A front-tracking shock-capturing method for two fluids

By

MEHDI VAHAB

B.S. (Khaje Nasir Toosi University of Technology, Tehran) 2005
M.S. (Chalmers University of Technology, Gothenburg) 2008

THESIS

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Engineering - Applied Science

in the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

Gregory H. Miller, Chair

François Gygi

Elbridge G. Puckett

Committee in Charge

2014

Contents

Mehdi Vahab
June 2014
Engineering - Applied Science

A front-tracking shock-capturing method for two fluids

Abstract

This dissertation presents a new high-order front tracking method for two-phase hyperbolic systems of conservation laws separated by a contact discontinuity. A review of existing methods for moving and/or irregular boundaries shows the significance of accurate geometry data and flux calculation near the interface to achieve a high order method. A general method for hyperbolic systems of conservation laws is presented along with the implementations of numerical methods for simulations of gas dynamics in 2-D using the Euler equations. Convergence tests show the new method is second order accurate for smooth solutions and first order in presence of shocks. Also the new method is used for simulation of Richtmyer-Meshkov instability, in which results are in agreement with both theoretical and experimental approaches.

Acknowledgments

I would like to thank my adviser Professor Greg Miller. I really appreciate his patient guidance, earnest encouragement and helpful critiques of this research work. Also, I would like to thank my parents, Reza and Afsaneh, for their heartwarming motivations and supports.

Mehdi Vahab

CHAPTER 1

Introduction

Hyperbolic systems of conservation laws are fundamental in continuum physics, and the treatment of their discontinuities is an important subject in fluid dynamics. Such systems have various applications in different fields. For example, study of cell membrane and cell behavior has applications in wound healing and development of tissues [**1, 2**]. Another example is study of bubble dynamics which is relevant in investigation of boiling heat transfer, cloud cavitation, bubble columns and reactors in the chemical industry [**3, 4**]. One may also find the importance of this study in flame propagation, detonation, deflagration and their transitions [**5, 6**]. In essence, fundamental study of interface dynamics between two fluids and corresponding instabilities, such as Kelvin-Helmholtz, Rayleigh-Taylor and Richtmyer-Meshkov[**7, 8**], has significance in many scientific fields. Therefore, many numerical methods have been developed to study these systems with different approaches.

Ideally, numerical methods are convergent with respect to some power of the mesh size h for smooth solutions, that is, $\mathcal{O}(h^p)$ for a p-th order convergent method. In the presence of a discontinuity, e.g., a shock or a material interface, numerical methods may have lower rate of convergence. Many of them have $\mathcal{O}(1)$ truncation error, that is, are not convergent. This is one of the main concerns of designing a numerical method for systems of conservation laws and will be addressed in the dissertation. In addition, if the numerical method does not treat the discontinuity specifically, it may smear out the discontinuity, and as a result, lack a sharp representation of the discontinuity in the solution. Special treatments are required to make a solution convergent and keep the discontinuities sharp while enforcing conservation.

A moving material interface, the general case when two different phases are separated by a material front that moves due the surrounding phases dynamics, is the case discussed in this dissertation. The numerical methods for modeling such system may encounter some difficulties such as front representation in two and higher dimensions, small volume fraction CFL limitation for fixed grid methods, loss of order of local truncation error at the material interface, and complex heuristic algorithms for front handling. These issues are discussed below by explaining some of the numerical methods, and possible design criteria to avoid them.

A new 2-D front-tracking method for hyperbolic systems of conservation laws is introduced. In this chapter a general definition of the problem and an overview of the previous studies and different approaches are presented. The second chapter contains the definition for the new method and details of the discretization procedure. The general algorithm and details of the implementations in 2-D are covered in chapter three. Chapter four contains the convergence tests and verifications studies, finished with the summary and conclusion.

1.1. Problem definition

The main focus in this study is on the numerical methods for hyperbolic systems of conservation laws. A general formulation for such systems, in D spatial dimensions and with m conserved quantities, is as follows:

$$\frac{\partial \mathbf{U}}{\partial t} + \nabla \cdot \vec{\mathbf{F}} = 0, \tag{1.1}$$

$$\mathbf{U} = \mathbf{U}(\mathbf{x}, t), \quad \mathbf{x} \in \mathbf{\Omega} \subset \mathbb{R}^D, \tag{1.2}$$

$$\vec{\mathbf{F}} = (\mathbf{F}^1 \ldots \mathbf{F}^D) = \vec{\mathbf{F}}(\mathbf{U}), \tag{1.3}$$

$$\mathbf{U},\, \mathbf{F}^d \in \mathbb{R}^m, \tag{1.4}$$

where $\mathbf{U}$ is a vector of conserved variables, and $\vec{\mathbf{F}}$ is the corresponding flux vector defined in the problem domain $\mathbf{\Omega}$. A material interface $\mathcal{F}(t)$ separates the problem domain into two subdomains $\mathbf{\Omega}_1(t)$ and $\mathbf{\Omega}_2(t)$. Each phase is governed by the equation system stated above while fulfilling the Rankine-Hugoniot jump condition on the front,

$$\vec{n}_s \cdot \vec{\mathbf{F}}_1 - s^f \mathbf{U}_1 = \vec{n}_s \cdot \vec{\mathbf{F}}_2 - s^f \mathbf{U}_2, \tag{1.5}$$

where $\vec{n}_s$ is the spatial normal vector on the front from $\mathbf{\Omega}_1$ to $\mathbf{\Omega}_2$, and s^f is the front velocity in the direction of $\vec{n}_s$ (Figure 1.1).

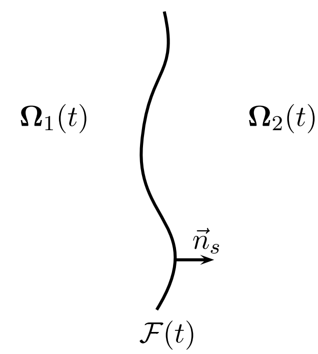

FIGURE 1.1. Two-phase flow domain and the moving interface.

1.2. Previous studies

The flows of immiscible fluids, separated by a resolved material interface, is a common phenomenon in fluid mechanics, and various numerical methods have been developed and successfully used to study it. A broad distinction between these methods arises from their mechanical system point of view. The two mainstream viewpoints are Lagrangian and Eulerian. In the Lagrangian description of a moving fluid, mesh points are defined on material particles and move with them, see Figure 1.2(a), which results in equations without convective terms. Aligning the interface on the moving grid brings a simple description of the interface. Although, difficulties may arise when material distortion causes entanglement of the mesh, that may be resolved with remeshing/rezoning. See the Lagrangian methods of Dukowics [**9**], Fritts and Boris [**10**], and the arbitrary Eulerian-Lagrangian (ALE) method of Huerta [**11**] for examples of this approach. In the Eulerian description of a fluid, mesh grids are fixed and material distortions can be handled. Because of the fixed grid meshing numerical methods are more straight forward in areas away from the interface, see Figure 1.2(b). The Eulerian description has convective terms in the formulation which may add complications to the numerical methods.

Another distinction between these methods is the numerical algorithms they are using, which may be categorized in general as finite element methods (FEM), finite difference methods (FDM) and finite volume methods (FVM). Finite element methods try to find an approximate solution of the boundary value problem in each computational cell while minimizing the error for general solution when putting them together. Finite difference methods look at the problem in pointwise discretized domain and approximate differential operators with discretized difference operators.

Finite volume methods discretize the problem domain with computational cells and approximate the solution as an average value in each cell. These values are changed based on the fluxes calculated between cells. In each approach, accuracy of the numerical approximation procedures determine the overall error introduced into the solution.

Generally, FEMs are based on unstructured meshes while FDMs and FVMs are solved on structured grids. Although irregular meshes are more capable of representing arbitrary geometries, mesh generation and maintenance for time dependent domains are more complex than dealing with a lower dimension entity, such as a front in 2-D domain on a regular Cartesian grid in FDMs/FVMs. Also, it is easier to apply high order shock capturing methods on the Cartesian grid rather than on a unstructured mesh.

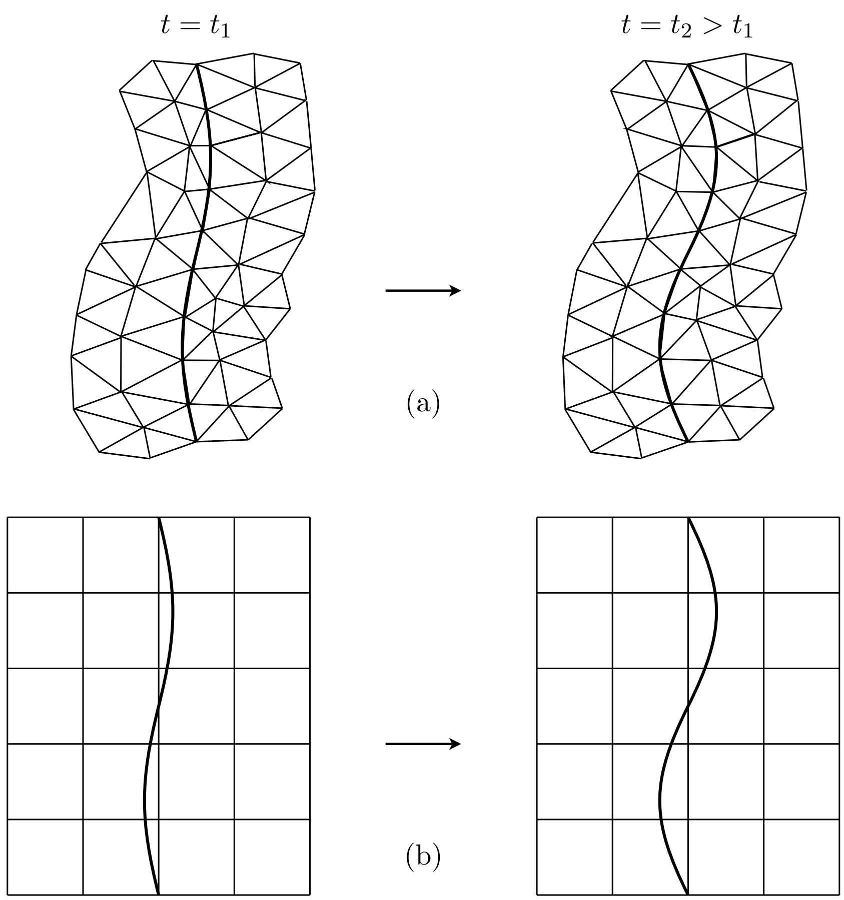

FIGURE 1.2. Front tracking with (a) an irregular mesh and Lagrangian method (the computational mesh deforms and move with the front) (b) a regular mesh and Eulerian method (the computational mesh is static).

Also, it is possible to classify these methods by their front representation and the way they handle the phase calculations near and on the front. In general, front tracking methods treat the interface explicitly and front capturing methods take the front as a steep gradients over a short distance. See examples of methods for front representation in Figure 1.3. One class of front-tracking methods is surface tracking [**12**]. Many developments have been done using surface-tracking methods [**13, 14**], although complex algorithms are involved for front entanglement and difficulties could arise for generalization to higher dimensions (see §1.3.2 for more details). For instance, the topology of the solution to Riemann problems is not known for the general case [**15**]. Volume-tracking methods introduce a simpler front representation. The domain of a phase is defined by the volume fraction values that a phase occupies in each computational cell. Therefore, the interface position is identified by the cells which have multiple phases in them, and evolved by solving an auxiliary evolutionary PDE. This approach is useful for interfaces with lots of topology changes, but lacks subgrid scale resolution [**16**]. Interface reconstruction methods were used to extract geometrical information from volume-tracking methods by finding the rectangular, piecewise linear or piecewise parabolic representation of the interface from volume fraction information [**16, 17**]. With the introduction of level set methods new front-tracking methods were developed for high order implicit front evolution [**18, 19**]. New interface reconstruction methods have been developed to extract geometry information from level set functions with arbitrary accuracy [**20, 21**].

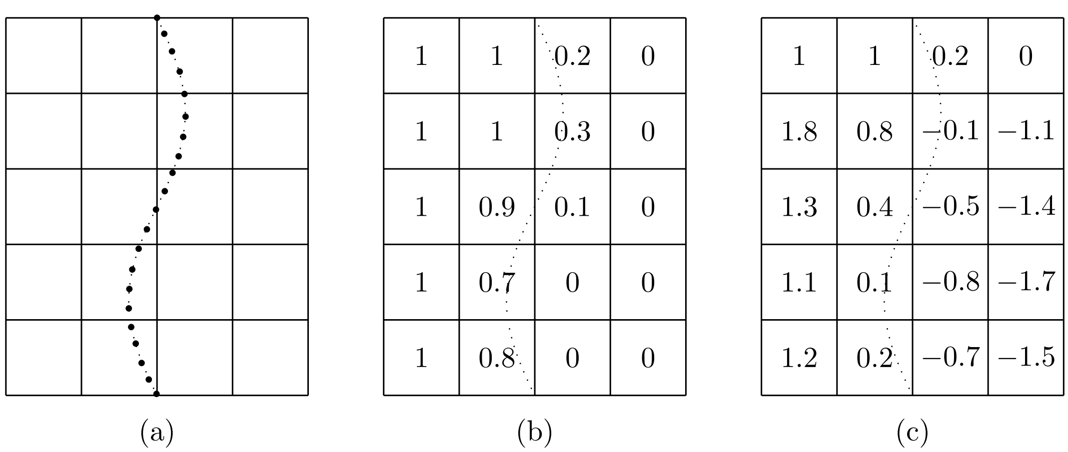

FIGURE 1.3. Examples of front representation (a) using particles for surface tracking type of methods (b) using volume fraction for volume tracking type of methods (c) distance function for level set type of methods.

Depending on available information on front position, finite volume and finite difference methods developed different ways to calculate state variables and fluxes near and on the front. The material interface moves, and for fixed grid methods, it may make small fractional cells near the interface. The small volume of those cells force severe limitation on unmodified finite volume and finite difference methods. As an example, a conservative finite volume discretization of equation (1.1) with a moving boundary for $D = 1$, grid size h in space and time step Δt gives

$$\mathbf{U}_i^{n+1} = \frac{\Lambda_i^n}{\Lambda_i^{n+1}}\mathbf{U}_i^n - \frac{\Delta t}{\Lambda_i^{n+1}h}\mathbf{DF}_i^{n,n+1}, \tag{1.6}$$

where $\mathbf{U}_\mathbf{i}^n$ is the discretized value of $\mathbf{U}$ at cell i, Λ_i^n is the volume fraction of the cell i at time step n and $\mathbf{DF}_i^{n,n+1}$ is the flux difference for control volume of cell i between time steps n and $n+1$ (Figure 1.4). The small volume fraction issue may be described as the numerical error associated with the right hand side of (1.6). When $\Lambda_i^{n+1} \to 0$, an unmodified finite difference or finite volume discretization may become unstable. This can also be shown by the CFL condition,

$$\Delta t \leq \frac{\min(\Lambda_i^n, \Lambda_i^{n+1})h}{v_i^{\max}} \tag{1.7}$$

where $v_i^{\max}$ is the maximum magnitude of the wave speed in cell i. For a regular cell, with $\Lambda_i^{n+1} = 1$, a typical time step is $\mathcal{O}(\frac{h}{v_i^{\max}})$. However, for a fractional cell near the front this may be much smaller if the volume fraction for that cell is small. Therefore, an unmodified method would severely limit the time step, or it will become unstable if it violates this necessary condition. Several approaches have been developed to address, resolve or circumvent this problem, such as cell merging [**22**], the h-box method [**23**], the ghost fluid method [**19**], and the single phase approximation [**24, 25**]. One other approach is the hybrid conservative method of Chern and Colella [**26**] and Bell et al. [**27**], which combines the conservative finite volume method with a nonconservative but stable update, and maintains global conservation using a redistribution algorithm. This idea has been successfully used for embedded boundary methods for static [**28**] and time-dependent [**20**] domains, and for a second-order conservative front-tracking method in one dimension [**29**].

The above classifications are not strict and there are several hybrid methods that benefit from multiple approaches. Examples of some these methods are discussed below in more detail, concluding with the approach used for the new method.

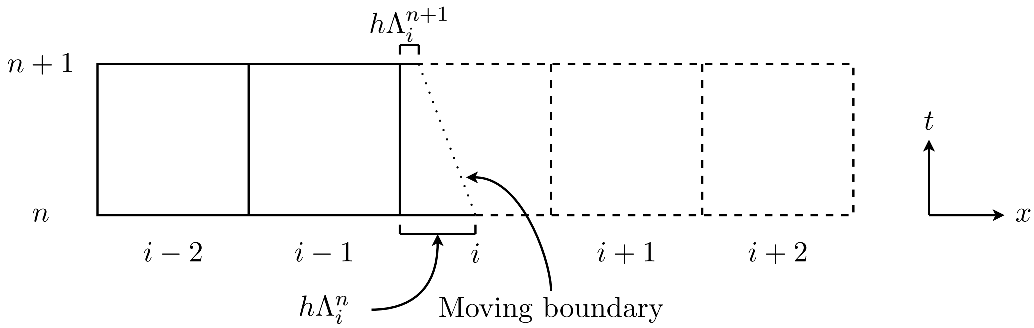

FIGURE 1.4. Discretization of a finite volume method in 1-D and time with a moving boundary.

1.3. Selected studies

1.3.1. XFEM - A finite element method. One of the issues in the field of moving interfaces is the way to define the interface with respect to computational grids. The same problem comes up in the analysis of cracks and fractures and their propagation. The extended finite element method (XFEM) was first introduced by Belytschko and Black [**30**] for modeling arbitrary discontinuities in a function. With a similar methodology, Chessa and Belytschko [**31**] developed a method to model the moving interface in a two-phase flow system. Based on the XFEM method, some enriched shape functions are added to the FEM base functions to have the desired type of discontinuity, e.g., discontinuity in the function or its derivatives. So the solution in the cells which include parts of an interface can approximate the discontinuity at the interface while normal shape functions are applicable to the cells away from the interface (Figure 1.5).

Tracking the interface was done by approximating the interface structure with a level set function. In this method, the level set function is the signed distance to the interface and approximated by the same mesh and shape functions as the variables which have the discontinuity on the interface. The standard level set evolution equation, in the form of a variable coefficient advection equation, governs the motion of the interface. For updating the interface position, a cut-off function is also used to reduce the computational contribution from the parts of domain that are far from the interface. Using these structures, the Navier-Stokes equations were solved by a characteristic based split algorithm using the projection method of Chorin [**32**]. Examples such as a bubble rising to a free surface and a drop falling onto a thin film show the capability of this method for modeling the dramatic topological changes in the interface [**31, 33**].

