s+}$ represent respectively the maximum tensile and compressive forces in the spring, where $\vec{x}_-$ and $\vec{x}_+$ represent the loaded lengths at respectively the elastic limits of the compressive and extensive loads in the spring, and where all other symbols have the same denotations given for them in the description of the contact-force model. Again, the inequalities relating the displacement vectors have definite and unambiguous meanings because all three displacement vectors defined above are always parallel. The relation expressed in the above equation represents what is perhaps the simplest non-strain-hardening, elasto-plastic behavior. The springs do not provide any torsional stiffness; they only resist extension or compression from their original, unloaded lengths. Nevertheless, the composite spring system can support bending moments and shear loads, and with an accuracy that improves with increasing spatial resolution of the discretization.

The forces and moments from each of the three types of force modeled in this test case; namely, the aerodynamic, contact, and solid-dynamic types, that are applied to each control point in each boundary in the test case are computed and added together to provide the total force applied to that control point. The appropriate equations of motion are then evolved by time-integration. As described in detail in Chapter V, the specific treatment of the equation(s) of motion depends on whether the boundary is rigid or deformable. If the boundary is rigid, the forces that are applied to the individual control points of the boundary are summed and pivoted to give a single global force and a single global moment for the entire boundary,

and the single force and single moment so obtained are then integrated to evolve the equation of motion of the rigid boundary to determine its trajectory. If the boundary is deformable, the equation of motion of each control point, which in this test case coincides with a point mass in the mass-spring system, is independently time-integrated to give the independent trajectory of that control point under the action of the forces directly applied to it. The independence of the trajectories of the individual control points allows the complete dynamic response (including the bulk motion and local relative deformation), of a composite elasto-plastic mass-spring system, such as the one described above, to be modeled accurately. In order to keep the structural-dynamic model as simple as possible in this test case, the connectivity patterns of the springs are not allowed to change during a deformation, no matter how severe the deformation becomes, and no account is taken of any coalescence, disintegration, or inter-penetration effects in boundaries.

The Boundary Conditions on all four sides of the Root Square for this test case are specified to be of the "inflow-outflow" type described for the Propulsive Separation test case presented in the preceding sub-section. The reasons for choosing these Boundary Conditions for this test case are the same as the corresponding reasons cited for the Propulsive Separation test case.

The Initial Conditions for this test case are uniform, with the pressure and temperature respectively set to 1.0e+05N/m^2 and 300K everywhere in the Computational Region except in the cavity of the rear part of the projectile: there, the thermodynamic state, specified in terms of the ratios of the pressure and the temperature relative to the background state, are respectively 100 and 10. The pressurized region occupies the entire cavity, extending up to to the throat (that is, to the smallest section) of the nozzle of that cavity. The initial velocity of the gas throughout the

Computational Region, including the pressurized cavity sub-region, is set to zero. When the speed of the projectile reaches what corresponds to a Mach Number of approximately 2.5, the thermodynamic state in the front cavity is instantaneously re-initialized to have pressure and temperature ratios (relative to the uniform initial background state) of 50 and 5 respectively, but without any alteration in the velocity distribution computed in the cavity at the time of re-initialization. The Initial Conditions applied in the cavity of the rear part of the projectile, and the re-initialization applied in the cavity of the front part of the projectile are clearly intended to simulate the effects of rocket engines.

The computations for this test case are performed with second-order accuracy in time and space, using the Least Squares procedure for reconstruction of the gradients, and the Venkatakrishnan Limiter for limiting of the gradients. The Numerical Flux Functions are computed with the Exact Riemann Solver. These techniques and procedures are described in detail in Chapter III.

Figures 8.98 and 8.99 show the two-stage projectile as it accelerates towards the target, shortly after its release from rest. The figures show the blast wave created from the rear-ward expulsion of the compressed gas from the cavity of the rear part of the projectile, and show the shock wave and expansion fan systems created ahead of the front part of the projectile which has already attained a supersonic speed.

Figures 8.100 and 8.101 show the motion and evolution of the blast system created from release of the compressed gas in the cavity of the rear part of the projectile, and the shock and expansion system created around the front part of the cavity, just after the re-initialization of the pressure and temperature in the front cavity. The release of the compressed gas from the front cavity is barely visible, but its effect can most easily be detected by the increased separation between the front and rear

parts of the cavity as the expelled gas acts to accelerate the front part of the cavity and to decelerate the rear part of the cavity.

Figures 8.102 and 8.103 show the motion and evolution of the blast systems created from release of the compressed gas from the cavities of both the front and the rear parts of the the projectile. The circular shape of the leading shock wave from the blast system emitted from the front cavity is especially evident. The figures also show that at the given time, the front part of the projectile has just made contact with the wall of the target, as evident from the compression and expansion waves emitted on the opposite sides of the lower front wall of the target, as that wall accelerates into the enclosure of the target under the effect of the impact force.

Figures 8.104 and 8.105 show the continued motion and evolution of the blast systems created from release of the compressed gas from the cavities of both the front and the rear parts of the the projectile, and again, the circular shape of the leading shock of the blast system from the front cavity is especially evident. The figures also show that at the given time, the impact of the front part of the projectile has extensively deformed the lower front wall of the target, while it has relatively only slightly deformed the upper front wall. Again, the rapid acceleration of the walls is evident from the strong compression and expansion waves created on opposite sides of these walls. The differences in the behavior and in the extent of deformation of the upper and lower front walls are attributable to the angle at which the impact force is applied to these walls: for the lower front wall, the geometry of the wall and the projectile at the slit are such that the impact force is largely perpendicular to the center-line of the wall, while for the upper front wall, the impact force is largely parallel to the center-line of the wall. For this reason, the lower front wall appears to deform up to the point in time shown predominantly in a bending mode, while the

upper front wall appears to deform up to the point in time shown predominantly in a buckling mode.

Figures 8.106 and 8.107 show the continued motion and evolution of the blast systems created from release of the compressed gas from the cavities of both the front and the rear parts of the the projectile, and the continued deformation of the walls of the target under the action of the applied impact and aerodynamic forces. At the time captured in these two figures, the elasto-plastic walls have deformed appreciably, but with the initial buckling deformation mode in the upper front wall giving way to a more dominant bending deformation mode. These figures also show that the front portion of the projectile is spinning at a relatively high angular speed as it propagates along the interior of the target. The implied angular momentum is actually generated by the asymmetry of the force system applied to the front part of the projectile during the impact process. This asymmetry arises because the projectile makes contact with the upper and lower front walls at slightly different times, and with differently-directed contact forces. These figures also show the reflections and diffractions of the expanding and traveling blast systems against the outer and inner walls of the target.

The grid plots of Figures 8.99, 8.101, 8.103, 8.105, and 8.107 clearly show the responsiveness of the solution-adaptation methodology to the motion and deformation of boundaries and to the evolution of the flow features, and clearly show how the "packing" or density distribution of the computational cells in the Computational Region closely matches the desired resolution at all times during the evolution of the solution.

This test case demonstrates the automatic handling of boundaries that move and deform under different types of force, including the structural-dynamic, contact,

and aerodynamic types of force. However, because the emphasis in this test case is on simulating the motion and deformation of the various boundaries involved, rather than on fully resolving the gasdynamic phenomena involved, the maximum refinement level is lower than that used for most of the computations shown in the preceding sub-sections, and the flowfield is resolved to a far lesser extent than in those test cases. The total number of cells for this test case ranged from about 20K near the start of the computation to about 70K near the end of the computation.

Several different variants of this test case, some with topological transformations in the boundaries, were also tried and evaluated in a similar manner to that described above.

8.5.3.2 Projectile Flight and Impact, II

The test case presented in this sub-sub-section, like the test case presented in the preceding sub-sub-section, simulates the acceleration from rest of a two-stage projectile under the thrust of its first-stage rocket motor, followed by separation of the two stages of the projectile under the thrust of its second-stage motor, followed by collision of the second stage of the projectile with a target, resulting in the breaching of that target.

As would be expected from the similarity outlined in the preceding paragraph, the key physical phenomena involved in the test case of this sub-sub-section are similar to those involved in the test case of the preceding sub-sub-section. There are differences between the two test cases in the geometry target (and hence in the stiffnesses of, and the mass distribution within the target), in the geometry and the masses of the front and rear parts of the projectile, and in some of the physical properties and the computational sub-models used, most notably in the greater sophistication

of the model for dynamic solid deformation used in the test case of this sub-sub-section. The test case of this sub-sub-section is also more resolved in space and time than the test case of the preceding sub-sub-section. However, all these physical and computational differences (and others like them) between the two test cases are functionally of little consequence. Rather, the similarities between the two test cases are so numerous and fundamental that this test case is most conveniently and compactly described as a variant of its predecessor.

The most important difference between the test case of this sub-sub-section and the test case of the preceding sub-sub-section is not in the physical phenomena involved, but in the modeling capabilities used: the test case of this sub-sub-section, unlike its predecessor, allows and simulates topologic transformations in boundaries, and incorporates no special arrangements to circumvent such transformations. In particular, three topologic transformations occur in the boundaries of the test of this sub-sub-section. The first of these arises in the separation of the two stages of the projectile, which is originally a single body, and which disintegrates into two bodies during the separation process. The second of these topologic transformations arises during the impact of the front part of the projectile with the target, which results in the fusing or coalescence of the front part of the projectile and the target into a single body. The third of these topologic transformations arises when the front part of the projectile is separated from the wall of the target after piercing the wall, causing the two objects which were united together into a single object by the impact to regain their original identities as two distinct objects.

The figures 8.108, 8.110, 8.112, 8.114, 8.116, 8.118, and 8.120 show the contour-line plots of the density distribution in the Computational Region, with about 50 contour lines in each, at several times during the computation, while the Figures

8.109, 8.111, 8.113, 8.115, 8.117, 8.119, and 8.121 show the grid plots corresponding respectively to the contour line plots just cited. These seven pairs of corresponding figures also show the geometry and the dimensions of the projectile and the target, and show the overall flight trajectory of the projectile and the impact process described above. In order to display the most important events as clearly as possible, some of these pairs of figures zoom in on different portions of the Computational Region. These figures are next described in more detail.

Figures 8.108 and 8.109 show the two-stage projectile, in its original form as a single body, as it accelerates towards the target, shortly after its release from rest. The figures show the complex blast wave system created from the rear-ward expulsion of the compressed gas originally stored in the cavity of the rear part of the projectile, and show the shock wave and expansion fan systems created in the vicinity of the front part of the projectile which has already attained a supersonic speed at the time captured in the figures.

Figures 8.110 and 8.111 show the motion and evolution of the blast wave system created from the release of the compressed gas from the cavity of the rear part of the projectile, and the motion and evolution of the shock wave and expansion fan systems created around the front part of the cavity, as the projectile continues to accelerate towards the target. The figures show the solution shortly before the commencement of the separation of the projectile.

Figures 8.112 and 8.113 show the density-distribution solution shortly after the separation of the projectile into two parts. The figures show the motion and evolution of the blast wave system created from the release of the compressed gas from the cavity of the rear part of the projectile, the shock wave and expansion fan systems created around the front part of the cavity, and the second blast wave system created

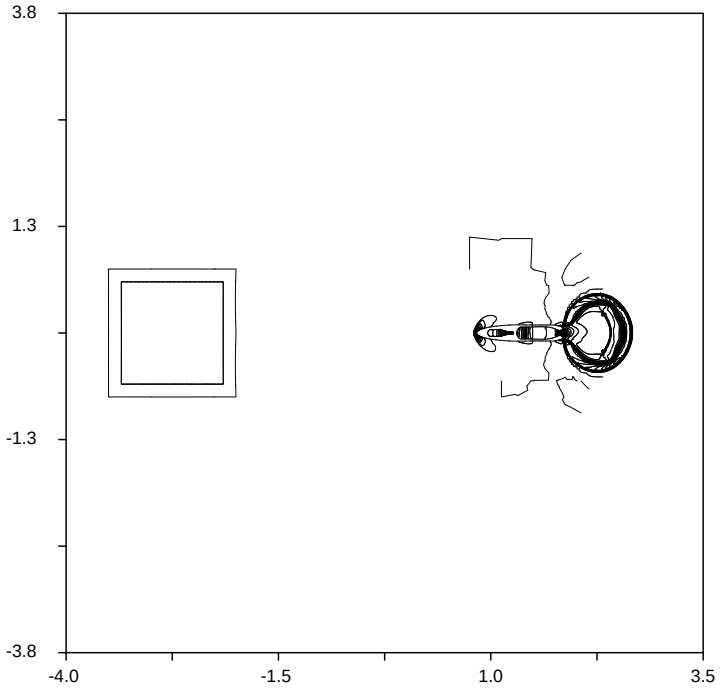

Figure 8.108: A contour-line plot of the density distribution shortly after the release from rest of a two-stage projectile towards its target.

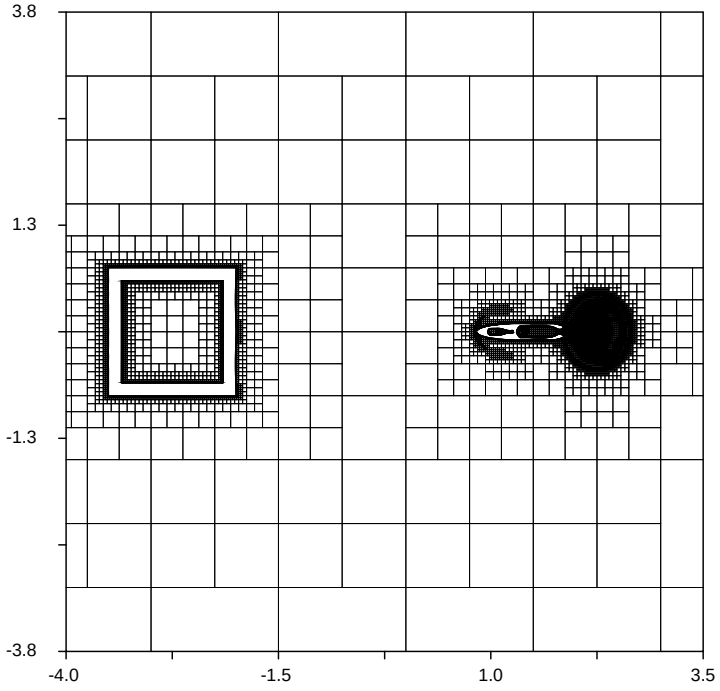

Figure 8.109: The grid plot corresponding to the contour-line plot of Figure 8.108.

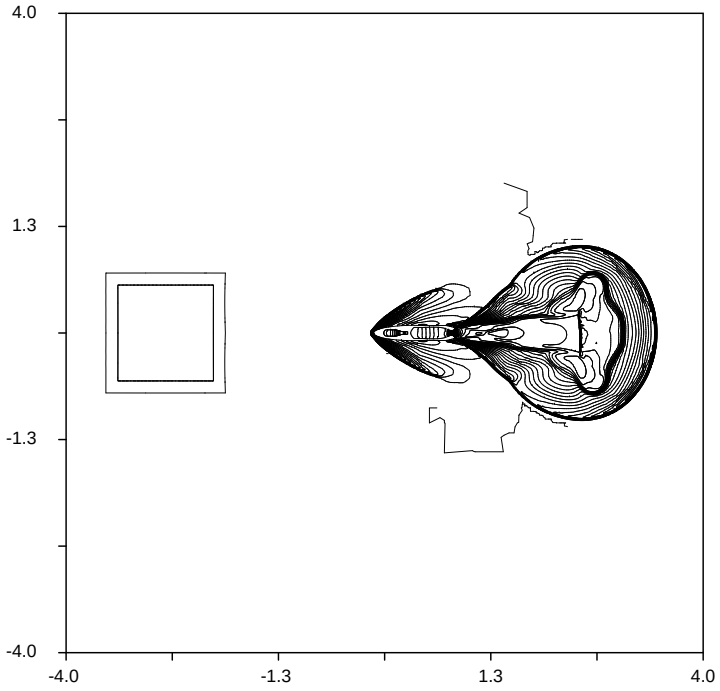

Figure 8.110: A contour-line plot of the density distribution just before the attainment of the separation velocity by a two-stage projectile.

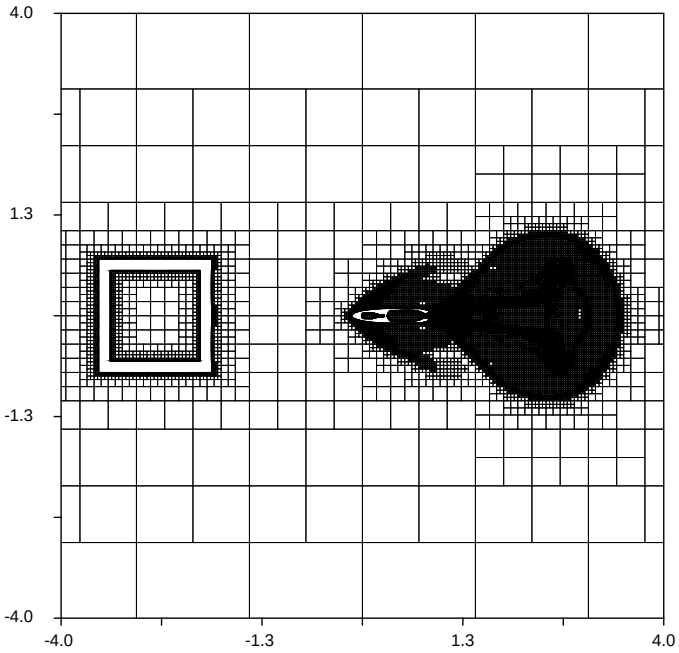

Figure 8.111: The grid plot corresponding to the contour-line plot of Figure 8.110.

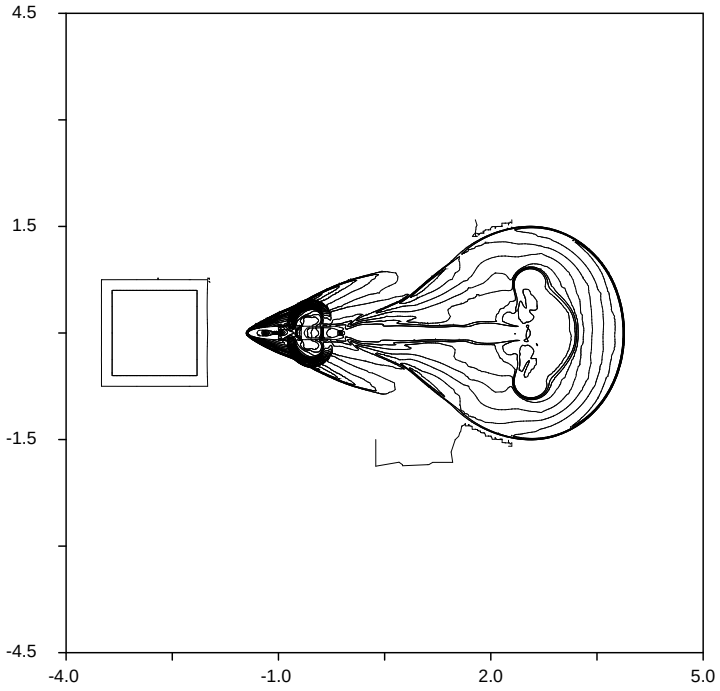

Figure 8.112: A contour-line plot of the density distribution shortly after the separation of the front and rear parts of a two-stage projectile.

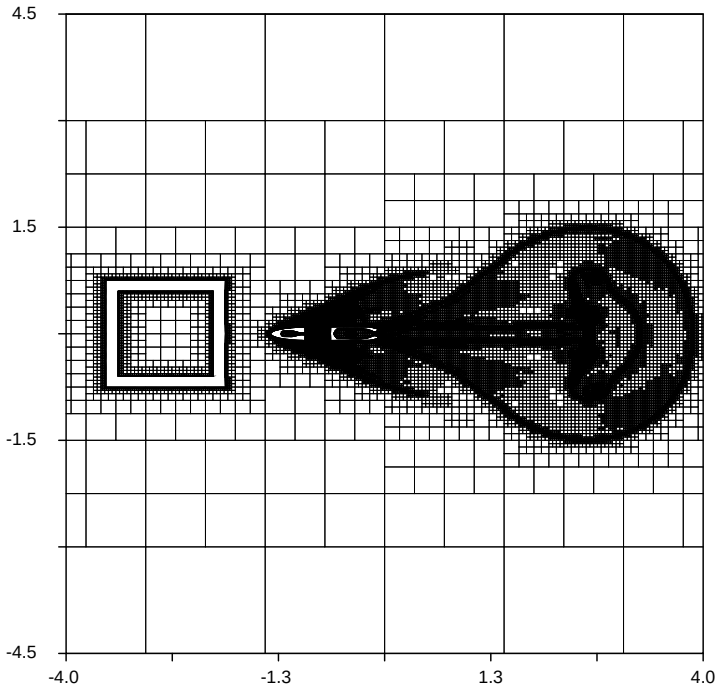

Figure 8.113: The grid plot corresponding to the contour-line plot of Figure 8.112.

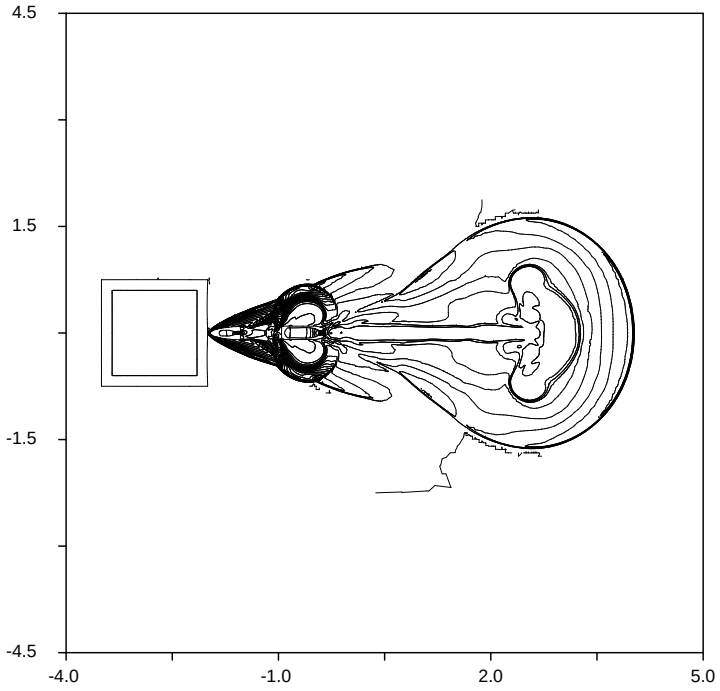

Figure 8.114: A contour-line plot of the density distribution shortly before the collision of the front part of a two-stage projectile with its target.

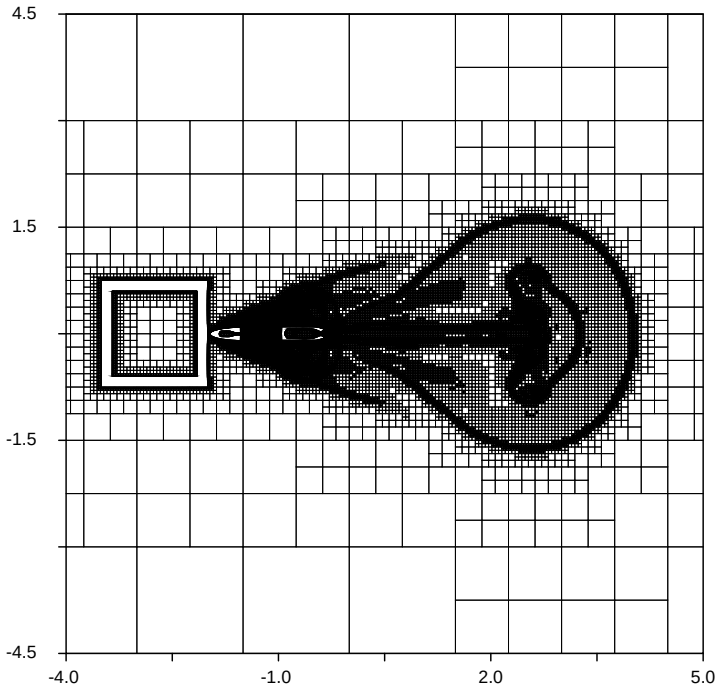

Figure 8.115: The grid plot corresponding to the contour-line plot of Figure 8.114.

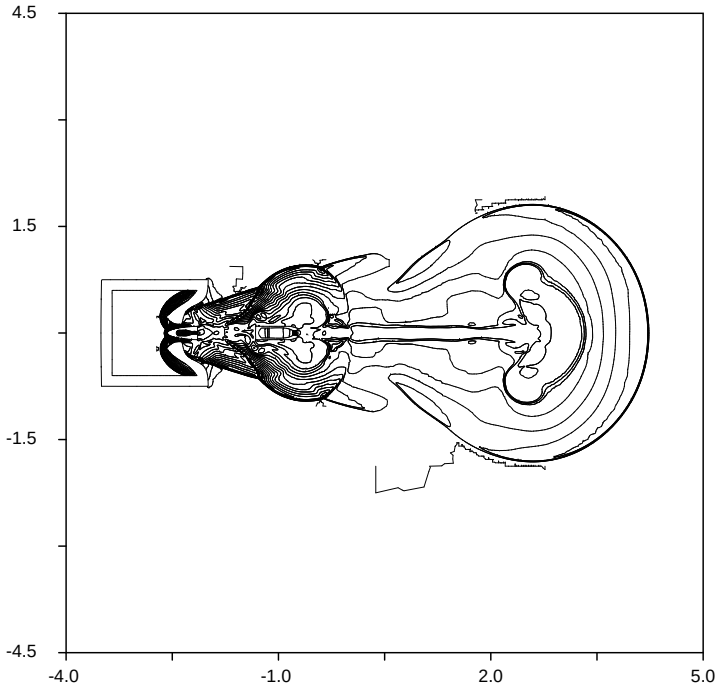

Figure 8.116: A contour-line plot of the density distribution shortly after penetration of a target by the front part of a two-stage projectile.

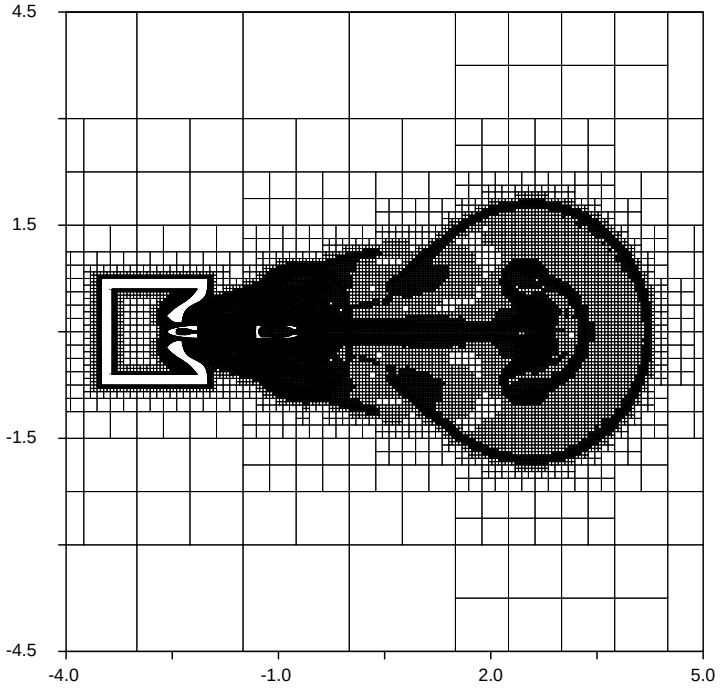

Figure 8.117: The grid plot corresponding to the contour-line plot of Figure 8.116.

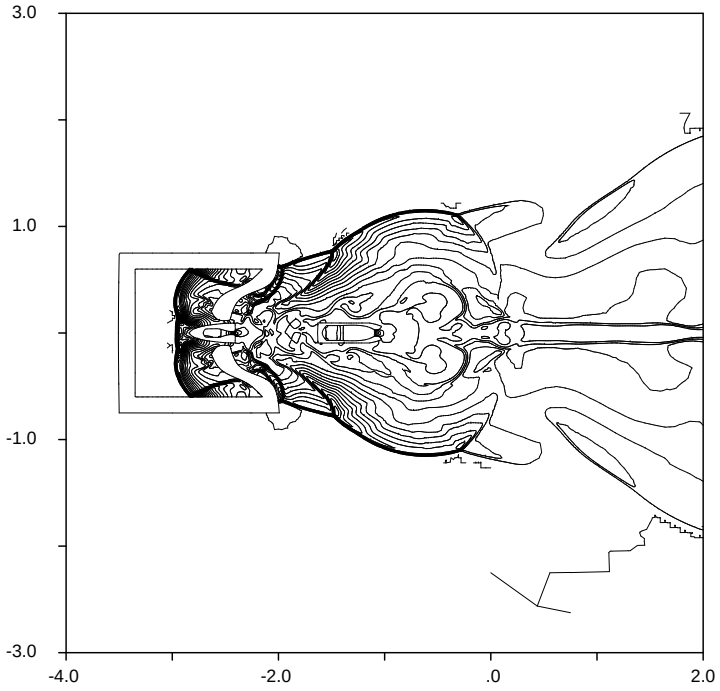

Figure 8.118: A contour-line plot of the density distribution shortly after the end of deformation of a target after being struck by a two-stage projectile.

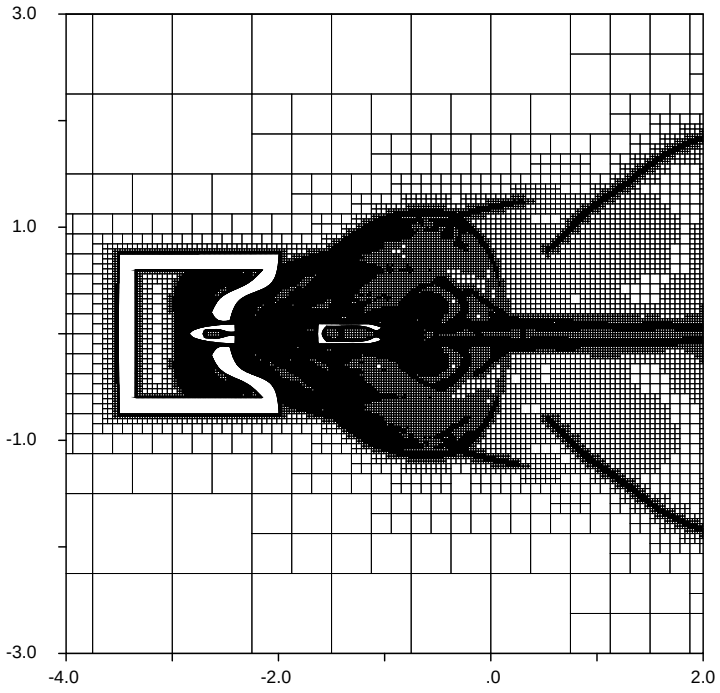

Figure 8.119: The grid plot corresponding to the contour-line plot of Figure 8.118.

