Conservative Front Tracking in Higher Space Dimensions

James Glimm *

Department of Applied Mathematics and Statistics

University at Stony Brook, Stony Brook, NY 11794-3600;

Center for Data Intensive Computing

Brookhaven National Laboratory, Upton, NY 11793-5000,

Xiao Lin Li, † and Yingjie Liu ‡

Department of Applied Mathematics and Statistics

University at Stony Brook, Stony Brook, NY 11794-3600;

September 12, 2001

Abstract

We propose a fully conservative Front Tracking algorithm in two space dimension. The algorithm first uses the point shifted algorithm [12] on two adjacent time levels and then constructs space time hexahedra as computational units. We develope and prove a successful geometric construction under certain interface requirement. This algorithm has a first order local truncation error for cells near the tracked discontinuity, which is an improvement by one order of accuracy over most finite difference schemes, which have $\mathcal{O}(1)$ local truncation errors near discontinuities.

*Supported by the MICS program of the U.S. Department of Energy DE-FG02-90ER25084, the Department of Energy Contract DE-AC02-98CH1-886 and the Office of Inertial Fusion, the Army Research Office, grant DAAD19-01-1-0642, the National Science Foundation, grant DMS-0102480 and Los Alamos National Lab Sponsor ID #26730001014L

†This research is supported by the U.S. Department of Energy, Sponsor ID #DEFG0398DP00206 and Los Alamos National Lab Sponsor ID #26730001014L

‡This research is supported by the U.S. Department of Energy, Sponsor ID #DEFG0290ER25084 and Los Alamos National Lab Sponsor ID #26730001014L

1 Introduction

We propose and demonstrate a tracking finite difference algorithm which is (a) fully conservative and (b) improves local truncation error by one order (from $\mathcal{O}(1)$ to $\mathcal{O}(\Delta x)$ near tracked discontinuities.

Discontinuities in the solutions of systems of nonlinear hyperbolic conservation laws are widely recognized as a primary difficulty for numerical simulation. Surprisingly, the nonlinearities actually help, as they cause information, which flows along solution characteristics, to flow into the discontinuity, and disappear there. The nonlinear discontinuities (shock waves) function much as a black hole in this regard. Included in this flow of information are the solution errors generated by the nonlinear discontinuity. Because the nonlinear discontinuities absorb errors associated with their numerical approximation, these errors do not grow or spread with time.

Nonlinear equations also have linear discontinuities. In gas dynamics these are the contact discontinuities, across which temperature or shear velocities can be discontinuous. Errors in these modes are never forgiven and never dissipated. For this reason the linear modes are more difficult to control numerically. The dominant numerical solution error is typically associated with discontinuities in these linear modes and occurs as diffusion of mass, vorticity, and temperature. These errors increase with time.

Front Tracking was introduced to give special treatment to solution with discontinuities. Perceptions that Front Tracking software difficulties would be insurmountable were too pessimistic, and a robust, validated code has been developed and used in production simulation of fluid instabilities [5, 7, 6, 4]. See also the URL http://www.ams.sunysb.edu/∼shock/FTdoc/FTmain.html. Here we address an algorithmic issue: formulation of a conservative tracking algorithm. In its original formulation, conservation was enforced only in regular grid cells, those not cut by the tracked front. The missing points of the computation stencil, in the case of a front cutting through the stencil, are filled in as ghost cells, with the state values obtained by extrapolation from nearby front states of the same component. Thus the state values are double valued near the front, with the left-component states extending by extrapolation for a small distance into the right component, and *vice versa*. The use of ghost cell states was introduced into Front Tracking in 1980 [9]. With the ghost states thus defined, the interior solver follows a conventional finite difference algorithm.

In the level set method [3] and the original Front Tracking, ghost cells constructed near the front (but using entropy extrapolation) allows a standard difference operation update. As with Front Tracking, the ghost cell extrapolation is non conservative and leads to $\mathcal{O}(1)$ local truncation error.

Here we propose an algorithm which is conservative for all grid cells, including the irregular ones cut by the front. The algorithm we propose is related to earlier work of Swartz and Wendroff [13], Harten and Hyman [11], and Colella and Chern [2], but differs from these works in several ways. We emphasize here tracking of a contact, rather than the shock tracking of [2]. Our support for fronts is fully general and can be used for unstable, convoluted, and bifurcating interfaces. The 1D version of this algorithm [8] is formally second order accurate in the L_1 norm except for interactions of tracked waves. An important difference with [13] is our discussion of the extension to higher dimensions.