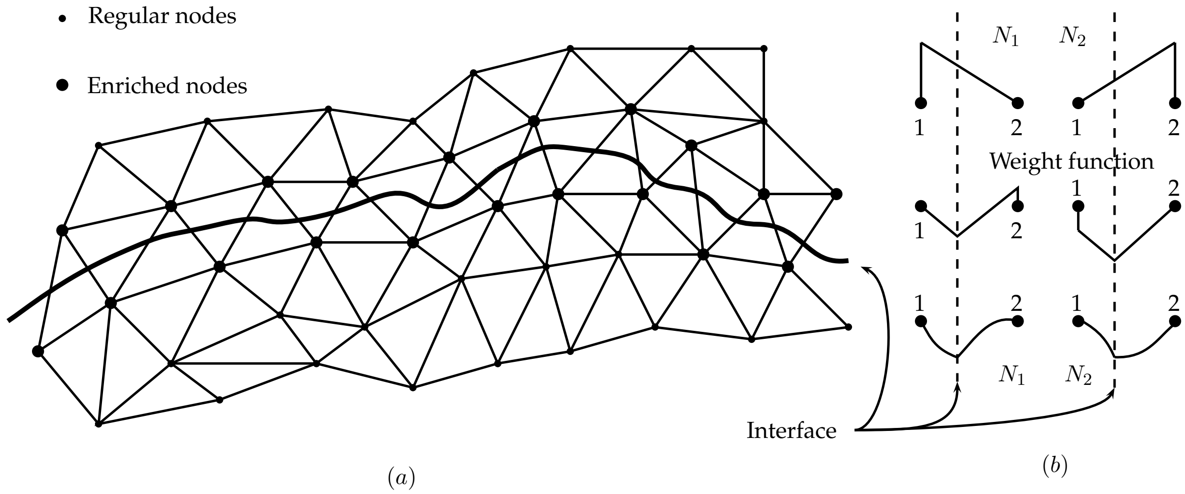

FIGURE 1.5. (a) Finite element mesh with enriched nodes around the interface. (b) Example of finite element shape functions in R^1 .

1.3.2. Glimm's method. In surface tracking methods, the interface is represented with set of marker points and interpolation between them. Glimm et al. developed a method based on this approach [**12, 13, 14**]. This algorithm is a combination of grid-based and grid free methods. A marker particle method, in which particles are located on the interface, is used. They proposed to update the front dynamics with a grid free method and the fluid dynamics away from the interface with a grid based method. This hybrid method, called the *locally grid-based method*, converts the grid free representation of interface to the grid-based model to apply the grid based method in bifurcation regions. To avoid the small control volume problem, they used cell-merging. That is, a small cell on the front is merged with a regular neighbor cell to create a cell large enough to avoid small cell CFL limitation. Although it is a straight-forward problem in one dimension, the extension to higher dimensions is not always a well-behaved procedure and may lose resolution near the interface.

1.3.3. Peskin's method. Peskin developed the Immersed Boundary (IB) method to simulate cardiac mechanics and blood flow dynamics around heart (for references and applications see the review paper by Peskin [**34**]). A method was developed in which a Cartesian grid is used for simulation with an Eulerian method. However, the interface is represented with a set of elastic fibers whose orientation is determined by massless particles which are evolved with a Lagrangian method, i.e., the particle velocity is the velocity of the surrounding fluid. The front-fluid interaction is completed by the force from a constitutive law such as Hooke's law. Since this force is pointwise and local, a smoothed Dirac delta function is used for coupling it with the fluid dynamics. Such methods are widely applied in the cases that interface geometry is complex such as simulation of biological systems, since the geometry representation is decoupled from the general grid structure.

1.3.4. *h*-box method. Here, it is noteworthy to mention the h-box method developed by Berger and LeVeque [**23, 35**] for the approximation of hyperbolic conservation laws on irregular grids. They resolved the time step limitation of a small cell by defining a new region, the h-box, with length h equal to the regular cell size, on the edges of the small cell. Variables on an h-box are derived by the weighted average of the values of cells which that h-box covers; that is a local averaging. Then variables and corresponding fluxes on those edges are found by the solution of the Riemann problem which is initialized by the states derived from the h-box cells instead of the original small cell (Figure 1.6). By this procedure the small cell volume problem is fixed.

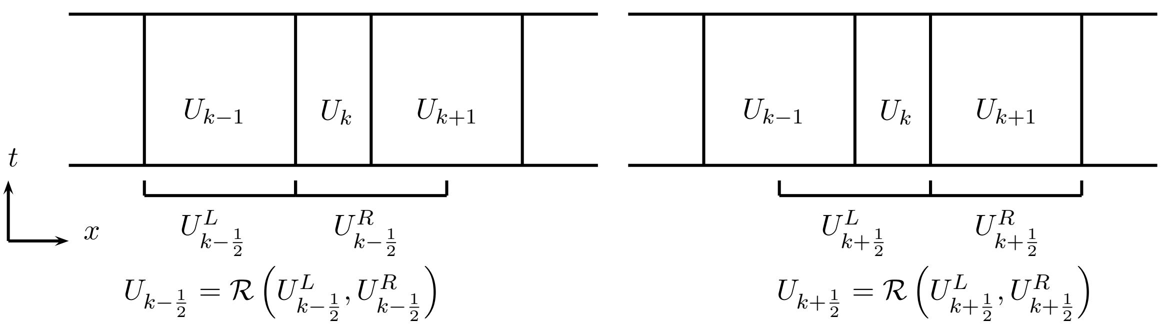

FIGURE 1.6. h-box method to drive the state variables on the edges of an irregular cell.

1.3.5. Mass redistribution methods. A group of Eulerian based methods are the finite volume methods, defined on Cartesian grids. The interface is expressed by its position on the grid or a volume-of-fluid representation and updated using front tracking/capturing approaches. A short summary of the method introduced by Chern and Colella [**26**] brought here as an example of these approaches.

This method is developed for a two-phase fluid with a moving interface. Each step starts with updating the position of the front. The interface divides the domain into two parts and crosses some of the computational cells, making two partitions in those cells. The front speed is needed for finding the position of the front at the next time step. This is found by solving the Riemann problem on the front. Initial values for the Riemann problem are the average values over space of the variable on each side of the front in the crossed cell (Figure 1.7.a). Special attention is given to the case when the volume of one of the partitions is very small, since that may lead to an inaccurate average value. To overcome this problem, the average value is calculated not only on the partition of the cell but also on the nearest neighbor cell. The solution to the Riemann problem also provides the state variables on the front which are needed to derive the continuous flux across the front based on the Rankine-Hugoniot relations.

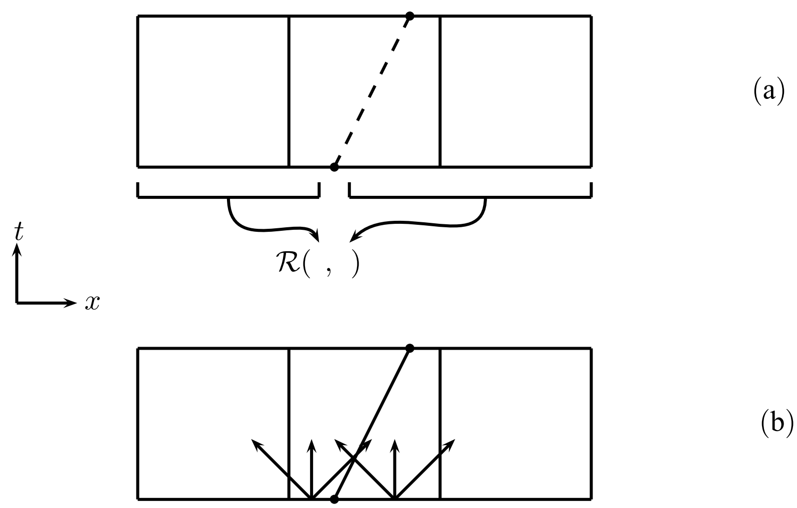

FIGURE 1.7. Chern-Colella method in 1-D, (a) Averaging process for solving Riemann problem. (b) Mass redistribution based on characteristics.

To update the solution away from the front an explicit conservative finite volume method is applied that only needs the fluxes on the faces of the cell. These fluxes are accessible through the solution of the Riemann problem on each face. If a cell is close to the front, average values of variables on the front, as stated before, are used. To update the cells containing the front, the same conservative method may be used while considering that it only applies to a part of the cell. As explained before, this calculation has a division by the size of the partition, which may cause a problem for small partitions. It is possible to use a linear combination of this flux with another stable flux, and use the fraction size as the linear coefficient of conservative to resolve unstable flux difference term. This flux modification makes the method nonconservative. Therefore, the mass difference between the conservative and nonconservative update should be calculated, and redistributed to the nearby cells to maintain the global conservation. Following the physical intuition, the remaining mass is propagated according to the characteristics. This is done by projecting the excess mass on eigenvectors of the system in primitive variables, and modifying the nearby cell values (Figure 1.7.b). Bell et al. [**27**] extended this method by using the unsplit second-order Godunov algorithm coupled with local adaptive mesh refinement.

1.3.6. Effective single phase method. Miller and Puckett developed a numerical method for front capturing in multiple condensed phases [**25**] based on the work of Colella, Glaz and Ferguson [**24**]. They used the volume-of-fluid interface representation. To update the solution on the interface, they used the effective single phase definition for the variables in the cells that contain more than one phase [**24**]. This approach becomes useful in the solution of the approximate Riemann problem on multiphase cell edges. A second-order volume-of-fluid interface reconstruction algorithm was used to update the cells on the front. On a multiphase edge, an edge of a cell which is multiphase or may become multiphase, fluxes for each phase are estimated through a linear approximation to the interface boundary and then finding the contribution of each phase in the upstream direction. The linear approximation is a least-squares optimization to the volume-of-fluid representation of the interface. The conservation equations can be decoupled, to some extent, for each phase considering the appropriate constraints on volume fraction of phases and its temporal change in each computational cell. This leads to a self-consistent advection equation for volume fraction which can be discretized along with the conservation equations for each phase to develop the update procedure. The only concern is when the adjacent phases have very different physical properties, e.g., solid and fluid.

1.3.7. Interface reconstruction method. Miller and Colella developed a method for 3D shock capturing of coupled solid-fluid multiphase systems [**36**]. For each phase, the dynamics are solved separately in parallel to a volume-of-fluid representation for the interfaces. Along with the material properties, all of the state variables are multivalued in a cell that crossed by an interface. This method updates the state variables using a finite volume method for each phase. Therefore, small volume fraction and CFL timestep limitations may occur which are resolved by the algebraic method of Bell et al. [**27**]. The update procedure needs fluxes and variables at the cell edges. A two-step hybrid update and excess mass redistribution is applied to maintain general conservation [**26, 27**].

In this method, away from the interface, appropriate single-phase solvers were applied, e.g., the solver developed by Colella and Woodward [**37**] for fluids and the solver developed by of Miller and Colella [**38**] for solids. In each step, the interface is reconstructed as the best-fit piecewise planar representation to the volume-of-fluid description of the interface. To advance the fluid fraction, an advection equation is used for each phase. In this algorithm, the volume of phases advected through faces of a cell is calculated using a spatially unsplit advection algorithm, and the advection equation is solved in a conservative form. It is also notable that for flux calculation a phase extension done by averaging in two passes with special treatment of vector and tensor properties. This approach extrapolated vector length along with the vector components in each direction, and scaled the extrapolated vector.

1.3.8. A second order accurate front tracking method. Gatti-Bono et al. developed a front tracking method in 1-D [**29**] based on the previous hybrid update and mass redistribution methods [**26, 27, 36**]; a finite volume method with a redistribution algorithm to avoid CFL time-step limitation while maintaining global conservation. In this method extra attention was given to the average of the variables on front cells and their difference with the values at the center of the computational cell, that is, the difference between a center and a centroid value, which may be different to $\mathcal{O}(h)$ in the front cells.

In this method, the front is explicitly known with its position and speed on the grid. In the update procedure, a modified finite volume method, the front speed is evaluated in two steps. A second-order extrapolation of conserved variables was computed and the corresponding Riemann problem was solved to provide the first approximation of front speed and geometry change. Then, variables at the half time step were derived, which gives the fluxes at the half time step. Using a non-conservative update an intermediate approximation of the variables at the next time step are calculated and consequently the front velocity at the next time step is estimated. The high order front velocity at half time step is the average of this speed and the first approximation, permitting a second-order accurate calculation of the geometry. A standard Godunov method with the van Leer limiter was employed to find fluxes at time centers and centroids to advance the variables in time using conservative and nonconservative flux differences. After updating variables, the mass difference between the conservative and nonconservative method are redistributed to the neighbor cells with respect to the characteristics while considering reflection and refraction of the waves on the front. This approach resulted in a method with second order solution error while tracking the material interface, which is promising for showing the effectiveness of using more accurate, geometry based flux calculation near the front.

1.3.9. Embedded boundary method on a Cartesian grid. Although embedded boundary methods do not consider the moving interfaces directly, the approaches they use to handle the complex geometries are inspiring for methods used in free interface problems. For example, Gatti-Bono et al. [**29**] derived their 1-D front tracking method based on the approach of the embedded boundary method of Colella et al. [**28**].

The embedded boundary method developed by Colella et al. [**28**] resolves the small-cell instability problem by using a combination of conservative and non-conservative but stable fluxes on irregular control volumes and redistributing the difference in mass to neighboring cells, similar to front tracking methods [**26, 27**]. The interesting part of this method is the approach to find the fluxes on the covered faces and algorithms for extrapolating state variables on covered faces.

1.3.10. Geometry information extraction. The accuracy of a finite volume method has a strong dependence on the accuracy of flux calculations, and hence the accuracy of the geometrical information of the front. The methods developed by Ligocki et al. [**21**] and Miller and Trebotich [**20**] introduced an algorithm to calculate the geometrical details of an embedded boundary with an arbitrary accuracy. Using an implicit definition of the domain, this method defines irregular computational cells that are cut by the embedded boundary, writes the divergence theorem for the flux terms, and expands the fluxes using the Taylor expansion around the appropriate centroids to have a definition of flux divergence based on integrals of polynomials over the volume and surface of the control volume. Using the divergence theorem recursively it is possible reduce the degree of the moments terms while keeping the accuracy intact. At the bottom of the recursion procedure, the solution of the one-dimensional case is found explicitly using 1-D root-finding.

1.4. Method criteria

As described above, several methods have been developed for modeling a system with a moving interface. They have similarities and differences in every aspect. The goal here is to choose a method for simulation of a moving material interface in 2-D while keeping the possibility of extension to higher dimensions.

Examples of methods with either regular grids or unstructured meshes are covered above. For a method on an unstructured mesh that conforms to the interface, the re-griding procedure to overcome mesh entanglement is a problem which only adds complexity to the algorithm and is not suited for conservative systems. Also for a fixed grid Lagrangian method, the theoretical approach for analysis of error and accuracy is more complicated than for an Eulerian method. In addition, implementing the method on a Cartesian grid is less intricate and a Cartesian grid is more suitable for applying the parallel computation and adaptive mesh refinement. Therefore, it is preferred to use an Eulerian method on a Cartesian grid.

The other objective is to achieve second order global accuracy, at least for smooth solutions. The aforementioned methods generally cover this goal away from the interface. Therefore, the main concern is the accuracy and stability near the interface. Here, a method that achieves its accuracy with a precise treatment of state variables and geometries near the interface is preferred. This is done by Gatti-Bono et al. with special treatment of centered and centroid states in the Cartesian cells in 1-D [**29**]. A strategy is to extend on this approach, knowing the finite volume methods are well suited for the systems of conservation laws. Because of structural similarity to embedded boundary methods of Colella et al. [**28**], it is possible to employ comparable algorithms at irregular cells for flux calculations.

Because of well known issues such as front entanglement, explicit front modeling should be avoided. It is more practical to use level set methods, in which the interface is described implicitly. The level set method has the potential to represent complex geometries and provide accurate geometry information [**20, 21**].

To summarize, the new method is based on the hybrid conservative finite volume of Chern and Colella [**26**], Colella et al. [**28**] and Pember et al. [**39**], as a 2-D extension of the 1-D work of Gatti-Bono et al. [**29**]. The method depends on accurate flux calculation near the front for high-order accuracy. A level set method is applied for front representation and evolution [**40, 41, 42**], while applying the space-time geometry extraction [**20, 21**] for accurate geometry information near the front. This approach results in a second-order front-tracking method. Convergence tests show the second-order convergence for smooth phase solutions, and first-order convergence in the presence of shocks.

CHAPTER 2

Problem definitions and discretization

2.1. Definitions

2.1.1. Gas dynamics. Although the new method is developed for general hyperbolic systems of conservation laws, the presentation here is restricted to the 2-D Euler equations. Following the equation (1.1), the conservative variables and corresponding fluxes for 2-D Euler equations are defined as

$$\mathbf{U} = (\rho,\, \rho u,\, \rho v,\, E)^T, \tag{2.1}$$

$$\mathbf{F}^1 = \left(\rho u,\, \rho u^2 + p,\, \rho uv,\, (E+p)u\right)^T, \tag{2.2}$$

$$\mathbf{F}^2 = \left(\rho v,\, \rho uv,\, \rho v^2 + p,\, (E+p)v\right)^T, \tag{2.3}$$

where ρ is the gas density, u and v are the velocities in the x and y direction respectively, p is the pressure and E is the total energy and defined by the equation of states for ideal polytropic gas,

$$E \equiv \frac{p}{\gamma - 1} + \frac{1}{2}\rho\left(u^2 + v^2\right), \tag{2.4}$$

where γ is the ratio of specific heats. Primitive variables $\mathbf{W} = (\rho,\, u,\, v,\, p)^T$ are used for flux calculations. The equation system in primitive variables is

$$\frac{\partial \mathbf{W}}{\partial t} + \sum_{d=1}^{D} \mathbf{A}^d \frac{\partial \mathbf{W}}{\partial x_d} = 0, \tag{2.5}$$

with $D = 2$ and

$$\mathbf{A}^1 = \begin{pmatrix} u & \rho & 0 & 0 \\ 0 & u & 0 & \frac{1}{\rho} \\ 0 & 0 & u & 0 \\ 0 & \gamma p & 0 & u \end{pmatrix}, \qquad \mathbf{A}^2 = \begin{pmatrix} v & 0 & \rho & 0 \\ 0 & v & 0 & 0 \\ 0 & 0 & v & \frac{1}{\rho} \\ 0 & 0 & \gamma p & v \end{pmatrix}. \tag{2.6}$$

The physical problem domain is discretized with a Cartesian grid with size h in space and Δt in time. That is, cell $\mathbf{i}$ is defined as $\Upsilon_{\mathbf{i}} = [\mathbf{i}, (\mathbf{i}+\mathbf{1})h]$, $\mathbf{i} \in \mathbb{Z}^D$ where $\mathbf{1}$ is a vector of ones. The spatial and space-time control geometries are defined as

$$V_{\mathbf{i},\alpha}(t) = \Upsilon_{\mathbf{i}} \cap \mathbf{\Omega}_\alpha(t), \tag{2.7}$$

$$C^n_{\mathbf{i},\alpha} = V_{\mathbf{i},\alpha}(t) \times [t^n, t^{n+1}], \tag{2.8}$$

where $\alpha \in \{1, 2\}$ is the phase indicator (Figure 2.1). A regular space-time control *volume*[1] is a

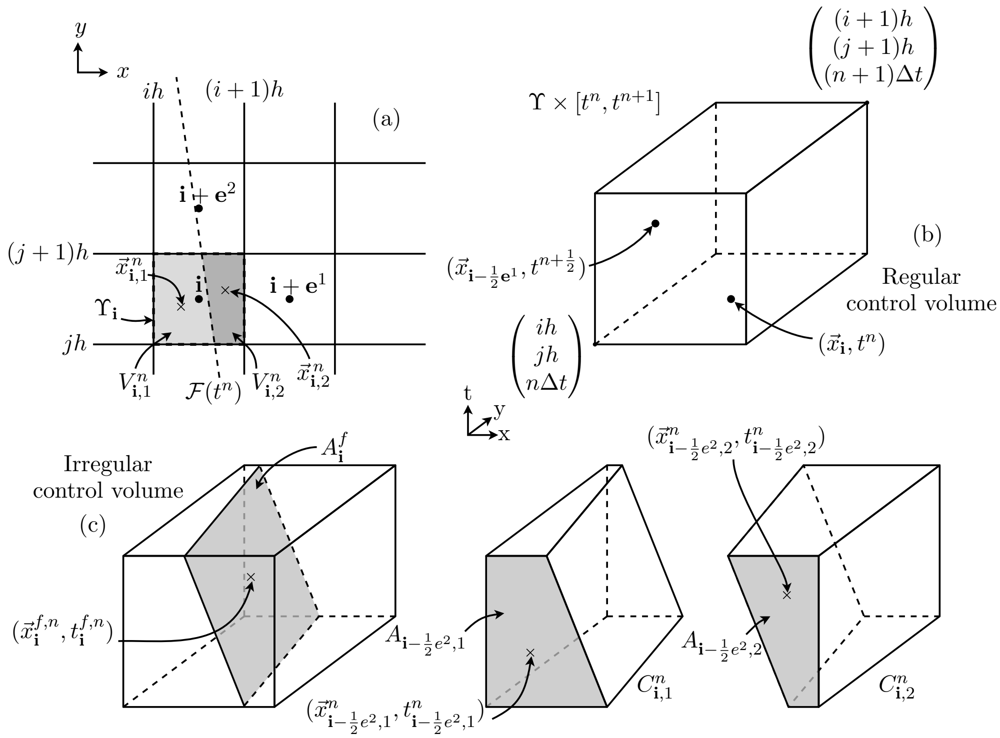

FIGURE 2.1. Geometry discretization and definitions.

rectangular cube in $\mathbb{R}^D \times T$ with $2D$ *faces*[2] and two *cells*[3]. Here, $(\vec{x}^n_{\mathbf{i}}, t^n)$ and $(\vec{x}^{n+1}_{\mathbf{i}}, t^{n+1})$ are the centers of the cell $\Upsilon_{\mathbf{i}}$ at time n and $n+1$ respectively. Centers of the faces are located at $(\vec{x}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d}, t^{n+\frac{1}{2}})$ where $\mathbf{e}^d$ is the unit vector in direction d and sign $\pm$ indicates that the face located on lower or higher side in direction d, and are at time $n + \frac{1}{2}$ (Figure 2.1.b).