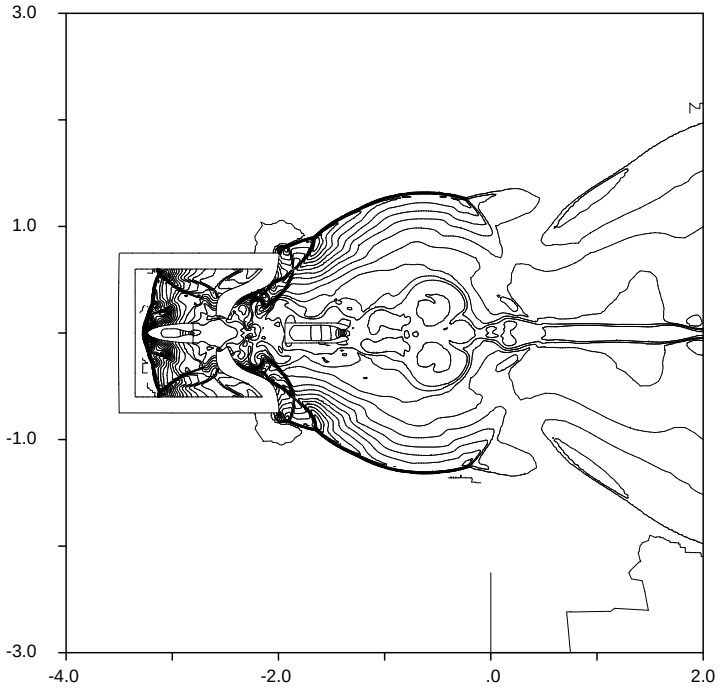

Figure 8.120: A contour-line plot of the density distribution shortly before the front part of a two-stage projectile reaches the rear wall of its target.

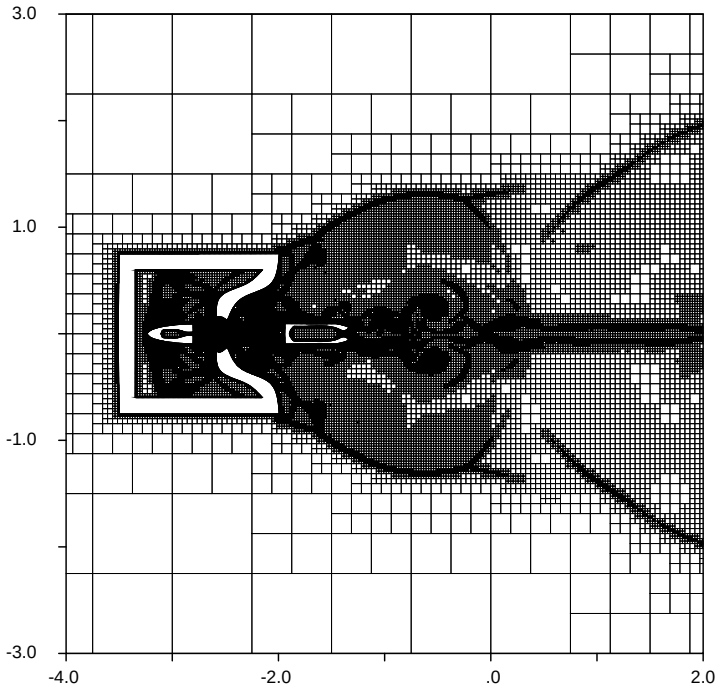

Figure 8.121: The grid plot corresponding to the contour-line plot of Figure 8.120.

from the release of the compressed gas from the cavity of the front part of the projectile. As the figures show, this release occurs initially across the separation gap between the two parts of the projectile, and starts only after the projectile separates into two parts. At the time captured in the figures, the second blast system, which has a characteristic elliptical shape, roughly centered about the point at which the separation starts, has expanded far enough to begin interacting with the shock wave and expansion fan systems emanating from the front of the second stage of the projectile, deflecting the fan and the leading shock of the latter systems in the process.

Figures 8.114 and 8.115 show the continued motion and evolution of the blast wave systems created from the release of the compressed gas from the cavities of both the front and the rear parts of the projectile. The figures show the effects of the interaction of the second blast system with not only the shock wave and expansion fan systems emanating from the leading edge region of the front part of the projectile, but also with the moving rear part of the projectile. The figures show the solution shortly before the front part of the projectile collides with the walls of the target, and before the coalescence of the boundaries of the latter two objects begins.

Figures 8.116 and 8.117 show the solution after the front part of the projectile has re-separated from the walls of the target and regained its identity as a separate object, distinct from the target, with which it had coalesced into one object during the impact. The figures show the continued motion and evolution of the blast wave systems created from the release of the compressed gas from the cavities of both the front and the rear parts of the projectile, and the complex flowfield developing as a result of the interactions of these blast wave systems with each other and with the

moving boundaries. The figures also show that the walls of the target have undergone large-scale deformation, even including a change in the thickness distribution along the length of the deformable part of the wall. The figures also show the strong shock waves emitted from the inner sides of the front walls of the target as they rapidly deform into the interior of the target under the effect of the forces created by the impact with the front part of the projectile.

Figures 8.118 and 8.119 show the solution after further deformation of the front walls of the target and further penetration of the projectile into the interior of the target. The figures show the continued motion and evolution of the blast wave systems created from the release of the compressed gas from the cavities of both the front and the rear parts of the the projectile, and again, show the interactions of these systems with each other, with the front walls of the target, and with the expansion fan systems created on the outer sides of the front walls of the target as a result of the motion of these walls into the interior of the target. The figures also show how the shock waves created in the enclosure of the target by the inward motion of the walls of the target have begun to reflect, first against the inner sides of the upper and lower walls of the target, and then against the inner sides of the deformed front walls of the target. Three different pairs of Mach Stems developing on the inner and outer sides of the walls of the target as a result of shock wave reflections against these walls are clearly visible in these figures.

Figures 8.120 and 8.121 show the continued motion and evolution of the blast wave systems created from the release of the compressed gas from the cavities of both the front and the rear parts of the projectile, and the interaction of these blast systems with each other and with the features emitted by the moving walls of the target. The figures show the complex reflection and refraction patterns created on

the outer sides of the front walls of the target, and within the enclosure of the target. At the point in time captured in the figures, the bulk deformation of the front walls of the target has almost completely ceased, but some residual vibration continues further well into the calculation.

The grid plots of Figures 8.109, 8.111, 8.113, 8.115, 8.117, 8.119, and 8.121, just like their analogs for the test case of the preceding sub-sub-section, again clearly reveal the effectiveness of the solution-adaptation methodology, and its responsiveness to the motion and deformation of boundaries and to the evolution of the flow features, and clearly show how the density distribution of the computational cells in the Computational Region remains closely matched to the desired resolution distribution at all times during the evolution of the solution.

The Numerical Flux Function, and the reconstruction and limiting algorithms chosen for the test case of this sub-sub-section are identical to the corresponding ones chosen for the test case of the preceding sub-sub-section.

The Boundary Conditions and Procedures applied on all four sides of the Root Square for the test case of this sub-sub-section correspond to the "inflow-outflow" type, as for the test cases of the preceding two sub-sub-sections. The reasons for choosing these specific Boundary Conditions and Procedures for the test case of this sub-sub-section are also identical to the reasons for choosing them for the two other test cases just cited, and these reasons are explained above, in the discussion of the Propulsive Separation test case.

The Initial Conditions (and re-initializations) of the test case of this sub-sub-section are also identical to those of the Projectile Flight and Impact test case of the preceding sub-sub-section, except for one difference: the pressurization of the cavity of the second stage is applied at the start of the computation for the test

case of this sub-sub-section, instead of when the projectile attains a speed of Mach 2.5 for the test case of preceding sub-sub-section. The pressure and temperature ratios relative to the background state in the cavity of the second stage; namely, 50 and 5 respectively, are identical for the two Projectile Flight and Impact test cases (and, as implied above, so are the pressure and temperature ratios in the cavity of the first stage). The cavity of the second stage can be pressurized at the start of the computation in the test case of this sub-sub-section because the first and second stages of the projectile are initially attached to each other (unlike the situation for the test case of the preceding sub-sub-section), allowing the pressurized gas in the second stage to be fully confined.

The pressurization of the cavity of the second stage of the projectile at the start of the computation in the test case of this sub-sub-section plays a more profound role in the separation of the projectile than does the mid-flight pressurization of that cavity in the test case of the preceding sub-sub-section. In particular, the separation event in the test case of this sub-sub-section, which is modeled as a "fracture" of the projectile, occurs as a direct result of the tensile force established across the junction between the front and rear parts of the projectile by the pressure in the cavity of the front part of the projectile. The fracture is initiated during the flight of the projectile as a result of a sudden reduction in the local stiffness and strength of the material at the junction between the first and second stages of the projectile, which is programmed to occur when the Mach Number of the projectile reaches 2.5, as explained in more detail below. This mechanism of simulating the separation process, which requires the handling of topologic transformations, is more realistic than the corresponding mechanism adopted in the test case of the preceding sub-sub-section.

As mentioned above, one difference between the test case of this sub-sub-section and the test case of the preceding sub-sub-section is that contact between different boundaries is allowed in the test case of this sub-sub-section, making it unnecessary to apply all contact forces remotely. Moreover, because coalescence between different boundaries is modeled "at the physical level" in the test case of this sub-sub-section, as will be discussed in more detail below, no explicit model is even needed for contact forces, and no such model is therefore used.

Another difference between the test case of this sub-sub-section and the test case of the preceding sub-sub-section is that the projectile is not modeled as a rigid body in the test of this sub-sub-section. Instead, it is modeled as a deformable body, but with a far higher stiffness than that of the walls of the target. The need to model the projectile as a deformable body in the test case of this sub-sub-section arises for two reasons. The first reason is that the fracture model used in the test case requires some deformation in the material of the projectile in order to enable its separation into two parts, as explained in more detail below. The second reason is that after the front part of the projectile combines with the walls of the target, the combined body must move and deform as a single object with a single, unified model and treatment procedure for predicting its dynamic response. The alternative of treating the projectile as a rigid object which combines with a deformable solid to form a "hybrid", moving solid with some parts rigid, and others deformable is also possible, but is far more laborious to implement than a unified-treatment model.

The treatment for the walls of the target in the test case of this sub-sub-section is analogous to the treatment for the walls of the middle part of the target in the test case of the preceding sub-sub-section. In particular, only the front wall of the target in the test case of this sub-sub-section is treated as a deformable solid, while all the

other (three) walls of the target are treated as stationary, rigid bodies. The combining together of a rigid part and a deformable part of a body in this manner does not present any of the difficulties described in the preceding paragraph for the combining together of a moving rigid projectile with a deformable solid. This is because any rigid parts of the target wall, unlike a moving rigid projectile, remain stationary, and therefore do not require any special or non-unified treatment to correctly compute their trajectory or dynamic response while attached to a deformable solid. Instead, any forces applied to the stationary, rigid parts of the target wall are ignored, and no updates to their accelerations, velocities, or locations are even computed.

The motions and deformations of deformable solids are computed in the test case of this sub-sub-section using the same mass-spring model used for computing such motions and deformations in the test case of preceding sub-sub-section, except for the introduction of two new additional capabilities: (i) a capability to model the effects of plastic fracture; and, (ii) a capability to add new springs to form new connections between pre-existing point masses, in order to enable the modeling of the physical effects of topologic transformations. The mass-spring model is described in the preceding sub-sub-section.

The time-integration and solution procedures used to evolve the Equations of Motion of the control points and the boundaries in the test case of this sub-sub-section are identical to the corresponding ones used for the test case of the preceding sub-sub-section, and are completely unaltered by the presence of topologic transformations or by the introduction of a model for plastic fracture.

The effect of plastic fracture is modeled in the test case of this sub-sub-section by "rupturing" any spring (connecting any two point masses) if the strain in that spring, which is given by the expression $\epsilon = \frac{l-l_0}{l_0}$, where l and l_0 are respectively the

current and the (unloaded) original lengths of the spring, exceeds a certain threshold, called ϵ_{max}, and here set to 1.4. Since ϵ_{max} is chosen so that is satisfies $\epsilon_{max} > 1$, the springs are ruptured only in tensile extension. The rupturing of a spring is computationally modeled by eliminating it from the mass-spring system, so that it no longer contributes to the internal forces generated in a solid. No provisions are made here for re-introducing the spring into the model if the strain reverses and falls below 1. This implies that the compressive resistance of a fractured material is ignored in the model. Despite the overall simplicity of the fracture model adopted here and its many limitations and shortcomings, some of which can be easily relaxed or remedied if desired, the model is adequate for the purposes of this test case.

The plastic fracture model described above is used in simulating the fracture of the projectile into two parts, and in simulating the fracture of the wall of the target and the re-separation of the second stage of the projectile from this wall after their combination together during the impact process.

The separation of the projectile, which is originally a single body, into two parts is modeled in the test case of this sub-sub-section, as outlined above, by instantaneously reducing the stiffness of every spring that connects a point mass in what would become the front part of the projectile with a point mass in what would become the rear part of the projectile. This is done when the Mach Number reaches 2.5. The result of reducing the stiffness in those springs is that the internal pressure in the cavity of the front part of the projectile (which is about 50 atmospheres) causes the affected springs to extend beyond their plastic fracture limits, resulting in disruption of the corresponding point-mass-to-point-mass connections as explained above. This action, however, only removes the "physical bond" between the material of the front part of the projectile and the material of the rear part of the projectile, and is alone

not sufficient to effect a topological transformation in which the single-body projectile is separated into two new bodies. This topological transformation is accomplished by also disrupting the connectivities between the appropriate control points as the distance between them increases, as explained in detail in Chapter V, and by also explicitly changing the data-structure representing the boundary geometry to reflect the creation of a new body, as also explained in detail in Chapter V.

The coalescence of the front part of the projectile with the front wall of the target is modeled by introducing appropriate new, data-structural connections between some of the control points of the front part of the projectile and some of the control points of the front wall of the target as the distance between these control points falls below the dimensions of the local computational cells separating the two boundaries, and also by eliminating some of the control points in these two boundaries. The number of boundaries or bodies in the computation is also automatically reduced by one. The overall procedure is explained in detail in Chapter V. However, this introduction of new data-structural connections and changing of the number of bodies in the computation only models the topological aspect of the coalescence; it is also necessary to model the physical aspects of the coalescence. This is done by introducing new springs between at least all the point masses whose control points have been newly connected together as part of the topological transformation that reflects the coalescence. These new springs must be introduced to reflect the natural resistance to compression or extension of any element within a solid, including the new solid that is "formed" from the combination of two originally-distinct solids.

The new springs that are introduced into the mass-spring representation of a deformable solid as a result of a coalescence type of topologic transformation, as described in the preceding paragraph, introduce a material resistance to the penetration

of one body into another, effectively representing the effect of contact resistance, and therefore making it unnecessary to add an external contact-force model, as mentioned above. It is recognized, however, that the physical accuracy of the ad hoc procedure of introducing springs in the manner described above is expected to be low, and to be highly dependent on the geometric discretization of the boundaries involved.

After the coalescence of the front part of the projectile and the front wall of the target into a single body, the resulting combined body continues to move and deform as a result of the kinetic energy carried by the point masses originally belonging to the front part of the projectile.

As the front wall of the target continues to deform, its inner side eventually exceeds its plastic fracture limit and starts rupturing, almost directly across that wall from the impact point. This rupturing is accompanied by automatic reconnection of the affected control points, so as to simultaneously modify the local geometric definition of the wall to account for the formation of a local feature resembling a crack. Meanwhile, the point masses that originally belonged to the front part of the projectile continue to move forward at a higher speed than the point masses that originally belonged to the wall of the target, and eventually the spring connections formed between the two sets of point masses also extend beyond their plastic limits and start rupturing. As the distances between the control points originally belonging to the front part of the projectile and the control points originally belonging to the walls of the target continue to increase, the individual spline connections that were established between these two sets of points as part of the coalescence process are also eventually disrupted, as explained in detail in Chapter V, as part of the process of disintegration of a boundary into separate ones.

Eventually, the control points that originally belonged to the front part of the

projectile are reconstituted back into a single body having a shape very similar to that of the original projectile, while the control points that originally belonged to the wall of the target are re-connected back together to form the boundary of the breached and deformed wall of the target. Because the stiffness of the material of the projectile is much higher than the stiffness of the material of the target, the reconstituted front part of the projectile largely retains its shape, while the walls of the target suffer significant permanent deformation.

The models described above for simulating topologic transformations and dynamic deformations reflect many of the important physical phenomena involved in the corresponding physical problem. They also appear to give physically plausible results, but this was only achieved, however, after significant trial-and-error testing and calibration of the various spring properties and the spline connection and disruption algorithms. For example, the re-separation of the front part of the projectile from the wall of the target so that it largely regains its original shape is achieved by making the stiffnesses of the springs connecting the point masses in the front part of the projectile about five orders of magnitude higher than the stiffnesses of the springs connecting the point masses of in the walls of the target, even though this implies the assignment of material properties that are not physically realistic. It is also necessary to avoid making certain spring connections to prevent unrealistic fractures in the walls of the target, and also to prevent the separation of small fragments from the walls of the target. The clear implication of all the trial-and-error, and adjustment and calibration required is that the specific parameters used in the various models are customized for this test case, and are not applicable to the generic problem. An even more important implication is that the dynamic deformation model described above is not a general-purpose model. The creation of a generic model for elasto-

plastic deformation and boundary coalescence and break-up is, however, outside the scope of this work as given in Chapter I.

The test case presented in this sub-sub-section demonstrates the automatic handling of motion and deformation of boundaries, and the automatic handling of topologic transformations in boundaries. The test case specifically demonstrates the ability of the computational methodology developed in this work to handle computational simulations involving approach, separation, or impact of bodies or boundaries, even if the number of bodies or boundaries changes during the computation. The test case also demonstrates the ability of the methodology to handle problems with fluid-structure interactions. In terms of the computational-fluid-dynamic capabilities utilized, the test case presented in this sub-sub-section belongs to the most general category that can be attempted with the methodology developed in this work, and also belongs, again from the fluid-dynamic viewpoint, to the most general category possible in two-dimensional, inviscid, compressible flow. As explained above, however, the specific sub-models used in this test case to model the non-fluid-dynamic parts of the problem, such as the dynamic deformation of solids, and the control of the coalescence and break-up of boundaries are specialized and calibrated for this test case, and are not applicable to the generic case.

8.5.4 Motion with Change in Boundary Topology

The Projectile Flight and Impact test case presented in the preceding sub-sub-section features three instances of topologic change in boundaries; namely, one instance of a simple disintegration, one instance of a simple coalescence, and one instance of a complex disintegration. In addition, that test case features an implied change in the topology of the target, which originally encloses an isolated region that

then becomes connected to the rest of the Computational Region after the target is breached.

Other simpler test cases involving topologic changes were also attempted, but are not presented here for brevity, partly because they are similar to the test cases of the preceding two sub-sub-sections, and partly because they show no new features or capabilities.

8.6 Computational Resource Requirements and Performance

The requirements of the new computational methodology developed in this work for storage space (that is, for computer memory), and for processing effort (that is, for CPU operations, or CPU time) are summarized in Table 8.3. That table also compares those requirements with the corresponding ones for two other related, standard methods, as described in more detail below.

Most of the storage space occupancy in the computational methodology developed in this work is normally taken up by the Quadtree data-structure representing the grid, which is here taken to include all the sub-ordinate data-structures and sub-components of the full Quadtree, including, for example, the objects that store the attribute data of the constitutive nodes of the Quadtree, as described in Sub-Section 6.3.1. The storage space occupied by a Quadtree data-structure can be accurately estimated from the formula $S = s_l n_l + s_i n_i$, where S is the total storage space occupancy, s_l and s_i are respectively the node-averaged storage space occupancies for a leaf node and for an interior node of the Quadtree, and n_l and n_i are respectively the number of leaf and interior nodes in the Quadtree. As explained in Section 6.1, the ratio $\frac{n_i}{n_l}$ is very close to $\frac{1}{3}$ for almost any Quadtree that is of practical use, so the formula given above may be re-expressed in the more convenient form $S = \frac{3s_l+s_i}{3} n_l$. In

	Memory (in 4-byte words)	**Processing Speed** (Cell / Update Cycle / CPU Second) (on a 50MFlop HP9000/735/125 Machine)
Structured Grid	12-15	4,000
Unstructured Grid	50-100	1,000-3,000
New Method with Static Boundaries	29	3,000
New Method with Moving Boundaries	39	2,000

Table 8.3: The storage-space and processing-effort requirements of the new computational methodolgy developed in this work, with comparisons to other standard methodologies.

computing the values of s_l and s_i, care must be taken to avoid any double-counting of the space occupied by objects that are shared by nodes, such as the links connecting each node to its superior and sub-ordinate nodes.

Table 8.3 shows the overall storage space requirements for each leaf node in the specific Quadtree data-structure implemented in this work (that is, for each cell in the grid). In particular, the table shows that these requirements are approximately 29 4-byte words for computations with stationary boundaries, and approximately 39 4-byte words for computations with moving boundaries. A detailed listing of the data objects that may be stored for each node in the Quadtree is given in Sub-Section 6.3.1, but it should be noted that depending on the type of computation performed,

some of these data objects are not used and not allocated, and that some of them are, as indicated in that section, purely optional, and are normally omitted. The figures given in Table 8.3 correspond to the ratio $\frac{S}{n_l} = \frac{3s_l+s_i}{3}$, which is defined and explained in the preceding paragraph, that is, the figures are obtained by dividing the total required storage space by the number of leaf nodes, and therefore include the additional overhead of storage space needed to store the non-leaf nodes of the Quadtree and the attribute data of these non-leaf nodes (which is a reduced set of the attribute data of the leaf nodes, as explained in Sub-Section 6.3.1). It should be noted that these figures, as implied in the formula just given, are practically independent of the number of nodes in the tree, or the number of cells in the grid.

The only data-structure other than the Quadtree that usually requires an appreciable amount of storage space is the output-data link-list. This link-list is created periodically (by mapping the Quadtree data-structure), then used for outputting the required grid and solution data, and then destroyed once the data-output processing has been completed. Because this data-structure is short-lived, and because its presence and form are dictated by external requirements on the format of the output data, rather than by the intrinsic properties of the computational methodology developed in this work, the memory requirements of this data-structure (which could be up to 25% of the total) are excluded from the figures given in Table 8.3.

The other data-structures used in the implementation, such as those used to store the geometries of boundaries, usually require far less memory than the Quadtree data-structure, and are therefore neglected in the figures given in Table 8.3.

As explained in Sub-Section 6.3.1, considerable flexibility exists in the methodology developed in this work to reduce the storage space occupied by the Quadtree at the expense of increasing the processing effort, and vice-versa. Therefore, the

storage space values given in Table 8.3 should be regarded as "optimal" values for the specific "operating conditions" in this work, rather than fixed values.

The storage space requirements described above and given in Table 8.3 were estimated and measured using three different methods: (i) adding the sizes of the individual active members of the "node" data-structure of the Quadtree, with appropriate averaging, as given in the detailed listing of Sub-Section 6.3.1, and assuming efficient alignment of these members on word boundaries; (ii) maintaining an internal counter that tracks how each dynamic allocation and de-allocation of memory changes the total memory used during execution of the program in which the new methodology is implemented; and, (iii) using the standard UNIX commands **ps** and **top**. The estimates produced from all three of these methods rarely differed by more than 10-15% of each other.

As mentioned above, Table 8.3 also presents the computational-effort requirements for the new computational methodology developed in this work. These requirements are expressed in that table in terms of the number of computational cells in the grid (or leaf nodes in the Quadtree) processed for one update cycle, in one second of CPU time, on a Workstation rated at SPECfp92:150.6. The figures are approximately 3,000 for problems with stationary boundaries, and approximately 2,000 for problems with moving boundaries. These estimates reflect the effort required for all operations except for the data-output operation, including, as implied above, all operations performed on non-leaf nodes. The specific machine type on which these measurements were taken was a Hewlett Packard (HP) 9000/735/125 Workstation with approximately 500 MBytes of RAM, running the HP-UX Operating System, which is a customized version of the standard Unix Operating System.

The computational-effort requirements given in the preceding paragraph were

estimated and measured by accumulating the number of computational cells (which is output after every update cycle) processed over a given number of update cycles, and from the cumulative CPU time devoted to the computation, which was measured using three different methods: (i) measuring the corresponding elapsed "wall-clock time" with a dedicated processor; (ii) using the standard UNIX command **time**; and, (iii) using the intrinsic C function **clock**, called at appropriate locations from within the computer program in which the computational methodology developed in this work is implemented. The estimates produced from all three of these methods rarely differed by more than 20% of each other.

The proportions of the total computational effort devoted to the different sub-algorithms or different operations of the overall computational methodology developed in this work were obtained and estimated by calling the intrinsic C function **clock**, which measures the elapsed time between two successive calls to that function, at appropriate locations from within the computer program in which the methodology is implemented, and, independently, by using the standard UNIX utility **prof**.

Using the **prof** utility requires linking the individual "object-language" files comprising the executable with the **-p** option (possibly also with some compiler-dependent or system-dependent variations or additional parameters), then running the executable to generate a file which contains the time-profile data (which by default is given the name "mon.out"), and then running the UNIX **prof** utility to extract the required run-time statistics (stored in the "mon.out" file) of the CPU time or CPU number of cycles expended in the various functions and sub-units of the executable.

The data obtained using the intrinsic C function **clock** can provide accurate measurements of the time spent in, for example, the merging operation, or the gas-

dynamic update operation, regardless of the specific C functions or program sub-units invoked from these operations. In contrast, the data from the **prof** command identifies the time spent in the various functions or sub-units of the computer program or executable, regardless of which high-level operation or sub-algorithm they were invoked from. Thus, for example, the time spent in the function "TraverseGrid" was typically 2-5% of the total time for the relatively large computations performed in this work, and this not only reflects the time spent in finding cell neighbors, but also the time spent in the traversal process for every major operation in the algorithm. This is because the function "TraverseGrid" is used whenever a global visiting of the cells is performed, whether the purpose is to generate the initial grid, to compute the fluxes, to compute the solution gradients, or to perform the update to the gasdynamic state. Thus, the **clock** function and the **prof** utility can be seen to provide complementary timing measurements that can be used not only to study the distribution of CPU time among the different sub-algorithms or functions of the overall methodology, but to also improve the performance of the computer program.

The timing measurements made with both the **clock** function and the **prof** utility show that for problems with moving boundaries, typically half of the CPU time is spent in computing the fluxes, reconstructing the gradients of the primitive variables, limiting the gradients of the primitive variables, and performing the gasdynamic update; and, about one quarter to one third of the CPU time is spent in the geometric calculations, including those invoked for merging, for updating the boundary positions, and for evaluating the cell-cutting patterns. These estimates are typical for the test cases examined in this work and presented in this chapter involving aerodynamic problems or shock-obstacle-interaction problems using several hundred thousand computational cells. The ratio of time spent in the geometry-related cal-

culations is higher for problems involving impacts or topological transformations, and is lower for problems with stationary boundaries than it is for problems with moving boundaries. Both of these trends are as expected, because problems with impacts or topological transformations require additional geometric calculations, while problems with stationary boundaries only require geometric computations for the merging operation and to account for changes in the grid caused by adaptation.

It should be noted that the timing distributions for individual problems may deviate significantly from the estimates given in the preceding paragraph, depending on the total length of the impermeable boundaries in the problem, on the overall adaptation pattern, on the density of cells around the impermeable boundaries, and even on the specific geometric definition of the impermeable boundaries present. For example, increasing the number of segments in an individual straight edge of an impermeable boundary up to a certain point would decrease the total time spent in the geometry calculations, because each segment would cross a fewer number of cells, and would require fewer intersection evaluations.

The processing speed quoted in Table 8.3 for the "structured grid" was obtained from measurements on a two-dimensional structured-grid computer program written by the author of this dissertation, while the processing speeds given for the "unstructured grid" were obtained from measurements taken on two different two-dimensional unstructured-grid computer programs specialized for triangular cells [16, 256]. All the computer programs used in these comparisons solved the Finite-Volume discretizations of The System of Euler Equations using a scheme which is second-order-accurate in space, and which performs the time-integration using a multi-stage Runge-Kutta procedure. Also, all the computer programs used in these comparison were compiled with the highest level of optimization, and all of them were run on the same hardware

platform; namely, the HP 9000/735/125 Workstation described above.

8.7 Summary and Conclusions

In this chapter, several computations which serve to verify, validate, or demonstrate the new computational methodology developed in this work, or to investigate its properties and capabilities are presented and discussed. These computations collectively exercise all the major capabilities of the new computational methodology, and cover the entire range of problem categories and sub-categories for which the new methodology was developed. The computations specifically tested, verified, validated, or demonstrated the new methodology for problems in which the fluid-dynamic behavior is governed by The System of Euler Equations, and which have one or more of the following key characteristics or features: (i) a simple, smooth, subsonic flow; (ii) a complex flow with multiple, interacting discontinuities; (iii) a steady solution; (iv) an unsteady solution with stationary boundaries; (v) an unsteady solution with moving or deforming boundaries; (vi) topologic transformations in boundaries, including coalescence or disintegration of boundaries; (vii) geometric complexity (of boundaries or bodies); (viii) multiple, or multiply-embedded, isolated fluid regions; (ix) large-scale separation or approach (of boundaries or bodies); and, (x) widely differing length scales.

Most of the computational solutions presented in this chapter were closely examined to assess their correctness and quality, which were evaluated by quantifying as far as possible the accuracy, the order of accuracy, and the spatial and temporal resolutions of these solutions, and their overall convergence to the corresponding correct solutions, either at specific points, or globally across the entire Computational Region. For the simpler flows studied, the correct solutions were taken to be the cor-

responding analytical solutions, while for the more complex flows studied, especially those with multiple interacting discontinuities, the correct solutions were taken to be the corresponding, independently-obtained, "benchmark" computational solutions if available, or those implied by the corresponding experimental test results. Because of the very important roles of cell cutting and cell merging and unmerging in the new methodology developed in this work, special attention was focused on their effects on the quality of the solution, especially on the smoothness and the overall accuracy of the solution near impermeable boundaries. Special attention was also focused on the effectiveness of the solution-adaptation methodology adopted and developed in this work in tracking traveling and evolving solution features, especially for problems with multiple interacting discontinuities.

The main general conclusion gathered from the evaluation of the various test cases studied in this work is that the tests, verifications, validations, and demonstrations confirm the viability of the new computational methodology developed in this work, and confirm that it effectively and correctly treats problems from all the categories in its intended field of application, including problems with moving or deforming boundaries or topological transformations, and that it successfully attains all the intended capabilities and objectives outlined for it in Chapter I. The most important specific findings and conclusions related to the individual properties and capabilities of the new computational methodology are as follows:

1. The computational results show that the new methodology treats cell merging, cell unmerging, and cell intersection at boundaries correctly and consistently, and that cell merging and unmerging is a viable and effective means of enabling the motion of exactly-defined boundaries on a fixed grid, and of resolving the intersected-small-cell problem for both stationary or moving boundaries.

2. The computational results show that the combined effect at or near boundaries of the application of spatially first-order-accurate Boundary Conditions, the special treatment of intersected cells, and the repeated application of cell merging and unmerging operations does not degrade the spatial order of accuracy of the solution to below first-order locally, nor (for a second-order-accurate interior scheme) to below second-order globally. More particularly, the effect on the local spatial order of accuracy of cell merging and unmerging is found to be comparable with that of the application of spatially first-order-accurate Boundary Conditions. This is consistent with what would be expected from the equivalence between the effects of cyclical cell merging and unmerging on the one hand, and the effects of a cyclical local averaging operation on the other. The entropy generation from cell merging and unmerging is also found to be comparable with the entropy generation from the application of spatially first-order-accurate Boundary Conditions.

   The spatial order of accuracy in the investigations leading to the above conclusions was determined by using the classical method of studying the effect on the solution error in a smooth problem of increasing the spatial (and for transient problems here also the temporal) refinement of the discrete solution.

3. The computational results show that cell merging and unmerging cause small losses of smoothness on boundaries, which, again, are comparable with those arising from the application of spatially first-order-accurate Boundary Conditions. The computational results also consistently exhibit reduced smoothness in the solution in regions of non-uniform cell size. Both of these observations are discussed in more detail below.

4. The computational results confirm that the "local sub-stepping" time-integration procedure used for intersected composite cells with moving or deforming boundaries, as explained in Chapter III, enables satisfaction or preservation of the two fundamental requirements expressed in The Discrete Geometric Conservation Laws (for moving or deforming boundaries, or time-varying Subregion Sets).

5. The various test cases demonstrate the high degree of effectiveness and efficiency of the solution-adaptation algorithm adopted and developed in this work in tracking individual flow features as they travel, evolve, and interact with other features throughout the Computational Region, and the ability of this algorithm to maintain such features without loss within refinement zones of appropriate thickness throughout a computation.

6. The new computational methodology is found to require slightly more computational effort (say, to advance the solution by one time-step for a given number of computational cells) than traditional unstructured-grid methods for problems with stationary boundaries, but the additional expense seems to be adequately justified by the gains from the high efficiency of the solution adaptation, and from the full automation of the grid generation and adaptation processes, especially for problems with moving boundaries.

Even though the main emphasis of the work presented in this dissertation is on the correct and accurate handling of moving or deforming boundaries, and on those properties of the computational methodology that are directly related to this handling, it is nevertheless necessary to also establish, at least for the purposes outlined in Chapter I, that the new computational methodology with cell cutting,

cell merging, and moving or deforming boundaries also correctly and accurately solves the governing equations chosen to model the behavior of the fluid medium: The System of Euler Equations.

As emphasized in several places in this dissertation, a discrete solution scheme for The System of Euler Equations is expected to (and is often judged by how well it does) accurately and reliably preserve and reproduce all the important analytical properties and features that this system of equations is capable of supporting, as these are described in Chapter II, and especially in Section 2.2. This is particularly the case for properties associated with discontinuities such as shock waves and vortex sheets, but it is also true for smooth features such as stagnation points and expansion fans.

The most important findings and conclusions from the verifications, validations, and tests carried out in this work, in relation to the discrete solution scheme for The System of Euler Equations, as it is implemented in the computational methodology developed in this work, are as follows:

1. All three types of discontinuity supported by The System of Euler Equations, whether oriented "normally" or "obliquely" to the grid lines, are correctly and accurately captured and computed, with the correct jump conditions and the correct propagation speeds. The interactions between two different shock waves, a shock wave and a shear wave, a shock wave and a contact wave, and a shock wave and an expansion fan are also all correctly captured and computed. Smooth features such as expansion fans and isentropic compressions are also correctly computed and resolved. However, the discrete solution scheme does exhibit a slight dependence of the resolution of features, especially the thickness of discontinuities, on their propagation angle relative to the grid lines, as would

be expected with any Sub-Region Set having a strong geometric alignment.

2. The Boundary Condition treatments adopted in this work, including the treatment for the far-field condition, and for stationary and moving impermeable boundaries, are correct and accurate for both supersonic and subsonic flows. However, the Boundary Condition treatments adopted in this work for impermeable boundaries often degenerate to first-order accuracy in space. This degeneration is discussed in more detail below.

3. All of the four Numerical Flux Functions implemented in this work are correct (but not always ideally-suited) for all types of flow features, including ones in subsonic or supersonic flow, whether with or without discontinuities.

4. The formulation and implementation of the time-integration scheme, and the overall discrete solution procedure, for both steady and unsteady computations (whether with or without moving boundaries), and whether for subsonic or supersonic flow, are correct, in both the global time-stepping, and the local time-stepping modes.

5. The discrete solution scheme appears to correctly impose the Kutta Condition for geometrically-induced flow separations.

The evaluation of the results from the various test cases studied in this work also shows that the new computational methodology developed in this work exhibits several defects, deficiencies, and disadvantages (some of which are mentioned above in this section). The most important of these defects, deficiencies and disadvantages, together with their known or suspected causes, and with the most promising approaches to remedying, circumventing, or compensating for some of them are as

follows:

1. A general lack of smoothness across "refinement boundaries" (or in regions of non-uniform cell size), even well away from impermeable boundaries. This is believed to be due mostly to the reduction in the local spatial accuracy and in the local spatial order of accuracy around such internal "boundaries". In particular, as explained in more detail in Chapter III, although the solution scheme adopted and developed in this work is formally second-order accurate, strict second-order accuracy is achieved only in uniform Sub-Region Sets: whenever size non-uniformities in a Sub-Region Set exist, such as around two neighboring computational cells with different refinement levels, the cancellation of the truncation errors in the spatial discretization is diminished, causing the local spatial order of accuracy there to degenerate from two towards one.

   From the explanation in the preceding paragraph, the most promising solution to the loss of smoothness across internal refinement boundaries is believed to lie in increasing the formal spatial order of accuracy of the discrete solution scheme from two to three, since this would ensure that "true" second-order accuracy is retained even around jumps in the refinement level. This can be accomplished by using a 2-exact, or piece-wise quadratic, reconstruction (instead of the 1-exact, or piece-wise linear, reconstruction adopted in this work), following, for example, the methodology outlined in [387]. Of course, the development of a spatially third-order-accurate scheme would be far more difficult for three-dimensional spaces than it is for two-dimensional spaces.

   The specific deficiency described above may more appropriately be regarded as a fundamental problem or issue related to "grid quality" in grid generation or

grid adaptation, especially for adaptive Cartesian grids (which feature relatively large ratios of cell area or cell volume across refinement boundaries), rather than a specific deficiency of the computational methodology developed in this work.

2. A slight loss in the smoothness of the solution along and near impermeable boundaries that are accelerating rapidly, or, more generally, that bound a region in which the solution gradient is high. This is believed to be due to the application of the Boundary Conditions and the solution projection of the unmerging operation with effectively only first-order accuracy in space on or near such boundaries. Any non-uniformities in the cell size along or near boundaries also tend to independently reduce the local spatial order of accuracy of the solution towards one, as explained in the preceding item. Again, all three of the effects just described arise even though, as explained in Chapter III, the solution scheme used in this work is formally second-order accurate (in both space and time). It should be noted, however, that the formal second-order accuracy in time of the solution scheme is still preserved on or near impermeable boundaries, regardless of their acceleration or the solution gradient near them.

   The reason for the local degeneration to first-order accuracy in space of the solution in regions of non-uniform cell size is explained in the preceding item. The reason for the local degeneration to first-order accuracy in space of the Boundary Condition treatment for an impermeable boundary bounding regions with finite solution gradients is that any "monotonicity enforcing" limiter applied to a 1-exact reconstruction will have the effect of reducing at least the component of the gradient normal to the boundary to near zero. This would occur even

if the cells near the boundary were of uniform size (and this is usually not the case). The reason for the local degeneration to first-order accuracy in space of the prolongation step of the unmerging procedure is, similarly, the reduction by the limiter of the unlimited reconstruction gradients computed in composite cells to near zero in the vicinity of accelerating boundaries before the prolongation of the solution in the composite cells to their constitutive Cartesian cells. As explained in detail in Chapters III and VII, this reduction in the unlimited gradients is performed to prevent the creation of non-physical values or new extrema in the Cartesian cells to which the solution is being prolonged.

The focusing of attention on the Boundary Condition treatment and the prolongation step of the unmerging procedure as the two suspected causes of the loss of smoothness explained in the preceding two paragraphs is encouraged by the observation that the loss of smoothness with uniform cell size is most readily detectable in the entropy and the Mach Number contours near an impermeable boundary. This encouragement arises because, in general, the order of accuracy of the Boundary Condition treatment affects the entropy and the Mach Number distributions on bounding surfaces the most, while the order of accuracy of the interior scheme treatment affects the pressure distributions on bounding surfaces the most [303].

The most promising potential remedy for overcoming or largely eliminating the deficiency described above for rapidly accelerating boundaries is, as implied by the above discussion, believed to be the raising to two of the actual spatial order of accuracy for both the treatment of the Boundary Conditions, and the prolongation step of the unmerging procedure. In both cases, increasing the actual spatial order of accuracy can be accomplished, as described in the

preceding item, by increasing the formal spatial order of accuracy of the discrete solution scheme from two to three, by the use of 2-exact reconstruction instead of 1-exact reconstruction at and near boundaries, in accordance with the procedures and techniques described, for example, in detail in [387]. It is also necessary to modify the limiters to allow the appearance of new (but physically-valid) extrema on impermeable boundaries. The proposed quadratic interpolants together with the more compressive limiters would enable more accurate extrapolation of the solution values to impermeable boundaries, and more accurate prolongation from the state vectors in composite cells to the state vectors in Cartesian cells, while still preventing the introduction of new extrema at points that do not fall on boundaries, and new extrema that are not physically valid.

3. The generation of isolated or partly-isolated "islands" of finer cells (in regions of coarser cells) or "islands" of coarser cells (in regions of finer cells), as can be seen in the grid plots of many of the test cases presented in this chapter. This defect is caused by the dependence of the solution-adaptation process on the solution gradients, and the (far smaller) response of the latter to changes in the grid introduced by the former. For example, if the local solution gradient is disturbed sufficiently by low amplitude fluctuations (or even noise) in the solution, the change may trigger action by the solution-adaptation algorithm, causing local grid refinement or coarsening. This refinement or coarsening may in turn introduce additional small local disturbances in the solution because, in refinement, the conserved variables are redistributed to the newly-created cells according to the solution gradient in the cell being refined, while in coarsening, the solution gradients of the cells to be eliminated are weight-averaged to obtain

the solution gradient in the cell to be coarsened. Thus, both actions result in spatial re-distribution of the conserved variables, leading in general to changes in the solution gradients (after the next update) in the directly-affected cells, as well as their close neighbors. The resulting changes in the solution gradients may then trigger further solution-adaptation actions, completing the cycle of interaction outlined above.

When it occurs, the feedback cycle described above typically rises and dies down quickly, with the isolated islands of high or low refinement typically appearing and disappearing within several dozen time-steps. Even more importantly, the amplitudes of the disturbances in the solution that accompany the appearance and disappearance of these "islands" are relatively very low, and have little effect on the time-averaged or the space-averaged values of the local solution, or on the evolution of the local solution, even though the refinement "islands" visibly and adversely affect the smoothness of the Sub-Region Set.

The most intuitive approach to eliminating the "islands" described above; namely, introducing a damping effect into the solution-adaptation operator so as to slow its response to the values of one or more of the adaptation indicators, surprisingly fails to yield the expected degree of improvement, and also increases the number of cells in a computation by thickening the refinement zones near traveling features. The approach of suppressing any solution-adaptive refinement or coarsening in response to the values of one or more of the adaptation indicators if that refinement or coarsening would result in a loss of smoothness in the local cell-size distribution works well, but also increases the total number of cells in a computation by thickening the refinement zones around traveling features, and has therefore not been adopted as a permanent

extension or feature of the solution-adaptation methodology adopted and developed in this work. The approach of raising the refinement thresholds (which are described in detail in Chapter VI) is also effective, but not always appealing, especially if it is desired to also capture weaker solution features as strictly as possible.

4. The loss of stability in the solution across highly-resolved strong shocks propagating exactly along one of the two coordinate directions. This occurs with the Exact Riemann Solver, and with Roe's Approximate-Riemann Numerical Flux Function, but not with the HLLE Numerical Flux Function. This deficiency appears to be due [293] to the dependence of the internal dissipation in many Numerical Flux Functions on the normal velocity across an interface, and hence to the reduction of this dissipation to an insufficiently low level when the velocity component in a flow field along one of the two principal Cartesian directions is vanishingly small. In turn, the reduction of the dissipation is believed to lead [293] to the generation of strong spurious oscillations behind and perpendicular to a propagating shock wave. This deficiency, which is encountered typically only with highly-resolved solution-adaptive computations, is more of a fundamental difficulty with the Godunov-based Numerical Flux Functions and discrete solution schemes, rather that with the new computational methodology developed in this work.

   As explained in this chapter and in Chapter III, the technique proposed in [293] of using the HLLE Numerical Flux Function "within" strong shock waves while using a less dissipative and more accurate Numerical Flux Function everywhere else, as adopted and implemented in a slightly modified form in this work,

appears to work well for all the test cases attempted in this work, and to adequately resolve or overcome the deficiency described above.

5. The failure to preserve exact symmetry for solutions that should be perfectly symmetric. In most cases, this defect is due to the slight asymmetry normally introduced into the Sub-Region Set by the cell merging process which, as explained in Chapter VII, does not necessarily generate a geometrically symmetric set of composite cells from a geometrically symmetric set of Cartesian cells. In other cases, the asymmetry in the solution is introduced or amplified by asymmetries in the Sub-Region Set introduced by the geometry-adaptation or the solution-adaptation procedures. In all cases, however, the asymmetry introduced into a computational solution which is stably symmetric is slight (having a magnitude typically less than a few percent in the primitive-variable mean values for typical levels of resolution of the solution, even with rapidly evolving features), and serves to confirm the well-posedness of the problem, and to reveal the degree of sensitivity of the solution to small changes in the Sub-Region Set or the solution-adaptation parameters. The magnitudes of any asymmetries generated in what should be a stably symmetric solution are expected to diminish at least linearly with decreasing cell dimension.

   For problems which generate valid asymmetric solutions from symmetric geometries, symmetric Boundary Conditions, and symmetric Initial Conditions, the asymmetries introduced into the Sub-Region Set by the three mechanisms described above will certainly bias the computational solution and favor its convergence to one or more of the possible asymmetric solutions. A similar biasing effect is expected for problems in which a high degree of asymmetry in

the solution may be produced by very slight asymmetries or mis-alignments in the Sub-Region Set, the Boundary Conditions, or the Initial Conditions, as for the problem of sub-critical flow over a cylinder [289].

Other defects, deficiencies, disadvantages, or imperfections in the new computational methodology developed in this work that are more minor than those cited above are noted and discussed in the test case presentations in preceding sections of this chapter.

The defects, deficiencies, and disadvantages listed above and in preceding parts of this chapter, and the remedies suggested for them may be investigated further than done in this work, possibly as the subjects of future research. There is no reason to believe that these defects, deficiencies, or disadvantages are permanent or unimprovable features of the new computational methodology developed in this work.

The emphasis in this section was on the findings and conclusions directly related to the evaluation of the computational results obtained with the new computational methodology developed in this work. The more general findings and conclusions, including those related to the overall role of the computational methodology, and to the overall contribution of the work described in this dissertation are presented and discussed in Chapter IX.

CHAPTER IX

Review, Conclusions, and Furtherance

This chapter presents a critical review of the research performed and the results obtained in this work, with special emphasis on the contributions and the longer-term consequences of the work. The computational technique developed and studied in this work is briefly described, by outlining the principles and implementation method for each of the main elements of the technique. The main findings of the work are summarized, especially the mathematical constraints that were found necessary for correct numerical implementation of Arbitrary Lagrangian-Eulerian schemes involving moving boundaries on stationary Cartesian grids. Several improvements, and promising potential extensions and generalizations of the methodology developed and studied in this work, are also briefly described and discussed, including the extension to three-dimensional spaces and for solution of different systems of equations.

9.1 Summary of Work Done

A new Lagrangian-Eulerian computational solution technique for inviscid compressible flows with moving boundaries was developed and studied in this work. The technique is also applicable to time-dependent and to steady-state inviscid com-

pressible flows with stationary boundaries. Compared to the techniques that have heretofore been developed for flows with moving boundaries, the main distinguishing characteristic of the new technique developed in this work is the ability to retain the geometric representation of boundaries as infinitesimally-thin discontinuities separating regions with arbitrarily different states and properties, while allowing these boundaries to move or deform arbitrarily on a *stationary* grid.

The computational technique is built from five main components:

1. **An Adaptive, Cartesian, Quadtree-based grid-generation algorithm for automatic meshing of arbitrary geometries.** The adaptation is inbuilt into the grid-generation algorithm to ensure that the spatial discretization of the Computational Region is everywhere automatically adjusted to provide the spatial resolution required to describe the geometry of the boundaries of the Computational Region to a specified truncation error. If boundaries move, then the grid is coarsened and refined as necessary in the vicinity of regions that are vacated or invaded by moving boundaries so that the boundaries remain resolved to the required accuracy at all times as they move across the stationary grid. The spatial localization of the refinement and coarsening operations ensures that the number of computational cells remains close to the minimum possible during each time-step and throughout the Computational Region;

2. **A representation of boundaries as exact discontinuities separating two arbitrarily different states in any Cartesian cells that they intersect.** The boundaries are represented using composite parameterized analytic splines of the first, second, or third degree. Boundaries are allowed to arbi-

trarily intersect (or "cut") the stationary grid. There are also no *grid-related* restrictions on the motion or deformation of boundaries. The motion of boundaries is effected through displacement of the endpoints of the splines, followed by re-interpolation between the updated end-point positions. The boundaries retain their definition as exact discontinuities, regardless of their motion on the grid, eliminating any side effects from numerical diffusion. The boundaries present in a computation are dynamically re-splined or re-defined if necessary, to account for changes in their length-scale or curvature or for any topologic changes, such as those involved in the coalescence or disintegration of boundaries;

3. **A Finite-Volume, Arbitrary Lagrangian-Eulerian discretization of The System of Euler Equations, and an explicit, characteristic-based algorithm for solution of the discretized equations.** The discretization and the solution algorithm are applicable to arbitrary polyhedral cells, such as those generated by non-boundary-conformal cutting of the cells of the Cartesian grid. The solution scheme is formally second-order accurate in space and time, and satisfies the Geometric Conservation Laws for moving boundaries;

4. **An algorithm for merging individual Cartesian cells into composite cells, and for "un-merging" the composite cells into their constitutive Cartesian cells.** The individual Cartesian cells that are merged into a composite cell are selected to ensure that the resulting composite cell has a satisfactory volume, and that its retains the same topologic configuration during a motion step of the boundary. The latter means that during a motion step of a boundary, any motion of the points at which the boundary and the edges of a

composite cell intersect is guaranteed to remain confined along the same edges of the composite cell, so that the composite cell only changes its volume, but not its topologic intersection pattern with the boundary. In this manner, the "re-assembly" of the computational cells by cell merging transforms the problem of moving boundaries on a stationary grid into a problem of deforming cells, a problem that can be solved with a standard Arbitrary Lagrangian-Eulerian solution technique. Un-merging is the process of decomposing the composite cells into their Cartesian-grid components, and projecting the solution in the composite cells back into the Cartesian grid cells. The merging and un-merging operations are carried out dynamically, one motion step at a time. The merging and un-merging algorithm enables the motion of boundaries through a stationary grid while maintaining conservation and accuracy, and while enabling the use of large time-steps in the solution algorithm. Cell merging is also used for problems with stationary boundaries, to solve several difficulties arising from the non-boundary-conformality of the Cartesian Quadtree grid; and,

5. **An algorithm for solution-adaptation of the grid.** This algorithm automatically adjusts the local spatial resolution in the Computational Region (by refinement or coarsening of the local Cartesian cells), to resolve and to track flow features with large gradients (such as discontinuities) as these features develop or move across the Computational Region.

Each of the five components listed above is described in detail in an appropriate chapter in this dissertation. Various sample computations are also shown to verify, validate, and demonstrate the applicability, accuracy, and capabilities of the overall methodology and its individual components for a wide variety of problems, ranging

from steady-state problems with simple, stationary flow features and boundaries, to problems with traveling flow features and flow-induced motions and deformations of boundaries. The methodology developed and the work performed here fully satisfy the objectives of the research, which are outlined in Chapter I.

9.2 Summary of Research Contributions and Main Findings

The major contributions and findings of the work presented in this dissertation are as follows:

1. **Demonstration of the concept and capabilities of cell merging, and development of an algorithm for merging of computational cells for boundaries of arbitrary geometry on Adaptive Cartesian grids**. This includes the demonstration that cell merging can be used to perform two critical functions for an explicit time-integration scheme on a Cartesian grid: (i) the removal of the "small-cell" CFL constraint (which is described further in Appendix A.1); and, (ii) the creation before each motion step of composite cells that remain topologically invariant during that motion step, and can therefore be treated as deforming computational cells. The deforming-cell treatment requires the application of a traveling boundary condition, and is formulated using an Arbitrary Lagrangian-Eulerian conservative discretization. In this manner, cell merging allows the standard Arbitrary Lagrangian-Eulerian discretization to be applied for boundaries that move on a stationary grid, as described further below. As a result of the two functions served by it, cell merging is demonstrated to be a successful means of allowing the formulation and computation of moving-boundary problems using stationary Cartesian grids;

2. **Demonstration of the applicability of non-boundary-conformal Cartesian grids for stationary and moving geometries, and development of an algorithm for automated generation of adaptive Cartesian grids for arbitrarily-complex geometries, and for automated, dynamic adaptation of the grids for arbitrarily complex motions of those geometries**. The use of either "truncation" or "polygon clipping" cell-cutting procedures to define the geometry of computational cells that are intersected by boundaries were both found to allow development of robust grid generation algorithms. With both methods, an accurate representation of all boundaries present in the Computational Region could be achieved and maintained during any motion of the boundaries, by automatically varying the local spatial resolution of the Cartesian grid to match the local length scales of the geometry of boundaries. For stationary boundaries (with both steady, and transient flows), the work done here is more properly viewed as a confirmation of earlier work done for stationary Cartesian grids, for example, in [46, 291, 101, 4];

3. **Demonstration of the feasibility and advantages of using an Arbitrary Lagrangian-Eulerian formulation in conjunction with cell merging for discretization and discrete solution of The System of Euler Equations for moving boundaries on a stationary grid**. This includes demonstration of the feasibility of solving The System of Euler of Equations in a straightforward and standard manner while satisfying The Geometric Conservation Laws simply and economically for all admissible motion patterns of boundaries. It was shown that in order to satisfy The Geometric Conservation Laws with boundaries that are represented by piecewise polynomials, the degree, d, of the polynomials must satisfy the condition $d \geq (o-1)$, where o is the formal

order of accuracy of the flux quadrature. Thus, for example, for a scheme that is second-order-accurate in time and space, the boundary geometry may be represented by piecewise straight-line segments, and account should be taken of the exact crossing times of segment endpoints as they travel into and out of stationary composite cells. No other constraints need to be imposed on the smoothness of the motion or on the differentiability of the representation of the boundary geometry. As a consequence of this requirement, it was shown that even a first-order-accurate scheme cannot be made to always satisfy The Geometric Conservation Laws for arbitrary directions of motion of a boundary if the geometry of the boundary is obtained by truncating the exact representation through intersecting that representation with the grid lines, rather than accounting for the exact representation. For computational schemes that are second-order accurate, as used in this work, for example, the use of piecewise-linear representations of the boundary geometry is recommended, and for the following main reasons:

(a) The Geometric Conservation Laws can be satisfied simply and economically for arbitrary boundary motions;

(b) Computing the intersections of grid lines that are here always parallel to the Cartesian axes with arbitrarily-oriented straight lines requires a smaller operation count than computing the corresponding intersections with arbitrarily-curved lines; and

(c) The intersection times of segment end-points with the faces of a composite cell can be computed by linear interpolation between the initial and final positions of the end-points, giving an exact calculation for boundaries

moving at constant speeds, and a second-order-accurate approximation for boundaries moving at non-constant speeds.

For third-order accurate computational schemes, the use of parabolic representations of boundaries is recommended. The computational cost associated with geometric computations will therefore increase compared to the second-order-accurate case because of the more laborious time-averaged integration evaluations for volumes and areas required to satisfy The Geometric Conservation Laws;

4. **Demonstration that in an explicit adaptive scheme for transient problems, it is sufficient to adapt the grid once every time-step to retain all "sharp" flow features within suitable refinement zones, and without having to add surrounding "buffer" refinement regions**. This allows the size of refined regions to be kept to the minimum possible, minimizing the total number of required computational cells, and minimizing the overall cost of the refinement and coarsening operations;

5. **Confirmation of the advantages and the effectiveness of grid adaptation for computations of transient and steady-state flows with multiple lengths scales**;

6. **Confirmation that the Barth Reconstruction and Limiting schemes perform adequately for moving flow features and for moving boundaries, and are simple to adapt for these applications**; and,

7. **Confirmation of the advantages and benefits of local selection of the flux computation scheme from several popularly-used ones to avoid**

their individual weaknesses under specific flow conditions, as suggested in [293]. This strategy was found to work well for many of the applications studied and tested in this work.

The main findings and contributions given above were listed (with the least possible degree of subjectivity) in decreasing order of significance and long-term impact on the current state of knowledge. Confirmations or demonstrations of well-established, secondary, or indirectly-relevant facts were omitted for brevity.

The contributions and findings given above are relevant to five different categories of methods or techniques used in Computational Physics: (i) methods for moving-boundary problems; (ii) methods using Cartesian grids; (iii) time-accurate methods; (iv) methods for the solution of The System of Euler Equations; and, (v) methods using adaptive-grid techniques. The remainder of this section discusses the practical contributions and likely impact of the work presented here for each of these five categories. For the sake of giving an exhaustive description for each category, some overlap between the individual contributions has been allowed. These descriptions are followed by a brief outline of the most as well as the least promising application areas of the technique developed and studied in this work, and of the limitations of the technique.

With regard to computational methods for the simulation of interfacial phenomena and moving boundaries, the main contribution of this work is the development of a new, successful, automated Lagrangian-Eulerian method. Because it relies on cell merging to allow boundaries to move on a stationary Cartesian grid instead of relying on grid deformation as do the traditional Lagrangian-Eulerian methods, the new method eliminates the need for local or global re-meshing and their attendant disadvantages. The new method exhibits the most desirable features of both

the Lagrangian and the Eulerian approaches: boundaries are represented as exact discontinuities with arbitrary property jumps and a precisely-defined location (the advantages of the Lagrangian approach), while the use of a non-boundary-conformal stationary grid allows arbitrary boundary motions and arbitrary topologic transformations in boundaries (the advantages of the Eulerian approach). The new method enables precise control over the motion of an interface, with no concern for the side-effects of numerical diffusion of the discontinuous representation of the interface. This allows the formulation and implementation of accurate schemes for modeling the behavior and motion of boundaries. These schemes can be fully dictated by the physics of the flow-field, the mechanics of the boundary, or the coupling of the physical problem on the two sides of the boundary. Examples of such treatments and couplings, including fluid-structure interactions, were demonstrated in some of the computations shown in Chapter VIII, and in [8]. The new method also enables the development and investigation of specific rules or models for execution of topological transformations in boundaries.

With regard to computational methods using Cartesian grids, the main contribution of this work is that it extends and demonstrates the applicability of these methods to moving-boundary problems. The work also demonstrates how the problems of non-boundary-conformality can be overcome for unsteady flows, confirms the applicability and suitability of these methods for inviscid flows with complex geometries and complex flow patterns, and confirms the suitability and success of the adaptive variants of these methods, especially for transient problems. Because cell-merging depends strongly on the axis-aligned orientation of the grid lines in a Cartesian grid, the extensions and demonstrations described above can also be viewed as demonstrations of the role of cell-merging with non-boundary-conformal

Cartesian grids. Most of the demonstrations and confirmations for problems without moving boundaries re-iterate the results of previous works, especially [101, 102, 103], and [73].

With regard to computational methods for time-accurate problems, and for The System of Euler Equations, the main contribution of this work was to develop a new, linearly-exact, general-purpose, Lagrangian-Eulerian variant which is especially "calibrated" for flows with moving-boundaries, and more generally for unsteady flows with strong discontinuities. The accuracy and conservation of the formulation (except during topologic transformations in boundaries), and its preservation of The Geometric Conservation Laws were all demonstrated for steady, unsteady, and moving-boundary flows, and so was the applicability of the method to complex, steady or unsteady flows in arbitrary geometries. It was shown that solution errors are most concentrated near boundaries, due to the effects of cell merging and un-merging.