[1]An object in $\mathbb{R}^D \times T$
[2]An object in $\mathbb{R}^{D-1} \times T$
[3]An object in $\mathbb{R}^D$

A cell is called irregular if it intersects with the front. $V^n_{\mathbf{i},\alpha}$ and $V^{n+1}_{\mathbf{i},\alpha}$ are cut cells in $\mathbb{R}^D$. Since such cells are fractional the position of the center and centroid are different. The centroid of $V^n_{\mathbf{i},\alpha}$ is located at $(\vec{x}^n_{\mathbf{i},\alpha}, t^n)$. For the faces, if a face only coincides with the Cartesian grid, it is denoted as $A^n_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha}$, and its centroid as $(\vec{x}^n_{\mathbf{i}\pm\frac{1}{2}e^2,\alpha}, t^n_{\mathbf{i}\pm\frac{1}{2}e^2,\alpha})$. If a face coincides with the front it is written as $A^f_{\mathbf{i}}$ with the centroid at $(\vec{x}^{f,n}_{\mathbf{i}}, t^{f,n}_{\mathbf{i}})$ (Figure 2.1.c). Based on the above definitions the cell and face fractions are specified as follows:

$$\Lambda^n_{\mathbf{i},\alpha} = \frac{|V^n_{\mathbf{i},\alpha}|}{h^D}, \qquad a^n_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha} = \frac{|A^n_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha}|}{\Delta t\ h^{D-1}}, \qquad a^{f,n}_{\mathbf{i}} = \frac{|A^{f,n}_{\mathbf{i}}|}{\Delta t\ h^{D-1}}. \tag{2.9}$$

The cell centroid $\vec{x}^n_{\mathbf{i},\alpha}$ is defined as the center of $V^n_{\mathbf{i},\alpha}$,

$$\vec{x}^n_{\mathbf{i},\alpha} = \frac{1}{|V^n_{\mathbf{i},\alpha}|} \int_{V^n_{\mathbf{i},\alpha}} \vec{x}\ \mathrm{d}V, \tag{2.10}$$

and face centroids

$$\left(\vec{x}^n_{\mathbf{i}+\frac{1}{2}\mathbf{e}^d,\alpha}, t^n_{\mathbf{i}+\frac{1}{2}\mathbf{e}^d,\alpha}\right) = \frac{1}{|A^n_{\mathbf{i}+\frac{1}{2}\mathbf{e}^d,\alpha}|} \int_{A^n_{\mathbf{i}+\frac{1}{2}\mathbf{e}^d,\alpha}} (\vec{x}, t)\ \mathrm{d}A\,\mathrm{d}t, \tag{2.11}$$

$$\left(\vec{x}^{f,n}_{\mathbf{i}}, t^{f,n}\right) = \frac{1}{|A^{f,n}_{\mathbf{i}}|} \int_{A^{f,n}_{\mathbf{i}}} (\vec{x}, t)\ \mathrm{d}A\,\mathrm{d}t, \tag{2.12}$$

where $\mathrm{d}V = \mathrm{d}x^D$ and $\mathrm{d}A = \mathrm{d}x^{D-1}$ for $D > 1$. The average space-time normal vector on the front for cell $\mathbf{i}$ is defined as

$$\vec{n}^n_{\mathbf{i},\alpha} = \frac{1}{|A^{f,n}_{\mathbf{i}}|} \int_{A^{f,n}_{\mathbf{i}}} \vec{n}_\alpha\ \mathrm{d}A\,\mathrm{d}t, \tag{2.13}$$

where $\vec{n}_\alpha$ is the outward normal vector for phase α.

2.1.2. Front dynamics. A level set function is used to represent the front. The level set $\phi(\vec{x}, t)$ is a continuous function with

$$\begin{cases} \phi(\vec{x}, t) < 0 & \text{in } \mathbf{\Omega}_1(t) \\ \phi(\vec{x}, t) = 0 & \text{on } \mathcal{F}(t) \\ \phi(\vec{x}, t) > 0 & \text{in } \mathbf{\Omega}_2(t). \end{cases} \tag{2.14}$$

Therefore, the interface is represented by the zero level set $\mathcal{F}(t) \equiv \phi_0 = \{\vec{x}|\phi(\vec{x},t) = 0\}$. The level set is updated using the level set equation [**18**],

$$\phi_t + \vec{\nu}_{ext} \cdot \nabla\phi = 0, \tag{2.15}$$

where $\vec{\nu}_{ext}$ is the extended velocity and defined in $\mathbb{R}^D$. The extended velocity represents the movement of the whole level set function and is defined to match the front velocity on the front,

$$\vec{\nu}_{ext} = \vec{\nu}^f \quad \text{on } \mathcal{F}(t). \tag{2.16}$$

The level set function is used for calculating accurate geometric features of the front. Therefore, it is helpful to define and maintain ϕ as a smooth function. Level set function ϕ is initialized as a signed distance function that satisfies (2.14),

$$\phi(\vec{x},0) = \pm l(\vec{x}), \tag{2.17}$$

where l is the distance of point $\vec{x}$ to the front $\mathcal{F}(0)$. The level set equation (2.15) moves the zero level set correctly but may change the level set function away from being a distance function. Level set function ϕ is kept as a distance function, that is $|\nabla\phi| = 1$, by carefully generating the extended velocity [**43**], and by using a redistancing procedure [**42**]. The details are described in §3.4 and §3.5.

2.2. Discretization

2.2.1. Gas dynamics. The conservation equation may be written in the following compact form

$$\left(\nabla, \frac{\partial}{\partial t}\right) \cdot \left(\vec{\mathbf{F}}, \mathbf{U}\right) = 0, \tag{2.18}$$

or in the integral form

$$\int_{C^n_{i,\alpha}} \left(\nabla, \frac{\partial}{\partial t}\right) \cdot \left(\vec{\mathbf{F}}, \mathbf{U}\right) \,\mathrm{d}V\,\mathrm{d}t = 0. \tag{2.19}$$

To derive the finite volume method, the divergence theorem is applied to (2.19) in space and time over the space-time control volume $C^n_{i,\alpha}$,

$$\oint_{\partial C^n_{\mathbf{i},\alpha}} \vec{n}_\alpha \cdot \left(\vec{\mathbf{F}}, \mathbf{U}\right) \, \mathrm{d}A = 0, \tag{2.20}$$

where $\vec{n}_\alpha$ is the outward space-time normal vector on the surface of control volume $\partial C^n_{\mathbf{i},\alpha}$ and $\mathrm{d}A$ is in $\mathbb{R}^{D-1} \times T$. Separating this integral to three parts based on the surface types results in

$$\int_{\partial C^n_{\mathbf{i},\alpha}\big|_{t=t^n \,\vee\, t=t^{n+1}}} \vec{n}_\alpha \cdot \left(\vec{\mathbf{F}}, \mathbf{U}\right) \, \mathrm{d}A + \int_{\partial C^n_{\mathbf{i},\alpha} \cap (\Upsilon_{\mathbf{i}}(t) \times [t^n, t^{n+1}])} \vec{n}_\alpha \cdot \left(\vec{\mathbf{F}}, \mathbf{U}\right) \, \mathrm{d}A \\ + \int_{\partial C^n_{\mathbf{i},\alpha} \cap \mathcal{F}(t)} \vec{n}_\alpha \cdot \left(\vec{\mathbf{F}}, \mathbf{U}\right) \, \mathrm{d}A = 0. \tag{2.21}$$

Discretizing in space and time gives

$$|V^{n+1}_{\mathbf{i},\alpha}| \mathbf{U}^{n+1}_{\mathbf{i},\alpha} - |V^n_{\mathbf{i},\alpha}| \mathbf{U}^n_{\mathbf{i},\alpha} \\ + \sum_{\pm, d=1}^{D} \left(\pm |A^n_{\mathbf{i} \pm \frac{1}{2}\mathbf{e}^d, \alpha}| \mathbf{F}^{cd,n}_{\mathbf{i} \pm \frac{1}{2}\mathbf{e}^d, \alpha} \right) + |A^{f,n}_{\mathbf{i}}| \vec{n}^n_{\mathbf{i},\alpha} \cdot \left(\vec{\mathbf{F}}^{f,n}_{\mathbf{i},\alpha}, \mathbf{U}^{f,n}_{\mathbf{i},\alpha} \right) = \mathcal{O}(h^{D+1} \Delta t). \tag{2.22}$$

Here, $\mathbf{U}^n_{\mathbf{i},\alpha}$ represents the average value of $\mathbf{U}(\vec{x}, t)$ in $V^n_{\mathbf{i},\alpha}$,

$$\mathbf{U}^n_{\mathbf{i},\alpha} = \frac{1}{\Lambda^n_{\mathbf{i},\alpha} h^D} \int_{V^n_{\mathbf{i},\alpha}} \mathbf{U}(\vec{x}, n\Delta t) \, \mathrm{d}V + \mathcal{O}\left(\frac{h^2}{\Lambda^n_{\mathbf{i},\alpha}} \right), \qquad \forall \Lambda^n_{\mathbf{i},\alpha} > 0, \tag{2.23}$$

and is calculated at $\vec{x}^n_{\mathbf{i},\alpha}$, the centroid of $V^n_{\mathbf{i},\alpha}$. $\mathbf{F}^{cd,n}_{\mathbf{i} \pm \frac{1}{2}\mathbf{e}^d, \alpha}$ is the average flux at the space-time centroid of the corresponding faces in direction d, and $\vec{\mathbf{F}}^{f,n}_{\mathbf{i},\alpha}$ and $\mathbf{U}^{f,n}_{\mathbf{i},\alpha}$ are the average front flux and state variable at the centroid of the front, respectively. Following the free-stream-preserving discretization of Pember et al. [**39**] the calculation on the front may be written in terms of fractional face areas on the Cartesian grid instead of relying on an estimation of the front area itself:

$$|V^{n+1}_{\mathbf{i},\alpha}| \mathbf{U}^{n+1}_{\mathbf{i},\alpha} - |V^n_{\mathbf{i},\alpha}| \mathbf{U}^n_{\mathbf{i},\alpha} + \sum_{\pm, d=1}^{D} \left(\pm |A^n_{\mathbf{i} \pm \frac{1}{2}\mathbf{e}^d, \alpha}| \mathbf{F}^{cd,n}_{\mathbf{i} \pm \frac{1}{2}\mathbf{e}^d, \alpha} \right) \\ - \sum_{d=1}^{D} (|A^n_{\mathbf{i} + \frac{1}{2}\mathbf{e}^d, \alpha}| - |A^n_{\mathbf{i} - \frac{1}{2}\mathbf{e}^d, \alpha}|) \mathbf{F}^{f,n}_{\mathbf{i},d,\alpha} - (|V^{n+1}_{\mathbf{i},\alpha}| - |V^n_{\mathbf{i},\alpha}|) \mathbf{U}^{f,n}_{\mathbf{i},\alpha} = \mathcal{O}(h^{D+1} \Delta t). \tag{2.24}$$

Dividing by h^D and substituting cell and face areas with unitless cell and face fractions gives

$$\Lambda_{\mathbf{i},\alpha}^{n+1}\mathbf{U}_{\mathbf{i},\alpha}^{n+1} - \Lambda_{\mathbf{i},\alpha}^{n}\mathbf{U}_{\mathbf{i},\alpha}^{n} + \frac{\Delta t}{h}\sum_{\pm,d=1}^{D}\left(\pm a_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha}^{n}\left(\mathbf{F}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha}^{cd,n} - \mathbf{F}_{\mathbf{i},d,\alpha}^{f,n}\right)\right)$$
$$- \left(\Lambda_{\mathbf{i},\alpha}^{n+1} - \Lambda_{\mathbf{i},\alpha}^{n}\right)\mathbf{U}_{\mathbf{i},\alpha}^{f,n} = \mathcal{O}(h\Delta t). \quad (2.25)$$

An interim conserved variable $\mathbf{U}^{n,n+1}$ is defined as the conserved variable evaluated at time t^n and at the cell centroid position of time t^{n+1},

$$\mathbf{U}_{\mathbf{i},\alpha}^{n,n+1} = \mathbf{U}(\vec{x}_{\mathbf{i},\alpha}^{n+1}, t^n). \quad (2.26)$$

Using (2.26), an explicit update equation form of (2.25) is

$$\mathbf{U}_{\mathbf{i},\alpha}^{n+1} = \mathbf{U}_{\mathbf{i},\alpha}^{n,n+1} - \Delta t\mathbf{DF}_{\mathbf{i},\alpha}^{n,C}, \quad (2.27)$$

where $\mathbf{DF}_{\mathbf{i},\alpha}^{n,C}$ is the conservative flux difference defined as

$$\mathbf{DF}_{\mathbf{i},\alpha}^{n,C} = \frac{\Lambda_{\mathbf{i},\alpha}^{n+1}\mathbf{U}_{\mathbf{i},\alpha}^{n,n+1} - \Lambda_{\mathbf{i},\alpha}^{n}\mathbf{U}_{\mathbf{i},\alpha}^{n}}{\Lambda_{\mathbf{i},\alpha}^{n+1}\Delta t}$$
$$+ \frac{1}{\Lambda_{\mathbf{i},\alpha}^{n+1}h}\left(\sum_{\pm,d=1}^{D}\pm\left(a_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha}^{n}\left(\mathbf{F}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha}^{cd,n} - \mathbf{F}_{\mathbf{i},d,\alpha}^{f,n}\right)\right)\right.$$
$$\left.-\frac{h}{\Delta t}\left(\Lambda_{\mathbf{i},\alpha}^{n+1} - \Lambda_{\mathbf{i},\alpha}^{n}\right)\mathbf{U}_{\mathbf{i},\alpha}^{f,n}\right). \quad (2.28)$$

Note that $\mathbf{DF}_{\mathbf{i},\alpha}^{n,C}$ may become unstable for small cell fraction $\Lambda_{\mathbf{i},\alpha}^{n+1}$. Therefore a nonconservative but stable flux difference is introduced,

$$\mathbf{DF}_{\mathbf{i},\alpha}^{n,NC,cr} = \frac{1}{h}\sum_{\pm,d=1}^{D}\left(\pm\mathbf{F}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha}^{cr,n}\right), \quad (2.29)$$

where $\mathbf{F}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha}^{cr,n}$ is the flux evaluated on the face centers. This calculation results in a cell center estimate of the nonconservative flux. Then, it is extrapolated to the cell centroid position at time t^{n+1}, which gives $\mathbf{DF}_{\mathbf{i},\alpha}^{n,NC}$. A linear combination of $\mathbf{DF}_{\mathbf{i},\alpha}^{n,C}$ and $\mathbf{DF}_{\mathbf{i},\alpha}^{n,NC}$ is used for the update equation to avoid the small cell problem of the finite volume methods [**44**]. By picking $\Lambda_{\mathbf{i},\alpha}^{n+1}$ as the linear coefficient of $\mathbf{DF}^{n,C}_{\mathbf{i},\alpha}$ a stable update method is achieved:

$$\mathbf{U}^{n+1}_{\mathbf{i},\alpha} = \mathbf{U}^{n,n+1}_{\mathbf{i},\alpha} - \Delta t \left[\Lambda^{n+1}_{\mathbf{i},\alpha}\, \mathbf{DF}^{n,C}_{\mathbf{i},\alpha} + \left(1 - \Lambda^{n+1}_{\mathbf{i},\alpha}\right) \mathbf{DF}^{n,NC}_{\mathbf{i},\alpha}\right]. \tag{2.30}$$

To maintain the conservation property of the method the mass difference between mass increment of the conservative update method (2.27) and hybrid update method (2.30) is calculated

$$\begin{aligned}\delta \mathbf{M}^{n}_{\mathbf{i},\alpha} &= \Lambda^{n+1}_{\mathbf{i},\alpha}\left(\left[\mathbf{U}^{n+1}_{\mathbf{i},\alpha} - \mathbf{U}^{n,n+1}_{\mathbf{i},\alpha}\right]_{\text{Conservative}} - \left[\mathbf{U}^{n+1}_{\mathbf{i},\alpha} - \mathbf{U}^{n,n+1}_{\mathbf{i},\alpha}\right]_{\text{Hybrid}}\right),\\ \delta \mathbf{M}^{n}_{\mathbf{i},\alpha} &= \Delta t \Lambda^{n+1}_{\mathbf{i},\alpha}\left(1 - \Lambda^{n+1}_{\mathbf{i},\alpha}\right)\left(\mathbf{DF}^{n,NC}_{\mathbf{i},\alpha} - \mathbf{DF}^{n,C}_{\mathbf{i},\alpha}\right),\end{aligned} \tag{2.31}$$

and the excess mass is redistributed to the appropriate neighbor cells of cell $\mathbf{i}$.