With regard to solution-adaptive methods, the main contribution of this work was to confirm the highly effective role that solution adaptation can play in enabling accurate and highly-resolved, yet affordable solutions for flows with widely differing length-scales. The value of solution adaptation was shown to be even greater when there are moving boundaries in the flow-field, especially when the motion of the boundaries is coupled to the flow. This work demonstrated again that although adaptation increases the operation count for each cell, it enables a proportionately far more significant reduction in the number of cells required to perform a computation with a given resolution.

The main theoretical detraction of the method developed in this work is the current lack of a theory for the bounding or estimation of the dispersive or diffusive errors introduced by repeated cell merging and un-merging. The main practical de-

traction of the method developed in this work is the increased computational cost associated with performing the cell cutting, and the cell merging and un-merging operations once every motion step. These operations may be considered equivalent to local re-meshings. For some of the computations with the smallest grids tested in this work, the computational cost of the method was found to almost treble compared to a structured-grid implementation. However, since the cost of these "additional" operations scales with the total arc-length for all the boundaries, this cost becomes proportionately less significant as the grid density or the total number of cells increases. The computational cost for effecting boundary motions or transformations was confirmed to be relatively insignificant.

Compared to standard Eulerian methods such as the Level-Set or the Volume-Of-Fluid methods, the method developed in this work is computationally more intensive, and requires an additional, explicit, somewhat arbitrary, non-conservative treatment for interface disintegrations or coalescences. Because of these two factors, the method developed in this work would not be the most suitable for problems in which the interface repeatedly or continuously undergoes merging or breakup and in which the precise geometry of the interface is not critical (for example, the breaking of waves on barriers, or the large-scale disintegration of a body of liquid into small fragments). On the other hand, the method developed in this work would be ideal for situations in which the precise geometry of the interface is crucial, either because high accuracy is required (for example, the motion of a solid projectile through a fluid), or because the precise shape of the interface plays a strong role in the evolution or the effects of the flow-field (for example, the surface-tension-induced flow-field in a micro-fluidic device, and the motions of the control-surface of an aerospace vehicle). Another category of problems that will benefit strongly from the technique developed in this

work is the category of transient problems with stationary boundaries of complex geometry with multiple length-scales in the flow.

9.3 Recommendations for Improvements

The technique devised in this work for computational solution of The System of Euler Equations with and without moving boundaries has been developed and explored largely to its full potential in two-dimensional spaces. Only relatively minor improvements are possible within the context of two-dimensional, inviscid, compressible flow, and these improvements are more related to extensions or refinements in the implementation strategies or methods, or to increases in the computational efficiency, than they are to fundamental principles or capabilities of the methodology. The most important improvements are as follows:

1. **Introduction of sub-cell resolution.** As implied in the name, sub-cell resolution requires single computational cells from the Cartesian grid to be *geometrically* divided into two or more computational cells. In its most general form, sub-cell resolution also requires the ability to assign more than one attribute set (which is typically a state-vector) to each Cartesian cell, and also the ability to perform cell merging with fragments of Cartesian cells. At present, a Cartesian cell may have at most two attribute sets, corresponding to the two regions separated by the boundary that intersects the cell, and at most two fragments, or sub-cells. The improvement described here would generalize the number of sub-regions into which a cell is split, and the number of cell fragments with which cell merging operates to more than two. An example of where these capabilities would be beneficial is shown in Figure 9.1 below. In the sub-figure on the left, the ability to split Cartesian cells and assign multi-

ple attribute sets to them (in this case, three, since the region in the interior of the boundary could be occupied by a fluid) can clearly be seen to remove the strong constraint that currently exists between the smallest length-scales in the geometry (in this example, these correspond to the width of the sharp spike) and the minimum cell size. Relieving this constraint would not only enable larger time-steps to be taken, but may also significantly reduce the total number of cells for geometries with narrow gaps and sharp features. The subfigure on the right illustrates why the cell-merging algorithm must be extended to allow the formation of composite cells from cell fragments as well as whole cells, with all the implied additional treatments for averaging of composite cell states and for projecting the composite cell states back to the original cells or cell fragments. While the extensions described above are feasible and relatively straightforward in two-dimensional spaces, the additional complications in the corresponding algorithm for three-dimensional spaces are significant, mostly because of the increased difficulty of identifying disconnected regions in three-dimensional cells. For both the two-dimensional and the three-dimensional variants, sub-cell resolution increases the total storage requirements.

2. **Implementation of better schemes for simulation of events involving topologic transformations in boundaries.** The main limitations of the current schemes, which are described in Chapters V and VI, are the simplicity of the rules used to determine the onset and the type of topologic transformation, and the arbitrary, non-conservative treatment of the local solution in regions in which coalescences or disintegrations of boundaries occur. While these two limitations or disadvantages are of little consequence to the type of moving-boundary application studied in most detail in this work, improve-

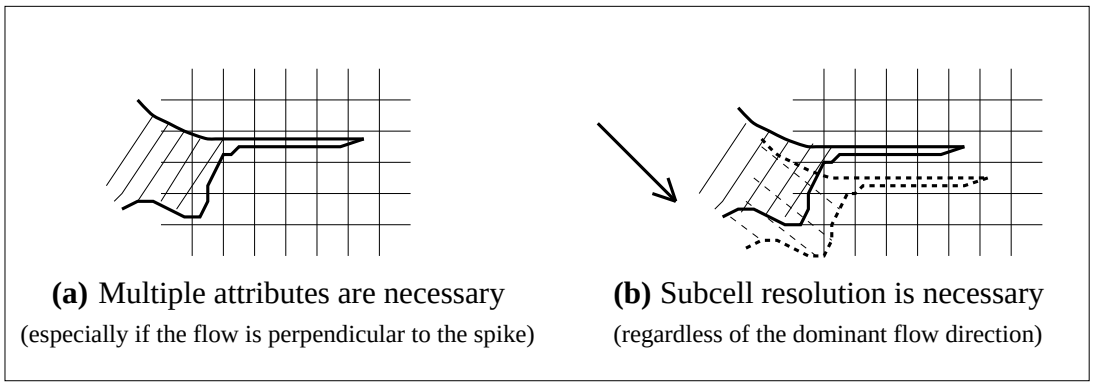

Figure 9.1: Complications Inherent with the Non-Boundary-Conformal Cartesian Grid Approach, and the Role of Sub-Cell Resolution in Resolving these Complications for Both Stationary- and Moving-Boundary Problems.

ments will be required for problems where interactions between interfaces occur frequently or where they play an important role in the overall solution. The rules for determining when to initiate coalescence or disintegration should be quantified more generally than through distances between boundaries, but it is clear that more refined and precise models can be systematically arrived-at only from a detailed understanding of the physics of the interfacial phenomena being simulated, and that even for a comprehensive set of rules, the specific parameters of a model will remain highly problem-dependent. In regard to improving the local solution procedure in the vicinity of a topologic transformation, the most promising approach appears to be the use of local flow models to impose an analytical flow-field in the vicinity of a topologic transformation. Correct implementation of this approach requires identification of the type of transformation as well as the main features of the geometry in the extended locality of the region where the topologic transformation is occurring.

3. **Implementation of adaptive local time-stepping algorithms.** In an Adaptive Cartesian grid with regions of differing refinement levels, it is possi-

ble to advance the different refinement levels with different time-steps, so that on average, a smaller number of larger time-steps are taken in the coarser cells. Such a procedure can be seen to decrease the disparities between the CFL numbers in differently-sized cells compared to what they would otherwise be with a uniform time-step. Of course, in order to preserve the time-accuracy of a computation for a transient problem, the total integration times in different refinement zones must be appropriately synchronized, and appropriate procedures are also required for correcting the fluxes across "refinement boundaries". Because of the need for these corrections, the potential computational savings from eliminating the unnecessarily large number of integration steps in the larger cells can actually only be realized if the refinement zones are at least several cells wide in each coordinate direction. The implementation and the advantages of such techniques have been demonstrated in [54], [291], and [73]. However, such a strategy was not adopted in this work, largely because the grid generation process in this work does not rely on the creation of rectangular regions of recursively-embedded zones with successively-graduated refinement, but instead creates arbitrarily-shaped, minimally-thin refinement zones, as shown throughout Chapter VIII. Because of this, the gains from adaptive local time-stepping in the approach used in this work would be relatively small. Another reason for not adopting such a time-stepping strategy is that most of the computational cells usually belong in the highest two refinement levels, further limiting the potential savings from such an approach. Adaptive local time-stepping would nevertheless be useful, even in the context of the approach adopted in this work, primarily in applications in which blocks of uniform refinement are purposely created, typically to exercise a-priori control

over the accuracy or resolution in one or more specific regions of the flow-field.

4. **Implementation of a parallel-execution version of the overall algorithm.** The most appropriate strategy for load-balancing for this application would be the use of a graph-partitioning approach [281, 332] for domain decomposition. For purely practical reasons, the most appropriate hardware configuration at present is probably a Distributed-Memory MIMD architecture. Since the solution algorithm is explicit, almost complete scalability, except for the effects from inter-processor communication boundaries, can be achieved in principle.

5. **Improvement of computational optimization.** The programming of the individual geometry and flow-solver algorithms in the computer code created in this work emphasized robustness and fault checking rather than speed. The optimization that was carried out was also customized for a specific hardware system. The utility of the resulting software, especially for a specific application, can be improved by optimizing the algorithms and data-structures to improve run-time performance and memory utilization.

9.4 Promising Extensions

As described and explained in Section 9.2 above, the two categories of problems that would benefit most significantly from the technique developed in this work are: (i) the category of transient problems with complex geometries, complex flows, and multiple length-scales; and, (ii) the category of moving-boundary problems for which the exact geometry of boundaries is important and must be preserved. The most promising extensions of the technique developed and studied in this work are

also closely associated with the same two categories. The variety of problems in these two categories is wide, ranging, for example, from problems with high-speed, compressible, discontinuous flows over fixed and moving obstacles, to problems with incompressible, low-speed interfacial phenomena such as crystal growth, evaporation and condensation, or surface-tension-induced flows, to problems with fluid-structure interactions, such as acoustic emissions, projectile penetrations, and missile-stage separations. The remainder of this section describes the areas of extension of the methodology developed in this work that would have the greatest impact on current computational capabilities and contemporary areas of active development, divided for convenience into four groups.

9.4.1 Extension to Three-Dimensional Spaces

Extension of the Adaptive Cartesian grid methodology to three-dimensional spaces has been successfully carried out and demonstrated, for example, in [247, 245], [70], [4], [44], and other works. Effective treatments for handling complex geometries and complex cell-cutting configurations have been devised, making the method applicable to interior boundaries of arbitrary geometry. The most crucial remaining issue for moving-boundary simulations is the ability to extend the cell-merging procedure to three-dimensional spaces.

In principle, cell merging would perform the same primary functions in three-dimensional Cartesian grids that it performs in two-dimensional Cartesian grids: composite cells that have satisfactory volume and that remain topologically-invariant during a motion step must be created from individual Cartesian cells. Although the basic idea carries forward in a straightforward manner, the implementation is far more complex in three-dimensional spaces, mainly because of two factors: (i) the

additional geometric complexity in selecting and forming composite cells; and, (ii) the need for an explicit ordering of all the Cartesian cells intersected by a closed boundary. These two factors are now described further in the next three paragraphs.

With regard to the geometric-complexity factor, many of the key procedures described in Chapter VII that were applied in the two-dimensional case carry over analogously to the three-dimensional case. These include the extension or contraction (of edges in two-dimensional spaces, or faces in three-dimensional spaces) to form closed composite cells, and the change of length-scale to adapt to local changes in grid resolution (around or along the boundary). However, the implementation details are more complex in the three-dimensional case. As described in Chapter VII, composite cells in two-dimensional spaces are constructed from appropriately-selected topologically-invariant edges. Extension to three-dimensional spaces requires a major extra step: the composite cells must be constructed from appropriately-selected topologically-invariant faces, which in turn may be constructed from topologically-invariant edges (just as such faces are formed from topologically-invariant edges in the two-dimensional case). This implies that all the algorithms described in Chapter VII for construction of topologically-invariant cells (which correspond to cell faces in three-dimensional spaces) can be re-used for the three-dimensional algorithm.

With regard to the cell-ordering-requirement factor, the two-dimensional and three-dimensional cases differ markedly. In two-dimensional spaces, a natural, sequential processing order can be imposed by traveling along the arc of each boundary: once a cell has been processed, the next cell is selected to be the one into which the boundary emerges from the current cell. In three-dimensional spaces, no such natural unique ordering exists. Instead, an explicit algorithm is required that will traverse the cells that are intersected by the boundary, visiting each exactly

one time. In order to minimize the memory and the computational requirements, the ordering algorithm should re-use as much of the currently-available data for local topologically-invariant faces as possible. This implies that consecutive cells in the ordering should have (at least partially) shared topologically-invariant faces, in analogy with the situation for the two-dimensional algorithm.

One feasible algorithm would order the Cartesian cells intersected by a closed boundary by first determining the largest cell intersected by that boundary, and then pursuing a traversal pattern that follows a widening spiral on the surface of the boundary, passing through all the nodes in the Octree that are at the refinement level of the largest cell and that are also cut by the given boundary. Some of those nodes will not be leaf nodes, and in that case, their sub-nodes are traversed in standard order, so that all their subordinate Cartesian cells are first visited before going on to the next Octree node in the ordering. When going from one Cartesian cell to the next, the "altitude" is allowed to change so that the "ordering spiral" remains close to the boundary surface, in analogy with the corresponding procedure for the two-dimensional case. It is easy to prove that the above algorithm will visit all the cells intersected by the boundary, and that it will result in re-use of many shared faces except, to a slight extent, toward the end of the traversal path. Another feasible algorithm is described in Figure 9.2 below, where the ordering principle is to take the sharpest possible "turn" at every occasion to identify the next Cartesian cell to be visited (without regard to the refinement level of the cell), again following an approximately spiral pattern along the boundary surface. A third alternative would be to perform the merging in "planar rings", determined by intersecting the boundary surface with parallel Cartesian planes. Although the least efficient of the three algorithms described above, the last one is the closest to the two-dimensional

one used in this work.

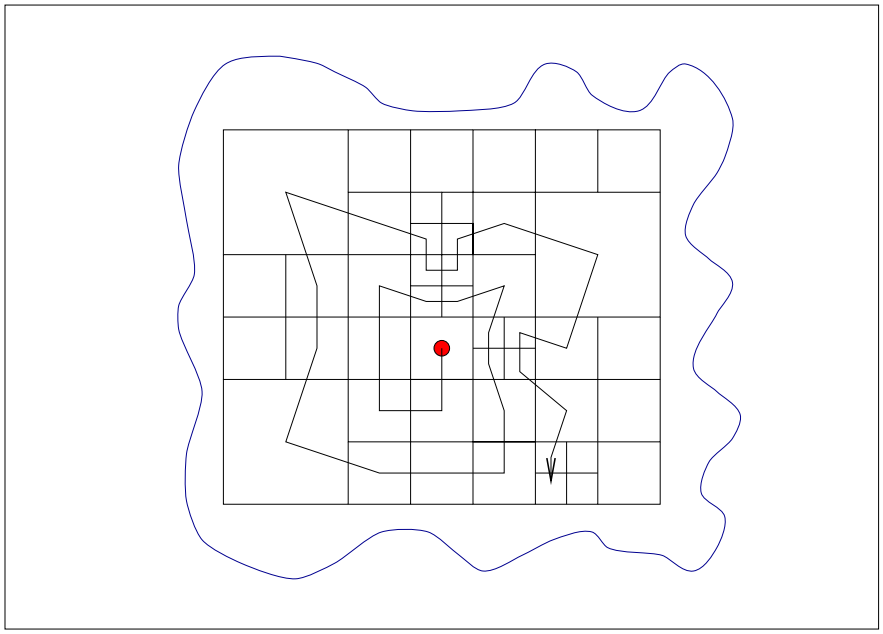

Figure 9.2: An Exhaustive Traversal Path on a Boundary Surface in Three-Dimensional Space Ensuring that all Cartesian Cells Intersected by the Surface are Incorporated into Composite Cells. The Figure Shows the Projection of the Surface and Cut Cells Onto One of the Cartesian Planes.

Because of the need for recurring geometric computations with moving boundaries (whether in two-dimensional or three-dimensional spaces), an attractive option in three-dimensional spaces would be to use only faceted boundary surfaces instead of NURBS-based or other types of polynomially-defined boundary surfaces.

Satisfaction of The Geometric Conservation Laws (described Chapter III) in three-dimensional spaces follows the same principles that it does in two-dimensional spaces. In particular, the motion step in each composite cell must be broken down into sub-steps during which the intersection pattern of each face of the composite cell with the boundary surface remains invariant. While the principle involved is analogous to that in the two-dimensional case, the implementation is again far

more involved in the three-dimensional case. The use of faceted boundary surfaces in three-dimensional spaces will again simplify the geometric computations involved and keep the computational cost to the minimum possible.

Despite the feasibility of the extensions discussed above, it should be borne in mind that the inability to perform anisotropic refinement in the standard Cartesian approach will become even more inhibiting in three-dimensional spaces than it is in two-dimensional spaces, and it will largely limit the technique to inviscid flows or low Reynolds Number viscous flows, as described further below.

9.4.2 Extension to Higher-Order Schemes

As described in Chapter IV generally, and in Section 4.4.8 particularly, an important concern with unstructured grids, and one of the main obstacles to their widespread use for error-sensitive applications (such as combustion computations), is the high discrepancy between the formal order of accuracy and the observed order of accuracy of discrete solutions on such grids. This discrepancy is often as large as 1, more than twice or four times what it is for typical structured grids. The main reason for this behavior is that unstructured grids usually have large variations in volumes between adjacent cells, and these cause significant reductions in the cancellation effects of the leading truncation error terms (whether diffusive or dispersive). The effect is especially pronounced with Adaptive Cartesian grids because of the severe abruptness of changes in cell volumes (by factors of 4 in two-dimensional spaces and 8 in three-dimensional spaces), often leading to large perturbations, inaccuracies, and phase errors across refinement boundaries, even with formally second-order-accurate schemes.

Chapter III presented another line of reasoning that explains the phenomenon

described above; namely, that k-exact schemes generally have an order of accuracy of $(k+1)$ only on uniform subregion sets, and degenerate by about one order of accuracy in non-uniform subregion sets [106]. This situation has motivated the development of quadratic (and higher) reconstruction techniques on arbitrary subregion sets that ensure that at least the second-order (and higher if need be) accuracy required for accurate evaluation of viscous and turbulent terms and quantities is maintained regardless of variations in the volumes or dimensions of adjacent cells [28, 29], [1], and [249].

The development, investigation, and implementation of such higher-order schemes would not only be valuable as a theoretical extension of the work carried out here, but would also advance the practical objectives of improving flow-solvers for Adaptive Cartesian grids, and of resolving many of the numerical problems associated with refinement boundaries in such grids.

9.4.3 Extension to Other Systems of Equations

Of all the possible feasible candidates for other systems of equations to be solved using the methodology developed in this work, including, for example, The Maxwell Equations, only two specific ones are singled out for further consideration here: (i) The Navier-Stokes Equations; and, (ii) The Three-Dimensional Steady Supersonic Euler Equations.

With regard to the extension to The Navier-Stokes Equations, distinctions are drawn between the treatments for high and low Reynolds Number, and between the treatments for compressible and incompressible flows.

For compressible flows with high Reynolds Number, three fundamental shortcomings of the non-boundary-conformal grids generated by Cartesian methods that

are not apparent, for example, with The System of Euler Equations, assume dominant importance: (i) the arbitrary variation in cell volumes and face areas near cutting surfaces and interfaces; (ii) the isotropic shape of the computational cells, and the impossibility of alignment or stretching near cutting surfaces; and, (iii) the non-orthogonality of the grid lines near cutting surfaces. While the first shortcoming may be controlled by cell merging to within a prescribed range, there is no obvious way to overcome the other two within the standard Cartesian approach. The second shortcoming is not important for the Euler Equations because the latter typically do not have relatively wide highly-anisotropic regions. The third shortcoming is mostly relevant to the accuracy of turbulence models. These shortcomings and their consequences are identified and discussed in detail in [85, 136].

The approaches that have been investigated so far to resolve or overcome the problems discussed in the preceding paragraph fall into two main categories, both of which require further exploration: (i) the application of the Boundary-Layer Integral Equations to solve the flow in the boundary-layer region, and then the coupling of the boundary-layer solution with the far-field inviscid-flow solution [399, 255]; and, (ii) the combining of multiple-grid solutions, including the use of a Cartesian-Prism grid (that is, a hybrid grid) approach [192, 397] with stretched prism grids, or a chimera approach in which the boundary-layer region is enclosed within either a stretched extruded prism layer or a stretched structured grid. Clearly, only the first approach confronts the target shortcomings within the "pure" Cartesian-grid approach. In the context of moving-boundaries, the Cartesian-Prism approach is not as attractive as a chimera approach, since it would inhibit the strong freedom that currently exists with executing topologic transformations. By comparison, the Integral-Equation approach would be the most effective one for moving-boundary simulations, but still

requires the most development.

For incompressible flows with high Reynolds Number, the shortcomings and relevant considerations are similar to those for compressible flows with high Reynolds Number.

For compressible flows with low Reynolds Number, all three of the shortcomings described above are not significant, as is the case for The System of Euler Equations. Standard convective and diffusive discretizations may be successfully used, as described and demonstrated, for example, in [85].

Extension of the methodology developed in this work to incompressible viscous flows with low Reynolds Number is also feasible and has been successfully demonstrated in [8], where it was shown that the most critical requirement is for derivation of correct stability limits if a staggered-grid discretization is adopted. A colocated discretization is also feasible and has been demonstrated.

The second extension considered in this sub-section is to The Steady, Supersonic Euler Equations in three-dimensional spaces. This extension can be carried out almost wholly within the two-dimensional framework described in this dissertation, by exploiting the mathematical similarity that exists between the time-dependent, two-dimensional version of The System of Euler Equations, and its steady, three-dimensional version [196], with one of the coordinate axes in three-dimensional space used to replace the time axis in the two-dimensional transient version. This isomorphism holds provided the flow remains supersonic throughout the computational region along the coordinate axis which replaced the time axis, that is, provided the hyperbolic character along that axis is maintained. The computational analogy described above is often called "space-marching", and has been extensively utilized to simulate steady, three-dimensional flows in that regime, especially flows over super-

sonic and hypersonic vehicles. With this analogy, successive planes perpendicular to the main flow axis in three-dimensional space correspond to successive times in the two-dimensional analogue, and the change in the geometry of the boundaries from each of these planes to the next in three-dimensional space corresponds to boundary motion in time in the two-dimensional analogue. The space-marching extension of the methodology developed in this work has been successfully carried out and demonstrated [196], but further development is still needed [196].

9.4.4 Specialization for Specific Applications

The computational technique developed and studied in this work was designed and implemented for the most general problem in its target class, which is described in detail, for example, in Chapter I. However, it can be specialized and optimized for automated, and more efficient simulation or study of specific categories or types of applications within that target class. The specialization process would involve calibration of the geometric computations, the cell merging algorithm, the flow solver, and the adaptation criteria to suit the specific category or type of problem being studied and its parameter ranges. In some cases, additional mathematical models, for example, a model for flight control, or an accurate model for structural dynamics, would also be required within the overall solution algorithm, and many of the specializations can be more effectively performed after implementing specific improvements to the general methodology, such as those discussed in Section 9.3 above.

One of the most attractive application categories for specialization as described above is the category of inviscid aerodynamic problems with moving boundaries. This includes applications involving missile or projectile trajectories, ejection of projectiles from guns, separation of objects from each other or their approach toward each other

(as in tank or projectile releases from aircraft), fluid-structure interactions, such as penetration problems generally, and impact-penetration problems specifically, and other problems where there is large-scale relative motion between solid objects. The essential common feature among these problems is that the flow is inviscid, and the moving boundaries are those of solid objects that travel over relatively large distances, requiring the numerical diffusion effects in the boundary representation to have as little effect on the solution as possible. In this category, the need to be able to handle topologic transformations automatically, or the elimination of the need for a re-meshing step, which is typically required by the traditional techniques currently used for such problems, is a useful advantage of the new methodology. Several problems from this general category are shown in Chapter VIII above.

With regard to specialization for fluid-structure interactions, it should be noted that only an elementary Finite-Element structural mechanics algorithm utilizing only point masses and spring and dash-pot elements has been implemented in this work. For accurate and reliable simulation of fluid-structure interaction problems, inclusion of a nonlinear explicit Finite-Element structural mechanics solver utilizing at least shell elements is necessary. Such an extension would enable better exploitation of the accurate manner in which the fluid mechanics and structural mechanics are currently coupled in the methodology developed in this work, allowing simulation of a large variety of important industrial and military problems, such as the prediction of dynamic fracture due to a blast wave, or the prediction of the stresses and deformations in fluid-filled vessels during impact or under earthquake-induced loading. Together with a flow solver for incompressible non-Newtonian flows, such a development or specialization would also enable simulation of several important biomedical problems, such as the flow of blood through the heart, the heart valves,

and the major arteries.

Another attractive application category for specialization is that concerned with the use of the methodology as a high-resolution, computationally-efficient technique for the study of specific problems in compressible, inviscid fluid dynamics, especially problems involving shock-shock or shock-solid interactions, whether the solids are moving or stationary. Several examples from this category are also shown in Chapter VIII.

Other categories of applications that are attractive for specialization include applications requiring accurate tracking of the motion or deformation of infinitesimally-thin interfaces, or applications requiring a general Lagrangian-Eulerian capability, especially where the variations in inter-boundary distance are large. Examples include the study of surface-tension-induced flows, the study of the effects of opening and closing of valves in machinery or biological organs, and the filling of cavities and molds during molding operations. The extension of the methodology developed in this work, for example, to the simulation and study of adhesion effects in biological cells has been demonstrated in [8].

APPENDICES

APPENDIX A

Arbitrary Intersection of Cells in Cartesian Grids

A.1 The Small-Cell Problem

Arbitrary intersection of cells by boundaries is the greatest detraction of Cartesian-grid-based methods because it necessitates special handling for cut cells, with penalties in programming simplicity and computational efficiency. Cut cells raise the following two main concerns, with the second being normally the more urgent and limiting:

1. **Accuracy:** Finite-Difference schemes that fail to take into account the possible irregular spacing near boundaries will suffer losses in accuracy, generally to the extent of loss of consistence. Finite-Volume-based solvers are less sensitive to this irregularity: with no special treatment, the reduction in the order of accuracy seems to be slight [1]; with simply no more than uniform quadratic reconstruction, second order accuracy seems tenable [53].

   In extreme cases, such as with sliver-like cell shapes, the gradient and limiting computations may become ill-conditioned. Precluding this adds to the programming complexity and computational overhead.

[1] for the steady-state test-case shown in [86], from 2.0 to about 1.4 in the L_{inf} norm, and almost none in the L_1 norm.

2. **Stability:** The area of a cut cell may be arbitrarily small. Regardless of its shape, this implies (by the CFL stability constraint: $\Delta t \leq \frac{d}{[\vec{a}_{max}+|\vec{u}_{max}|]}$, where d is a suitable length-scale for the cell, typically given the value $\sqrt{A}$, where A is the cell's area, and where the other symbols have their usual meanings) that the maximum time-step that can stably be taken in such a cell, Δt, may be arbitrarily small. The consequence of this is at least decreased convergence efficiency for steady-state computations, or arbitrarily inefficient time-marching for explicit unsteady computations.

Techniques for dealing with the above two problems are presented in Appendix A.2.

Arbitrarily intersected cells also pose (minor) algorithmic difficulties. For example, flow-solvers must be able to deal with cells having 3-5 faces, even with the simplest boundary representation. Also, unlike it is with structured grids and most unstructured ones, boundary conditions must be applied at faces that cannot be specified a-priori.

A.2 Methods for Handling the Small-Cell Problem

The most commonly-used methods for attacking the "small-cell problem" are based on one or more of the following techniques:

1. local time-stepping and linear reconstruction [102];

2. cell merging [79, 74];

3. large time-stepping [220, 221, 51];

4. flux redistribution [273]; and

5. extrapolation of variables across boundaries to interior points [120].

For reliable and accurate computations with arbitrary geometries, only methods based on the first, second or fourth techniques are viable for steady flows, while only methods based on the second or fourth ones are viable for unsteady flows. The details are given below.

A.2.1 Steady-State Computations

Method 1: In [102], it was proposed and demonstrated that for steady flows local time-stepping and linear reconstruction alone are sufficient to stabilize small cells with no observable loss in accuracy. In principle, the approach should not encounter difficulty with any type of cut cell. For cells having all dimensions comparable, this is clearly true. For those that are so slender that for stability d in the CFL constraint given in Appendix A.1 must be restricted to the smallest dimension of the cell, rather than to $\sqrt{A}$, the multiplier of the residual of update formula, $\tilde{U}^{n+1} = \tilde{U}^n + (\nu\frac{\Delta t}{A})\mathbf{Res}(\tilde{U}^n)$ can be approximated by $\frac{\nu}{[\vec{a}_{max}+|\vec{u}_{max}|]D}$ where ν is the CFL number, and D the larger dimension of the cell. Since the residual for such cells scales with D, rather than d, the update procedure will still be effective, regardless of the relative magnitudes of d and D, provided the arithmetic operations are performed in a manner that prevents disrupting effects from truncation error. Otherwise, the small cell's state may be arithmetically decoupled from those of its neighbors, leading to an error not only in the cell's state, but at least in those sharing the longer edges of the cell.

Method 2: This approach [79, 74] relied on merging the state vectors of cells that are considered to have too small an area with appropriate neighbors so that the combined area is within pre-specified bounds. Merging neighbors were selected for

each small cell by finding the uncut cell most closely aligned with the normal to the local boundary.

Method 3: The "large time-step" technique, proposed in [220, 221], and further discussed in [51], allows waves that propagate through the edges of a small cell to be integrated over time intervals too large for stability in the small cell, but only after accounting for their reflection from any solid boundaries back into the computational region. Any cells which are penetrated by the reflected waves have their updates incremented in accordance with the type and strength of the wave and the volume or area it sweeps. While this method works extremely well in 1-D, allowing local CFL numbers of several orders of magnitude to be used [36], it inexplicably does not generalize well to higher dimensions. In 2-D, the method gives poor results for cells that are, depending on their shape, less than about 10-20% of an uncut cell in area [36]. In addition, the geometric computations required to evaluate reflection directions and area overlaps are coupled to the flow-solver and are computationally expensive [36].

Method 4: An extension of the main idea of Method 3 is to simply "redistribute" the "excess" flux for any small cell to its neighbors in accordance with the lengths or areas of shared edges or faces respectively. In practice, this extension, proposed and demonstrated in [273] and related publications, does not suffer from restrictions on the minimum cell area of Method 3. Nevertheless, the basis of the accuracy of the redistribution among neighbors may be questionable.

Method 5: In [120], the same computational stencil is used for all vertices lying in the computational region. For stencils that penetrate solid boundaries, the state vector assigned to a vertex within a solid body is obtained by extrapolation from the interior of the computational region. In particular, the velocity is linearly

extrapolated from the velocity at a vertex close to the local normal and from the flow-tangency condition at another vertex lying on the surface, thereby applying the analytically-required boundary condition at the interior vertex. The interior vertex is then used in the normal way in the stencil to which it belongs, ignoring its interior status. The most obvious shortcomings of this method are its poor accuracy and its violation of conservation.

A.2.2 Time-Accurate Computations

For computations of unsteady flows using an explicit solution algorithm, the requirement for global time-stepping implies that the CFL restriction imposed by small cells cannot be circumvented by adaptations of Method 1 above. However, extensions of Methods 2 and 4 are obviously viable alternatives. This has already been demonstrated: for Method 2 in [74]; for Method 4, even with moving boundaries, in [273]. Extensions based on Method 3 alone will inherent the same limitations observed with steady-state computations.

APPENDIX B

Alternative Discretization Methods for The System of Euler Equations

The strongly nonlinear nature of The System of Euler Equations, the possibility of discontinuous solutions, and the frequent need for solutions in domains that are finite, excludes the options of discretizations based on Spectral or Boundary-Integral Methods, or any other methods resulting in discrete equations that explicitly encode non-local support. The need to perform calculations with the minimum possible use of storage space and computational effort also precludes the use of Particle-Based Methods. In general, there are three major remaining "pure" options for a system like the Euler Equations: (i) the Finite-Difference Method; (ii) the Finite-Element Method; and, (iii) the Finite-Volume Method.

The Finite-Difference Method is based on replacing derivatives in one of the Differential Forms of the Governing Equations, say,

$$\frac{\partial \vec{U}}{\partial t} + \nabla \cdot \vec{\vec{F}} = \nabla \cdot \vec{\vec{S}} \tag{B.1}$$

by ratios of Finite Differences using Taylor Series expansions [12, 169]. Because it presumes the existence of a Differential Form, and because it uses Taylor Series expansions, this method is strictly applicable only to smooth functions (or only in the smooth regions of functions with discontinuties). However, this restriction is

routinely circumvented or ignored in practice, and the Finite-Difference Method is often applied to systems of equations that support discontinuities in their solutions, including The System of Euler Equations. The Finite-Difference Method is also not applicable for the irregular or unstructured grids (which are explained and defined in Chapter IV) that are used in this work.

The Finite-Element Method [266, 430, 78] is based on discretizing the Weak Formulation of the Governing Equations, typically,

$$\frac{\partial}{\partial t}\int_{\Omega}\phi\vec{U}+\int_{\partial\Omega}\phi\vec{\vec{F}}_r\cdot\vec{n}-\int_{\Omega}\nabla\phi\cdot\vec{\vec{F}}_r=\int_{\partial\Omega}\phi\vec{\vec{S}}\cdot\vec{n}-\int_{\Omega}\nabla\phi\cdot\vec{\vec{S}}, \tag{B.2}$$

with continuous piecewise-linear or higher-order approximations within each cell in the Computational Region. The method is readily applicable to unstructured grids. The main variants of the Finite-Element Method that have been successfully demonstrated for the Euler Equations are the following six: (i) The Standard Galerkin Formulation with Artificial Dissipation [184]; (ii) The Least-Squares Galerkin Formulation [186]; (iii) The Taylor-Galerkin Formulation [275, 229, 276]; (iv) The Richtmeyer-Galerkin Formulation [15]; (v) the Petrov-Galerkin Formulation; and, (vi) The Discontinuous Galerkin Formulation [234].

The Finite-Volume Method [244, 169, 170] is based on discretizing directly one of the Integral Forms of the Governing Equations, say,

$$\frac{\partial}{\partial t}\int_{\Omega}\vec{U}+\int_{\partial\Omega}\vec{\vec{F}}_r\cdot\vec{n}=\int_{\partial\Omega}\vec{\vec{S}}\cdot\vec{n}, \tag{B.3}$$

for each cell in the Computational Region, to give an algebraic equation for that cell. The resulting set of algebraic equations for the assembly of all the cells is then solved by any appropriate discrete-solution method. The Finite-Volume Method is readily applicable to unstructured grids and to cells of arbitrary shape. Three main families of Finite-Volume method may be identified: (i) The Cell-Vertex Methods

[244, 262], in which the discrete solution is associated with cell vertices; (ii) the Cell-Centered Methods, in which the discrete solution is associated with cells, or with points within the cells [185]; and, (iii) the Vertex-Centered Methods [25], in which the discrete solution is associated with cell vertices reconfigured as cell centers in a dual grid.

A variety of "hybrid" methods, such as the Control-Volume-Based Finite-Element Method [236], are also applicable to arbitrary unstructured grids and are feasible alternative discretization methods for this work. The recent development of Fluctuation Splitting Schemes [105], which attempt to model the multi-dimensional propagation of characteristic waves directly, also offers another feasible discretization option.

APPENDIX C

Segregated- and Coupled-Solution Approaches

In the segregated-solution approach, which for fluid mechanics is also known as the so-called "pressure-based" approach, each of the governing equations is associated with one of the independent variables, and is solved separately, typically using an iterative technique. The individual equations in the whole system are solved in sequence, one after the other, and while each equation is being solved, the values of the dependent variables in it that are associated with any of the other governing equations are frozen. A whole sequence of individual solutions is typically called a "sweep". The sequential solution procedure is repeated until the solutions for each of the individual governing equations converge to a fixed solution (for a given time-step, or for the overall steady-state solution). The fixed solution which is attained is then independent of the iterations taken within the solution procedure of that equation, and also independent of the sweeps carried out over the entire set of equations. While each of the governing equations is being separately solved, the flux evaluation across cell faces is confined to that of the conserved quantity associated with that governing equation.

In the coupled-solution approach, which for fluid-mechanics is also known as the so-called "density-based" approach, all the governing equations are solved simulta-

neously. The flux evaluation across cell faces is obtained for the conserved quantities associated with all of the governing equations simultaneously.

For compressible flows, the coupled-solution approach is generally considered to be more accurate and based on a stronger theoretical foundation. However, the performance of methods using this approach rapidly deteriorates as the Mach Number approaches zero. The segregated-solution approach is generally considered to be more robust overall, and is particularly well-suited for incompressible flows, where the velocity and pressure fields must be coupled over the entire flowfield. However, the performance of methods using this approach slowly deteriorates as the Mach Number increases beyond 2 or 3. A comparison of the relative strengths and weaknesses of these two approaches is discussed in [241].

APPENDIX D

Centered Discretizations or Flux Functions

In centered discretizations [185, 349, 306], the instantaneous flux, $\vec{H}_\perp$, at a point in a polyhedral face separating two computational cells is evaluated using a formula of the form

$$\vec{H}_\perp = \vec{H}_\perp(\vec{U_L}) + \vec{H}_\perp(\vec{U_R}) + \vec{D}(\vec{U_L}, \vec{U_R}),$$

where $\vec{U_L}$ and $\vec{U_R}$ refer to the state vectors on the "left" and on the "right" sides of the cell face at the point at which the flux is being evaluated. The "left" and "right" orientations change depending on which of the two cells adjoining that face is being updated with the flux: the "left" direction is associated with the interior of the cell being currently considered, while the "right" direction is associated with the exterior of that cell. The vector $\vec{D}$ is a mandatory dissipation term.

The flux in a centered discretization may also be expressed in the form

$$\vec{H}_\perp = \vec{H}_\perp(\theta\vec{U_L} + (1-\theta)\vec{U_R}) + \vec{D}(\vec{U_L}, \vec{U_R}),$$

where θ is a parameter satisfying $0 \leq \theta \leq 1$, and all the other terms are as defined above.

The addition of dissipation here is mandatory because the centered discretization has no dissipation, and so would otherwise be unconditionally unstable. The amount

of dissipation that is added should ideally be just sufficient to stabilize the scheme, to control oscillations around discontinuities, and to eliminate entropy-violating solutions. Clearly, this amount is problem-dependent.

In order to confine attention to the main concepts and the most important mathematical properties of centered flux discretizations, the discussion of the dissipation term in the remainder of this subsection will be confined to the one-dimensional case, assuming that the left and right states and their neighboring states are all cell-centered averages on a structured grid (which is explained and defined in Chapter IV).

In the scalar dissipation approach [185, 287], the dissipation vector is evaluated using a formula of the form

$$\vec{D}(\vec{U_L}, \vec{U_R}) = \lambda \left[\epsilon_2 \delta^2 \vec{W} - \epsilon_4 \delta^4 \vec{W} \right],$$

where the terms $\delta^2 \vec{W}$ and $\delta^4 \vec{W}$ are the second and fourth differences in the vector $\vec{W} = \vec{U} + (0, 0, 0, 0, p)^T$ for the three-dimensional case, where the term λ is given by

$$\lambda = \frac{1}{2} (\lambda_L + \lambda_R),$$

where λ_L and λ_R are the maximum spectral radii in the flux Jacobian matrices of the left and right states, respectively, and where the terms ϵ_2 and ϵ_4 are given by

$$\epsilon_2 = \kappa_2 \; max(\nu_L, \nu_R), \qquad \text{and} \qquad \epsilon_4 = max(0, \kappa_4 - \epsilon_2),$$

where κ_2 and κ_4 have constant values, typically chosen to be $\frac{1}{2}$ and $\frac{1}{64}$, respectively, where the value of ν_L is given by

$$\nu_L = \left\| \frac{p_{L_-} - 2p_L + p_{L_+}}{p_{L_-} + 2p_L + p_{L_+}} \right\|,$$

where p_{L_-} and p_{L_+} refer to the pressures further to the left and further to the right of the state $\vec{U_L}$, while the value of ν_R is given by an expression analogous to that for ν_L.

As can be seen by inspection of the dissipation equation, the purpose of the formulation is to smoothly increase the second-order term near pressure discontinuities while decreasing the fourth-order term. This enables satisfaction of the jump conditions and eliminates oscillations near discontinuities. Away from pressure discontinuities, the second order term is seen to vanish while the fourth-order term supplies the dominant contribution, providing the required reduced dissipation that is sufficient to stabilize the computation in smooth regions of the flow.

The scalar dissipation model in many forms and variants have been widely used for The System of Euler Equations, but all these retain the disadvantage that the same dissipation scaling is used for all the governing equations in the system. This deficiency is largely remedied in the matrix dissipation approach [349], which is based on scaling the dissipation for each of the individual governing equation in a system in accordance with the eigenvalues rather than the spectral radii of flux Jacobian matrices. Indeed, it is sufficient [349] to simply replace the λ values in the scalar dissipation model by the absolute values of the flux Jacobian matrices, $|A|$, to achieve this selective dissipation. Also, it is not necessary to compute the $|A|$ values, since it is normally required only to evaluate the product of the absolute value matrix with a vector [349], and this can be done more efficiently without directly computing $|A|$. In practice, the dissipation scalings have to be increased beyond the magnitudes of the eigenvalues since otherwise, the vanishing of one of these eigenvalues [170] near stagnation points or sonic points would largely eliminate the dissipation and render the scheme unstable there.

D.1 Comparison of the Central and Upwind Flux Functions

In a discretization based on upwind-differencing, the dissipation addition for each of the equations in a system (of equations) is independently and automatically determined, requiring no external input, but rendering it difficult to control the extent of damping. On the other hand, the amount of dissipation added in a discretization based on central-differencing requires the specification of the dissipation model parameters, which are often difficult to determine, but offering the opportunity of direct control over the damping behavior. While the two techniques have been shown to be largely equivalent under special circumstances [181], upwind schemes are still generally believed to capture shocks and other discontinuities with higher resolution, albeit at additional computational cost.

APPENDIX E

Flux-Vector Splitting

In Flux-Vector Splitting (FVS), the total flux, $\vec{H}_\perp$, is composed from two components, in accordance with the general formulation

$$\vec{H}_\perp = \vec{H}_\perp\left(\overline{\vec{U}_L}, \overline{\vec{U}_R}\right) = \vec{H}_\perp^+\left(\overline{\vec{U}_L}\right) + \vec{H}_\perp^-\left(\overline{\vec{U}_R}\right).$$

The characterizing feature of the separation or "splitting" expressed in the above equation is that while the total flux, $\vec{H}_\perp\left(\overline{\vec{U}_L}, \overline{\vec{U}_R}\right)$ is a function of the two states across the interface, each of the components into which it is split is a function of only one of the two states. The component $\vec{H}_\perp^+\left(\vec{U}_L\right)$ propagates along the outward-pointing normal of the interface separating the two states, from left to right, while the component $\vec{H}_\perp^-\left(\vec{U}_R\right)$ propagates in the opposite direction. The propagation directions of participating waves are determined by the eigenvalues of the Flux Jacobian matrix.

Several schemes sharing the fundamental FVS character described above have been developed and demonstrated, but only the most popular of these in the literature will be briefly reviewed here.

The first reported FVS scheme [345] was constructed by using the Homogeneity property of the flux vectors of the System of Euler Equations, namely;

$$\vec{H}\left(\alpha\vec{U}\right) = \alpha\vec{H}\left(\vec{U}\right).$$

By differentiating this equation, it can be shown [345] that the Flux Jacobian Matrix for the flux normal to an interface (see the derivation of the Characteristic quasi-linear form in Chapter II) can be split into two parts,

$$\frac{\partial \vec{H}_\perp}{\partial \vec{U}} = \frac{\partial \vec{H}_\perp^+}{\partial \vec{U}} + \frac{\partial \vec{H}_\perp^-}{\partial \vec{U}},$$

such that the matrices $\frac{\partial \vec{H}_\perp^+}{\partial \vec{U}}$ and $\frac{\partial \vec{H}_\perp^-}{\partial \vec{U}}$ are simultaneously diagonalizable into matrices that have only nonnegative and nonpositive eigenvalues respectively. In this manner the wave contribution from each state to the flux at the interface can be computed using only the two states across the interface.

Although simple and physically appealing, this FVS scheme possesses the undesirable property of non-differentiability of the split fluxes wherever the eigenvalues change sign [62], including all stagnation and sonic points. Unless overcome, by, for example, offsetting the eigenvalues away from zero by a small amount, this property may cause small non-physical oscillations and convergence difficulties [62, 13].

The formulae and properties of the FVS scheme just described are valid for arbitrarily moving and deforming cells and grids provided the state vectors are computed in a frame of reference that is instantaneously moving at the speed of the interface between the two states. The same is true of the second FVS scheme, presented below.

The second FVS scheme to be published [379] is also a continuous function of the Mach Number, but is constructed with the explicit requirement for differentiability at sign changes of the eigenvalues. In particular, the flux splittings in that scheme

are given by

$$\vec{H}_{\perp}^{+}\left(\vec{U}_L\right) = \frac{\rho c(1+M_{\perp})^2}{4}\begin{pmatrix} 1 \\ u+(2-M_{\perp})\frac{c}{\gamma}n_x \\ v+(2-M_{\perp})\frac{c}{\gamma}n_y \\ w+(2-M_{\perp})\frac{c}{\gamma}n_z \\ \frac{c^2}{\gamma+1}\left(\frac{2}{(\gamma-1)}+2M_{\perp}-M_{\perp}^2\right)+\frac{q^2}{2} \end{pmatrix},$$

and

$$\vec{H}_{\perp}^{-}\left(\vec{U}_R\right) = -\frac{\rho c(1-M_{\perp})^2}{4}\begin{pmatrix} 1 \\ u-(2-M_{\perp})\frac{c}{\gamma}n_x \\ v-(2-M_{\perp})\frac{c}{\gamma}n_y \\ w-(2-M_{\perp})\frac{c}{\gamma}n_z \\ \frac{c^2}{\gamma+1}\left(\frac{2}{(\gamma-1)}-2M_{\perp}-M_{\perp}^2\right)+\frac{q^2}{2} \end{pmatrix}.$$

The differentiability in this splitting was achieved by construction of the split fluxes as polynomials of the lowest degree in the Mach Number needed to guarantee uniqueness. As exhibited in the above formulae, the mass, momentum, and energy fluxes are respectively polynomials of degree 1, 2, and 3 in the Mach Number. The Flux Jacobian Matrices $\frac{\partial \vec{H}_{\perp}^{+}}{\partial \vec{U}}$, and $\frac{\partial \vec{H}_{\perp}^{-}}{\partial \vec{U}}$ may be derived in the usual manner, and verified to respectively always have nonnegative and nonpositive eigenvalues, implying that they can therefore be differenced in an upwind manner. The following properties can also be easily verified from the above formulations:

$$\forall M_{\perp}: \qquad \vec{H}_{\perp} = \vec{H}_{\perp}^{+} + \vec{H}_{\perp}^{-},$$

$$\forall M_{\perp} \geq +1: \qquad \vec{H}_{\perp} = \vec{H}_{\perp}^{+}; \qquad \vec{H}_{\perp}^{-} = 0, \quad \text{and}$$

$$\forall M_{\perp} \leq -1: \qquad \vec{H}_{\perp} = \vec{H}_{\perp}^{-}; \qquad \vec{H}_{\perp}^{+} = 0.$$

Another property of the splitting in this scheme is that both of the flux Jacobians have exactly one zero eigenvalue in the range $-1 \leq M_{\perp} \leq 1$, allowing resolution of

a steady shock with only two interior states with first-order schemes, and only one interior state with a higher-order scheme [13].

An unexpected property of both of the splittings described above is that since the energy and mass fluxes are differenced differently, it is possible for the total enthalpy to be computed non-conservatively (that is, it is possible for the computed stagnation enthalpy to be non-constant, in violation of the "Isenthalpic Property" described in Section 2.2). This "non-conservation" may occur even while the mass, energy, and momentum equations are computed conservatively. In [149], this difficulty was rectified for the splitting of [379] by modifying the energy flux to be computed as the product of the mass flux and the stagnation enthalpy, instead of by the original formulation given above. Another undesirable property of both of the above splittings is that they are excessively diffusive for shear layers. This occurs, because unlike the case for shock waves, shear waves have no inherent anti-diffusive effects to oppose the numerical diffusivity.

Several attempts have been made to modify and reformulate the original Flux Vector Splitting approach, in some cases even discarding the defining properties of FVS schemes, but often leading to significant improvements in performance for some problems. A review of most of the classical FVS schemes and some of the recent attempts at improving and combining the original schemes with others is given in [87]. Recent efforts have focused on combining the Flux Vector Splitting idea with other approaches. In [224], an FVS scheme (called the Advection Upstream Splitting Method, AUSM) is proposed which discards some of the unfavorables properties of FVS methods, and which can more readily be linearized for implicit operators. Two other methodologies that are very similar to AUSM have also been published, under the names WPS [147] and CUSP [181]. In a more recent development, [416], a novel

technique using a kinetic theory approach, but based on an FVS computational method, was developed. The scheme satisfies the property of Positive Conservation, which is defined and discussed in Chapter III. In [89], a hybrid technique is proposed in which an FVS scheme is used for the nonlinear waves, while an FDS scheme (which is described in Chapter III) is used for the linear waves, leading to flux functions that exhibit the robustness of the FVS approach and the low diffusivity of the FDS approaches.

APPENDIX F

Other Upwind Methods

The **Random-Choice Method** was originally developed [132] for use in proving existence theorems for the weak formulation of hyperbolic conservation laws. Eventually, it was adapted and applied to the evaluation of flux functions and the formulation of computational schemes [76, 77]. The basic procedure of this method is to solve either an exact or an approximate Riemann problem at each computational-cell interface and then to define the solution within each computational cell to be the value of that exact or approximate Riemann solution at a randomly selected point within that cell. Although formally not conservative and only first-order accurate in space, this scheme can capture discontinuities with high resolution, mostly because it is has no dissipation. A general review and analysis of this method, its properties, and its possible improvements can be found in [76, 77, 80]. Primarily because of its limitation to one-dimensional problems, the method has limited practical significance.

In all the upwind schemes described in Chapter III, the extension from one-dimensional to multi-dimensional computations is accomplished by assuming that waves always propagate normal to cell faces. As remarked in Chapter III, other than in the very special case in which flow-features are aligned with the cell faces

on a geometrically-regular grid, this approach to extension from single to multiple dimensions inherently introduces a grid-alignment dependence in the solution. For example, as described in [382], a shear wave oblique to a cell face is treated as a superposed compression wave and shear wave that are respectively perpendicular to and aligned with the face. In practice this forcing of alignment results in smearing of all discontinuities, and in the appearance of spurious pressure and shear waves throughout the domain. The family of **Multi-dimensional** or **Fluctuation Splitting** techniques was motivated largely by the need to counter this effect, and most of the methods devised under this category to date are reviewed in [346] and [382].

The **The PieceWise Parabolic Method (PPM)** [84] features an inbuilt higher-order spatial accuracy that is achieved by reconstruction of the members of the state vector in each computational cell using piecewise parabolic functions. This method adds some dissipation to that computed from upwind-differencing, and therefore appears to be more stable and robust than other upwind methods.

APPENDIX G

The Mathematical Bases of Boundary Condition Treatments for The System of Euler Equations

A fundamental analytical requirement for all boundary-condition treatment techniques is that no new unconstrained values of the dependent variables may be introduced at any point in the boundary of the solution domain. Instead, the values of the dependent variables at boundaries must either be specified as given values, or determined from the values of the dependent variables within the interior of the domain. The discrete analog of this requirement is that no new unconstrained discrete unknowns may be introduced in association with boundary treatments.

Since, as shown in Section 2.2, The System of Euler Equations is purely hyperbolic in time, and either purely hyperbolic, partially parabolic, or partially elliptic in space, the analytic treatment of its boundary conditions must conform with the requirements of the Theory of Characteristics. The situation is analogous for the discrete case.

Referring to the Characteristic Quasi-Linear Form of The Sytem of Euler Equations, which is presented in Section 2.2, the consequences of the Theory of Characteristics are simple [206, 207]: each wave that propagates *into* the Computational Region, as given by the wave-speeds of Equation 2.49, requires the Riemann Variable for that wave to be specified as a "free" or "external" (that is, given) boundary

condition value; each wave that propagates *out* of the Computational Region requires the Riemann Variable for that wave to have its value not freely specified, but instead computed from the numerical solution within the Computational Region. In this manner, the Numerical Domain of Dependence is ensured to lie within the Physical Domain of Dependence for each wave, and all wave propagation effects are guaranteed to be discretized in a consistent and linearly stable manner [207].

In accordance with the above discussion, and in the context of a Finite-Volume Method, waves are considered to propagate into and out of the Computational Region only across and perpendicularly to boundary faces, and the arbitrary vector $\vec{n}$ associated with Equations 2.48 and 2.49 refers to the outward-pointing normal of the boundary face for which the boundary conditions are being analyzed. An analogous interpretation holds for a Finite-Difference Method. Since the local wave-propagation analysis is carried out using a quasi-one-dimensional form of the Characteristic Equations, the theory as described above can be considered to be valid for multi-dimensional situations. For multi-dimensional computations, it turns out that a similar analysis to that described above can be derived along an arbitrary direction away from a boundary [315], so that the interpretation of $\vec{n}$ given above can be relaxed if this interpretation is reflected in the formulation.

Specifications of "free" boundary values are conventionally called **Physical Boundary Conditions**. Specifications of boundary values extracted from the interior of the Computational Region are conventionally called **Numerical Boundary Conditions**, and the procedures used to implement them are conventionally called **Numerical Boundary Procedures** [253]. These naming conventions are used for all boundary-condition treatment techniques.

One of the limitations of the theoretical foundation described above is self-evident: the Riemann Variables only become Riemann Invariants in simple-wave regions. In non-simple-wave regions, the Riemann Invariants do not exist, and the individual waves of the Euler System cannot be decoupled. Nevertheless, the theory is usually generalized to regions with arbitrary wave-family combinations in analytical work by performing a local linearization. Under these conditions, it turns out that the simple Characteristic-Based principle described above is the only strict general principle which is necessary [206, 207] (but not always sufficient) for well-posedness of computational schemes for The System of Euler Equations. Another limitation arises from the fact that the foregoing ignores any interactions between outgoing waves that occur outside the domain and that are then reflected back into the domain. The fact that such "external" interactions are ignored implies that, strictly, it is impossible to locally formulate remote or far-field Boundary Conditions for problems having two or more spatial dimensions. Ignoring this implication, however, seems to have limited effects for practical purposes, except when highly-accurate boundary conditions for long intergration times are sought.

As described above, the Theory of Characteristics provides a complete and rigorous mathematical basis for the linearized treatment of boundary conditions for The System of Euler Equations. The theory not only specifies the number of discretized Physical Boundary Conditions and the number of discretized Numerical Boundary Conditions that are required at each point in the boundary (or correspondingly, at each boundary face in a Finite-Volume Method), but it also identifies the formulation of the variables in the boundary treatment (namely, the Riemann Invariants) that must be specified or extracted from the interior solution. More specifically, the boundary conditions required, and a consistent, stable manner in which they may be

specified for all possible flow states with respect to the outward-pointing normal at a boundary point (or correspondingly, with respect to the outward-pointing normal of a boundary face in a Finite-Volume Method) may be summarized as follows, with all symbols having the meanings defined for them in Chapter II:

1. **Supersonic Outflow:** Here, all of the five characteristic eigenvalues $u_\perp - c$, $u_\perp$, $u_\perp$, $u_\perp$, and $u_\perp + c$ are positive. Therefore, no physical boundary conditions may be specified, and five numerical boundary conditions corresponding to the five Riemann Invariants $u_\perp - \frac{p}{\rho_o c_o}$, $\rho - \frac{p}{c_o^2}$, $u_{\parallel_1}$, $u_{\parallel_2}$, and $u_\perp + \frac{p}{\rho_o c_o}$ must be determined from the interior of the Computational Region.

2. **Supersonic Inflow:** Here, all of the five characteristic eigenvalues $u_\perp - c$, $u_\perp$, $u_\perp$, $u_\perp$, and $u_\perp + c$ are negative. Therefore, no numerical boundary conditions may be specified, and five physical boundary conditions corresponding to the five Riemann Invariants $u_\perp - \frac{p}{\rho_o c_o}$, $\rho - \frac{p}{c_o^2}$, $u_{\parallel_1}$, $u_{\parallel_2}$, and $u_\perp + \frac{p}{\rho_o c_o}$ must be specified.

3. **Subsonic Outflow:** Here, the single characteristic eigenvalue $u_\perp - c$ is negative, while the four characteristic eigenvalues $u_\perp$, $u_\perp$, $u_\perp$, and $u_\perp + c$ are all positive. Therefore, four numerical boundary conditions corresponding to the four Riemann Invariants $\rho - \frac{p}{c_o^2}$, $u_{\parallel_1}$, $u_{\parallel_2}$, and $u_\perp + \frac{p}{\rho_o c_o}$ must be determined from the interior of the Computational Region, and only one physical boundary condition corresponding to the Riemann Invariant $u_\perp - \frac{p}{\rho_o c_o}$ must be specified.

4. **Subsonic Inflow:** Here, the four characteristic eigenvalues $u_\perp$, $u_\perp$, $u_\perp$, and $u_\perp - c$ are all negative, while the single characteristic eigenvalue $u_\perp + c$ is positive. Therefore, one numerical boundary condition corresponding to the

Riemann Invariant $u_\perp + \frac{p}{\rho_o c_o}$ must be determined from the interior of the Computational Region, and four physical boundary conditions corresponding to the four Riemann Invariants $\rho - \frac{p}{c_o^2}$, $u_{\parallel_1}$, $u_{\parallel_2}$, and $u_\perp - \frac{p}{\rho_o c_o}$ must be specified.

5. **Impermeable Boundary:** Here, the four characteristic eigenvalues $u_\perp$, $u_\perp$, $u_\perp$, and $u_\perp + c$ are all non-negative, while the single characteristic eigenvalue $u_\perp - c$ is negative. Therefore, four numerical boundary conditions corresponding to the four Riemann Invariants $\rho - \frac{p}{c_o^2}$, $u_{\parallel_1}$, $u_{\parallel_2}$, and $u_\perp + \frac{p}{\rho_o c_o}$ must be determined from the interior of the Computational Region, and one physical boundary condition corresponding to the Riemann Invariant $u_\perp - \frac{p}{\rho_o c_o}$ must be specified.

Generalization of the principles described above to boundary conditions on boundaries that are moving or deforming is also simple: the only additional requirement [66] is that the speeds that determine whether waves are incoming or outgoing across a boundary must be computed relative to a reference frame moving at the instantaneous local speed of the boundary being considered. For this purpose, the Characteristic Eigenvalue Equation 2.49 can be re-written in the more general form

$$\vec{\lambda} = \left(u_{\perp r} - c, u_{\perp r}, u_{\perp r}, u_{\perp r}, u_{\perp r} + c\right)^T, \tag{G.1}$$

where $u_{\perp r}$ indicates the normal relative speed of the fluid, and all other symbols have the meanings defined for them in Chapter II.

The principles described above for treatment of boundary conditions imply that regardless of any changes in PDE-type in The System of Euler Equations that may occur within the Computational Region, the application of the boundary conditions that are necessary to ensure global numerical properties such as stability and well-posedness can remarkably be carried out entirely by local analysis of the flow con-

ditions in the immediate vicinity of boundaries. The only known exceptions to this general principle are situations in which the flow states are identical at an the inlet and exit in quasi-one-dimensional reversible flows [415]: such situations require the inflow and outflow boundary conditions to be specified by using different variables to avoid non-uniqueness and hence ill-posedness in the solution.

Improper treatment of discretized boundary conditions may sometimes lead to negligible errors in the computational results; at other times, it may cause instability or a severe degradation in the convergence performance of the overall computational scheme. Upwind-biased flow solvers are usually more tolerant than other solvers with regard to improper treatment of boundary conditions. For example, for supersonic outflow, even if up to five physical boundary conditions are erroneously specified, as long as the specified values still result in the flux being computed by pure upwinding from the interior of the Computational Region, then the flux at the boundary will correctly be computed as if it were entirely determined from the solution values within the Computational Region, and no adverse effects on the solution will be introduced. Although it is possible to deviate from the general principles dictated by linear analysis as described above, analytical results [146] as well as empirical observation indicate that adverse effects on stability increase with increasing departure from the Characteristic-Theory formulation described above or its equivalents.