2.2.2. Front dynamics. The level set equation (2.15) is discretized in space using the WENO method for Hamilton-Jacobi equations by Jiang and Peng [**41**]. The level set function at any time step is only needed in a band around the zero level set (Figure 2.2). The local level set method of Peng et al. [**40**] is applied to reduce the computational work of updating the level set function. This means the update method applies to the cells in a band around the zero level set. The update band should be wide enough to provide enough information to calculate the gradient for few cell around zero level set.

For time discretization a two-step Adams-Bashforth method is used. For the ODE system

$$\phi_t = L(\phi), \qquad \phi(\vec{x}, 0) = \phi_0, \tag{2.32}$$

the update algorithm at time step n is

$$\begin{cases} \tilde{\phi}^{n+1} = \phi^n + \Delta t L(\phi^n) \\ (D\phi)^n = \tilde{\phi}^{n+1} - \phi^n \\ \phi^{n+1} = \phi^n + \frac{3}{2}(D\phi)^n - \frac{1}{2}(D\phi)^{n-1}. \end{cases} \tag{2.33}$$

The above method is second-order accurate in time with the initial starting value $(D\phi)^{-1} = (D\phi)^0$. For the level set update equation the L operator is defined as $L(\phi) = -\vec{v}_{ext} \cdot \nabla\phi$.

The algorithm to calculate the front velocity on the front centroids at time t^n is described in §3.3 . The extension method to find the extended velocity $\vec{\nu}_{ext}$ in the cell centers of the update band is explained in §3.4.

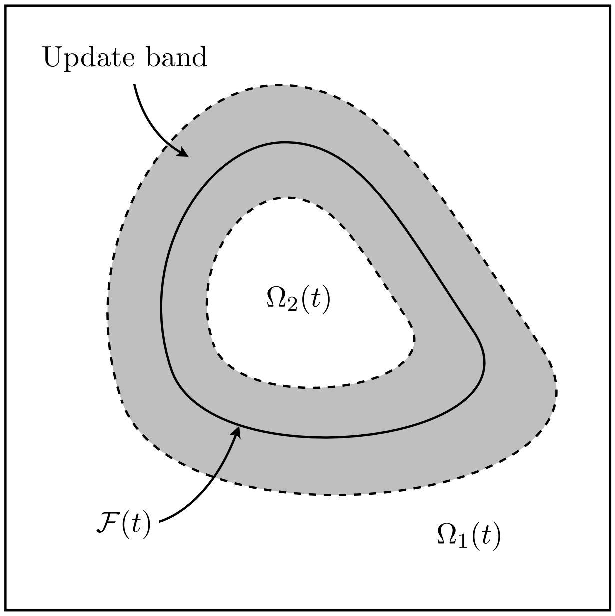

FIGURE 2.2. The local level set method: calculation are done only on the cells in the update band.

CHAPTER 3

Algorithm

3.1. General method

At the beginning of each time step it is assumed that the following data are available.

- Conserved variable $\mathbf{U}^n_{\mathbf{i},\alpha}$ for each phase evaluated at cell centroid, $\vec{x}^n_{\mathbf{i},\alpha}$.
- Front position $\mathcal{F}(t^n)$ and corresponding level set ϕ^n at time step n.
- Geometrical features: cell, face and interface centroids, cell fractions and interface normal vectors for control volumes between time t^n and t^{n+1}. The geometry algorithm of Miller and Trebotich [**20**] is applied to calculate these terms. Depending on the required accuracy for geometrical features, a number of level set functions at previous time steps are needed.

The goal is to find a second-order estimate of the interface position $\mathcal{F}\left(t^{n+1}\right)$ and conserved variables $\mathbf{U}^{n+1}_{\mathbf{i},\alpha}$ at time t^{n+1}. The algorithm for one time step is as follows:

(1) For each phase, extrapolate cell centered variables from cell centroid data and convert to primitive variables $\mathbf{W}^n_{\mathbf{i},\alpha}$. A linear extrapolation is sufficient. See §3.2 for details.

(2) Extrapolate from $\mathbf{W}^n_{\mathbf{i},\alpha}$ to the front centroids at time n and solve the 1-D Riemann problem in the front-normal direction. This procedure includes rotating the system of equations on the front to the front-normal direction and solving the Riemann problem in that direction. See §3.2 and §3.3 for details.

(3) Calculate $\vec{v}^f$ from the solution of the Riemann problem on the front and extend the velocity field to a band around the zero level set. See §3.4 for details.

(4) Update the level set function to ϕ^{n+1} using the method (2.33).

(5) Calculate geometrical features using $\phi^{n+1}, \phi^n, \dots \phi^{n-k}$ where k determines the number of level set functions used from previous time step calculations. Find the cell, face and front centroids, normal vector and cell and face fractions needed to calculate the conservative flux difference and hybrid update equation.

(6) If a cell may become *uncovered*,

$$\left\{ \Upsilon_{\mathbf{i}} \cap \mathbf{\Omega}_a(t^n) = \varnothing,\ \Upsilon_{\mathbf{i}} \cap \mathbf{\Omega}_a(t^{n+1}) \neq \varnothing \right\},$$

determine its value $\mathbf{W}^n_{\mathbf{i},\alpha}$ by extrapolation. See §3.2 for details.

(7) For all cells in which $\mathbf{W}^n_{\mathbf{i}}$ is known, find high-order slopes using symmetric fourth-order stencils where possible, and one-sided stencils otherwise. Apply van Leer slope limiting. See §3.6 for details.

(8) For all cells in which $\mathbf{W}^n_{\mathbf{i},\alpha}$ is known, use upwind-filtered characteristic tracing to find half time step face center states $\mathbf{W}^{n+1/2}_{\mathbf{i},\pm,d}$. See §3.7.1 for details.

(9) If a given face is exterior, that is it has only one face state, determine the missing face state by extrapolation (See §3.7.2 for details) and solve the Riemann problem on all faces.

(10) Include transverse flux terms (corner coupling) [**45, 46**]. Details are described in §3.7.1. This is straight forward for interior faces. However the data may not be available to do corner coupling for an exterior face. So, after updating the interior faces, redo the extrapolation for missing side of exterior faces as explained in the previous step.

(11) Solve the Riemann problem on all face centers and compute the cell-centered nonconservative flux difference.

(12) Extrapolate the nonconservative flux difference from cell centers to cell centroids. See §3.2 for details.

(13) If a face is irregular, find the left and right states on the face centroid using the space-time extrapolation from face center values and solve the Riemann problem. See §3.7.3 for details.

(14) Extrapolate to the front centroids from the cell centroid state from each phase, and solve the Riemann problem to calculate the state variables and flux terms on the front. See §3.7.4 for details.

(15) Compute the conservative flux difference.

(16) Extrapolate $\mathbf{U}^{n,n+1}_{\mathbf{i},\alpha}$ from cell centroid data. See §3.2 for details.

(17) Calculate $\mathbf{U}^{n+1}_{\mathbf{i},\alpha}$ using the hybrid update equation (2.30).

(18) Balance the excess mass between phases on the front and redistribute the balanced excess mass to neighbor cells in each phase. See §3.8 for details.

The above algorithm is done for each time step. The CFL condition is considered to pick a stable value for Δt. The Courant number is defined as

$$\vartheta = \frac{\Delta t}{h} \max_{\mathbf{i},p} |\lambda_{\mathbf{i}}^p|, \tag{3.1}$$

where $|\lambda_{\mathbf{i}}^p|$ is the p-th wavespeed of the cell center primitive variables in control volume $\mathbf{i}$. The time step is chosen to satisfy

$$\vartheta \leq 1, \tag{3.2}$$

in each timestep.

3.2. Local grid interpolation

The interpolation steps used in the algorithm are linear estimates based on the local grid data. To interpolate the value of function G at target point $\vec{x}_t$ based on the support data point at $\vec{x}_s$, the Taylor expansion of $G(\vec{x}_s)$ centered at $\vec{x}_t$ is used,

$$G(\vec{x}_s) \approx G(\vec{x}_t) + \nabla G(\vec{x}_t)(\vec{x}_s - \vec{x}_t). \tag{3.3}$$

By gathering enough support point data around the target, a least-squares problem is created to solve for $G(\vec{x}_t)$,

$$\begin{pmatrix} 1 & (\vec{x}_1 - \vec{x}_t)/h \\ \vdots & \vdots \\ 1 & (\vec{x}_m - \vec{x}_t)/h \end{pmatrix} \begin{pmatrix} G(\vec{x}_t/h) \\ \nabla G(\vec{x}_t/h) \end{pmatrix} = \begin{pmatrix} G(\vec{x}_1/h) \\ \vdots \\ G(\vec{x}_m/h) \end{pmatrix}, \tag{3.4}$$

where $\vec{x}_1, \ldots, \vec{x}_m$ are support points. The support points are chosen based on the nearest cell criteria to the target cell position. The algorithm starts with the most compact stencil possible and try solve the above least-square system with QR decomposition (Figure 3.1). More support data points are added if values on the diagonal of R are too small and therefore error bound is too big for solution to be acceptable. The heuristic cut-off value that used for this limit in the simulations is 10^{-10} for each element and 10^{-15} for multiplication of all diagonal elements of R.

The above algorithm is used for steps 1, 2, 6, 12 and 16 of the algorithm described in §3.1. Note that the neighbor cell data is not used if it is an irregular cell for steps 1 and 16. However the neighbor cell data is used for support points in steps 2, 6 and 12 of the algorithm regardless of being from a regular or an irregular cell.

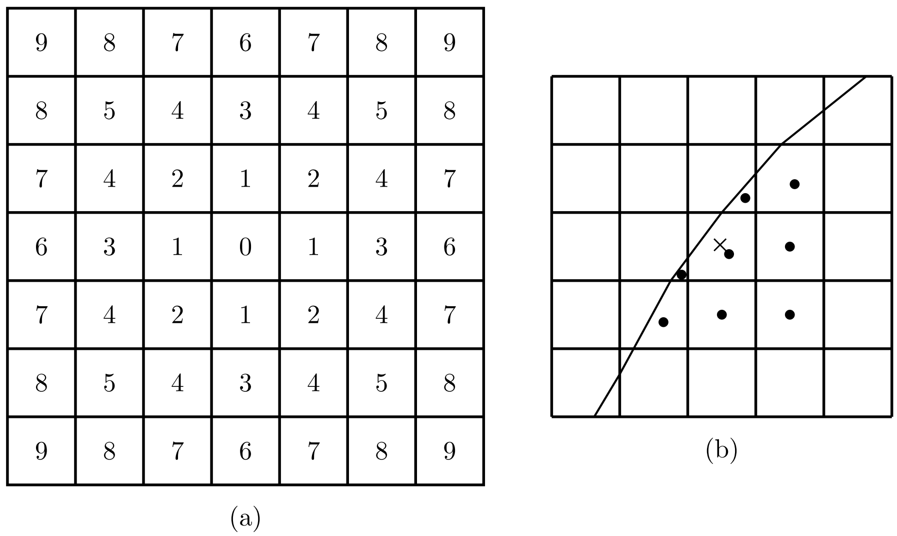

FIGURE 3.1. (a) Sets of nearest neighbor cell stencils. Each number represent a group of neighbor cells with the same approximate distance to the center cell. (b) Interpolation from cell centroid data to cell center: Target point (the cell center) is shown with $\times$ and support points are shown with $\bullet$. In this example only set 1 and 2 of nearest neighbor cells are used.

3.3. Solution on the front

The state variables on the front are needed to calculate the front speed and the flux on the front. The states variables from each phase are extrapolated to the front centroids where the Riemann problem is solved in the front normal direction to determine the states on the front.

Assume $\mathbf{W}_\alpha = (\rho_\alpha,\, u_\alpha,\, v_\alpha,\, p_\alpha)^T$ for $\alpha = 1, 2$ are the extrapolated states and $\vec{n}_s = (n_x\ n_y)^T$ is the spatial normal vector on the front. The rotation matrix is defined as

$$\mathbf{R} = \begin{pmatrix} n_x & n_y \\ -n_y & n_x \end{pmatrix}, \tag{3.5}$$

where $\mathbf{R} \cdot \vec{z}$ rotates $\vec{z}$ from the laboratory frame to the coordinate system in which first axis is aligned in the front normal direction. Vector components of the state variables, velocity in case of

Euler equation, are then rotated using this rotation matrix $\mathbf{R}$ (Figure 3.2)

$$(\widehat{u}_\alpha,\, \widehat{v}_\alpha)^T = \mathbf{R} \cdot (u_\alpha,\, v_\alpha)^T\,, \quad \alpha = 1, 2. \tag{3.6}$$

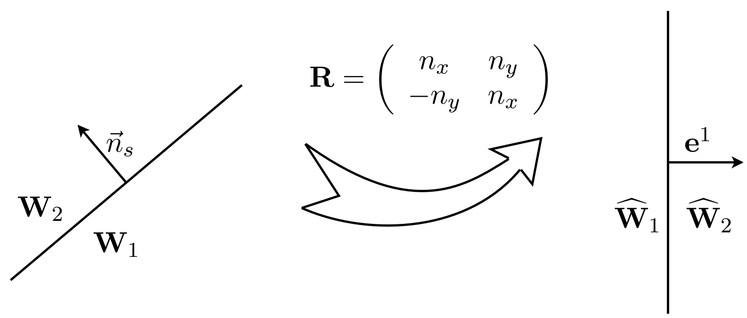

FIGURE 3.2. Laboratory frame rotation on the front

The Riemann problem is solved in the x direction with $\mathbf{W}_L = \widehat{\mathbf{W}}_1$ and $\mathbf{W}_R = \widehat{\mathbf{W}}_2$ to calculate the neighboring states on the contact discontinuity $\mathbf{W}_L^*$ and $\mathbf{W}_R^*$, see Figure 3.3 and Appendix B,

$$\left((\rho_L^*,\, u^*,\, v_L^*,\, p^*)^T , (\rho_R^*,\, u^*,\, v_R^*,\, p^*)^T \right) = \mathfrak{R}_x^f \left((\rho_1,\, \widehat{u}_1,\, \widehat{v}_1,\, p_1)^T , (\rho_2,\, \widehat{u}_2,\, \widehat{v}_2,\, p_2)^T \right). \tag{3.7}$$

Vector components of the solution are rotated back to the laboratory frame by multiplying by $\mathbf{R}^{-1} = \mathbf{R}^T$,

$$(u_1^*,\, v_1^*)^T = \mathbf{R}^{-1} \cdot (u^*,\, v_L^*)^T\,, \tag{3.8}$$

$$(u_2^*,\, v_2^*)^T = \mathbf{R}^{-1} \cdot (u^*,\, v_R^*)^T\,. \tag{3.9}$$

This procedure is used twice in each iteration of the method. First, it is applied to find the state variables on the front at time n (step 2 of the algorithm §3.1). The solution of the Riemann problem is used to determine the front velocity for updating the level set equation. The normal velocity component is unique for the solution on front. The tangential velocites are different but they do not contribute in the evolution of the level set equation. The average value is picked for the calculation

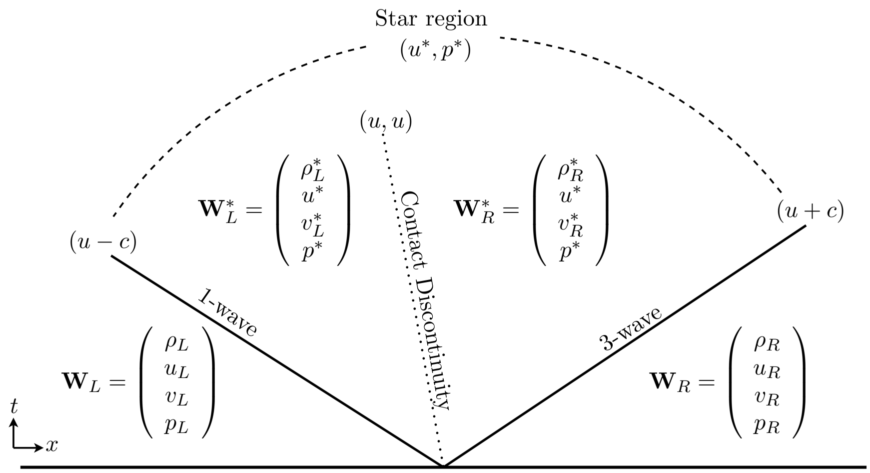

FIGURE 3.3. Structure of the solution of the 2-D split Riemann Problem. The solution has three waves in regard to the eigenvalues $u-c$, u and $u+c$. The middle wave is always a contact discontinuity and 1-wave and 3-wave are either shock or rarefaction waves. The solution is separated into four regions. The left-most and right-most regions preserve the initial states. The two middle regions (star region) connect the two side regions with some discontinuities on each wave family. The contact discontinuity separates the two middle regions with a jump in density and tangential velocity while keeping normal velocity and pressure constant.

of the front velocity on the corresponding front centroid,

$$\vec{v}^f = \mathbf{R}^{-1} \cdot \left(u^*, \frac{v_L^* + v_R^*}{2}\right)^T. \tag{3.10}$$

Second, this procedure is applied to find the state variables on the front centroid $(\vec{x}_{\mathbf{i}}^{f,n}, t_{\mathbf{i}}^{f,n})$ to calculate the flux between phases (step 14 of the algorithm §3.1). Normal velocity at the star region u^* is replaced with the interface velocity that is calculated from the geometry information of moving front,

$$s^f = \frac{-n_t}{\sqrt{n_x^2 + n_y^2}}, \quad \vec{n}_{\mathbf{i}}^n = (n_x, n_y, n_t)^T. \tag{3.11}$$

With this procedure, it is ensured that the cancellation due to the moving front in the Rankine-Hugoniot jump condition is done completely and the method is conservative. See Appendix C for more details.

3.4. Velocity extension

To update the level set equation, the velocity field is needed in a band around the zero level set. The solution on the front, see §3.3, gives the velocity on front centroids, which should be extended to the center of the cells in the band around the zero level set.

To keep ϕ a signed distance function after updating the level set equation one needs to impose the following condition on the velocity [**43**]:

$$\nabla(\vec{\nu}_{ext,d}) \cdot \nabla\phi = 0, \quad d = 1, \ldots, D, \tag{3.12}$$

which means that velocity is constant along the level set gradient. Here, the extension algorithm developed by Peng et al. [**40**] and Zhao et al. [**47**] is used. To extend the quantity q with condition $\nabla q \cdot \nabla\phi = 0$,

$$q_t + S(\phi)\frac{\nabla\phi}{|\nabla\phi|} \cdot \nabla q = 0 \tag{3.13}$$

is solved as a internal boundary value problem, where $S(\phi)$ is the sign function defined as

$$S(\phi) = \begin{cases} -1 & \text{if } \phi < 0, \\ 0 & \text{if } \phi = 0, \\ +1 & \text{if } \phi > 0, \end{cases} \tag{3.14}$$

and approximated by

$$S(\phi)_\delta = \frac{\phi}{\sqrt{\phi^2 + \delta^2}}, \tag{3.15}$$

where δ is a small smoothing parameter which is taken as h. The characteristics of (3.13) are normal to the front and pointing outward from the front, that is, it is only sufficient to initiate q on a narrow band around ϕ_0 and then solve (3.13).