The order of accuracy of boundary schemes is often in practice not given sufficient care. This is somewhat justified from a practical point of view since in general it is observed that the overall order of accuracy of a computational solution will not be degraded if the boundary scheme has an order of accuracy lower by no more than one than the order of accuracy of the interior scheme. This observation has been proved [143] for the arbitrary linear computational scheme, and is discussed and explained

further in [145].

The strong association between the Characteristic-Theory basis of treatment of boundary conditions and the construction principles of upwind-biased interior schemes is evident in the above, and is probably another contributor to the wide appeal of upwind-biased methods.

APPENDIX H

Techniques of Boundary Condition Treatments for The System of Euler Equations

A general review of the practical implementation of boundary condition treatments for The System of Euler Equations is given in Reference [401], which also includes discussions on the mathematical foundations of these treatments. Only the most successful and general-purpose of boundary-condition implementation techniques for The Euler Equations will be discussed in this Appendix.

Appendix G describes how the Theory of Characteristics provides the mathematical foundation for the treatment of boundary conditions for The Euler Equations. However, the question of how to obtain the discrete values of the Characteristic Variables at boundaries is left open, allowing the development of alternative methods of implementing boundary condition treatments. All the methods described below can be implemented either in explicit or implicit form, to match the interior scheme.

One of the alternative approaches [253, 288, 362] is to obtain the values of the Characteristic Variables at boundaries by solving the system of linearized characteristic equations, which is a system of ordinary differential equations, usually using some iterative technique. This approach and its variants is called the **Characteristic Boundary Condition Technique**.

Another alternative is to compute the values of the Characteristic Variables by extrapolation to the boundaries from the interior values. This approach and its variants is called the **Characteristic Extrapolation Technique**, and is less precise than the Characteristic Boundary Condition Technique but more popular because it circumvents the need to solve the Characteristic or Compatibility equations. However, even with this simplification, methods that use Characteristic Variables remain inconvenient. This is because Characteristic Variables and Riemann Invariants are not commonly used as the dependent unknowns in computational schemes, and because it is more convenient to be able to specify the boundary conditions in terms of the conserved variables, or, usually even more conveniently, in terms of the primitive variables. This motivates the need for a special variant of the Characteristic Extrapolation Technique, which is based on exploiting the non-singularity of the transformation Jacobians between the characteristic variables and the conserved variables and between the characteristic and the primitive variables [419]. This non-singularity can be used to show that any combination of primitive and conserved variables may be specified as given boundary conditions instead of the Riemann Invariants, provided that the number of physical and numerical boundary conditions given by the Theory of Characteristics is complied with. The only exception to this general rule is that the density cannot be excluded from the possible combinations of Physical Boundary Condition variables at a subsonic inflow. The resulting extrapolation technique is among the most useful yet still rigorous of the Characteristic Extrapolation Techniques.

Another alternative is to solve discrete forms of the differential form of the Compatibility Equations, including the physical boundary conditions as additional constraints [66, 104]. This approach and its variants is called the **Compatibility-**

Equation Technique. The approach may be completely unified with the interior scheme for both explicit and implicit schemes [66, 104, 343].

Another alternative, closely related to the family of Characteristic Boundary Condition Techniques, is the family of **Non-reflecting Boundary Condition Techniques** [119, 167, 362, 129, 19], of which a general review is given in [131]. The main idea of this approach is that waves that would enter the domain are prevented from doing so at the level of the formulation of the boundary condition treatment, by forcing the strengths of these waves to vanish. The purpose of this is to prevent the boundary from "reflecting" back waves into the domain that would slow down the convergence of steady-state solutions. A novel approach along similar lines [239] demonstrates the use of non-reflecting far-field boundary conditions and the use of absorbing boundary conditions at impermeable surfaces that decrease the amplitude of reflected waves to improve convergence rates.

Another alternative [171, 394] is based on using asymptotic expansions of The Euler Equations, to couple an arbitrary linearized analytical solution in the interior of the Computational Region with the nonlinear applied Numerical Boundary Conditions. The technique is limited to steady-state computations but is valid for arbitrary flows, including ones with vorticity and shock waves.

The most popular technique for implementation of boundary conditions for The Euler Equations is based on simple extrapolation of either the conserved variables or the primitive variables, respecting only the number of Physical Boundary Conditions and the number of Numerical Boundary Conditions that are prescribed by the Theory of Characteristics, but without regard to the Characteristic Variables or the individual waves involved. This technique is know as the **Variable Extrapolation Technique** [140, 305]. In this technique, Physical Boundary Conditions

are most often specified as a combination of one or more primitive variables, since this establishes the most immediate correspondence with the physical problem being simulated, and since it allows the boundary conditions to be selected in the form that is most appropriate for the physical circumstances of the specific application being considered [253], for example, knowledge of upstream stagnation conditions. Numerical Boundary Conditions are specified by extrapolating some combination of either the primitive or the conserved variables from the interior of the Computational Region, according to an interpolating function of zeroth or higher order in space or time. Although this approach is the least strongly founded of all, any negative effects it has on the stability of the interior scheme in implicit implementations appears to be minimal [420].

H.1 Boundary Conditions Simulating Physical Models

Boundary conditions and procedures may be applied to introduce the effects of "lumped" or "reduced" models into the interior scheme. The models may be coupled to or independent of the global solution.

In the **Transpiration Boundary Condition**, a non-zero normal velocity is specified along impermeable boundaries to displace the inviscid mean flow away from boundaries. This provides a simple and effective means of approximately modeling some of the effects of viscous boundary layers within solutions of the Euler Equations [412]. The magnitude of the normal velocity may vary with time and with location in the boundaries, and may also be coupled to the solution.

The **Actuator Disc Model** and its variants [242, 243, 162] can be used to introduce some of the time-averaged effects of devices that interact with the flow, especially propulsion devices such as propellers or jet engines. The device is modeled

with a double-sided boundary-condition (often with vanishing thickness) applied on faces of interior cells. The conditions across the two opposite sides of the double-boundary are linked together to reflect the total jumps imposed by the device on, say, the stagnation pressure, the swirl, the total enthalpy, or any other property or combination of properties, possibly as a function of the radial distance from a given hub, and possibly in a manner that takes into account the geometry and angular speed of the device. In accordance with the analytical requirements described in Appendix G and Chapter III, the inlet of the device must be treated as an outflow boundary condition for the Computational Region, while the exit of the device must be treated as an inflow boundary condition for the Computational Region, but no other constraints are needed. The model can be precisely tailored for specific applications, and complete freedom exists in representing the internal action of the device.

REFERENCES

REFERENCES

[1] R. Abgrall, *An Essentially Non-Oscillatory Reconstruction Procedure for on Finite-Element Type Meshes: Application to Compressible Flows*, Computational Methods in Applied Mechanical Engineering, Volume 116, Pages 95–101, 1994.

[2] M. Aftosmis, D. Gaitonde, and S. Tavares, *On the Accuracy, Stability, and Monotonicity of Various Reconstruction Algorithms for Unstructured Meshes*, AIAA Paper 94-0415, AIAA, January, 1994.

[3] M. Aftosmis, D. Gaitonde, and S. Tavares, *Behavior of Linear Reconstruction Techniques on Unstructured Meshes*, AIAA Journal, AIAA, Volume 33, Number 11, Pages 2038–2049, 1995.

[4] M. Aftosmis, J. Melton, and M. Berger, *Adaptation and Surface Modeling for Cartesian Mesh Methods*, AIAA Paper Number 95-1725-CP, Proceeding of the 12th AIAA Computational Fluid Dynamics Conference, AIAA, June 1995.

[5] Advisory Group for Aerospace Research and Development (AGARD), North Atlantic Treaty Organization (NATO), *Test Cases for Inviscid Flow Field Methods*, Advisory Report Number 211 (AGARD-AR-211), Published by Specialized Printing Services Limited, Essex, U.K., May, 1985.

[6] Advisory Group for Aerospace Research and Development (AGARD), North Atlantic Treaty Organization (NATO), *A Selection of Experimental Test Cases for the Validation of CFD Codes*, Advisory Report Number 303 (AGARD-AR-303), Volume I, Published by Specialized Printing Services Limited, Essex, U.K., August, 1994.

[7] Advisory Group for Aerospace Research and Development (AGARD), North Atlantic Treaty Organization (NATO), *A Selection of Experimental Test Cases for the Validation of CFD Codes*, Advisory Report Number 303 (AGARD-AR-303), Volume II, Published by Specialized Printing Services Limited, Essex, U.K., August, 1994.

[8] G. Agresar, *A Computational Environment for the Study of Circulating Cell Mechanics and Adhesion*, Ph.D. Dissertation (in Biomedical Engineering and Scientific Computing), The University of Michigan, 1996.

[9] G. Agresar, J. Linderman, G. Tryggvason, and K. Powell, *An Adaptive, Cartesian, Front-Tracking Method for the Motion, Deformation, and Adhesion of Circulating Cells*, Journal of Computational Physics, Volume 143, Pages 346–380, 1998.

[10] S. Allwright, *Multi-Block Topology Specification and Grid Generation for Complete Aircraft Configurations*, in Application of Mesh Generation to Complex Three-Dimensional Configurations, AGARD Conference Proceedings Number 464 11.1-11.11, 1990.

[11] W. Anderson, *A Grid Generation and Flow Solution Method for the Euler Equations on Unstructured Grids*, Journal of Computational Physics, Volume 110, Pages 23–38, 1994.

[12] D. Anderson, J. Tannehill, and R. Pletcher, "Computational Fluid Mechanics and Heat Transfer", Hemisphere Publishing Corporation, 1984.

[13] W. Anderson, J. Thomas, and B. van Leer, *Comparison of Finite-Volume Flux Vector Splittings for the Euler Equations*, AIAA Journal, AIAA, Number 24, Pages 1453–1460, 1986.

[14] A. Arcilla, J. Hauser, P. Eiseman, and J. Thompson, (editors) *Numerical Grid Generation in Computational Fluid Dynamics and Related Fields*, North-Holland, Amsterdam, 1991.

[15] P. Arminjon, A. Dervieux, L. Fezoui, H. Steve, and B. Stoufflet, *Non-Oscillatory Schemes for Multidimensional Euler Calculations with Unstructured Grids*, in Notes on Numerical Fluid Mechanics, Volume 24, "Nonlinear Hyperbolic Equations - Theory, Numerical Methods, and Applications", Edited by J. Ballman, and R. Jeltsch, Vieweg, Braunschweig, 1989.

[16] G. Ashford, Private Communications, 1994, 1995, and 1996.

[17] G. Ashford, and K. Powell, *Adaptive Unstructured Triangular Mesh Generation and Flow Solvers for the Navier-Stokes Equations at High Reynolds Number*, AIAA Paper 95-1724-CP, Proceedings, 12th AIAA Computational Fluid Dynamics Conference, AIAA, July, 1995, Pages 869–880.

[18] N. Ashgriz, J. Poo, 1991, *FLAIR: Flux Line-Segment Model for Advection and Interface Reconstruction*, Journal of Computational Physics, Volume 93, Pages 449–468, 1991.

[19] H. Atkins, and J. Casper, *Nonreflective Boundary Conditions for High-Order Methods*, AIAA Journal, AIAA, Pages 512–518, Volume 32, Number 3, March 1994.

[20] I. Babuška, and A. Aziz, *On the Angle Condition in the Finite-Element Method*, SIAM Journal on Numerical Analysis, Volume 13, Number 2, 1976.

[21] P. Baehmann, S. Wittchen, M. Shephard, K. Grice, and M. Yerry, *Robust Geometrically Based Automatic Two-Dimensional Mesh Generation*, International Journal for Numerical Methods in Engineering, Volume 24, p. 1043, 1987.

[22] T. Baker, *Element Quality in Tetrahedral Meshes*, 7th International Conference on Finite Element Methods for Flow Problems, Hunstsville, Alabama, April 3-7, 1989.

[23] T. Baker, *Three-Dimensional Mesh Generation by Triangulation of Arbitrary Point Sets*, AIAA Paper 87-1124-CP, 8th AIAA Computational Fluid Dynamics Conference, AIAA, Hawaii, June, 1987.

[24] T. Baker, A. Jameson, and R. Vermeland, *Three-Dimensional Euler Solutions With Grid Embedding*, AIAA Paper 85-0121-CP, AIAA, 1985.

[25] T. Barth, *On Unstructured Grids and Solver*, Computational Fluid Dynamics, Von Karman Institute for Fluid Dynamics, Lecture Series 1990-04, 1990.

[26] T. Barth, *Aspects of Unstructured Grids and Finite-Volume Solvers for the Euler and Navier-Stokes Equations*, in Special Course on Unstructured Grid Methods for Advection Dominated Flows, Neuilly Sur Siene, France, May, 1992, AGARD Report 787.

[27] T. Barth, *Recent Developments in High-Order k-Exact Reconstruction on Unstructured Meshes*, AIAA Paper 93-0688, AIAA, 1993.

[28] T. Barth, and P. Frederickson, *Higher Order Solution of the Euler Equations on Unstructured Grids Using Quadratic Reconstruction*, AIAA Paper 90-0013, AIAA, January 1990.

[29] T. Barth, and D. Jesperson, *The Design and Application of Upwind Schemes on Unstructured Meshes*, AIAA Paper 89-0366, in Proceedings of the AIAA 27th AIAA Aerospace Sciences Meeting, AIAA, Reno, 1989.

[30] G. Batchelor, *An Introduction to Fluid Dynamics*, Cambridge University Press, 1967.

[31] J. Batina, *Unsteady Euler Algorithm With Unstructured Dynamic Mesh for Complex Aircraft Aeroelastic Analysis*, AIAA Paper 89-1189, AIAA, 1989.

[32] J. Batina, *A Gridless Euler/Navier-Stokes Solution Algorithm for Complex-Aircraft Applications*, AIAA Paper 93-0333, 31st Aerospace Sciences Meeting and Exhibit, AIAA, 11-14 January 1993, Reno, Nevada.

[33] J. Batina, *Unsteady Euler Algorithm with Unstructured Dynamic Mesh for Complex-Aircraft Aerodynamic Analysis*, AIAA Journal, AIAA, Volume 29, Number 3, Pages 327–333, 1991.

[34] J. Batina, *Implicit Flux-Split Euler Schemes for Unsteady Aerodynamic Analysis Involving Unstructured Dynamics Meshes*, AIAA Journal, AIAA, Volume 29, Number 11, Pages 1836–1843, 1991.

[35] J. Baum, H. Lou, and R. Löhner, *A New ALE Adaptive Unstructured Methodology for the Simulation of Moving Bodies*, AIAA Paper 94-0414, AIAA, 1994.

[36] S. Bayyuk, *Progress Report on Implementation and Testing of the Large Time-Step Method*, Internal Memorandum, Aerospace Engineering Department, The University of Michigan, 1989.

[37] R. Beam, and R. Warming, *Implicit Numerical Methods for the Compressible Navier-Stokes and Euler Equations*, Von Karman Institute Lecture Series 1982-04, Von Karman Institute, 1982.

[38] J. Bell, M. Berger, J. Saltzman, and M. Welcome, *Three-Dimensional Adaptive Mesh Refinement for Hyperbolic Conservation Laws*, SIAM Journal of Scientific and Statistical Computing, Volume 15, Number 1, Pages 127–138, January, 1994.

[39] J. Bell, P. Colella, J. Trangenstein, and M. Welcome, *Adaptive Mesh Refinement on Moving Quadrilateral Grids*, AIAA Paper 89-1979-CP, Proceedings, AIAA 9th Computational Fluid Dynamics Conference, AIAA, Buffalo, New York, 1989.

[40] J. Bell, D. Marcus, *A Second-Order Projection Method for Variable-Density Flows*, Journal of Computational Physics, Volume 101, Pages 334–348, 1992.

[41] G. Ben-Dor, *Shock Wave Reflection Phenomena*, Springer-Verlag, 1991.

[42] G. Ben-Dor, J. Dewey, and K. Takayama, *The Reflection of a Plane Shock Wave Over a Double Wedge*, Journal of Fluid Mechanics, Volume 176, Pages 483–520, 1987.

[43] J. Benek, P. Buning, and J. Steger, *A 3-D Chimera Grid-Embedding Technique*, AIAA Paper 85-1523-CP, AIAA, July, 1985.

[44] J. Benko, *3-D Cartesian Octree Method for Inviscid Flows*, Ph.D. Dissertation (in Aerospace Engineering), The University of Michigan, in preparation (expected in 2002).

[45] W. Benz, *Smooth Particle Hydrodynamics: A Review*, Numerical Modeling of Stellar Pulsation: Problems and Prospects, Proceedings of NATO Workshop, Les Arcs, France, March 20-24, 1989.

[46] M. Berger, *Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations*, Ph.D. Thesis, Computer Science Department, Stanford University, 1982.

[47] M. Berger, *On Conservation at Grid Interfaces*, SIAM Journal of Numerical Analysis, Volume 42, Number 5, Pages 967–984, October, 1987.

[48] M. Berger, and P. Colella, *Local Adaptive Mesh Refinement for Shock Hydrodynamics*, Journal of Computational Physics, Volume 82, Pages 67–84, 1989.

[49] M. Berger, and A. Jameson, *Automatic Adaptive Grid Refinement for the Euler Equations*, AIAA Journal, AIAA, Volume 32, Number 4, Pages 561–568, April, 1985.

[50] M. Berger, and R. LeVeque, *An Adaptive Cartesian Mesh Algorithm for the Euler Equations in Arbitrary Geometries*, AIAA Paper 89-1930, AIAA 9th Computational Fluid Dynamics Conference, AIAA, 1989.

[51] M. Berger, and R. LeVeque, *A Rotated Difference Scheme for Cartesian Grids in Complex Geometries*, AIAA Paper 91-1602, AIAA, 1991.

[52] M. Berger, and R. LeVeque, *Stable Boundary Conditions for Cartesian Grid Calculations*, ICASE Report Number 90-37, 1990.

[53] M. Berger, and J. Melton, *An Accuracy Test of a Cartesian Grid Method for Steady Flow in Complex Geometries*, Proceedings of the Fifth International Conference on Hyperbolic Problems, 1994.

[54] M. Berger, and J. Oliger, *Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations*, Journal of Computational Physics, Volume 53, Pages 484–512, 1984.

[55] J. Boris, and D. Book, *Flux-Corrected Transport, I: SHASTA, A Fluid Transport Algorithm that Works*, Journal of Computational Physics, Volume 11, Pages 38–69, 1973.

[56] A. Bourlioux, *A Coupled Level-Set Volume-of-Fluid Algorithm for Tracking Material Interfaces*, Proceedings of the Sixth International Symposium on Computational Fluid Dynamics, Pages 15–22, Lake Tahoe, Nevada, 1995.

[57] A. Bowyer, *Computing Dirichlet Tessellations*, The Computer Journal, Volume 24, Number 2, Pages 162–166 (and also Pages 167–172), 1981.

[58] J. Brackbill, and J. Saltzman, *Adaptive Zoning for Singular Problems in Two Dimensions*, Journal of Computational Physics, Volume 46, Number 3, Pages 342–368, 1982.

[59] C. Brebbia, J. Telles, and L. Wrobel, *Boundary Element Techniques*, Springer-Verlag, New York, 1984.

[60] G. Browning, H.-O. Kreiss, and J. Oliger, *Mesh Refinement*, Mathematics of Computation, Volume 27, Number 121, Pages 29–39, 1973.

[61] A. Bryson, and R. Gross, *Diffraction of Strong Shocks by Cones, Cylinders, and Spheres*, Journal of Fluid Mechanics, Volume 10, Pages 1–16, 1961.

[62] P. Buning, and J. Steger, *Solution of the Two-Dimensional Euler Equations with Generalized Coordinate Transformations Using Flux Vector Splitting*, AIAA Paper Number 82-0971, AIAA, July, 1982.

[63] J. Castillo, *A Direct Variational Grid Generation Method: Orthogonality Control*, in [322].

[64] D. Catherall, *The Adaption of Structured Grids to Numerical Solutions for Transonic Flows*, International Journal of Numerical Methods in Engineering, Volume 32, Number 4, Pages 921–939, 1991.

[65] J. Cavendish, D. Field, and W. Frey, *An Approach to Automatic Three-Dimensional Finite Element Mesh Generation*, International Journal for Numerical Methods in Engineering, Volume 21, Pages 329–347, 1985.

[66] S. Chakravarthy, *Euler Equations - Implicit Schemes and Boundary Conditions*, AIAA Journal, AIAA, Volume 21, Number 5, Pages 699–706, 1983.

[67] S. Chakravarthy, and S. Osher, *High-Resolution Applications of the Osher Upwind Scheme for the Euler Equations*, AIAA Paper 86-1943, AIAA 6th CFD Conference, AIAA, 1986.

[68] S. Chakravarthy, and K-Y. Szema, *Euler Solver for Three-Dimensional Supersonic Flows with Subsonic Pockets*, Journal of Aircraft, Volume 24, Number 2, February 1987.

[69] K. Chan, K. Pericleous, and M. Cross, *Numerical Simulation of Flows Encountered During Mold-Filling*, Journal of Applied Mathematical Modeling, Volume 15, Pages 624–631, 1991.

[70] E. Charlton, *An Octree Solution to Conservation Laws over Arbitrary Regions (OSCAR) with Applications to Aircraft Aerodynamics*, Ph.D. Dissertation (in Aerospace Engineering), The University of Michigan, expected in 1997.

[71] T. Cherry, *Flow of a Compressible Fluid About a Cylinder*, Proceedings of the Royal Society of London, **A**, Volume 192, Pages 45–79, 1947.

[72] T. Cherry, *Numerical Solutions for Transonic Flow*, Proceedings of the Royal Society of London, **A**, Volume 196, Pages 32–36, 1949.

[73] Y.-L. Chiang, *Simulation of Unsteady Inviscid Flow on an Adaptively Refined Cartesian Grid*, Ph.D. Dissertation (in Aerospace Engineering), The University of Michigan, 1992.

[74] Y.-L. Chiang, B. van Leer, and K. Powell, *Simulation of Unsteady Inviscid Flow on an Adaptively Refined Cartesian Grid*, AIAA Paper 92-0443, 1992, in Proceedings of the 30th AIAA Aerospace Sciences Meeting and Exhibit, AIAA, 6-9 January 1992, Reno, Nevada.

[75] G. Chiocchia, *Exact Solutions to Transonic and Supersonic Flows*, in [5].

[76] A. Chorin, *Random-Choice Solution of Hyperbolic Systems*, Journal of Computational Physics, Volume 22, Pages 517–533, 1976.

[77] A. Chorin, *Random-Choice Methods with Application to Reacting Gas Flows*, Journal of Computational Physics, Volume 25, Pages 253–271, 1977.

[78] T. Chung, *Finite-Element Analysis in Fluid Dynamics*, McGraw-Hill, New York, 1978.

[79] D. Clarke, M. Salas, and H. Hassan, *Euler Calculations for Multielement Airfoils Using Cartesian Grids*, AIAA Journal, AIAA, Volume 24, Number 3, Pages 353–358, 1986.

[80] P. Colella, *Glimm's Method for Gas Dynamics*, SIAM Journal of Scientific and Statistical Computing, Volume 3, Pages 76–110, 1982.

[81] P. Colella, *Multidimensional Upwind Methods for Hyperbolic Conservation Laws*, Journal of Computational Physics, Volume 87, Pages 171–200, 1990.

[82] P. Colella, and H. Glaz, *Efficient Solution Algorithms for the Riemann Problem for Real Gases*, Journal of Computational Physics, Volume 59, Pages 264–289, 1985.

[83] P. Colella, and L. Henderson, *The von Neumann Paradox for the Diffraction of Weak Shock Waves*, Journal of Fluid Mechanics, Volume 213, Pages 71–94, 1990.

[84] P. Colella, and P. Woodward, *The Piecewise Parabolic Method for Gas-Dynamical Simulations*, Journal of Computational Physics, Volume 54, Pages 174–201, 1984.

[85] W. Coirier, *Adaptively Refined Euler and Navier-Stokes Solution on an Unstructured Quadtree-Based Grid*, Ph.D. Dissertation (in Aerospace Engineering), The University of Michigan, 1994.

[86] W. Coirier, K. Powell, and M. Berger, *An Accuracy Assessment of Cartesian-Mesh Approaches for the Euler Equations*, Journal of Computational Physics, Volume 117, Pages 121–131, 1995.

[87] W. Coirier, and B. van Leer, *Numerical Flux Formulas for the Euler and Navier-Stokes Equations: II. Progress in Flux Vector Splitting*, AIAA Paper 91-1566-CP, AIAA 10th CFD Conference, AIAA, 1991.

[88] G. Cooper, *The PARC Code: Theory and Usage*, Arnold Engineering Development Center, Tullahoma, Tennessee, TN, TR-87-24, October, 1987.

[89] F. Coquel, and M. Liou, *Field by Field Hybrid Upwind Splitting Methods*, AIAA Paper Number 93-3302-CP, AIAA, 1993.

[90] R. Courant, E. Isaacson, and M. Reeves, *On the Solution of Nonlinear Hyperbolic Partial Differential Equations by Finite Differences*, Communications on Pure and Applied Mathematics, Volume 5, Pages 243–255, 1952.

[91] R. Courant, and K. Friedrichs, *Supersonic Flow and Shock Waves*, Applied Mathematical Sciences, Volume 21, Springer-Verlag, 1948.

[92] R. Courant, and D. Hilbert, *Methods of Mathematical Physics*, Volume II, Wiley-Interscience, New York, 1962.

[93] P. Crumpton, and M. Giles, *Implicit Time-Accurate Solutions on Unstructured Dynamic Grid*, AIAA Paper 95-1671-CP, AIAA, 1995.

[94] A. Dadone, and B. Grossman, *Surface Boundary Conditions for the Numerical Solution of the Euler Equations*, AIAA Paper 93-3334, 11th AIAA CFD Conference, AIAA, Orlando, Florida, 6-9 July 1993.

[95] J. Dannenhoffer, *Computer-Aided Block Structuring Through the Use of Optimization and Expert System Techniques*, AIAA Paper 91-1585-CP, AIAA, 1991.

[96] J. Dannenhoffer, *A New Method for Creating Grid Abstractions for Complex Configurations*, AIAA Paper 93-0428, AIAA, January, 1993.

[97] J. Dannenhoffer, *A Comparison of Adaptive-Grid Redistribution and Embedding for Steady Transonic Flows*, International Journal for Numerical Methods in Engineering, Volume 32, Pages 653–663, 1991.

[98] J. Dannenhoffer, *Grid Adaptation for Complex Two-Dimensional Transonic Flows*, Doctoral Thesis, Massachusetts Institute of Technology, August 1987.

[99] J. Dannenhoffer, and J. Baron, *A Hybrid Expert System for Complex CFD Problems*, AIAA Paper 87-1111-CP, AIAA, June 1987.

[100] R. Davis, and J. Dannenhoffer, *Three-Dimensional Adaptive Grid Embedding Euler Technique*, AIAA Paper 93-0330, AIAA, January, 1993.

[101] D. De Zeeuw, *A Quadtree-Based Adaptively-Refined Cartesian-Grid Algorithm for Solution of the Euler Equations*, Ph.D. Dissertation (in Aerospace Engineering), The University of Michigan, 1993.

[102] D. De Zeeuw, and K. Powell, *An Adaptively-Refined Cartesian Mesh Solver for the Euler Equations*, Journal of Computational Physics, Volume 104, Number 1, Pages 56–68, 1993.

[103] D. De Zeeuw, and K. Powell, et. al. *Octree-Based Method for MHD*, Journal of Computational Physics, Volume 117, Pages 121–131, 1995.

[104] H. Deconinck, R. Struijs, *A Consistent Boundary Condition for Cell Centered Upwind Finite Volume Euler Solvers*, Numerical Methods for Fluid Dynamics III, Clarendon Press, Oxford, 1988.

[105] H. Deconinck, R. Struijs, and P. Roe, *Fluctuation Splitting for Multi-Dimensional Convection Problems: an Alternative to Finite Element and Finite Volume Methods*, Von Karman Institute Lecture Series 1990-03, Von Karman Institute, 1990.

[106] M. Delanaye, and J. Essers, *An Accurate Finite-Volume Scheme for Euler and Navier-Stokes Equations on Unstructured Adaptive Grids*, AIAA Paper Number 95-1710-CP, AIAA, Pages 736–745, 1995.

[107] M. Delanaye, and J. Essers, *Quadratic-Reconstruction Finite Volume Scheme for Compressible Flows on Unstructured Adaptive Grids*, AIAA Journal, AIAA, Volume 35, Number 4, pp 631–639, 1997.

[108] I. Demirdžić, and M. Perić, *Space Conservation Law in Finite Volume Calculations of Fluid Flow*, International Journal of Numerical Methods in Fluids, Volume 8, Pages 1037–1050, 1988.

[109] I. Demirdžić, and M. Perić, *Finite Volume Method for Prediction of Fluid Flow in Arbitrarily Shaped Domains with Moving Boundaries*, International Journal of Numerical Methods in Fluids, Volume 10, Pages 771–790, 1990.

[110] J. Donéa, *An Arbitrary Lagrangian-Eulerian Finite-Element Method for Transient Fluid-Structure Interactions*, Computational Methods in Applied Mechanical Engineering, Volume 33, Pages 689–723, 1982.

[111] J. Donéa, *Arbitrary Lagrangian-Eulerian Finite-Element Methods*, in *Computational Methods for Transient Analysis*, Chapter 10, Edited by T. Belytschko and T. Hughes, Elsevier, New York, 1983.

[112] F. Dougherty, and J.-H. Kuan, *Transonic Store Separation Using a Three-Dimensional Chimera Grid Scheme*, AIAA Paper 89-0637, AIAA, January, 1989.

[113] T. Edwards, *Noniterative Three-Dimensional Grid Generation Using Partial Differential Equations*, AIAA Paper 85-0485, AIAA, January, 1985.

[114] B. Einfeldt, *On Godunov-Type Methods for Gas Dynamics*, SIAM Journal of Numerical Analysis, Volume 25, Number 2, Pages 294–318, 1988.