To initialize a narrow band around the zero level set, cells that have at least one neighbor cell with opposite level set sign amongst their $3^D - 1$ neighbors are picked. Applying the extension condition (3.12) means the extended velocity field does not change in the direction of the level set gradient. Therefore, if a projection from a point outside the front, such as the center of a cell in the initialization band $\vec{x}_c$ (see Figure 3.4), in the direction of the level set gradient toward the front reaches a point on the front such as $\vec{x}_f$,

$$\vec{x}_f \approx \vec{x}_c - \phi(\vec{x}_c)\frac{\nabla\phi(\vec{x}_c)}{|\nabla\phi(\vec{x}_c)|}, \tag{3.16}$$

the extension condition (3.12) recovers

$$\vec{\nu}(\vec{x}_f) = \vec{\nu}(\vec{x}_c). \tag{3.17}$$

To find the velocity at $\vec{x}_f$, a linear interpolation is applied using the velocity values on the front centroids in the 3^D cells near $\vec{x}_f$. If m front centroid points reside in the vicinity of $\vec{x}_f$, the least-squares equation system (3.18) is solved for $\nu_d(\vec{x}_f)$,

$$\begin{pmatrix} 1 & \vec{x}_1 - \vec{x}_f \\ \vdots & \vdots \\ 1 & \vec{x}_m - \vec{x}_f \end{pmatrix} \begin{pmatrix} \nu_d(\vec{x}_f) \\ \nabla\nu_d(\vec{x}_f) \end{pmatrix} = \begin{pmatrix} \nu_d(\vec{x}_1) \\ \vdots \\ \nu_d(\vec{x}_m) \end{pmatrix}, \quad d = 1, \ldots, D. \tag{3.18}$$

The singular value decomposition (SVD) method is used to solve the interpolation equation system. Since, it is possible that support points be placed on a line, and make system (3.18) underdetermined.

The extension equation (3.13) is solved by iteration as described by Peng et al. [**40**] while applying the high-order WENO discretization of Jiang and Peng [**41**] to increase the accuracy of the method and decrease the number of needed iterations.

3.5. Redistancing

The level set method inherently has some diffusive nature which may cause deviation from a distance function or loss of area or volume. Sussman et al. [**48**] introduced the redistancing equation in artificial time τ

$$\phi_\tau + S(\tilde{\phi})(|\nabla\phi| - 1) = 0, \tag{3.19}$$

where S is the sign function and $\tilde{\phi} = \phi(\vec{x}, \tau = 0)$. The redistancing equation (3.19) is solved using the improved Hamilton-Jacobi WENO algorithm with Godunov approximation in space and the third-order TVD Runge-Kutta method in time [**42**].

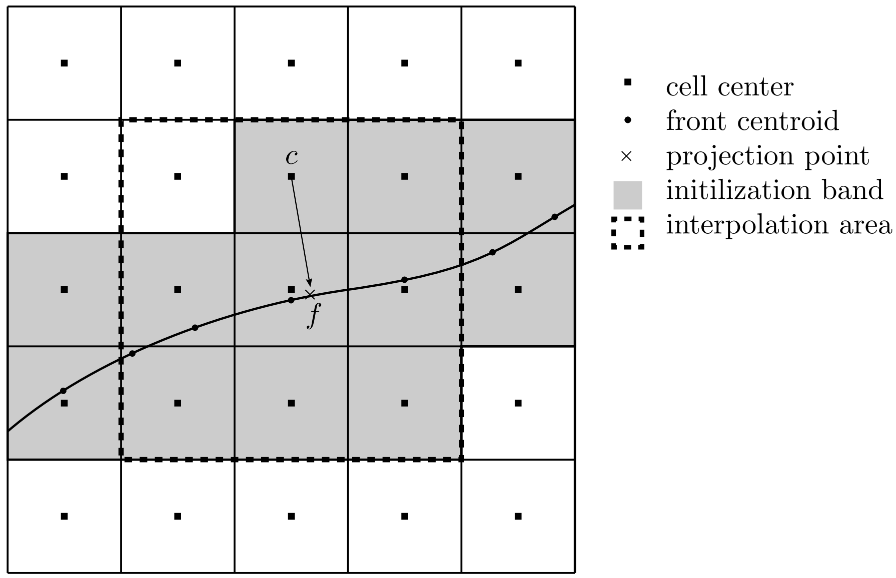

FIGURE 3.4. Initialization of velocity field in a narrow band around zero level set.

3.6. Slope calculation

The slopes are calculated in cell center primitive variables similar to the procedure by Colella et al. [**28**]. For a cell that has at least two neighbor cells on each side, the fourth order centered difference is calculated. A lower order differences is used for a cell which has fewer available neighbor cells. The van Leer limiter $\Delta^{vL}(\delta\mathbf{W}_C, \delta\mathbf{W}_L, \delta\mathbf{W}_R)$ is applied when the center, left and right differences are available. It is defined on the expansion in characteristic variables for centered and one-sided differences:

$$
\begin{aligned}
&\Delta^{vL} = \sum_k \zeta^k \mathbf{r}^k, \\
&\zeta^k = \begin{cases} \min(2|\zeta_L^k|,\, 2|\zeta_R^k|,\, |\zeta_C^k|) & \text{if } \zeta_L^k \cdot \zeta_R^k > 0, \\ 0 & \text{otherwise,} \end{cases} \\
&\zeta_L^k = \mathbf{l}^k.\delta\mathbf{W}_L, \\
&\zeta_R^k = \mathbf{l}^k.\delta\mathbf{W}_R, \\
&\zeta_C^k = \mathbf{l}^k.\delta\mathbf{W}_C,
\end{aligned}
\tag{3.20}
$$

where $\mathbf{l}^k$ and $\mathbf{r}^k$ are the corresponding left and right orthonormal eigenvectors of $\mathbf{W}^n_{\mathbf{i},\alpha}$. The second order center difference and first and second order one sided differences in direction d are defined as

$$\begin{aligned}
\Delta^C \mathbf{W}_{\mathbf{i},\alpha} &= \frac{1}{2}(\mathbf{W}^n_{\mathbf{i}+\mathbf{e}^d,\alpha} - \mathbf{W}^n_{\mathbf{i}-\mathbf{e}^d,\alpha}),\\
\Delta^L \mathbf{W}_{\mathbf{i},\alpha} &= \mathbf{W}^n_{\mathbf{i},\alpha} - \mathbf{W}^n_{\mathbf{i}-\mathbf{e}^d,\alpha},\\
\Delta^R \mathbf{W}_{\mathbf{i},\alpha} &= \mathbf{W}^n_{\mathbf{i}+\mathbf{e}^d,\alpha} - \mathbf{W}^n_{\mathbf{i},\alpha},\\
\Delta^{LL} \mathbf{W}_{\mathbf{i},\alpha} &= \frac{1}{2}(3\mathbf{W}^n_{\mathbf{i},\alpha} - 4\mathbf{W}^n_{\mathbf{i}-\mathbf{e}^d,\alpha} + \mathbf{W}^n_{\mathbf{i}-2\mathbf{e}^d,\alpha}),\\
\Delta^{RR} \mathbf{W}_{\mathbf{i},\alpha} &= \frac{1}{2}(-3\mathbf{W}^n_{\mathbf{i},\alpha} + 4\mathbf{W}^n_{\mathbf{i}+\mathbf{e}^d,\alpha} - \mathbf{W}^n_{\mathbf{i}+2\mathbf{e}^d,\alpha}).
\end{aligned} \tag{3.21}$$

When at least two cells on each side in direction d are available, Figure 3.5.a, the fourth order limited difference is calculated

$$\begin{aligned}
\Delta^d_x \mathbf{W}^n_{\mathbf{i},\alpha} &= \Delta^{vL}(\Delta^B \mathbf{W}_{\mathbf{i},\alpha}, \Delta^L \mathbf{W}_{\mathbf{i},\alpha}, \Delta^R \mathbf{W}_{\mathbf{i},\alpha}),\\
\Delta^B \mathbf{W}_{\mathbf{i},\alpha} &= \frac{2}{3}\left(\left(\mathbf{W} - \frac{1}{4}\Delta^d_2 \mathbf{W}\right)_{\mathbf{i}+\mathbf{e}^d,\alpha} - \left(\mathbf{W} + \frac{1}{4}\Delta^d_2 \mathbf{W}\right)_{\mathbf{i}-\mathbf{e}^d,\alpha}\right),\\
\Delta^d_2 \mathbf{W}^n_{\mathbf{i},\alpha} &= \Delta^{vL}(\Delta^C \mathbf{W}_{\mathbf{i},\alpha}, \Delta^L \mathbf{W}_{\mathbf{i},\alpha}, \Delta^R \mathbf{W}_{\mathbf{i},\alpha}).
\end{aligned} \tag{3.22}$$

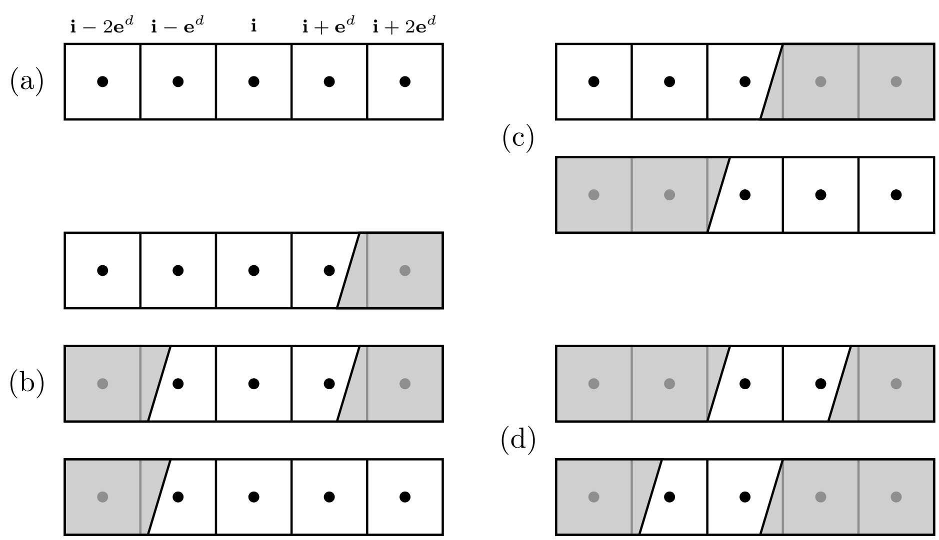

FIGURE 3.5. Possible stencil variations for slope calculation. The shaded area is the domain of the other phase.

When only one cell is available on each side in direction d, Figure 3.5.b, the second order limited difference is applied.

$$\Delta_x^d \mathbf{W}_{\mathbf{i},\alpha}^n = \Delta^{vL}(\Delta^C \mathbf{W}_{\mathbf{i},\alpha}, \Delta^L \mathbf{W}_{\mathbf{i},\alpha}, \Delta^R \mathbf{W}_{\mathbf{i},\alpha}). \tag{3.23}$$

When there is no cell available on one side but at least two cells are available on the other side, Figure 3.5.c, the one sided second order limited difference is used.

$$\Delta_x^d \mathbf{W}_{\mathbf{i},\alpha}^n = \begin{cases} \min(\Delta^{LL}\mathbf{W}_{\mathbf{i},\alpha}, \Delta^L \mathbf{W}_{\mathbf{i},\alpha}) & \text{if } \Delta^{LL}\mathbf{W}_{\mathbf{i},\alpha} \cdot \Delta^L \mathbf{W}_{\mathbf{i},\alpha} > 0, \\ 0 & \text{otherwise,} \end{cases} \tag{3.24}$$

$$\Delta_x^d \mathbf{W}_{\mathbf{i},\alpha}^n = \begin{cases} \min(\Delta^{RR}\mathbf{W}_{\mathbf{i},\alpha}, \Delta^R \mathbf{W}_{\mathbf{i},\alpha}) & \text{if } \Delta^{RR}\mathbf{W}_{\mathbf{i},\alpha} \cdot \Delta^R \mathbf{W}_{\mathbf{i},\alpha} > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{3.25}$$

And if there is no cell available on one side and only one cell is available on the other side, Figure 3.5.d, the one sided first order difference $\Delta^L \mathbf{W}_{\mathbf{i},\alpha}$ or $\Delta^R \mathbf{W}_{\mathbf{i},\alpha}$ is used to calculate $\Delta_x^d \mathbf{W}_{\mathbf{i},\alpha}^n$

3.7. Flux calculation

There are four different fluxes needed to apply the hybrid update method. These fluxes are defined based on the type of the face that the flux calculated on, which are *regular*, *covered*, *irregular* and *front*. See Figure 3.6 for an example of face types on an irregular control volume.

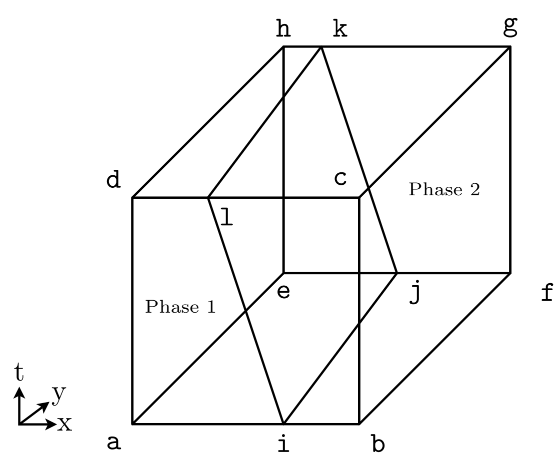

Figure 3.6. Face definitions on an irregular control volume. Respect to phase 1, `aehd` is a regular face, `adli` and `ehkj` are irregular faces, and `bcgf` is a covered face. Respect to phase 2, `bcgf` is a regular face, `bcli` and `fgkj` are irregular faces, and `aehd` is a covered face. Quadrilateral `ilkj` is the front face for both phases.

3.7.1. Flux on a regular face. The most common flux calculated is $\mathbf{F}^{cr,n}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha}$ for a regular face, that is the flux calculated on the face center $(\vec{x}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d}, t^{n+\frac{1}{2}})$. This is calculated using the high-order Godunov method with transverse flux correction in primitive variables [**28**, §4]. In summary

$$\widetilde{\mathbf{W}}^{n+\frac{1}{2}}_{\mathbf{i},\pm,d,\alpha} = \mathbf{W}^n_{\mathbf{i},\alpha} + \frac{1}{2}\left(\pm\mathbf{I} - \frac{\Delta t}{h}\mathbf{A}^d_{\mathbf{i},\alpha}(\mathbf{W}^n_{\mathbf{i},\alpha})\right)\mathbf{P}_\pm(\Delta^d_x\mathbf{W}^n_{\mathbf{i},\alpha}), \tag{3.26}$$

$$\mathbf{P}_\pm(\Delta^d_x\mathbf{W}^n_{\mathbf{i},\alpha}) = \sum_{\pm\lambda_k>0}\left(\mathbf{l}_k.\Delta^d_x\mathbf{W}^n_{\mathbf{i},\alpha}\right)\mathbf{r}_k, \tag{3.27}$$

where $\Delta^d_x\mathbf{W}^n_{\mathbf{i},\alpha}$ is the centered spatial difference calculated in direction d, λ_k are the eigenvalues of $\mathbf{A}^d_{\mathbf{i},\alpha}$, and $\mathbf{l}_k$ and $\mathbf{r}_k$ are the corresponding left and right eigenvectors. An initial Riemann problem is solved based on these extrapolation of primitive variables at the half time step and an initial estimate for the flux is calculated,

$$\widetilde{\mathbf{F}}^{cr,n}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha} = \mathbf{F}^d\left(\mathfrak{R}_d\left(\widetilde{\mathbf{W}}^{n+\frac{1}{2}}_{\mathbf{i},+,d,\alpha}, \widetilde{\mathbf{W}}^{n+\frac{1}{2}}_{\mathbf{i}+\mathbf{e}^\mathbf{d},-,\mathbf{d},\alpha}\right)\right). \tag{3.28}$$

The transverse flux is applied to find the corrected extrapolation of primitive variables on the face centers,

$$\mathbf{W}^{n+\frac{1}{2}}_{\mathbf{i},\pm,d,\alpha} = \widetilde{\mathbf{W}}^{n+\frac{1}{2}}_{\mathbf{i},\pm,d,\alpha} - \frac{\Delta t}{2h}\nabla_\mathbf{U}\mathbf{W}\left(\widetilde{\mathbf{F}}^{cr,n}_{\mathbf{i}+\frac{1}{2}\mathbf{e}^d,\alpha} - \widetilde{\mathbf{F}}^{cr,n}_{\mathbf{i}-\frac{1}{2}\mathbf{e}^d,\alpha}\right), \tag{3.29}$$

and the second Riemann problem is solved to calculate the unique values of primitive variables and fluxes on the face centers,

$$\mathbf{W}^{cr,n}_{\mathbf{i}+\frac{1}{2}\mathbf{e}^d,\alpha} = \mathfrak{R}_d\left(\mathbf{W}^{n+\frac{1}{2}}_{\mathbf{i},+,d,\alpha}, \mathbf{W}^{n+\frac{1}{2}}_{\mathbf{i}+\mathbf{e}^\mathbf{d},-,\mathbf{d},\alpha}\right), \tag{3.30}$$

$$\mathbf{F}^{cr,n}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha} = \mathbf{F}^d\left(\mathbf{W}^{cr,n}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha}\right). \tag{3.31}$$

3.7.2. Flux on a covered face. A covered face does not reside in the space-time domain of the corresponding phase. The upwinding extrapolation method that is used for flux calculation on a regular face only provides the state variable on one side of the covered face. A local extrapolation method is applied to calculate the state variables on the other side [**28**, §5.2]. For cell $\mathbf{i}$ with a covered face with respect to phase α and unit normal vector $\vec{n}$ defined to point outside of the corresponding phase domain the up, side and corner neighbor cells are defined as

$$\mathbf{i}^u = \mathbf{i} - s^{d'}\mathbf{e}^{d'} + s^d\mathbf{e}^d,$$

$$\mathbf{i}^s = \mathbf{i} - s^d\mathbf{e}^d,$$

$$\mathbf{i}^c = \mathbf{i} - s^{d'}\mathbf{e}^{d'},$$

where d and d' are 1 or 2, $d \neq d'$ and $s^d = \text{sign}(n_d)$. Based on the side of the unknown face value, support points on the same face side of the neighbor cells are defined as

$$\mathbf{W}^u = \widetilde{\mathbf{W}}^{n+\frac{1}{2}}_{\mathbf{i}^u,\pm,d,\alpha},$$

$$\mathbf{W}^s = \widetilde{\mathbf{W}}^{n+\frac{1}{2}}_{\mathbf{i}^s,\pm,d,\alpha} + s^d\Delta^d\mathbf{W},$$

$$\mathbf{W}^c = \widetilde{\mathbf{W}}^{n+\frac{1}{2}}_{\mathbf{i}^c,\pm,d,\alpha},$$

where $\Delta^d\mathbf{W}$ is the linear interpolation of the difference variables at the neighbor cells used for the extrapolation,

$$\Delta^{d''}\mathbf{W} = \begin{cases} \frac{|n_{d'}|}{|n_d|}\Delta_2^{d''}\mathbf{W}^n_{\mathbf{i}^c,\alpha} + \left(1 - \frac{|n_{d'}|}{|n_d|}\right)\Delta_2^{d''}\mathbf{W}^n_{\mathbf{i}^s,\alpha} & \text{if } |n_d| \geq |n_{d'}|, \\ \frac{|n_d|}{|n'_d|}\Delta_2^{d''}\mathbf{W}^n_{\mathbf{i}^c,\alpha} + \left(1 - \frac{|n_d|}{|n'_d|}\right)\Delta_2^{d''}\mathbf{W}^n_{\mathbf{i}^u,\alpha} & \text{if } |n_d| < |n_{d'}|, \end{cases}, \tag{3.32}$$

for $d'' = 1,\, 2$. The covered face extrapolation is calculated by

$$\mathbf{W}^f = \begin{cases} \frac{|n_{d'}|}{|n_d|}\mathbf{W}^c + \left(1 - \frac{|n_{d'}|}{|n_d|}\right)\mathbf{W}^s + \left(\frac{|n_{d'}|}{|n_d|}s^{d'}\Delta^{d'}\mathbf{W} + s^d\Delta^d\mathbf{W}\right) & \text{if } |n_d| \geq |n_{d'}|, \\ \frac{|n_d|}{|n'_d|}\mathbf{W}^c + \left(1 - \frac{|n_d|}{|n'_d|}\right)\mathbf{W}^u + \left(\frac{|n_d|}{|n'_d|}s^d\Delta^d\mathbf{W} + s^{d'}\Delta^{d'}\mathbf{W}\right) & \text{if } |n_d| < |n_{d'}|. \end{cases} \tag{3.33}$$

The first two terms on left are the interpolation between neighbor state variables, and the rest is the extrapolation in the normal direction (see Figure 3.7). Depending on the availability of the neighbor cell data the covered face extrapolation value is assigned by

$$\widetilde{\mathbf{W}}^{n+\frac{1}{2}}_{\mathbf{i}\mp\mathbf{e}^d,\pm,d,\alpha} = \begin{cases} \mathbf{W}^f & \text{if both required neighbor cells exist,} \\ \mathbf{W}^u \text{ (or } \mathbf{W}^s) & \text{if only } \mathbf{W}^u \text{ (or } \mathbf{W}^s) \text{ exists,} \\ \mathbf{W}^c & \text{if only } \mathbf{W}^c \text{ exists,} \\ \mathbf{W}^n_{\mathbf{i}\mp\mathbf{e}^d,\alpha} & \text{if neither neighbor cells exists.} \end{cases} \tag{3.34}$$

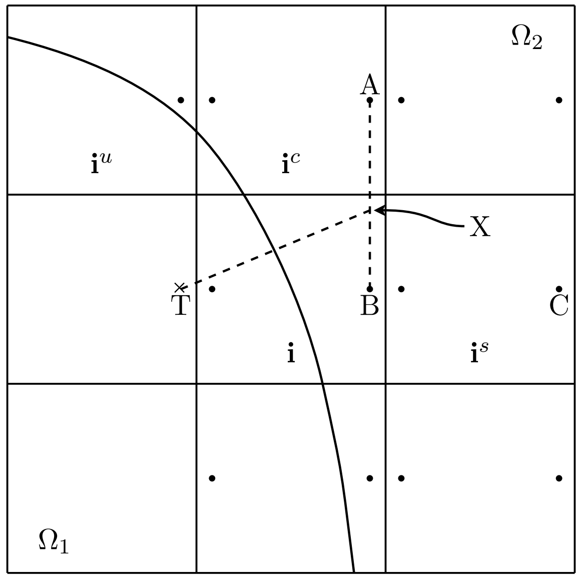

FIGURE 3.7. Extrapolation to a covered face. Here, the left face of cell $\mathbf{i}$ is a covered face respect to the phase domain Ω_2. Dots are the extrapolated state variables in $d = 1$ direction those are available using method described in §3.7.1. The state variables are not available at point T. Based on the direction of $\vec{n}$, neighbor cells $\mathbf{i}^c$ and $\mathbf{i}^s$ are used for the calculation. That is, the state variables are extrapolated from C to B. Then state variables are interpolated from A and B to X and extrapolated along $\vec{n}$ to point T.

3.7.3. Flux on an irregular face. An irregular face is created when the front intersects with a face of the control volume. The average flux is evaluated on a irregular face using the primitive variables on the face centroid,

$$\mathbf{F}^{cd,n}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha} = \mathbf{F}^d\left(\mathbf{W}^{cd,n}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha}\right). \tag{3.35}$$

The primitive variable on the face center, calculated in section 3.7.1, is extrapolated to the face centroid using the average slopes of two surrounding cells in time and space,

$$\begin{aligned}\mathbf{W}^{cd,n}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha} &= \mathbf{W}_\alpha\left(\vec{x}^{\,n}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha}, t^n_{\mathbf{i}+\frac{1}{2}\mathbf{e}^d,\alpha}\right) \\ &= \mathbf{W}^{cr,n}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha} + \left(\vec{x}^{\,n}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha} - \vec{x}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d}\right).\mathbf{e}^d\left(\frac{\Delta^{d'}_x\mathbf{W}^n_{\mathbf{i},\alpha} + \Delta^{d'}_x\mathbf{W}^n_{\mathbf{i}\pm\mathbf{e}^d,\alpha}}{2h}\right) \\ &\quad + \left(t^{cd,n}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha} - \left(n+\frac{1}{2}\right)\Delta t\right)\left(\frac{\Delta_t\mathbf{W}^n_{\mathbf{i},\alpha} + \Delta_t\mathbf{W}^n_{\mathbf{i}\pm\mathbf{e}^d,\alpha}}{2\Delta t}\right), \quad d \neq d',\end{aligned} \tag{3.36}$$

where $\Delta_t \mathbf{W}^n_{\mathbf{i},\alpha}$ is an estimate of time difference calculated as

$$\Delta_t \mathbf{W}^n_{\mathbf{i},\alpha} = \frac{1}{2}\left(\frac{1}{2D}\sum_{\pm}\sum_{d=1}^{D}\mathbf{W}^{cr,n}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha} - \mathbf{W}^n_{\mathbf{i},\alpha}\right). \tag{3.37}$$

3.7.4. Flux on the front. The flux on the front is calculated using the Riemann problem described in section 3.3 and Appendix B. The initial states for the Riemann problem are extrapolated from the cell center primitive states using the local slopes in space and time for each phase,

$$\begin{aligned}\widetilde{\mathbf{W}}^{f,n}_{\mathbf{i},\alpha} &= \widetilde{\mathbf{W}}_\alpha\left(\vec{x}^{f,n}_{\mathbf{i}}, t^{f,n}_{\mathbf{i}}\right) \qquad (3.38)\\ &= \mathbf{W}^n_{\mathbf{i},\alpha} + \frac{1}{h}\sum_{d=1}^{D}\left(\vec{x}^{f,n}_{\mathbf{i}} - \vec{x}^n_{\mathbf{i}}\right).\mathbf{e}^d\Delta^d_x\mathbf{W}^n_{\mathbf{i},\alpha} + \frac{1}{\Delta t}\left(t^{f,n}_{\mathbf{i}} - n\Delta t\right)\Delta_t\mathbf{W}^n_{\mathbf{i},\alpha}.\end{aligned}$$

The solution to the Riemann problem in the front normal direction with initial states $\widetilde{\mathbf{W}}^{f,n}_{\mathbf{i},1}$ and $\widetilde{\mathbf{W}}^{f,n}_{\mathbf{i},2}$ gives the primitive state on both sides of the front, $\mathbf{W}^{f,n}_{\mathbf{i},1}$ and $\mathbf{W}^{f,n}_{\mathbf{i},2}$, which are used for calculation of $\mathbf{F}^{f,n}_{\mathbf{i},\alpha}$ and $\mathbf{U}^{f,n}_{\mathbf{i},\alpha}$.

3.8. Excess mass balance and redistribution

To preserve the general conservation the mass difference between hybrid methods and conservative method (2.31) should be redistributed [**26**]. First, the excess mass on the front cells is balanced based on the characteristics. Then, the excess mass on each side is projected on the characteristics of the cell centered values,

$$\delta\mathbf{M}^n_{\mathbf{i},\alpha} = \sum_{k=1}^{m} b_{k,\alpha}\mathbf{r}_{k,\alpha}, \tag{3.39}$$

where $\mathbf{r}_{k,\alpha}$ are the right eigenvectors of $\dfrac{\partial\vec{\mathbf{F}}}{\partial\mathbf{U}}(\mathbf{U}^n_{\mathbf{i},\alpha})$. Then, the contribution of excess mass to each side is calculated:

$$\delta\mathbf{M}^n_{\mathbf{i},1} = b_{0,1}\mathbf{r}_{0,1} + b_{1,1}\mathbf{r}_{1,1} + b_{2,1}\mathbf{r}_{2,1} + b_{0,2}\mathbf{r}_{0,2}, \tag{3.40}$$

$$\delta\mathbf{M}^n_{\mathbf{i},2} = b_{1,2}\mathbf{r}_{1,2} + b_{2,2}\mathbf{r}_{2,2} + b_{3,2}\mathbf{r}_{3,2} + b_{3,1}\mathbf{r}_{3,1}. \tag{3.41}$$

Note that this balancing procedure is omitted for the density component, since it does not cross the interface.

In general, the redistribution method is

$$\mathbf{U}_{\mathbf{j},\alpha}^{n+1} := \mathbf{U}_{\mathbf{j},\alpha}^{n+1} + w_{\mathbf{i},\mathbf{j}}\,\delta\mathbf{M}_{\mathbf{i},\alpha}^{n}, \quad \mathbf{j} \in N(\mathbf{i}), \tag{3.42}$$

where $N(\mathbf{i})$ indicates a set of cells in neighborhood of cell $\mathbf{i}$ and weight coefficients satisfy following conditions [**28**],

$$w_{\mathbf{i},\mathbf{j}} \geq 0, \qquad \sum_{\mathbf{j}\in N(\mathbf{i})} w_{\mathbf{i},\mathbf{j}}\,\Lambda_{\mathbf{j}}^{n+1} = 1. \tag{3.43}$$

Here, the weight coefficient is

$$w_{\mathbf{i},\mathbf{j}} = \frac{1}{\sum_{\mathbf{l}\in N(\mathbf{i})} \Lambda_{\mathbf{l}}^{n+1}} \tag{3.44}$$

where $N(\mathbf{i})$ is the neighborhood of cell $\mathbf{i}$ in each phase, containing cell $\mathbf{i}$. Using the above weight coefficients gives

$$\Lambda_{\mathbf{j},\alpha}^{n+1}\mathbf{U}_{\mathbf{j},\alpha}^{n+1} := \Lambda_{\mathbf{j},\alpha}^{n+1}\mathbf{U}_{\mathbf{j},\alpha}^{n+1} + \frac{\Lambda_{\mathbf{j},\alpha}^{n+1}}{\sum_{\mathbf{k}\in N(\mathbf{i})} \Lambda_{\mathbf{k},\alpha}^{n+1}}\delta\mathbf{M}_{\mathbf{i},\alpha}^{n}, \quad \mathbf{j} \in N(\mathbf{i}), \tag{3.45}$$

which means the excess mass of cell $\mathbf{i}$ is redistributed into neighbor cells in proportion to their cell fraction at time $n+1$.

CHAPTER 4

Test and verification

4.1. Convergence test - Smooth perturbation

The new method is implemented using the algorithm described in previous chapter. First, it is tested for a 2-D perturbation problem. The problem domain is a 1 by 1 square. The initial front is a circle with radius 0.27 centered at the center of the problem domain. The phase inside the front has a smooth bell-shaped perturbation in the pressure component of the form

$$p_{in}(r) = p_{base}\left(1 + 256\alpha\left(\frac{r+r_p}{2r_p} - \left(\frac{r+r_p}{2r_p}\right)^2\right)^4\right), \qquad r \leq r_p, \tag{4.1}$$

where $p_{base} = 1$ is the background value of pressure, $\alpha = 0.15$ determines the peak to base value of the perturbation, $r_p = 0.25$ is the radius of the perturbation and r is the distance from the center of the problem domain. The initial density is constant and equals 1 for both phases and the initial velocities are zero. The initial pressure for the phase outside the front equals p_{base}. Both phases are the same material, $\gamma = 1.4$. With Courant number of 0.5, simulation is done to a fixed time $t = 0.6$ to allow the perturbation to pass the front. As shown in Figure 4.1 the final result has the expected symmetry. Since the same material is chosen for both phases, it is expected to have continuous states on the front, which is confirmed for the simulation results. Also, a simulation for a single phase setup with the same initial condition is done and results are compared to the two-phase simulation in Figure 4.1. Qualitatively, there are no difference between single phase and two-phase simulations.

To make quantitative comparisons, the error for conservative state variables are defined as

$$\mathbf{E}^{2h}_{\mathbf{i},\alpha} = \mathbf{U}^{2h}_{\mathbf{i},\alpha}(t) - \mathbf{U}^{e}_{\mathbf{i},\alpha}(t), \tag{4.2}$$

where $\mathbf{U}^{2h}_{\mathbf{i},\alpha}(t)$ is calculated with spatial grid size $2h$ and $\mathbf{U}^{e}_{\mathbf{i},\alpha}$ is the exact solution. Since the exact solution is not available, it is replaced with the solution from the simulation with a finer grid. That

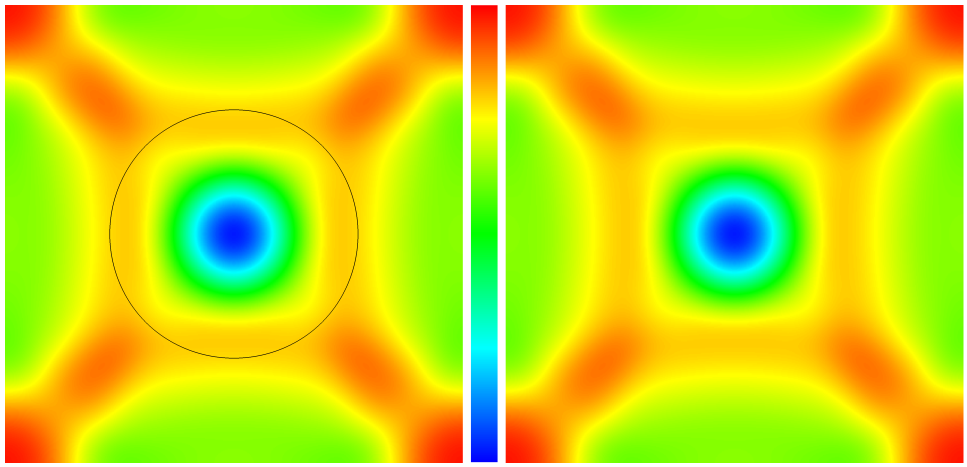

FIGURE 4.1. Density profile for smooth perturbation test at at $t = 0.6$ on a 512×512 grid. Left: dual phase simulation, front position shown with solid line. Right: result for single phase simulation with no front-tracking. The color table limits are [0.9, 1.04].

is, if the finer grid calculation is done by step size h, the exact solution is replaced with

$$\mathbf{U}^e_{\mathbf{i},\alpha}(t) \approx \frac{\sum_{\mathbf{j}\in M(\mathbf{i})} \Lambda^h_{\mathbf{j},\alpha} \mathbf{U}^h_{\mathbf{j},\alpha}(t)}{\sum_{\mathbf{j}\in M(\mathbf{i})} \Lambda^h_{\mathbf{j},\alpha}}, \tag{4.3}$$

where $M(\mathbf{i})$ are the cells in the finer grid solutions those are corresponding to the cell $\mathbf{i}$ in the coarser grid solution. In other words, $M(\mathbf{i})$ is the box defined by two cells $2\mathbf{i}$ and $2\mathbf{i}+\mathbf{1}$ at its lower left and upper right corners.

The total error in the L_1, L_2 and L_∞ norms are

$$\epsilon^h_{L_1} = \sum_\alpha \sum_{\mathbf{i}\in\mathbf{\Omega}_\alpha} h^d \Lambda^h_{\mathbf{i},\alpha} |\mathbf{E}^h_{\mathbf{i},\alpha}|, \tag{4.4}$$

$$\epsilon^h_{L_2} = \sum_\alpha \left(\sum_{\mathbf{i}\in\mathbf{\Omega}_\alpha} h^d \Lambda^h_{\mathbf{i},\alpha} {\mathbf{E}^h_{\mathbf{i},\alpha}}^2 \right)^{\frac{1}{2}}, \tag{4.5}$$

$$\epsilon^h_{L_\infty} = \sum_\alpha \max_{\mathbf{i}\in\mathbf{\Omega}_\alpha} |\mathbf{E}^h_{\mathbf{i},\alpha}|, \tag{4.6}$$

and the convergence rate is calculated by

$$p = \frac{\log\left(\frac{\epsilon^{2h}}{\epsilon^h}\right)}{\log(2)}. \tag{4.7}$$

The simulation for the expansion of the perturbation is done in six grid sizes from $\frac{1}{32}$ to $\frac{1}{1024}$, and the solution errors and convergence rates are calculated based on Richardson method explained above. As shown in Table 4.1, the new method shows second-order convergence in L_1, L_2 and L_∞ for all conserved variables. It is also verified that conserved variables are satisfying the discrete conservation identity for each phase in its domain,

$$\sum_{\mathbf{i}\in\mathbf{\Omega}_\alpha} |V^{n+1}_{\mathbf{i},\alpha}|\mathbf{U}^{n+1}_{\mathbf{i},\alpha} = \sum_{\mathbf{i}\in\mathbf{\Omega}_\alpha} |V^{n}_{\mathbf{i},\alpha}|\mathbf{U}^{n}_{\mathbf{i},\alpha} - \sum_{\mathbf{i}\in\partial\mathbf{\Omega}_\alpha}\left(\sum_{\pm,d=1}^{D}\left(\pm|A^n_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha}|\mathbf{F}^{cd,n}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha}\right) + |A^{f,n}_{\mathbf{i}}|\vec{n}^n_{\mathbf{i},\alpha}\cdot\left(\vec{\mathbf{F}}^{f,n}_{\mathbf{i},\alpha},\mathbf{U}^{f,n}_{\mathbf{i},\alpha}\right)\right), \quad (4.8)$$

to the machine precision.