[115] B. Einfeldt, C. Munz, P. Roe, and B. Sjögreen, *On Godunov-Type Methods Near Low Densities*, Journal of Computational Physics, Volume 92, Pages 273–295, 1991.

[116] P. Eiseman, *Alternating Direction Adaptive Grid Generation*, AIAA Paper 83-1937, AIAA, July 1983.

[117] P. Eiseman, *Solution Adaptivity Using a Triangular Mesh*, Lecture Notes in Physics, Volume 238, Pages 205–235, Springer-Verlag, 1985.

[118] P. Eiseman, *Grid Generation for Fluid Mechanics Computation*, Annual Review of Fluid Mechanics, Volume 17, Pages 487–522, 1985.

[119] B. Engquist, and A. Majda, *Absorbing Boundary Conditions for the Numerical Simulation of Waves*, Mathematics of Computation, Volume 31, Pages 629–651, 1977.

[120] B. Epstein, L. Luntz, and A. Nachshon, *Multigrid Euler Solver about Arbitrary Aircraft Configurations with Cartesian Grids and Local Refinement*, AIAA Paper 89-1960-CP, AIAA 9th Computational Fluid Dynamics Conference, AIAA, 1989.

[121] L.-E. Eriksson, *Generation of Boundary Conforming Grids Around Wing-Body Configurations Using Transfinite Interpolation*, AIAA Journal, AIAA, Volume 20, Number 10, Pages 1313–1320, 1982.

[122] J. Fang, and S. Kennon, *Unstructured Grid Generation for Nonconvex Domains*, AIAA Paper 89-1983-CP, AIAA, June 1989.

[123] C. Farhat, M. Lesoinne, and N. Maman, *Mixed Explicit / Implicit Time Integration of Coupled Aeroelastic Problems: Three-Field Formulation, Geometric Conservation and Distributed Solution*, International Journal of Numerical Methods in Fluids, Volume 21, Pages 807–835, 1995.

[124] J. Ferziger, and M. Perić, *Computational Methods for Fluid Dynamics*, Springer-Verlag, 1996.

[125] R. Finkel, and J. Benltey, *Quad Trees: A Data Structure for Retrieval on Composite Keys*, Acta Informatica, Volume 4, Pages 1–9, 1974.

[126] S. Fortune, *Voronoi Diagrams and Delaunay Triangulations*, in Computing in Eucledian Geometry, F. Hwang, and D. Du (editors), World Scientific Publishing Company, 1992.

[127] R. Gaffney, H. Hassan, and M. Salas, *Euler Calculations for Wings Using Cartesian Grids*, AIAA Paper 87-0356, AIAA, 1987.

[128] A. George, *Computer Implementation of the Finite-Element Method*, Stanford University, Department of Computer Science, Report Stan-CS-71-208, 1971.

[129] M. Giles, *Nonreflecting Boundary Conditions for Euler Equation Calculations*, AIAA Paper 89-1942, AIAA, 1989.

[130] M. Giles, *Generalized Conservation Cells for Finite Volume Calculations*, AIAA Paper Number 87-1118, AIAA, 1987.

[131] D. Givoli, *Review Article: Non-reflecting Boundary Conditions*, Journal of Computational Physics, Volume 94, Number 1, Pages 1–29, 1991.

[132] J. Glimm, *Solution in the Large for Nonlinear Hyperbolic Systems of Equations*, Communications on Pure and Applied Mathematics, Volume 18, Pages 697–715, 1965.

[133] J. Glimm, and O. McBryan, *A Computational Model for Interfaces*, Advances in Applied Mathematics, Volume 6, Pages 422–435, 1985.

[134] E. Godlewski, and P.-A. Raviart, *Numerical Approximation of Hyperbolic Systems of Conservation Laws*, Applied Mathematical Sciences, Volume 118, Springer-Verlag, 1996.

[135] S. Godunov, *A Difference Scheme for Numerical Computation of Discontinuous Solutions of Hydrodynamic Equations*, Translated by the US Joint Publication Research Service, JPRS Publication Number 7226, from Math. Sbornik, Number 47, Pages 271–306, 1959.

[136] C. Gooch, *Solution of the Navier-Stokes Equations on Locally-Refined Cartesian Meshes Using State-Vector Splitting*, Ph.D. Thesis, Department of Aeronautics and Astronautics, Stanford Unversity, 1993.

[137] W. Gordon, *Blending Function Methods of Bivariate and Multivariate Interpolation*, SIAM Journal of Numerical Analysis, Volume 8, Pages 158–177, 1971.

[138] A. Goswami, and I. Parpia, *Grid Restructuring for Moving Boundaries*, AIAA 10th Computational Fluid Dynamics Conference, AIAA, 1991.

[139] J. Gottlieb, and C. Groth, *Assessment of Riemann Solvers for Unsteady One-Dimensional Inviscid Flows of Perfect Gases*, Journal of Computational Physics, Volume 78, Number 2, Pages 437–458, 1988.

[140] M. Griffin, and J. Anderson, *On the Application of Boundary Conditions to Time Dependent Computations of Quasi One-Dimensional Fluid Flows*, Computers and Fluids, Volume 5, Pages 127–137, 1977.

[141] B. Grossman, *Numerical Procedure for the Computation of Irrotational Conical Flows*, AIAA Journal, AIAA, Volume 17, Pages 828–837, August, 1979,

[142] K. Guderley, *The Theory of Transonic Flow*, Pergamon Press, 1962.

[143] B. Gustafsson, *The Convergence Rate for Difference Approximations to Mixed Initial Value Problems*, Mathematics of Computation, Volume 29, Pages 396–406, 1975.

[144] B. Gustafsson, *The Choice of Numerical Boundary Conditions for Hyperbolic Systems*, Journal of Computational Physics, Volume 48, Pages 270–283, 1982.

[145] B. Gustafsson, H.-O. Kreiss, and J. Oliger, *Time-Dependent Problems and Difference Methods*, Wiley-Interscience, New York, 1995.

[146] B. Gustafsson, H.-O. Kreiss, and A. Sundström, *Stability Theory of Difference Approximations for Mixed Initial Boundary Value Problems, II*, Mathematics of Computations, Volume 26, Pages 649–686, 1972.

[147] D. Halt, and R. Agarwal, *A Novel Algorithm for the Solution of the Compressible Euler Equations in Wave/Particle (WPS) Form*, AIAA Paper Number 93-3392-CP, AIAA, 1993.

[148] R. Hamming, "Numerical Methods for Scientists and Engineers", McGraw-Hill, 1962.

[149] D. Hänel, R. Schwane, and G. Seider, *On the Accuracy of Upwind Schemes for the Solution of the Navier-Stokes Equations*, AIAA Paper 87-1105, AIAA, 1987.

[150] F. Harlow, *PIC and its Progeny*, Computer Physics Communications, Volume 48, Pages 1–11, 1988.

[151] F. Harlow, and A. Amsden, *A Simplified MAC Technique for Incompressible Fluid Flow Calculations*, Journal of Computational Physics, Volume 6, Pages 322–325, 1970.

[152] F. Harlow, and J. Welch, *Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with a Free Surface*, Physics of Fluids, Volume 8, Pages 2182–2189, 1965.

[153] F. Harlow, and J. Welch, *Numerical Study of Large-Amplitude Free-Surface Motions*, Physics of Fluids, Volume 9, Pages 842–851, 1966.

[154] A. Harten, *On a Class of High-Resolution, Total-Variation-Stable Finite-Difference Schemes*, SIAM Journal of Numerical Analysis, Volume 21, 1984.

[155] A. Harten, *On the Symmetric Form of Systems of Conservation Laws with Entropy*, ICASE Report Number 81-34, ICASE, 1981.

[156] A. Harten, *High-Resolution Schemes for Hyperbolic Conservation Laws*, Journal of Computational Physics, Volume 49, Number 3, Pages 357–393, 1983.

[157] A. Harten, B. Engquist, S. Osher, and S. Chakravarthy, *Uniformly High-Order Accurate Essentially Non-Oscillatory Schemes, III*, ICASE Report 86-22, ICASE, 1986, and Journal of Computational Physics, Volume 71, Number 2, Pages 231–303, 1983.

[158] A. Harten, and S. Chakravarthy, *Multi-Dimensional ENO Schemes for General Geometries*, ICASE Report 91-76, ICASE, 1991.

[159] A. Harten, P. Lax, and B. van Leer, *On Upstream Differencing and Godunov-Type Schemes for Hyperbolic Conservation Laws*, SIAM Review, Volume 25, Pages 35–61, 1983.

[160] A. Harten, and S. Osher, *Uniformly High-Order Accurate Non-Oscillatory Schemes, I*, MRC Technical Summary Report 2823, 1985.

[161] P. Hartwich, *Fresh Look at Floating Shock Fitting*, AIAA Journal, AIAA, Volume 29, Pages 1084–1091, 1991.

[162] P. Hartwich, and N. Frink, *Estimation of Propulsion-Induced Effects on Transonic Flows Over a Hypersonic Configuration*, AIAA Paper Number 92-0523, AIAA, 1992.

[163] A. Harvey, S. Acharya, and S. Lawrence, *Space Marching Calculations About Hypersonic Configurations Using a Solution-Adaptive Mesh Algorithm*, , AIAA Journal, AIAA, Volume 31, Number 10, Oct 1993.

[164] O. Hassan, K. Morgan, and J. Peraire, *An Implicit Finite-Element Method for High-Speed Flows*, International Journal for Numerical Methods in Engineering, Volume 32, Pages 183–205, 1991.

[165] J. Hauser, and C. Taylor (editors), *Numerical Grid Generation in Computational Fluid Dynamics*, Pineridge Press, Swansea, U.K., 1986.

[166] D. Hawken, J. Gottlieb, and J. Hansen, *Review of Some Adaptive Node-Movement Techniques in Finite-Element and Finite-Difference Solutions of Partial Differential Equaitons*, Journal of Computational Physics, Volume 95, Pages 254–302, 1991.

[167] G. Hedstrom, *Nonreflecting Boundary Conditions for Nonlinear Hyperbolic Systems*, Journal of Computational Physics, Volume 30, Pages 222–237, 1979.

[168] R. Hindman, *Generalized Coordinate Forms of the Governing Fluid Equations and Associated Geometrically Induced Errors*, AIAA Journal, AIAA, Volume 20, Number 10, Pages 1359–1367, 1982.

[169] C. Hirsch, *Numerical Computation of Internal and External Flows, Volume 1: Fundamentals of Numerical Discretization*, John Wiley and Sons, 1988.

[170] C. Hirsch, *Numerical Computation of Internal and External Flows, Volume 2: Computational Methods for Inviscid and Viscous Flows*, John Wiley and Sons, 1990.

[171] C. Hirsch, and A. Verhoff, *Far-Field Numerical Boundary Conditions for Internal and Cascade Problems*, AIAA Paper Number 89-1943, AIAA, June 1989.

[172] C. Hirt, A. Amsden, and J. Cook, *An Arbitrary Lagrangian-Eulerian Computing Method for All Flow Speeds*, Journal of Computational Physics, Volume 14, p. 227, 1974.

[173] C. Hirt, and B. Nichols, *Volume of Fluid (VOF) Method for the Dynamics of Free Boundaries*, Journal of Computational Physics, Volume 39, Pages 201–225, 1981.

[174] D. Holmes, and D. Snyder, *The Generation of Unstructured Triangular Meshes Using Delaunay Triangulation*, in [322].

[175] T. Hou, *Numerical Solutions to Free Boundary Problems*, Acta Numerica, Pages 335–415, 1995.

[176] J. Hyman, *Numerical Methods for Tracking Interfaces*, Physica 12D, Pages 396–407, Elsevier Science Publishers B.V., 1984.

[177] C. Ingram, K. Laflin, and D. McRae, *A Structured Multi-Block Solution-Adaptive Mesh Algorithm with Mesh Quality Assessment*, in ICASE/LaRC Workshop on Adaptive Grid Methods, Hampton, Virginia, November 7-9, 1994 - NASA Conference Publication 3316 (1994).

[178] K. Itoh, K. Takayama, and G. Ben-Dor, *Numerical simulation of the reflection of a planar shock wave over a double wedge*, International Journal for Numerical Methods in Fluids, Volume 13, Pages 1153–1170, 1991.

[179] D. Ives, *Conformal Grid Generation*, in [360].

[180] D. Ives, and J. Liutermoza, *Analysis of Transonic Cascade Flow Using Conformal Mapping and Relaxation Techniques*, AIAA Journal, AIAA, Volume 15, Pages 647–652, May, 1977.

[181] A. Jameson, *Computational Algorithms for Aerodynamic Analysis and Design*, MAE Report Number 1966, Princeton University, 1992.

[182] A. Jameson, *Airfoils Admitting Non-Unique Solutions of the Euler Equations*, AIAA Paper Number 91-1625, AIAA, 1991.

[183] A. Jameson, *Solution of the Euler Equations for Two-Dimensional Transonic Flow by a Multi-Grid Method*, Applied Mathematics and Computation, Volume 13, Pages 327–355, 1983.

[184] A. Jameson, T. Baker, and N. Weatherill, *Calculation of Inviscid Transonic Flow Over a Complete Aircraft*, AIAA Paper 86-0103, AIAA, January, 1986.

[185] A. Jameson, W. Schmidt, and E. Turkel, *Numerical Simulation of the Euler Equations by Finite-Volume Methods Using Runge-Kutta Time-Stepping Schemes*, AIAA Paper 81-1259, AIAA 5th CFD Conference, AIAA, 1981.

[186] C. Johnson, *Finite-Element Methods for Flow Problems*, in AGARD Special Course on Unstructured Grid Methods for Advection Dominated Flows, AGARD-R-787, May 1992.

[187] Y. Kallinderis, *A New Finite-Volume Navier-Stokes Scheme on 3-D Semi-Unstructured Prismatic Elements*, AIAA Paper 92-2697, AIAA, 1992.

[188] Y. Kallinderis, *Methods for Prismatic/Tetrahedral Grid Generation and Adaptation*, in ICASE/LaRC Workshop on Adaptive Grid Methods, 7-9 November, 1994, NASA Conference Publication 3316.

[189] Y. Kallinderis, and J. Baron, *Adaptation Methods for a New Navier-Stokes Algorithm*, AIAA Journal, AIAA, Volume 72, Number 1, Pages 37–43, January 1989.

[190] Y. Kallinderis, A. Khawaja, and H. McMorris, *Adaptive Hybrid Prismatic-Tetrahedral Grids for Viscous Flows*, NASA-CP-3291, 1995.

[191] Y. Kallinderis, and S. Ward, *Prismatic Grid Generation for Three-Dimensional Complex Geometries*, AIAA Journal, AIAA, Volume 31, Number 10, Pages 1850–1856, October, 1993.

[192] S. Karman, *SPLITFLOW: A 3-D Unstructured Cartesian/Prismatic Grid CFD Code for Complete Geometries*, AIAA Paper Number 95-0343, AIAA, 1995.

[193] S. Kennon, *A Vectorized Delaunay Trinagulation Scheme for Non-Convex Domains with Automatic Nodal Point Generation*, AIAA Paper 88-0314, AIAA, 1988.

[194] S. Kennon, J. Meyering, C. Berry, and J. Oden, *Geometry Based Delaunay Tetrahedralization and Mesh Movement Strategies for Multi-Body CFD*, AIAA Paper Number 92-4575, AIAA, 1992.

[195] J. Kim, and J. Thompson, *Three-Dimensional Adaptive Grid Generation on a Composite Block Grid*, AIAA Paper 88-0311, AIAA, 1988.

[196] D. Kinsey, *Toward the Direct Design of Waverider Airframes*, Ph.D. Dissertation (in Aerospace Engineering), The University of Michigan, 1998.

[197] G. Klopfer, *Solution Adaptive Meshes with a Hyperbolic Grid Generator*, in [322].

[198] P. Knupp, and S. Steinberg *Fundamentals of Grid Generation*, CRC Press, 1994.

[199] D. Knuth, *The Art of Computer Programming, Volume 1 / Fundamental Algorithms*, Addison-Wesley Publishing Company, 1968 (Second Edition, 1973).

[200] D. Knuth, *The Art of Computer Programming, Volume 2 / Seminumerical Algorithms*, Addison-Wesley Publishing Company, 1969.

[201] D. Knuth, *The Art of Computer Programming, Volume 3 / Sorting and Searching*, Addison-Wesley Publishing Company, 1973.

[202] B. Koren, and B. van Leer, *Analysis of Preconditioning and Multigrid for Euler Flows with Low Subsonic Regions*, Advances in Computational Mathematics, Volume 4, Number 1–2, 1995.

[203] D. Kothe, and A Mjolsness, *RIPPLE: A New Model for Incompressible Flows with Free Surfaces*, AIAA Journal, AIAA, Volume 30, Pages 2694–2700, 1996.

[204] D. Kothe, W. Rider, S. Mosso, J. Brock, and J. Hochstein, *Volume Tracking of Interfaces Having Surface Tension in Two and Three Dimensions*, AIAA Paper Number 96-0859, in 34th Aerospace Sciences Meeting and Exhibit, AIAA, January 15-18, 1996, Reno, Nevada.

[205] M. Kranyš, *Causal Theories of Evolution and Wave Propagation in Mathematical Physics*, Applied Mechanics Review, Volume 42, Number 11, Pages 305–322.

[206] H.-O. Kreiss, *Initial Boundary Value Problems for Hyperbolic Systems*, Communications on Pure and Applied Mathematics, Volume 23, Pages 273–298, 1970.

[207] H.-O. Kreiss, and J. Lorenz, *Initial Boundary Value Problems*, Academic Press, 1989.

[208] R. Kruse, *Data Structures and Program Design*, Prentice-Hall Inc., 1984, (and Second Edition, 1987).

[209] P. Kutler, *Computation of Three-Dimensional, Inviscid Supersonic Flows*, Lecture Notes in Physics, Progress in Numerical Fluid Dynamics, Springer-Verlag, 1975.

[210] B. Lafaurie, C. Nardone, R. Scardovelli, S. Zaleski, and G. Zanetti, *Modelling Merging and Fragmentation in Multiphase Flows with SURFER*, Journal of Computational Physics, Volume 113, Pages 134–147, 1994.

[211] L. Landau, and E. Lifshitz, *Course of Theoretical Physics, Volume 6, Fluid Mechanics*, Pergamon Press, 1959.

[212] A. Lapidus, *A Detached Shock Calculation by Second-Order Finite Differences*, Journal of Computational Physics, Volume 2, Pages 154–177, 1967.

[213] P. Lax, *Hyperbolic Systems of Conservation Laws and the Mathematical Theory of Shock Waves*, SIAM CBMS Regional Conference Series on Applied Mathematics, Volume 11, SIAM, Philadelphia, 1972.

[214] P. Lax, *Weak Solutions of Nonlinear Hyperbolic Equations and Their Numerical Computation*, Communications on Pure and Applied Mathematics, Volume 7, Pages 159–193, 1954.

[215] P. Lax, and B. Wendroff, *Systems of Conservation Laws*, Communications on Pure and Applied Mathematics, Number 13, Pages 217–237, 1960.

[216] K. Lee, and P. Rubbert, *Transonic Flow Computations Using Grid Systems with Block Structure*, in 7th International Conference on Numerical Methods in Fluid Dynamics, Springer-Verlag, New York, 1980, Pages 266–271.

[217] A. Lefebvre, *Atomization and Sprays*, Hemisphere Publishing Corporation, 1989.

[218] M. Lesoinne, and C. Farhat, *Geometric Conservation Laws for Aeroelastic Computations using Unstructured Dynamics Meshes*, AIAA Paper Number 95-1709-CP, 12th AIAA CFD Conference, San Diego, California, AIAA, June 19-22, 1995.

[219] R. LeVeque, *Cartesian Grid Methods for Flow in Irregular Regions*, Numerical Methods for Fluid Dynamics III, Edited by K. Morton and M. Baines, Pages 375–382, Clarendon Press, Oxford, 1988.

[220] R. LeVeque, *A Large Time Step Generalization of Godunov's Method for Systems of Conservation Laws*, SIAM Journal of Numerical Analysis, Volume 22, Pages 1051–1073, 1985.

[221] R. LeVeque, *High Resolution Finite Volume Methods on Arbitrary Grids Via Wave Propagation*, Journal of Computational Physics, Volume 78, Pages 36–63, 1988.

[222] R. LeVeque, "Numerical Methods for Conservation Laws", Birkhäuser Verlag, 1990.

[223] S.-Y. Lin, T.-M. Wu, and Y.-S. Chin, *Upwind Finite-Volume Method with a Triangular Mesh for Conservation Laws*, Journal of Computational Physics, Volume 107, Pages 324–337, 1993.

[224] M. Liou, and J. Steffen, *A New Flux Vector Splitting Scheme*, NASA Technical Memorandum 104452, 1991.

[225] R. Löhner, *Adaptive Remeshing for Transient Problems*, Computational Methods in Applied Mechanical Engineering, Volume 75, Pages 195–214, 1989.

[226] R. Löhner, *Matching Semi-Structured and Unstructured Grids for Navier-Stokes Calculations*, AIAA Paper 93-3348, AIAA, 1993.

[227] R. Löhner, *An Adaptive Finite Element Solver for Transient Problems with Moving Bodies*, Computers and Structures, Volume 30, Pages 303–317, 1988.

[228] R. Löhner, *Finite Element Methods in CFD, Grid Generation, Adaptivity, and Parallelism*, AGARD Report 787, NATO, 1992.

[229] R. Löhner, K. Morgan, and O. Zienkiewicz, *Adaptive Grid Refinement for the Compressible Euler equations*, in *Accuracy Estimates and Adaptive Refinements in Finite Element Computations*, Edited by Babuška, Zienkiewicz, Gago, and Oliviera, John Wiley and Sons, New York, 1986, Pages 281–297.

[230] R. Löhner, K. Morgan, and O. Zienkiewicz, *The Solution of Nonlinear Systems of Hyperbolic Equations by the Finite-Element Method*, International Journal of Numerical Methods in Fluids, Volume 4 , Pages 1043–1063, 1984.

[231] R. Löhner, and P. Parikh, *Generation of Three-Dimensional Unstructured Grids by the Advancing Front Method*, AIAA Paper 88-0515, AIAA, January 1988.

[232] R. Löhner, P. Parikh, and C. Gumbert, *Intractive Generation of Unstructured Grids for Three-Dimensional Problems*, in [322].

[233] I. Lottati, and S. Eidelman, *A Second-Order Godunov Scheme on a Spatial Adapted Triangular Grid*, Applied Numerical Mathematics, Volume 14, Pages 353–365, 1994.

[234] R. Lowrie, *Compact Higher-Order Numerical Methods for Hyperbolic Conservation Laws*, Ph.D. Dissertation (in Aerospace Engineering), The University of Michigan, 1996.

[235] M. Marchant, and N. Weatherill, *Adaptivity Techniques for Compressible Inviscid Flows*, Computer Methods in Applied Mechanics and Engineering, Volume 106, Pages 83–106, 1993.

[236] C. Masson, H. Saabas, and R. Baliga, *Co-Located Equal-Order Control-Volume Finite-Element Method for Two-Dimensional Axisymmetric Incompressible Fluid Flow*, International Journal of Numerical Methods in Fluids, Volume 18, Pages 1–26, 1994.

[237] D. Mavriplis, *Adaptive Mesh Generation for Viscous Flows Using Delaunay Trinagulations*, Journal of Computational Physics, Volume 90, Pages 271–291, October, 1990.

[238] D. Mavriplis, *An Advancing Front Delaunay Triangulation Algorithm Designed for Robustness*, Journal of Computational Physics, Volume 117, Pages 90–101, 1995.

[239] K. Mazaheri, *Numerical Wave Propagation and Steady State Solutions*, Ph.D. Dissertation (in Aerospace Engineering), The University of Michigan, 1992.

[240] R. Meakin, *Moving Body Overset Grid Methods for Complete Aircraft Tiltrotor Simulations*, AIAA Paper 93-3350-CP, AIAA, 1993.

[241] C. Merkle, S. Venkateswaran, and P. Buelow, *The Relationship Between Pressure-Based and Density-Based Algorithms*, AIAA Paper Number 92-0425, AIAA, 1992.

[242] B. McCormick, Jr., *Aerodynamics of V/STOL Flight*, Academic Press, 1967.

[243] B. McCormick, Jr., *Aerodynamics, Aeronautics, and Flight Mechanics*, John Wiley and Sons, 1979.

[244] P. McDonald, *The Computation of Transonic Flow Through Two-Dimensional Gas Turbine Cascades*, ASME Paper 71-GT-89, ASME, 1971.

[245] J. Melton, *Automated Three-Dimensional Cartesian Grid Generation and Euler Flow Solutions for Arbitrary Geometries*, Ph.D. Thesis, University of California at Davis, 1996.

[246] J. Melton, M. Berger, M. Aftosmis, and M. Wong, *3D Applications of a Cartesian Grid Euler Method*, AIAA Paper 95-0853, AIAA, 1995.

[247] J. Melton, F. Enomoto, and M. Berger, *3-D Automatic Cartesian Grid Generation for Euler Flows*, AIAA Paper Number 93-3386-CP, AIAA, 1993 (in 13th AIAA CFD Conference, July 1993).

[248] J. Melton, S. Pandya, and J. Steger, *3D Euler Flow Solutions Using Unstructured Cartesian and Prismatic Grids*, AIAA Paper 93-0331, AIAA, 1993.

[249] C. Mitchell, and R. Walters, *K-Exact Reconstruction for the Navier-Stokes Equations on Arbitrary Grids*, AIAA Paper 93-0536, AIAA, 1993.

[250] Y. Mizuta, *Generalized Boundary Conditions on the Basis of a Deformable-Cell Method: Free Surface, Density Interfaces and Open Boundaries*, Computers and Fluids, Volume 19, Pages 377–385, 1991.

[251] J. Monaghan, *Smoothed Particle Hydrodynamics*, Annual Reviews of Astronomy and Astrophysics, Volume 30, Pages 543–574, 1992.

[252] G. Moretti, *Conformal Mappings for Computations of Steady Three-Dimensional Supersonic Flows*, Numerical/Laboratory Computer Methods in Fluid Mechanics, American Society of Mechanical Engineers, New York, 1976.

[253] G. Moretti, *A Physical Approach to the Numerical Treatment of Boundaries in Gas Dynamics*, Numerical Boundary Condition Procedures, NASA CP Number 2201, Pages 73–95, 1981.

[254] M. Mortenson, *Geometric Modeling*, John Wiley and Sons, Inc., 1985.

[255] B. Mughal, and M. Drela, *A Calculation Method for the Three-Dimensional Boundary-Layer Equations in Integral Form*, AIAA Paper 93-0786, AIAA, 1993.

[256] J.-D. Müller, Private Communications, 1994, and 1995.

[257] J.-D. Müller, *Quality Estimates and Stretched Meshes Based on Delaunay Triangulations*, AIAA Journal, AIAA, Volume 32, Number 12, Pages 2372–2379, 1994.

[258] J.-D. Müller, P. Roe, and H. Deconink, *A Frontal Approach for Internal Node Generation in Delaunay Triangulations*, International Journal for Numerical Methods in Fluids, Volume 17, Number 3, Pages 241–257, August, 1993.

[259] K. Nakahashi, and G. Deiwert, *A Practical Adaptive-Grid Method for Complex Fluid-Flow Problems*, Lecture Notes in Physics, Volume 218, Pages 422–426, 1985.

[260] S. Nakamura, *Marching Grid Equations Using Parabolic Partial Differential Equations*, in [360].

[261] ICASE / NASA Langley Research Center (LaRC), *Workshop on Adaptive Grid Methods*, 7-9 November 1994, Hampton, Virginia, Edited by J. South, Jr., J. Thomas, and J. Vanrosendale, Published in October 1995, as NASA Conference Publication 3316.

[262] R. Ni, *A Multigrid Scheme for Solving the Euler Equations*, AIAA Journal, AIAA, Volume 20, Number 11, Pages 1565–1571, 1982.

[263] B. Nkonga, and H. Guillard, *Godunov Type Method on Non-Structured Meshes for Three-Dimensional Moving Boundary Problems*, Computational Methods in Applied Mechanical Engineering, Volume 113, Pages 183–204, 1994.

[264] R. Noack, and D. Anderson, *Solution Adaptive Grid Generation Using Partial Differential Equations*, AIAA Paper 88-0315, AIAA, January, 1988.

[265] W. Noh, and P. Woodward, *SLIC (Simple Line Interface Method)*, Lecture Notes in Physics, Volume 59, Pages 330–340, 1976.

[266] T. Oden, *Finite Elements of Non-Linear Continua*, McGraw-Hill, New York, 1972.

[267] E. Oran, and J. Boris, *Numerical Simulation of Reactive Flow*, Elsevier, 1987.

[268] S. Osher, *Riemann Solvers, the Entropy Condition, and Difference Solvers*, SIAM Journal of Numerical Analysis, Volume 21, Pages 289–315, 1984.

[269] S. Osher, and J. Sethian, *Fronts Propagating with Curvature-Dependent Speed: Algorithms Based on the Hamilton-Jacobi Formulation*, Journal of Computational Physics, Volume 79, Pages 12–49, 1988.

[270] S. Osher, and F. Solomon, *Upwind Difference Schemes for Hyperbolic Systems of Conservation Laws*, Mathematics of Computation, Volume 38, Number 158, Pages 339–374, 1982.

[271] H. Paillère, K. Powell, and D. De Zeeuw, *A Wave-Model-Based Refinement Criterion for Adaptive-Grid Computation of Compressible Flows*, in Proceedings of the 30th Aerospace Sciences Meeting and Exhibit, 6-9 January 1992, Reno, Nevada, AIAA Paper 92-0322, AIAA.

[272] J. Paraschivoiu, J.-Y. Trépanier, M. Reggio, and R. Camarero, *A Conservative Dynamic Discontinuity Tracking Algorithm for the Euler Equations*, AIAA Paper 94-0081, AIAA, January 1994.

[273] R. Pember, J. Bell, P. Colella, W. Crutchfield, and M. Welcome, *Adaptive Cartesian Grid Methods for Representing Geometry in Inviscid Compressible Flow*, AIAA Paper 93-3385-CP, Proceedings, 11th AIAA Computational Fluid Dynamics Conference, AIAA, July, 1993.