4.2. Convergence test - Shock channel

A shock wave test similar to the Sod [**49**] shock tube problem is picked to test our method for a problem with an initial captured discontinuity. The domain is a $[2, 0.5]$ channel. Initially the discontinuity is set in phase 1 at $x = 1$ and the interface is a sine wave $\phi_0(y) = \eta_0 \sin(\omega y + \theta) + x_0$ at $x_0 = 1.5$. The wavelength is the same as the width of the channel $\lambda = 0.5$ which gives $\omega = 2\pi/\lambda = 4\pi$. The interface amplitude is set to $\eta_0 = 0.16$ and phase is $\theta = \pi/2$ (Figure 4.2). The initial values for each phase are shown in Table 4.2. Solid wall and periodic boundary conditions are used for vertical and horizontal boundaries, respectively. The simulation is done up to time $t = 0.6$, allowing the shock wave to pass through the interface. Using five grid size from 1/32 to 1/512, error and convergence rates are calculated (see Table 4.3). Since there is a captured shock, a discontinuity that is not tracked, in the solution, a drop in the convergence rate to first-order is observed. The same drop of convergence rate is found when the single phase simulation is done with the similar initial conditions. The final result for finest resolution is shown in Figure 4.2.

4.3. Shock channel - Richtmyer-Meshkov instability

To test our algorithm in a case comparable to experiment the initial growth of Richtmyer-Meshkov instability (RMI) is simulated with the new method. RMI occurs when a sudden acceleration is forced on an interface separating fluids with different densities. A shock passing the

ρ						
h	L_1	rate	L_2	rate	L_∞	rate
1/32	1.59e-03		2.43e-03		1.56e-02	
1/64	1.92e-04	3.05	2.73e-04	3.15	1.96e-03	2.99
1/128	3.13e-05	2.61	4.42e-05	2.63	5.49e-04	1.83
1/256	6.25e-06	2.33	8.56e-06	2.37	8.61e-05	2.67
1/512	2.01e-06	2.01	2.18e-06	1.98	2.31e-05	1.90
ρu						
h	L_1	rate	L_2	rate	L_∞	rate
1/32	7.74e-04		1.12e-03		5.61e-03	
1/64	1.76e-04	2.14	2.50e-04	2.17	2.33e-03	1.27
1/128	2.57e-05	2.78	4.19e-05	2.58	9.44e-04	1.30
1/256	5.46e-06	2.23	8.01e-06	2.39	2.54e-04	1.89
1/512	1.44e-06	1.92	2.03e-06	1.98	4.89e-05	2.38
ρv						
h	L_1	rate	L_2	rate	L_∞	rate
1/32	5.54e-04		1.03e-03		5.08e-03	
1/64	1.42e-04	2.14	2.00e-04	2.17	1.92e-03	1.27
1/128	2.29e-05	2.78	3.63e-05	2.58	8.18e-04	1.30
1/256	5.40e-06	2.23	7.81e-06	2.39	2.27e-04	1.89
1/512	1.43e-06	1.92	2.02e-06	1.98	4.38e-05	2.38
E						
h	L_1	rate	L_2	rate	L_∞	rate
1/32	5.54e-03		8.54e-03		5.51e-02	
1/64	6.56e-04	3.08	9.53e-04	3.16	6.93e-03	2.99
1/128	1.07e-04	2.62	1.53e-04	2.64	1.92e-03	1.85
1/256	2.09e-05	2.36	2.94e-05	2.38	3.00e-04	2.68
1/512	5.21e-06	2.00	7.53e-06	1.96	8.29e-05	1.86

TABLE 4.1. Error and convergence rate for 2-D perturbation problem.

	Phase 1: post-shock	Phase 1: pre-shock	Phase 2
ρ	3	1	1
u	0	0	0
v	0	0	0
p	3	1	1
γ	1.4	1.4	1.276

TABLE 4.2. Initial values for shock channel problem.

interface of two different gasses is an example of RMI. It is observed that an initial small perturbation on the interface grows after passage of a shock wave in a wide range of Mach numbers [**50**]. Such growth has been characterized as having a linear phase at the beginning of the process [**51**], before the nonlinear evolution of spike and bubble development, and of turbulent mixing.

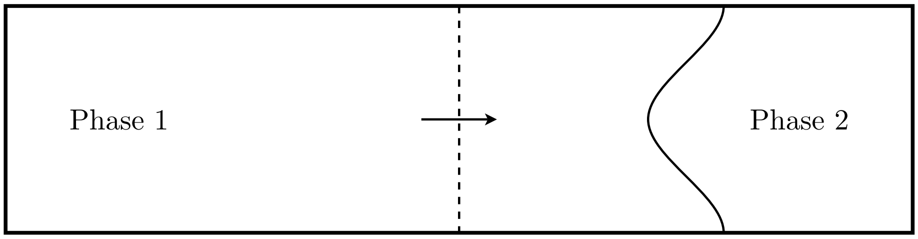

FIGURE 4.2. Shock channel initial configuration (top). Pressure profile at $t = 0.6$. The same color table as Figure 4.1 is used with limits [1.67, 3.0] for phase 1 and [1.0, 1.72] for phase 2 (bottom).

h	ρ	rate	ρu	rate
1/32	8.58e-03		8.27e-03	
1/64	3.84e-03	1.16	3.48e-03	1.25
1/128	2.09e-03	0.88	1.82e-03	0.94
1/256	1.13e-03	0.89	9.71e-04	0.90
h	ρv	rate	E	rate
1/32	1.89e-03		2.64e-02	
1/64	9.43e-04	1.01	1.01e-02	1.28
1/128	5.60e-04	0.75	5.20e-03	0.96
1/256	2.92e-04	0.94	2.71e-03	0.94

TABLE 4.3. Error in L_1 and convergence rate for the shock channel problem.

	Phase 1: post-shock	Phase 1: pre-shock	Phase 2
$\rho\,(\mathrm{g/cm^3})$	1.872×10^{-3}	1.351×10^{-3}	5.494×10^{-3}
$u\,(\mathrm{cm/s})$	1.013×10^{4}	0	0
$v\,(\mathrm{cm/s})$	0	0	0
$p\,(\mathrm{g/(cm.s^2)})$	1.453×10^{6}	9.650×10^{4}	9.650×10^{4}
γ	1.276	1.276	1.4

TABLE 4.4. Initial values for RMI problem.

The setup for the simulation is comparable with an experiment done by Collins and Jacobs [**52**]. The domain for this simulation is a $[23.73\,\mathrm{cm}, 5.93\,\mathrm{cm}]$ channel. The initial shock is located at $x = 10\,\mathrm{cm}$ and the interface is placed at $x = 12\,\mathrm{cm}$ with a pre-shock sinusoidal perturbation of amplitude $a_0^- = 0.18\,\mathrm{cm}$ and wavelength $\lambda = 5.93\,\mathrm{cm}$. The initial condition for phase 1 (pre-shock) and phase 2 gases are picked to be comparable to the cited experiment, and phase 1 (post-shock) initial conditions were chosen based on Rankine-Hugoniot jump condition to have a shock with Ma $= 1.21$ in phase 1. With Courant number 0.5 the dynamics of the front and phases are simulated up to time 0.4 ms and the convergence rate of the method is measured (see Table 4.5).

Richtmyer [**51**] derived the impulsive growth rate relation for the amplitude of the perturbation based on the linear theory which describes the development of the instability after refraction of shock while the perturbation is small enough to be considered in the linear regime of the process,

$$\frac{\mathrm{d}\eta}{\mathrm{d}t} = kA^{+}\Delta V_f \eta_0^{+}, \tag{4.9}$$

where k is the wavenumber of the perturbation, A^{+} is the post-shock Atwood number, ΔV_f is the velocity jump on the front following the shock refraction and η_0^{+} is the post-shock initial perturbation amplitude. For the simulation, it is initialized to have $k = 2\pi/\lambda$, $\rho_1^{+} = 2.07 \times 10^{-3}$, $\rho_2^{+} = 9.05 \times 10^{-3}$, $A^{+} = (\rho_2^{+} - \rho_1^{+})/(\rho_2^{+} + \rho_1^{+}) = 0.63$, $\Delta V_f = 6356.24$ and $\eta_0^{+} = 0.15$ in CGS units. The calculated amplitude growth using equation (4.9) is $\mathrm{d}\eta/\mathrm{d}t = 624.82$ cm/s.

N_x	ρ	rate	ρu	rate
128	7.87e-06		1.44e-01	
256	3.62e-06	1.12	6.49e-02	1.15
512	3.00e-06	0.27	5.21e-02	0.32
1024	1.22e-06	1.30	2.43e-02	1.10
N_x	ρv	rate	E	rate
128	2.68e-02		1.76e+04	
256	1.08e-02	1.31	8.14e+03	1.11
512	5.79e-03	0.90	6.68e+03	0.29
1024	3.47e-03	0.74	2.84e+03	1.23

TABLE 4.5. Error in L_1 and convergence rate for RMI problem. N_x is the number of cells in x direction

From the simulation results, the amplitude of the interface is plotted in its linear regime (Figure 4.4). A linear fit shows a growth rate of $\mathrm{d}\eta/\mathrm{d}t = 606.64$ cm/s while the amplitude growth rate from the experiment [**52**] is 628.64 cm/s. The simulation results show good match ($\approx 3\%$ error) with experimental and analytical results.

After the initial linear regime of the RMI test, where crests and troughs are symmetric, the interface grows nonlinearly. It becomes visible by appearance of an asymmetric bubble and spike,

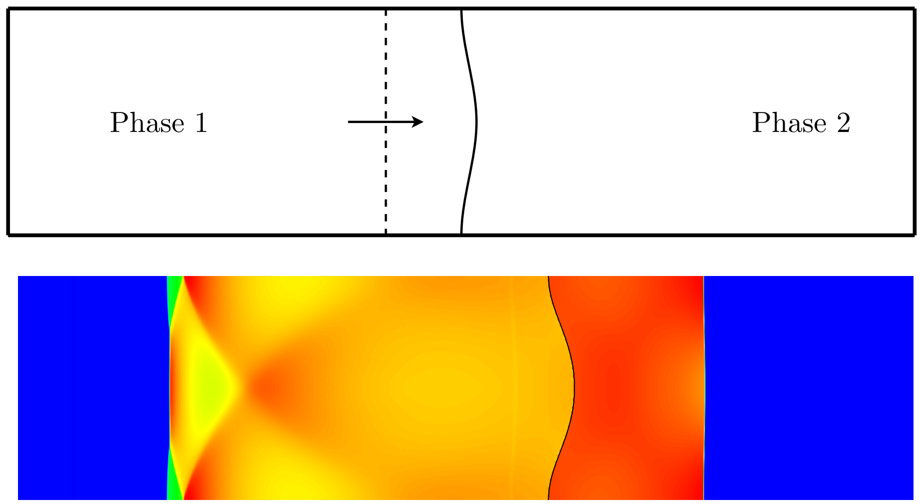

FIGURE 4.3. RMI problem initial configuration (top). Pressure profile at $t = 0.4\,\mathrm{ms}$. The same color table as Figure 4.1 is used with limits $[1.87\,2.11] \times 10^{-3}\,(\mathrm{g/cm^3})$ for phase 1 and $[5.49\,9.28] \times 10^{-3}\,(\mathrm{g/cm^3})$ for phase 2 (bottom).

followed by the spike rolling-up. The RMI simulation is continued to time $t = 2.8\,\mathrm{ms}$ to observe the nonlinear evolution of the front (Figure 4.5). Simulation is stopped when the front curvature increased near the tip of the spikes. At these regions, the level set function becomes non-differentiable, which is not applicable for geometry information extraction.

4.4. Summary and conclusion

A new front-tracking method is developed for contact discontinuities using the finite volume approach on a Cartesian grid. Including the geometry information at the irregular front cells is an essential part to this approach and achieved by applying the level set method for front representation and evolution, while profiting from the accurate geometry calculation algorithms [**20, 21**].

Algorithms needed to implement the method are presented and implemented for numerical test. Simple convergence and stability tests show the new method is second-order in L_1, L_2 and L_∞ norms when tracking the material interface and no other discontinuity is present in the solution. Also it is shown that in the presence of a captured shock the convergence reduces to first-order in L_1. Further validation is done with the simulation of initial developments of Richtmyer-Meshkov instability. Simulations results are in good agreement with theory and experimental results. A

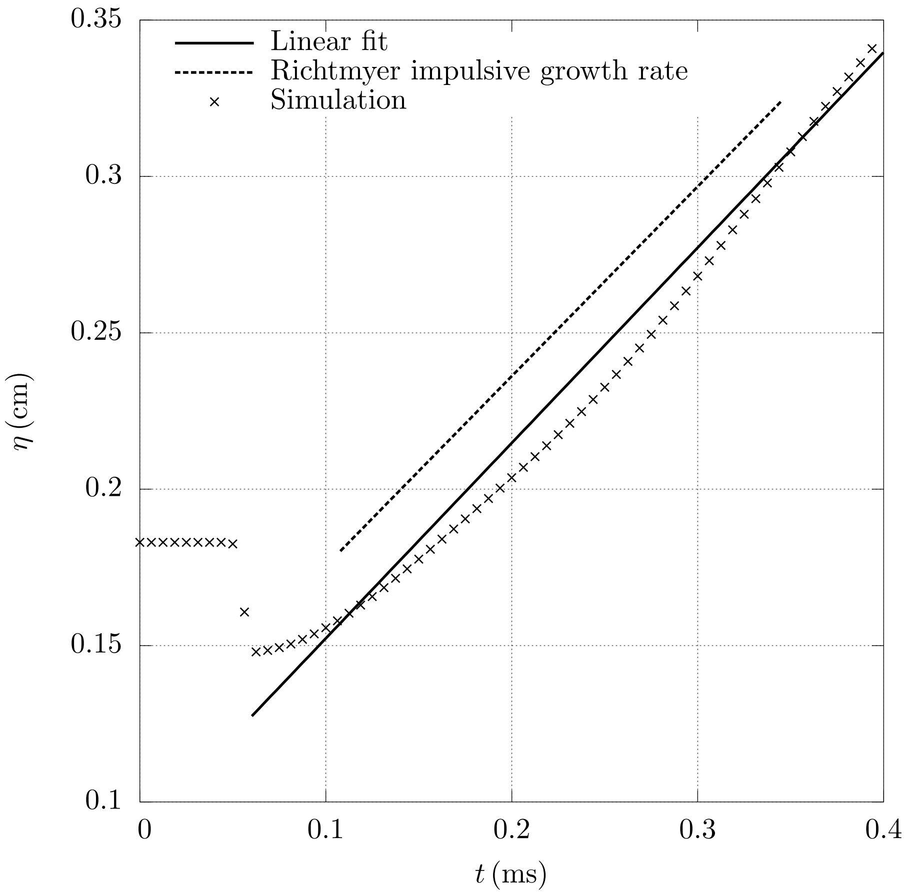

FIGURE 4.4. Amplitude of the perturbation on the front in RMI simulation

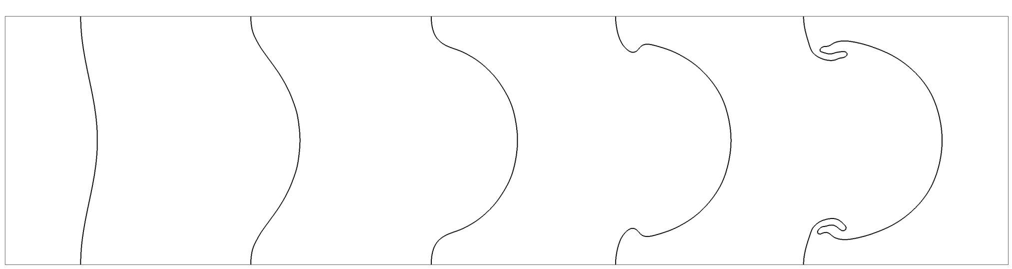

FIGURE 4.5. RMI problem front evolution. From left to right, $t = 0$ initial single mode perturbation , $t = 0.7$ ms linear growth of perturbation, $t = 1.4$ ms asymmetric growth of crests and troughs, $t = 2.1$ ms formation of bubble and spike, and $t = 2.8$ ms Spike roll-up.

extended simulation case shows the capability of the method for handling complex geometries to the limit of geometry information extraction method.

It should be noted that many parts of the above algorithms may be implemented differently. For example the interpolation and extrapolation algorithms may include cautionary limiting operators [**39**] or different redistribution algorithm may be applied [**28**]. Although such details may vary from case to case, here, it is shown that a second-order method is achievable, if accurate geometrical information are considered and incorporated in the algorithm.

Further work is needed to develop and study robust and accurate interpolation and extrapolation methods for the algorithm. It would be an advantage to implement a mesh refinement algorithm. An adaptive mesh refinement(AMR) algorithm would increase the resolution locally, where needed, while keeping computational cost low, respect to the whole domain resolution increase. Implementation in three spatial dimension is also another extension that would be useful for applying this method on 3-D problems which would have broader physical application.