[274] R. Pember, L. Howell, J. Bell, P. Colella, W. Crutchfield, W. Fiveland, and J.Jessee, *An Adaptive Projection Method for Unsteady, Low-Mach Number Combustion*, Combustion Science and Technology, Volume 140, Number 1-6, Pages 123–168, 1998.

[275] J. Peraire, J. Piero, L. Formaggia, K. Morgan, and O. Zienkeiwicz, *Finite-Element Euler Calculations in Three-Dimensions*, AIAA Paper 87-0032, AIAA, 1987.

[276] J. Peraire, J. Piero, L. Formaggia, K. Morgan, and O. Zienkeiwicz, *Finite Element Euler Computations in Three Dimensions*, International Journal of Numerical Methods in Engineering, Volume 26, Pages 2135–2159, 1988.

[277] J. Peraire, M. Vahdati, K. Morgan, and O. Zienkeiwicz, *Adaptive Remeshing for Compressible Flow Computations*, Journal of Computational Physics, Volume 72, Pages 449–466, 1987.

[278] K. Pericleous, K. Chan, and M. Cross, *Free Surface Flow and Heat Transfer in Cavities: The SEA Algorithm*, Journal of Numerical Heat Transfer, Part B, Volume 27, Pages 487–507, 1995.

[279] C. Peskin, *Numerical Analysis of Blood Flow in the Heart*, Journal of Computational Physics, Volume 25, Pages 220–252, 1977.

[280] S. Pirzadeh, *Viscous Unstructured Three-Dimensional Grids by the Advancing-Layers Method*, AIAA Paper 94-0417, AIAA, 1994.

[281] A. Pothen, H. Simon, and P. Liou, *Partitioning Sparse Matrices with Eigenvectors of Graphs*, SIAM Journal on Matrix Analysis and Applications, Volume 11, Pages 430–452, 1990.

[282] F. Preparata, and M. Shamos, *Computational Geometry, an Introduction*, Springer-Verlag, New York, 1985.

[283] E. Probert, O. Hassan, K. Morgan, and J. Peraire, *An Adaptive Finite Element Method for Transient Compressible Flows with Moving Boundaries*, *International Journal for Nummerical Methods in Engineering*, Volume 32, Pages 751–765, 1991.

[284] K. Powell, *A Tree-Based Adaptive Scheme for Solution of the Equations of Gas Dynamics and Magnetohydrodynamics*, Applied Numerical Mathematics, Volume 14, Pages 327–352, 1994.

[285] K. Powell, and B. van Leer, *A Genuinely Multi-Dimensional Upwind Cell-Vertex Scheme for the Euler Equations*, AIAA Paper Number 89-0095, AIAA, Reno, Nevada, 1989.

[286] K. Powell, P. Roe, R.-S. Myong, T. Gombosi, and D. De Zeeuw, *An Upwind Scheme for Magnetohydrodynamics*, AIAA Paper Number 95-1704-CP, AIAA, 1995.

[287] T. Pulliam, *Artificial Dissipation Models for the Euler Equations*, AIAA Paper 85-0438, AIAA, 1985.

[288] T. Pulliam, *Characteristic Boundary Conditions for the Euler Equations*, Numerical Boundary Condition Procedures, NASA CP Number 2201, Pages 165–181, 1981.

[289] T. Pulliam, *Computational Challenge: Euler Solutions for Ellipses*, AIAA Journal, AIAA, Volume 28, Number 10, Pages 1703–1704, October, 1990.

[290] J. Purvis, and J. Burkhalter, *Prediction of Critical Mach Number for Store Configurations*, AIAA Journal, AIAA, Volume 17, Pages 1170–1177, 1979.

[291] J. Quirk, *An Adaptive Grid Algorithm for Computational Shock Hydrodynamics*, Ph.D. Thesis, College of Aeronautics, Cranfield Institute of Technology, U.K., January, 1991.

[292] J. Quirk, *An Alternative to Unstructured Grids for Computing Gas Dynamic Flows Around Arbitrarily Complex Two-Dimensional Bodies*, Computers and Fluids, Volume 23,Number 1, Pages 125–142, 1994. Also published as ICASE Report Number 92-7, ICASE, NASA Langley Research Center, February, 1992.

[293] J. Quirk, *A Contribution to the Great Riemann Solver Debate*, ICASE Report Number 92-64, ICASE, NASA Langley Research Center, November, 1992.

[294] M. Rai, *A Conservative Treatment of Zonal Boundaries for Euler Equation Calculations*, Journal of Computational Physics, Volume 62, Number 2, p. 472, 1986.

[295] R. Rauch, J. Batina, and H. Yang, *Three-Dimensional Time-Marching Aeroelastic Analyses Using an Unstructured-Grid Euler Method*, AIAA Journal, AIAA, Volume 31, Number 9, 1993.

[296] S. Rebay, *Efficient Unstructured Mesh Generation by Means of Delaunay Triangulation and Boyer-Watson Algorithm*, Journal of Computational Physics, Volume 106, Issue 1, Pages 125–138, 1993.

[297] M. Reggio, J.-Y. Trépanier, H. Zhang, and R. Camarero, *Numerical Simulation of the Gas Flow in a Circuit Breaker*, International Journal for Numerical Methods in Engineering, Volume 34, Pages 607–618, 1992.

[298] T. Reyhner, *Cartesian Mesh Solution for Axisymmetric Transonic Potential Flow Around Inlets*, AIAA Journal, AIAA, Volume 15, Number 5, Pages 624–631, 1977,

[299] W. Rheinboldt, and C. Mesztenyi, *On a Data Structure for Adaptive Finite Element Mesh Refinements*, ACM Transactions on Mathematical Software, Volume 6, Number 2, June 1980, Pages 166–187.

[300] R. Richtmeyer, and K. Morton, "Difference Methods for Initial-Value Problems", 2nd Edition, Wiley Interscience, 1967.

[301] W. Rider, and D. Kothe, *Stretching and Tearing Interface Tracking Methods*, AIAA Paper 95-1717, AIAA, 1995.

W. Rider, and D. Kothe, *Reconstructing Volume Tracking*, Journal of Computational Physics, Volume 141, Pages 112–152, 1998.

[302] W. Rider, D. Kothe, S. Mosso, J. Cerruti, and J. Hochstein, *Accurate Solution Algorithms for Incompressible Multiphase Fluid Flows*, AIAA Paper 95-0699, AIAA, 1995.

[303] S. Rill, *A High-Resolution, Implicit Scheme for the Solution of the Euler Equations*, DGLR Paper 89-167, Pages 377–386, 1989.

[304] S. Rippa, *Minimal Roughness Property of the Delaunay Triangulation*, Computer Aided Geometric Design, North-Holland, Number 7, Pages 489–497, 1990.

[305] A. Rizzi, *Numerical Implementation of Solid-Body Boundary Conditions for the Euler Equations*, ZAMM, Volume 58, Pages 301–304, 1978.

[306] A. Rizzi, and L. Eriksson, *Computation of Flow Around Wings Based on the Euler Equations*, Journal of Fluid Mechanics, Volume 148, Pages 45–71, 1984.

[307] P. Roache, *Computational Fluid Dynamics*, Hermosa Publishers, 1972.

[308] P. Roe, *Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes*, Journal of Computational Physics, Volume 43, Number 2, Pages 357–372, 1981.

[309] P. Roe, *Characteristic-Based Schemes for the Euler Equations*, Annual Review of Fluid Mechanics, Volume 18, Pages 337–365, 1986.

[310] P. Roe, *Discrete Models for the Numerical Analysis of of Time-Dependent Multi-Dimensional Gas Dynamics*, Journal of Computational Physics, Volume 63, Pages 458–476, 1986.

[311] P. Roe, Private communication, November, 1994.

[312] P. Roe, Private communication, November, 1995.

[313] P. Roe, *The Use of the Riemann Problem in Finite-Difference Schemes*, Lecture Notes in Physics, Number 141, Springer-Verlag, 1981.

[314] P. Roe, *Some Contributions to the Modelling of Discontinuous Flows*, Lecture Notes in Applied Mathematics, Volume 22, American Mathematical Society, Springer-Verlag, Pages 163–193, 1985.

[315] P. Roe, *Remote Boundary Conditions for Unsteady Multidimensional Aerodynamic Applications*, ICASE Report Number 86-75, 1986.

[316] P. Roe, Private communication, April, 1996.

[317] P. Roe, and J. Pike, *Efficient Construction and Utilization of Approximate Riemann Solutions*, Computing Methods in Applied Sciences and Engineering, Ed. R. Glowinski, and J.-L. Lions, Volume 6, Pages 499–518, North-Holland.

[318] H. Samet, *The Design and Analysis of Spatial Data Structures*, Addison-Wesley Publishing Company, 1989 (and republished with corrections, 1990).

[319] H. Samet, *Applications of Spatial Data Structures: Computer Graphics, Image Processing, and GIS*, Addison-Wesley Publishing Company, 1990.

[320] M. Salas, *Shock-Fitting Method for Complicated Two-Dimensional Supersonic Flows*, AIAA Journal, AIAA, Volume 14, Pages 583–588, 1976.

[321] W. Seibert, W. Fritz, and S. Leicher, *On the Way to an Integrated Mesh Generation System for Industrial Applications*, Paper Number 14, AGARD Conference Proceedings Number 464, 1989.

[322] S. Sengupta, J. Hauser, P. Eiseman, and J. Thompson (editors), *Numerical Grid Generation in Computational Fluid Mechanics '88*, Pineridge Press, Swansea, U.K., 1988.

[323] J. Sethian, *Level Set Methods*, Cambridge University Press, 1996.

[324] J. Sethian, *A Review of the Theory, Algorithms, and Applications of Level Set Methods for Propagating Interfaces*, Acta Numerica, Cambridge University Press, 1996.

[325] A. Shapiro, *The Dynamics and Thermodynamics of Compressible Fluid Flow*, Volume I, John Wiley and Sons, 1953.

[326] A. Shapiro, *The Dynamics and Thermodynamics of Compressible Fluid Flow*, Volume II, The Ronald Press Company, 1954.

[327] M. Shepard, *Approaches to Automatic Generation and Control of Finite-Element Meshes*, Applied Mechanics Review, Volume 41, Number 4, Pages 169–185, 1988.

[328] M. Shephard, and N. Weatherill, (editors) *Adaptive Remeshing*, Special Issue of the International Journal of Numerical Methods in Engineering, Volume 32, Number 4, 1991.

[329] C. Shu, and S. Osher, *Efficient Implementation of Essentially Non-Oscillatory Shock-Capturing Schemes*, Journal of Computational Physics, Volume 77, Pages 439–471, 1988.

[330] G. Shubin, A. Stevens, and J. Bell, *Three-Dimensional Grid Generation Using Biharmonics*, in [360].

[331] W. Shyy, H. Udaykumar, M. Rao, and R. Smith, *Computational Fluid Dynamics with Moving Boundaries*, Taylor and Francis, 1996.

[332] H. Simon, *Partitioning of Unstructured Problems for Parallel Processing*, Computing Systems in Engineering, Volume 2 (2/3), Pages 135–148, 1991.

[333] G. Smith, "Numerical Solution of Partial Differential Equations: Finite Difference Methods", Third Edition, Oxford University Press (Clarendon Press), 1985.

[334] R. Smith, and E. Everton, *Interactive Grid Generation for Fighter Aircraft Geometries*, in [322].

[335] J. Smoller, *Shock-Waves and Reaction-Diffusion Equations*, Springer, 1983.

[336] J. Smoller, *On the Solution of the Riemann Problem with General Step Data for an Extended Class of Hyperbolic Systems*, Michigan Mathematical Journal, The University of Michigan, Pages 201–210, 1969.

[337] G. Sod, *A Survey of Several Finite Difference Methods for Systems of Nonlinear Hyperbolic Conservation Laws*, Journal of Computational Physics, Volume 27, Pages 1–31, 1978.

[338] J. Solomon, M. Clement, R. Ferguson, and J. Bell, *Inviscid Flowfield Calculations for Re-entry Vehicles with Control Surfaces*, AIAA Journal, AIAA, Volume 15, Number 12, December 1977.

[339] R. Sorenson, *A Computer Program to Generate Two-Dimensional Grids About Airfoils and Other Shapes by the Use of Poission's Equation*, NASA TM-81198, May 1980.

[340] S. Spekreijse, *Multigrid Solution of Monotone Second-Order Discretizations of Hyperbolic Conservation Laws*, Mathematics of Computing, Volume 49, Pages 135–155, 1987.

[341] S. Spekreijse, *Elliptic Grid Generation Based on Laplace Equations and Algebraic Transformations*, Journal of Computational Physics, Volume 118, Pages 28–61, 1995.

[342] J. Steger, F. Dougherty, and J. Benek, *Advances in Grid Generation: A Chimera Grid Scheme*, Edited by K. N. Ghia and U. Ghia, ASME Fluids Engineering Division, Volume 5, Pages 59–69, 1983.

[343] J. Steger, T. Pulliam, and R. Chima, *An Implicit Finite-Difference Code for Inviscid and Viscous Cascade Flows*, AIAA Paper Number 80-1427, AIAA 13th Fluid and Plasma Dynamics Conference, AIAA, July 1980.

[344] J. Steger, and Y. Rizk, *Generation of Three-Dimensional Body-Fitted Coordinates Using Hyperbolic Partial Differential Equations*, NASA TM-86753, June, 1985.

[345] J. Steger, and R. Warming, *Flux Vector Splitting of the Inviscid Gasdynamic Equations with Application to Finite-Difference Methods*, Journal of Computational Physics, Volume 40, Pages 263–293, 1981.

[346] R. Struijs, H. Deconinck, P. Depalma, P. Roe, and K. Powell, *Progress on Multidimensional Upwind Euler Solvers for Unstructured Grids*, AIAA Paper Number 91-1550-CP, AIAA, 1991.

[347] B-Q Su, and D-Y Liu, *Computational Geometry: Curve and Surface Modeling*, Academic Press, Inc., 1989.

[348] M. Sussman, A. Almgren, J. Bell, P. Colella, L. Howell, and M. Welcome, *An Adaptive Level-Set Approach for Incompressible Two-Phase Flows*, Journal of Computational Physics, Volume 148, Number 1, Pages 81–124, 1999.

[349] R. Swanson, and E. Turkel, *On Central Difference and Upwind Schemes*, Journal of Computational Physics, Volume 101, Pages 292–306, 1992.

[350] P. Sweby, *High-Resolution Schemes Using Flux Limiters for Hyperbolic Conservation Laws*, SIAM Journal on Numerical Analysis, Volume 21, Pages 995–1011, 1984.

[351] W. Szymczak, J. Rogers, J. Solomon, and A. Berger, *A Numerical Algorithm for Hydrodynamic Free-Boundary Problems*, Journal of Computational Physics, Volume 106, Pages 319–336, 1993.

[352] E. Tadmor, *The Numerical Viscosity of Entropy Stable Schemes for Systems of Conservation Laws, I*, Mathematics and Computation, Volume 49, Pages 91–103, 1987.

[353] C.-H. Tai, *Acceleration Techniques for Explicit Euler Codes*, Ph.D. Dissertation (in Aerospace Engineering), The University of Michigan, 1990.

[354] Y. Tamura, and K. Fujii, *Conservation Law for Moving and Transformed Grids*, AIAA Paper 93-3365, 11th AIAA CFD Conference, Orlando, Florida, AIAA, 6-9 July 1993.

[355] P. Thomas, and C. Lombard, *The Geometric Conservation Law - A Link Between Finite-Difference and Finite-Volume Methods of Flow Computation on Moving Grids*, AIAA Paper Number 78-1208, AIAA, 1978.

[356] P. Thomas, and C. Lombard, *Geometric Conservation Law and its Application to Flow Computations on Moving Grids*, AIAA Journal, AIAA, Volume 17, Number 10, Pages 1030–1037, 1979.

[357] P. Thomas, and J. Middlecoff, *Direct Control of the Grid Point Distribution in Meshes Generated by Elliptic Equations*, AIAA Journal, AIAA, Volume 18, Pages 652–656, June, 1980.

[358] P. Thomas, M. Vinokur, R. Bastianon, and R. Conti, *Numerical Solution for Three-Dimensional Inviscid Supersonic Flow*, AIAA Journal, AIAA, Volume 10, Number 7, July 1972.

[359] F. Thomasset, *Appendix to Navier-Stokes Problems*, Navier Stokes Equations: Theory and Numerical Analysis, R. Temam (Editor), North-Holland, Amsterdam, 1977.

[360] J. Thompson, (editor), *Numerical Grid Generation*, North-Holland, Amsterdam, 1982.

[361] J. Thompson, *Review on the State of the Art of Adaptive Grids*, AIAA Paper 84-1606, AIAA, 1984.

[362] K. Thompson, *Time Dependent Boundary Conditions for Hyperbolic Systems*, Journal of Computational Physics, Volume 68, Pages 1–24, 1987.

[363] J. Thompson, and J. Steger, *Three-Dimensional Grid Generation for Complex Configurations – Recent Progress*, AGARD-AG-309, 1988.

[364] J. Thompson, F. Thames, and W. Mastin, *Automatic Numerical Grid Generation of Body-Fitted Curvilinear Co-ordinate System of Field Containing any Number of Arbitrary Two-Dimensional Bodies*, Journal of Computational Physics, Volume 15, Pages 299–319, 1974.

[365] J. Thompson, Z. Warsi, and W. Mastin, *Numerical Grid Generation: Foundations and Application*, North-Holland, 1985.

[366] J. Thompson, and N. Weatherill, *Aspects of Numerical Grid Generation: Current Science and Art*, AIAA Paper 93-3539, 11th Applied Aerodynamics Conference, AIAA, August, 1993.

[367] D. Tidd, D. Strash, B. Epstein, A. Luntz, A. Nachson, and T. Rubin, *Application of an Efficient 3-D Multigrid Euler Method (MGAERO) to Complete Aircraft Configurations*, AIAA Paper 91-3236, AIAA, 1991.

[368] J.-Y. Trépanier, M. Reggio, and R. Camarero, *Automated Geometric-Based Mesh Requirements for Adaptive Flow Computations*, AIAA Paper 93-0674, AIAA, 1993.

[369] E. Turkel, *Accuracy of Schemes With Non-Uniform Meshes for Compressible Fluid Flows*, ICASE Technical Report Number 85-43, ICASE, 1985.

[370] S. Unverdi, and G. Tryggvason, *A Front-Tracking Method for Viscous, Incompressible Multi-Fluid Flows*, Journal of Computational Physics, Volume 100, Pages 25–37, 1992.

[371] S. Unverdi, and G. Tryggvason, *Computations of Multi-Fluid Flows*, Physica D, Volume 60, Pages 70–83, 1992.

[372] University of Toronto Institute of Aerospace Sciences, *Definition of Two Benchmark Problems*, (CFD 94). Note that the computations there were significantly under-resolved compared to what's shown, for example, here in this work or in [233].

[373] G. Van Albada, B. van Leer, and W. Roberts, *High-Resolution Schemes Using Flux Limiters for Hyperbolic Conservation Laws*, Jounral of Astronomy and Astrophysics, Volume 108, 1982.

[374] B. van Leer, *Towards the Ultimate Conservative Difference Scheme, I: The Quest of Monotonicity*, Springer Lecture Notes in Physics, Volume 18, Pages 163–168, 1973.

[375] B. van Leer, *Towards the Ultimate Conservative Difference Scheme, II: Monotonicity and Conservation Combined in a Second-Order Scheme*, Journal of Computational Physics, Volume 14, Pages 361–370, 1974.

[376] B. van Leer, *Towards the Ultimate Conservative Difference Scheme, III: Upstream-Centered Finite-Difference Schemes for Ideal Compressible Flow*, Journal of Computational Physics, Volume 23, Pages 263–275, 1977.

[377] B. van Leer, *Towards the Ultimate Conservative Difference Scheme, IV: A New Approach to Numerical Convection*, Journal of Computational Physics, Volume 23, Pages 276–299, 1977.

[378] B. van Leer, *Towards the Ultimate Conservative Difference Scheme, V: A Second-Order Sequel to Godunov's Method*, Journal of Computational Physics, Volume 32, Pages 101–136, 1979.

[379] B. van Leer, *Flux Vector Splitting for the Euler Equations*, Lecture Notes in Physics, Volume 170, Pages 501–512, 1982.

[380] B. van Leer, *On the Relation Between the Upwind-Differencing Schemes of Godunov, Engquist-Osher, and Roe*, SIAM Journal of Scientific and Statistical Computing, Volume 5, Pages 1–19, 1984.

[381] B. van Leer, Private Discussions, February and March 1994.

[382] B. van Leer, *Progress in Multi-Dimensional Upwind Differencing*, ICASE Report 92-43, 1992.

[383] B. van Leer, Private Communication, 1995.

[384] B. van Leer, *Numerical Fluid Dynamics, II*, ICASE Report Number 36, ICASE, 1987.

[385] B. van Leer, J. Thomas, P. Roe, and R. Newsome, *A Comparison of Numerical Flux Formulas for the Euler and Navier-Stokes Equations*, AIAA Paper Number 87-2033, AIAA, 1987.

[386] G. Van Wylen, and R. Sonntag, *Fundamentals of Classical Thermodynamics*, SI Version 2^e, Revised Printing, John Wiley and Sons, 1976.

[387] P. Vankeirsbilck, and H. Deconinck, *Higher Order Upwind Finite Volume Schemes with ENO-Properties for General Unstructured Meshes*, Pages 7–1 to 7–54, in *Special Course on Unstructured Grid Methods for Advection Dominated Flows*, AGARD Report 787 (AGARD-R-787), AGARD (Advisory Group for Aerospace Research and Development, NATO), May 1992.

[388] V. Venkatakrishnan, *On the Accuracy of Limiters and Convergence to Steady State Solutions*, AIAA Paper Number 93-0880, AIAA, 1993.

[389] V. Venkatakrishnan, *Convergence to Steady State Solutions of the Euler Equations on Unstructured Grids with Limiters*, Journal of Computational Physics, Volume 118, Pages 120–130, 1995.

[390] V. Venkatakrishnan, and D. Mavriplis, *Computation of Unsteady Flows Over Complex Geometries in Relative Motion*, First AFOSR Conference on Dynamic Motion CFD, New Brunswick, New Jersey, June 1996.

[391] A. Verhoff, *Modeling of Computational and Solid Surface Boundary Conditions for Fluid Dynamics Calculations*, AIAA Paper Number 85-1496, AIAA 7th CFD Conference, AIAA, 1985.

[392] A. Verhoff, T. Michal, and D. Stookesberry, *Solution of the Euler Equations for Two-Element Airfoils Using Asymptotic Methods*, AIAA Paper Number 94-0082, AIAA, 1994.

[393] A. Verhoff, D. Stookesberry, *Solution of the Euler Equations for Airfoils Using Asymptotic Methods*, AIAA Paper Paper Number 93-2931, AIAA, July 1993.

[394] A. Verhoff, D. Stookesberry, and S. Agrawal, *Far-Field Computational Boundary Conditions for the Two-Dimensional External Flow Problem*, AIAA Journal, AIAA, Volume 30, Number 11, Pages 2585–2594, 1992.

[395] W. Vincenti, and C. Kruger, Jr., *Introduction to Physical Gas Dynamics*, Robert E. Krieger Publishing Company, Malabar, Florida, 1986 Reprint with Corrections, (1965).

[396] M. Vinokur, *An Analysis of Finite-Difference and Finite-Volume Formulations of Conservation Laws*, Journal of Computational Physics, Volume 81, Number 1, Pages 1–52, 1989.

[397] Z.-J. Wang, *A Fast Nested Multi-Grid Flow Solver for Adaptive Cartesian / Quad Grids*, AIAA Paper Number 96-2091, AIAA, 1996.

[398] S. Wang, R. Sekerka, A. Wheeler, B. Murray, S. Coriell, and G. McFadden, *Thermodynamically Consistent Phase-Field Models for Solidification*, Physica D, Volume 69, Pages 189–200, 1993.

[399] A. Wardlaw, Jr., and F. Baltakis, *An Integral Boundary-Layer Procedure for Tactical Missiles*, AIAA Paper 92-1026, AIAA, 1992.

[400] A. Wardlaw Jr., J. Solomon, and F. Baltakis, *Supersonic Inviscid Flowfield Computations of Missile Type Bodies*, AIAA Journal, AIAA, Volume 19, Number 7, July 1981.

[401] R. Warming, R. Beam, and H. Yee, *Lecture Notes on the Stability of Difference Approximations for Initial Boundary Value Problems*, NASA Report TM Number 84318, 1983.

[402] G. Warren, K. Anderson, J. Thomas, and S. Krist, *Grid Convergence for Adaptive Methods*, AIAA Paper 91-1592, AIAA 10th Computational Fluid Dynamics Conference, AIAA, June 24-26, 1991.

[403] D. Watson, *Computing the n-Dimensional Delaunay Tesselation With Applications to Voronoi Polytopes*, The Computer Journal, Volume 24, Number 2, Pages 167–172, 1981.

[404] N. Weatherill, and O. Hassan, *Efficient Three-Dimensional Grid Generation Using the Delaunay Triangulation*, C. Hirsch (editor), First Computational Fluid Dynamics Conference, Brussels, September, 1992, Elsevier.

[405] N. Weatherill, and C. Forsey, *Grid Generation and Flow Calculations for Complex Aircraft Geometries Using a Multi-Block Scheme*, AIAA Paper 84-1665, AIAA, June 1984.

[406] N. Weatherill, *On the Combination of Structured-Unstructured Meshes*, in [322].

[407] N. Weatherill, *Grid Generation*, VKI Lecture Series in Numerical Grid Generation, June 1990.

[408] N. Weatherill, *A Method for Generating Irregular Computational Grids in Multiply-Connected Planar Domains*, International Journal for Numerical Methods in Fluids, Volume 8, Pages 181–197, 1988.

[409] N. Weatherill, *Mixed Structured-Unstructured Meshes for Aerodynamic Flow Simulation*, The Aeronautical Journal, Volume 94, Number 934, Pages 111–123, April, 1990.

[410] B. Wedan, and J. South, Jr., *A Method for Solving the Transonic Full-Potential Equation for General Configurations*, AIAA Paper 83-1889, AIAA, 1983.

[411] R. Winterstein, and M. Hafez, *Euler Solutions for Blunt Bodies Using Triangular Meshes: Artificial Viscosity Forms and Numerical Boundary Conditions*, AIAA Paper 93-3333-CP, AIAA, July 1993.

[412] D. Whitfield, J. Thomas, A. Jameson, W. Schmidt, *Computation of Transonic Viscous-Inviscid Interaction Flow*, Second Symposium on Numerical and Physical Aspects of Aerodynamic Flows, Springer-Verlag, 1983.

[413] G. Witham, *Linear and Nonlinear Waves*, Wiley-Interscience, New York, 1974.

[414] P. Woodward, and P. Colella, *The Numerical Simulation of Two-Dimensional Fluid Flow with Strong Shocks*, Journal of Computational Physics, Volume 54, Pages 115–173, 1984.

[415] S. Wornom, and M. Hafez, *A Rule for Selecting Analytical Boundary Conditions for the Conservative Quasi-One-Dimensional Nozzle Flow Equations*, AIAA Paper Number 84-0431, AIAA, 1984.

[416] K. Xu, L. Martinelli, and A. Jameson, *Gas-Kinetic Finite-Volume Methods, Flux-Vector Splitting, and Artificial Diffusion*, Journal of Computational Physics, Volume 120, Pages 48–65, 1995.

[417] J. Yang, Y. Liu, and H. Lomax, *Computation of Shock Wave Reflection by Circular Cylinders*, AIAA Journal, AIAA, Volume 25, Number 5, Pages 683–689, 1987.

[418] H. Yee, *A Class of High-Resolution Explicit and Implicit Shock-Capturing Methods*, NASA Technical Memorandum Number 101088, 1989.

[419] H. Yee, *Numerical Approximation of Boundary Conditions with Applications to Inviscid Equations of Gas Dynamics*, NASA Technical Memorandum TM Number 81265, 1981.

[420] H. Yee, R. Beam, and R. Warming, *Boundary Approximations for Implicit Schemes for One-Dimensional Inviscid Equations of Gasdynamics*, AIAA Journal, AIAA, Volume 20, Number 9, Pages 1203–1211, 1982.

[421] H. Yee, and A. Harten, *Implicit TVD Schemes for Hyperbolic Conservation Laws in Curvilinear Coordinates*, AIAA Paper Number 85-1513, AIAA, Cincinnati, 1985.

[422] H. Yee, R. Warming, and A. Harten, *Implicit Total Variation Diminishing (TVD) Schemes for Steady-State Calculations*, Journal of Computational Physics, Volume 75, Pages 327–360, 1985.

[423] M. Yerry, and M. Shephard, *Automatic Three-Dimensional Mesh Generation by the Modified Octree Technique*, International Journal for Numerical Methods in Engineering, Volume 20, Pages 1965–1990, 1987.

[424] D. Young, R. Melvin, M. Bieterman, F. Johnson, and S. Samant, *A Locally Refined Rectangular Grid Finite Element Method: Application to Computational Fluid Dynamics and Computational Physics*, Journal of Computational Physics, Volume 62, Pages 1–66, 1991.

[425] S. Zalesak, *Fully-Multidimensional Flux-Corrected Transport Algorithms for Fluids*, Journal of Computational Physics, Volume 31, Pages 335–362, 1979.

[426] Y. Zel'dovich, and Y. Raizer, *Elements of Gasdynamics and the Classical Theory of Shock Waves*, Edited by W. Hayes and R. Probstein, Academic Press, 1966.

[427] M. Zerroukat, and C. Chatwin, *Computational Moving Boundary Problems*, John Wiley and Sons, 1994.

[428] D. Zhang, and I. Glass, *An Interferometric Investigation of the Diffraction of Planar Shock Waves Over a Half-Diamond Cylinder in Air*, in University of Toronto Institute for Aerospace Studies (UTIAS) Report Number 322, March 1988.

[429] H. Zhang, M. Reggio, J.-Y. Trépanier, and R. Camarero, *Discrete Form of the GCL for Moving Meshes and its Implementation in CFD Schemes*, Computers and Fluids, Volume 22, Number 1, Pages 9–23, 1993.

[430] O. Zienkiewicz, *The Finite-Element Method*, McGraw-Hill, New York, 1977.

[431] M. Zucrow, and J. Hoffman, *Gas Dynamics*, Volume I, John Wiley and Sons, 1976.