APPENDIX A

Notation

Notation	Description
D	Spatial dimension
$\mathbf{U}$	Conservative variables
$\mathbf{F}$	Conservative fluxes
$\boldsymbol{\Omega}$	Phase domain
$\vec{n}_s$	Spatial normal vector
$\vec{n}$	Space-time normal vector
$\mathcal{F}$	Front/interface position
s^f	Magnitude of front velocity in the front normal direction
Δt	Time discretization step
h	Space discretization step
ρ	Density
u	Velocity in x direction
v	Velocity in y direction
p	Pressure
E	Total energy
γ	Ratio of specific heats
$\mathbf{W}$	Primitive variables
$\mathbf{A}$	Jacobian matrix of flux variables
Υ	Cell in Cartesian grid
V	Control cell (in space)
C	Control volume (in space-time)
A	Face of a control volume

Λ	Cell fraction
a	Area fraction
ϕ	Level set function
$\vec{\nu}_{ext}$	Extended velocity field
$\mathbf{DF}^{C}$	Cell centroid conservative flux difference
$\mathbf{DF}^{NC}$	Cell centroid nonconservative flux difference
$\mathbf{DF}^{NC,cr}$	Cell center nonconservative flux difference
$\delta\mathbf{M}$	Excess mass
$\mathbf{F}^{cd,n}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha}$	Flux at face centroid
$\mathbf{F}^{cr,n}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d,\alpha}$	Flux at face center
$\mathbf{F}^{f,n}_{\mathbf{i},d,\alpha}$	Flux at front centroid
$(\vec{x}_{\mathbf{i}}, t^n)$	Space-time position of a cell center
$\left(\vec{x}_{\mathbf{i}\pm\frac{1}{2}\mathbf{e}^d}, t^{n+\frac{1}{2}}\right)$	Space-time position of a regular face center
$\left(\vec{x}^{n}_{\mathbf{i},\alpha}, t^n\right)$	Space-time position of a cell centroid
$\left(\vec{x}^{n}_{\mathbf{i}\pm\frac{1}{2}e^2,\alpha}, t^{n}_{\mathbf{i}\pm\frac{1}{2}e^2,\alpha}\right)$	Space-time position of a fractional face centroid
$\left(\vec{x}^{f,n}_{\mathbf{i}}, t^{f,n}_{\mathbf{i}}\right)$	Space-time position of front centroid

APPENDIX B

Numerical solution to Riemann problem on the contact discontinuity

The solution to Riemann problems is needed to calculate the state variables and fluxes on the front centroid between the two phases. The Riemann problem for the 2-D Euler equations is defined

$$\mathfrak{R}_x^f(\mathbf{W}_L, \mathbf{W}_R) = (\mathbf{W}_L^*, \mathbf{W}_R^*), \tag{B.1}$$

$$\mathbf{W}_L = (\rho_L,\, u_L,\, v_L,\, p_L)^T, \tag{B.2}$$

$$\mathbf{W}_R = (\rho_R,\, u_R,\, v_R,\, p_R)^T, \tag{B.3}$$

$$\mathbf{W}_L^* = (\rho_L^*,\, u^*,\, v_L*,\, p^*)^T, \tag{B.4}$$

$$\mathbf{W}_R^* = (\rho_R^*,\, u^*,\, v_R*,\, p^*)^T, \tag{B.5}$$

with $\gamma = \gamma_L$ for the phase on the left and $\gamma = \gamma_R$ for the phase on the right side. An approximate Riemann solver [**53**] is used to calculate the state variable in the star region, Figure B.1. The sound speeds are

$$c_L = \sqrt{\frac{\gamma_L p_L}{\rho_L}}, \quad c_R = \sqrt{\frac{\gamma_R p_R}{\rho_R}}, \tag{B.6}$$

and the Lagrangian sound speeds are defined as

$$w_L = \rho_L c_L, \quad w_R = \rho_R c_R. \tag{B.7}$$

The pressure and normal velocity in the star region and densities are calculated as follows,

$$p^* = \frac{1}{w_L + w_R}\left(w_R p_L + w_L p_R + w_R w_L (u_L - u_R)\right), \tag{B.8}$$

$$u^* = \frac{1}{w_L + w_R}\left(w_L u_L + w_R u_R + (p_L - p_R)\right), \tag{B.9}$$

$$\rho_L^* = \rho_L + \frac{p^* - p_L}{c_L^2}, \tag{B.10}$$

$$\rho_R^* = \rho_R + \frac{p^* - p_R}{c_R^2}. \tag{B.11}$$

The tangential component of the velocity does not change passing the 1-wave or 3-wave discontinuity,

$$v_L^* = v_L, \quad v_R^* = v_R. \tag{B.12}$$

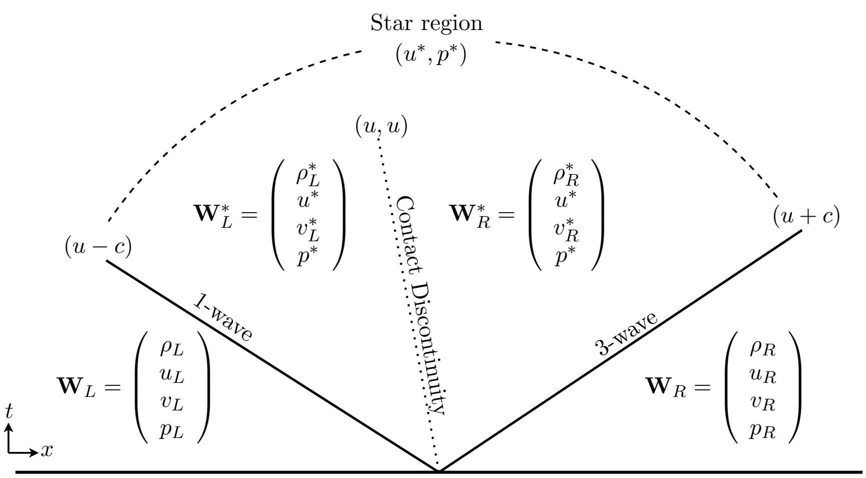

FIGURE B.1. Structure of the solution of the 2-D split Riemann Problem.

APPENDIX C

Front velocity and Riemann solution

Here, the reasoning for replacing u^* with s^f after solving the Riemann problem on the front centroid is shown. For ease of demonstration the $D = 2$ case is discussed, but this would be valid for any spatial dimension.

It is assumed that after solving the Riemann problem in the front normal direction the solution is

$$\mathfrak{R}_x^f\left(\widehat{\mathbf{W}}_1, \widehat{\mathbf{W}}_2\right) = \left(\left(\rho_L^*,\, u^*,\, v_L^*,\, p^*\right)^T, \left(\rho_R^*,\, u^*,\, v_R^*,\, p^*\right)^T\right). \tag{C.1}$$

The front flux on the front centroid is calculated from equation (2.22).

$$\mathbf{F}_{\mathbf{i},\alpha}^{front,n} = |A_{\mathbf{i}}^{f,n}|\vec{n}_{\mathbf{i},\alpha}^{n} \cdot \left(\vec{\mathbf{F}}_{\mathbf{i},\alpha}^{f,n}, \mathbf{U}_{\mathbf{i},\alpha}^{f,n}\right). \tag{C.2}$$

Since the free stream preservation method is applied, $|A_{\mathbf{i}}^{f,n}|\vec{n}_{\mathbf{i},\alpha}^{n}$ can be written as

$$|A_{\mathbf{i}}^{f,n}|\vec{n}_{\mathbf{i},\alpha}^{n} = \begin{pmatrix} |A^n_{\mathbf{i}-\frac{1}{2}\mathbf{e}^1,\alpha}| - |A^n_{\mathbf{i}+\frac{1}{2}\mathbf{e}^1,\alpha}| \\ |A^n_{\mathbf{i}-\frac{1}{2}\mathbf{e}^2,\alpha}| - |A^n_{\mathbf{i}+\frac{1}{2}\mathbf{e}^2,\alpha}| \\ |A^n_{\mathbf{i}-\frac{1}{2}\mathbf{e}^3,\alpha}| - |A^n_{\mathbf{i}+\frac{1}{2}\mathbf{e}^3,\alpha}| \end{pmatrix} = \begin{pmatrix} \Delta A_{x,\alpha} \\ \Delta A_{y,\alpha} \\ \Delta A_{t,\alpha} \end{pmatrix}. \tag{C.3}$$

Therefore, the spatial normal vector $\vec{n}_s$ and rotation matrix are

$$\vec{n}_s = \frac{1}{\sqrt{\Delta A_{x,\alpha}^2 + \Delta A_{y,\alpha}^2}} \begin{pmatrix} \Delta A_{x,\alpha} \\ \Delta A_{y,\alpha} \end{pmatrix}, \mathbf{R} = \frac{1}{\sqrt{\Delta A_{x,\alpha}^2 + \Delta A_{y,\alpha}^2}} \begin{pmatrix} \Delta A_{x,\alpha} & \Delta A_{y,\alpha} \\ -\Delta A_{y,\alpha} & \Delta A_{x,\alpha} \end{pmatrix}. \tag{C.4}$$

The solution of the Riemann problem is rotated back to the laboratory frame,

$$\mathbf{W}_{\mathbf{i},\alpha}^{f,n} = \begin{pmatrix} \rho_\alpha^* \\ \frac{\Delta A_{x,\alpha} u^* - \Delta A_{y,\alpha} v_\alpha^*}{\sqrt{\Delta A_{x,\alpha}^2 + \Delta A_{y,\alpha}^2}} \\ \frac{\Delta A_{y,\alpha} u^* + \Delta A_{x,\alpha} v_\alpha^*}{\sqrt{\Delta A_{x,\alpha}^2 + \Delta A_{y,\alpha}^2}} \\ p^* \end{pmatrix}. \tag{C.5}$$

Primitive variables in equation (C.5) is used to calculate the front flux value for density channel in equation (C.2),

$$F^{front,n}_{\mathbf{i},\alpha,mass} = \begin{pmatrix} \Delta A_{x,\alpha} \\ \Delta A_{y,\alpha} \\ \Delta A_{t,\alpha} \end{pmatrix} \cdot \begin{pmatrix} \rho^*_\alpha \frac{\Delta A_{x,\alpha} u^* - \Delta A_{y,\alpha} v^*_\alpha}{\sqrt{\Delta A^2_{x,\alpha} + \Delta A^2_{y,\alpha}}} \\ \rho^*_\alpha \frac{\Delta A_{y,\alpha} u^* + \Delta A_{x,\alpha} v^*_\alpha}{\sqrt{\Delta A^2_{x,\alpha} + \Delta A^2_{y,\alpha}}} \\ \rho^*_\alpha \end{pmatrix} \tag{C.6}$$

$$= \rho^*_\alpha \left(\frac{u^*(\Delta A^2_{x,\alpha} + \Delta A^2_{y,\alpha})}{\sqrt{\Delta A^2_{x,\alpha} + \Delta A^2_{y,\alpha}}} + \Delta A_t \right) \tag{C.7}$$

Since mass does not cross the contact discontinuity, $F^{front,n}_{\mathbf{i},\alpha,mass}$ should be zero, which gives

$$u^* = \frac{-\Delta A^2_{t,\alpha}}{\sqrt{\Delta A^2_{x,\alpha} + \Delta A^2_{y,\alpha}}}. \tag{C.8}$$

Dividing the nominator and denominator by $|A^{f,n}_{\mathbf{i}}|$ results in

$$u^* = \frac{-n_t}{\sqrt{n_x^2 + n_y^2}} = s^f. \tag{C.9}$$

Replacing u^* with s^f ensure the correct numerical evaluation of flux calculation on the front and consistent flux calculation with the evolution of the front geometry. This is not only for density but for all conserved variables, which can be shown using the similar method as above. Also, one can say that it is necessary to pick $u^* = s^f$ to satisfy Rankine-Hugoniot jump condition in a conservative and physical manner.

Bibliography

[1] G. Agresar, J. J. Linderman, G. Tryggvason, and K. G. Powell. An adaptive, cartesian, front-tracking method for the motion, deformation and adhesion of circulating cells. *J. Comput. Phys.*, 143(2):346–380, 1998.

[2] M. J. Saxton and K. Jacobson. Single-particle tracking: applications to membrane dynamics. *Annu. Rev. Bioph. Biom.*, 26(1):373–399, 1997.

[3] G. Tryggvason and R. Scardovelli. *Direct numerical simulations of gas-liquid multiphase flows.* Cambridge University Press, 2011.

[4] E. Olmos, C. Gentric, Ch. Vialh, G. Wild, and N. Midoux. Numerical simulation of multiphase flow in bubble column reactors. influence of bubble coalescence and break-up. *Chem. Eng. Sci.*, 56(21):6359–6365, 2001.

[5] R. P. Fedkiw, T. Aslam, and S. Xu. The ghost fluid method for deflagration and detonation discontinuities. *J. Comput. Phys.*, 154(2):393–427, 1999.

[6] V. Smiljanovski, V. Moser, and R. Klein. A capturing-tracking hybrid scheme for deflagration discontinuities. *Combustion Theory and Modelling*, 1(2):183–215, 1997.

[7] R. L. Holmes, G. Dimonte, B. Fryxell, M. L. Gittings, J. W. Grove, M. Schneider, D. H. Sharp, A. L. Velikovich, R. P. Weaver, and Q. Zhang. Richtmyer–meshkov instability growth: experiment, simulation and theory. *J. Fluid Mech.*, 389:55–79, 1999.

[8] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y-J Jan. A front-tracking method for the computations of multiphase flow. *J. Comput. Phys.*, 169(2):708–759, 2001.

[9] J. K. Dukowicz. A simplified adaptive mesh technique derived from the moving finite element method. *J. Comput. Phys.*, 56(2):324–342, 1984.

[10] M. J. Fritts and J. P. Boris. The Lagrangian solution of transient problems in hydrodynamics using a triangular mesh. *J. Comput. Phys.*, 31(2):173–215, 1979.

[11] A. Huerta and W. K. Liu. Viscous flow with large free surface motion. *Comput. Method Appl. M.*, 69(3):277 – 324, 1988.

[12] J. Glimm, E. Isaacson, D. Marchesin, and O. McBryan. Front tracking for hyperbolic systems. *Ma. Comput. Sci. Eng.*, 2(1):91–119, 1981.

[13] J. Glimm, X. Li, Y. Liu, Z. Xu, and N. Zhao. Conservative front tracking with improved accuracy. *SIAM J. Numer. Anal.*, 41(5):1926–1947, 2003.

[14] J. Du, B. Fix, J. Glimm, X. Jia, X. Li, Y. Li, and L. Wu. A simple package for front tracking. *J. Comput. Phys.*, 213(2):613 – 628, 2006.

[15] R. Menikoff and B. J. Plohr. The Riemann problem for fluid flow of real materials. *Rev. Mod. Phys.*, 61(1):75, 1989.

[16] C. W. Hirt and B. D. Nichols. Volume of fluid (vof) method for the dynamics of free boundaries. *J. Comput. Phys.*, 39(1):201–225, 1981.

[17] W. F. Noh and P. Woodward. SLIC (simple line interface calculation). In *Proceedings of the Fifth International Conference on Numerical Methods in Fluid Dynamics*, pages 330–340. Springer, 1976.

[18] S. Osher and J. A. Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *J. Comput. Phys.*, 79(1):12 – 49, 1988.

[19] R. P. Fedkiw, T. Aslam, B. Merriman, and S. Osher. A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method). *J. Comput. Phys.*, 152(2):457–492, 1999.

[20] G. H. Miller and D. Trebotich. An embedded boundary method for the Navier-Stokes equations on a time-dependent domain. *Commun. Appl. Math. and Comput. Sci.*, 7(1):1 – 31, 2012.

[21] T. J. Ligocki, P. O. Schwartz, J. Percelay, and P. Colella. Embedded boundary grid generation using the divergence theorem, implicit functions, and constructive solid geometry. *J. Phys. Conf. Ser.*, 125:012080 (5pp), 2008.

[22] J Glimm, X. L. Li, Y. Liu, and N. Zhao. Conservative front tracking and level set algorithms. *P. Natl. Acad. Sci.*, 98(25):14198–14201, 2001.

[23] M. J. Berger, C. Helzel, and R. J. Leveque. h-box methods for the approximation of hyperbolic conservation laws on irregular grids. *SIAM J. Numer. Anal.*, 41(3):893, 2003.

[24] P. Colella, H. M. Glaz, and R. E. Ferguson. Multifluid algorithms for Eulerian finite difference methods, 1996. manuscript.

[25] G. H. Miller and E. G. Puckett. A high-order Godunov method for multiple condensed phases. *J. Comput. Phys.*, 128(1):134 – 164, 1996.

[26] I.-L. Chern and P. Colella. A conservative front tracking method for hyperbolic conservation laws. *Lawrence Livermore National Laboratory*, LLNL Report No. UCRL-97200, 1987.

[27] J. B. Bell, P. Colella, and M. L. Welcome. Conservative front-tracking for inviscid compressible flow. In A D Vakili and C Gauthier, editors, *American Institute of Aeronautics and Astronautics Conference*, pages 24–26, 1991.

[28] P. Colella, D. T. Graves, B. J. Keen, and D. Modiano. A Cartesian grid embedded boundary method for hyperbolic conservation laws. *J. Comput. Phys.*, 211(1):347 – 366, 2006.

[29] C. Gatti-Bono, P. Colella, and D. Trebotich. A second-order accurate conservative front-tracking method in one dimension. *SIAM J. Sci. Comput.*, 31(6):4795–4813, 2010.

[30] T. Belytschko and T. Black. Elastic crack growth in finite elements with minimal remeshing. *Int. J. Numer. Meth. Eng.*, 45(5):601 – 620, 1999.

[31] J. Chessa and T. Belytschko. An extended finite element method for two-phase fluids. *J. Appl. Mech.*, 70(1):10–17, 2003.

[32] A. J. Chorin. Numerical solution of the Navier-Stokes equations. *Math. Comput.*, 22(104):745–762, 1968.

[33] J. Chessa and T. Belytschko. An enriched finite element method and level sets for axisymmetric two-phase flow with surface tension. *Int. J. Numer. Meth. Eng.*, 58(13), 2003.

[34] C. S. Peskin. The immersed boundary method. *Acta Numerica*, 11:479 – 517, 2002.

[35] M. J. Berger and R. J. LeVeque. Stable boundary conditions for Cartesian grid calculations. *Comput. Syst. Eng.*, 1(2):305 – 311, 1990.

[36] G. H. Miller and P. Colella. A conservative three-dimensional Eulerian method for coupled solid-fluid shock capturing. *J. Comput. Phys.*, 183(1):26 – 82, 2002.

[37] P. Colella and P. R. Woodward. The piecewise parabolic method (ppm) for gas-dynamical simulations. *Journal of computational physics*, 54(1):174–201, 1984.

[38] G. H. Miller and P. Colella. A high-order eulerian godunov method for elastic–plastic flow in solids. *Journal of computational physics*, 167(1):131–176, 2001.

[39] R. B. Pember, J. B. Bell, P. Colella, W. Y. Crutchfield, and M. L. Welcome. An adaptive Cartesian grid method for unsteady compressible flow in irregular regions. *J. Comput. Phys.*, 120(2):278–304, 1995.

[40] D. Peng, B. Merriman, S. Osher, H. Zhao, and M. Kang. A PDE-based fast local level set method. *J. Comput. Phys.*, 155(2):410 – 438, 1999.

[41] G.-S. Jiang and D. Peng. Weighted ENO schemes for Hamilton-Jacobi equations. *SIAM J. Sci. Comput.*, 21(6):2126–2143, 2000.

[42] A. du Chéné, C. Min, and F. Gibou. Second-order accurate computation of curvatures in a level set framework using novel high-order reinitialization schemes. *J. Sci. Comput.*, 35:114–131, 2008.

[43] D. Adalsteinsson and J. A. Sethian. The fast construction of extension velocities in level set methods. *J. Comput. Phys.*, 148(1):2 – 22, 1999.

[44] P. Colella. Volume-of-fluid methods for partial differential equations. In *Godunov Methods: Theory and Applications*, pages 161–176. Kluwer Academic/Plenum Publishers: New York, 2001.

[45] P. Colella. Multidimensional upwind methods for hyperbolic conservation laws. *J. Comput. Phys.*, 87(1):171 – 200, 1990.

[46] J. Saltzman. An unsplit 3D upwind method for hyperbolic conservation laws. *J. Comput. Phys.*, 115(1):153 – 168, 1994.

[47] H.-K. Zhao, T. Chan, B. Merriman, and S. Osher. A variational level set approach to multiphase motion. *J. Comput. Phys.*, 127(1):179 – 195, 1996.

[48] M. Sussman, P. Smereka, and S. Osher. A level set approach for computing solutions to incompressible two-phase flow. *J. Comput. Phys.*, 114(1):146 – 159, 1994.

[49] G. A. Sod. A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws. *J. Comput. Phys.*, 27(1):1 – 31, 1978.

[50] M. Brouillette. The Richtmyer-Meshkov instability. *Annu. Rev. Fluid Mech.*, 34(1):445–468, 2002.

[51] R. D. Richtmyer. Taylor instability in shock acceleration of compressible fluids. *Commun. Pur. Appl. Math.*, 13(2):297–319, 1960.

[52] B. D. Collins and J. W. Jacobs. PLIF flow visualization and measurements of the Richtmyer-Meshkov instability of an air/SF6 interface. *J. Fluid Mech.*, 464:113–136, 2002.

[53] E. F. Toro. *Riemann solvers and numerical methods for fluid dynamics*, volume 16. Springer, 1999.