2 The Two Dimensional Algorithm

Consider the two space dimensional system of conservation laws

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} + \frac{\partial g(u)}{\partial y} = 0, \tag{1}$$

defined in a spatial domain Ω. Assume that Ω can be partitioned by a uniform square grid and the boundary is along the grid lines. The side of a cell in the grid is of length Δx.

Our algorithm is organized into three main steps. The first is the construction of a spatial grid locally conforming to the old (time t_n) and new propagated (time t_{n+1}) *INTERFACE*. The second is construction of a space time grid joining these two, and the third is a finite volume discretization associated with the space time conforming grid.

In the following two sections, we emphasize the major results and leave the more technical lemmas and some proofs to the Appendix.

2.1 The Point Shifted Algorithm

In this section we describe a simple point shifted algorithm to achieve an *INTERFACE* conforming grid node displacement at a fixed time level. We begin with hypotheses which are requirements on the topology of the *INTERFACE* and the size of the cell. In the present study the *INTERFACE*s are topologically equivalent to a union of line segments or circles. Thus we postulate that triple or multiple *CURVE* intersection points do not occur. *INTERFACE*s that involve topological change during the time evolution can be resolved by premerging when the distance of the gap to be merged is within Δx. Analysis of this step is out of the scope of the present paper, but is supported in the numerical implementation of the point shifted construction.

The discretized *INTERFACE* [10] is a disjoint union of non intersecting *CURVE*s. Each *CURVE* is piecewise linear and connected, and composed of *BOND*s. Each *BOND* is a pair of *INTERFACE POINT*s or *POINT*s, and (conceptually) the straight line segment joining them. Each *CURVE* is assigned an orientation which remains unchanged during the propagation of the *INTERFACE*. If all the *POINT*s are on the interior of cell edges with at most one *POINT* occuring on the interior of any given grid cell edge, then the *INTERFACE* is called grid based [7]. Propagation of the *POINT*s of a grid based *INTERFACE* will yield a general *INTERFACE*, not grid based, as there is no reason for a propagated *POINT* to lie on a grid cell edge, just because it starts on one. According to the grid based construction of [10], we consider this propagated *INTERFACE* as a collection of polygonal *CURVE*s in $\Re^2$. Crossing points of the *CURVE* with grid cell edges are inserted as new *POINT*s. The propagated old *POINT*s will be deleted (named images of propagation in this sense), but their ordering along the *CURVE*s will be retained for later use in the construction of space time interface. The *CURVE* is then reconstructed, as straight line segments joining these new *POINT*s. In this process, the *CURVE* is displaced by an amount $\mathcal{O}(\Delta x^2)$, assuming that the *CURVE* is smooth, so that all angles between neighboring *BOND*s are $\mathcal{O}(\Delta x)$. Also all images of propagated *POINT*s on the original *CURVE* can be projected onto the grid based *CURVE*, with their ordering unchanged and a maximum displacement $\mathcal{O}(\Delta x^2)$.

This grid based *INTERFACE* is the starting point for the interface conforming volume grid which we construct here. An *INTERFACE* with a displaced rectangularly indexed volume grid is called a point shifted *INTERFACE*. It is point shifted if the grid corners have been displaced so that all *POINT*s are at displaced grid cell corners and all *BOND*s are either the edges or diagonals of displaced grid cells.

Here we construct an algorithm which yields a point shifted *INTERFACE* at each time step. In two space dimensions, we use the front propagation algorithm developed in [5, 7, 6, 4] to follow the *INTERFACE* evolution. We first projected the propagated *INTERFACE* to be grid based [7], by inserting new *POINT*s at cell edge crossings, and then removing old *POINT*s. To this *INTERFACE* we apply the point shifted algorithm [12] near the front on each time level to align the grid nodes nearest to the *INTERFACE* so that there is no intersection between the *INTERFACE* and the interior of cell edges (*i.e.*, the *INTERFACE* passes through displaced grid cell corners only and thus lies on the diagonals and edges of displaced grid cells). Some *POINT*s are deleted in this construction; again with maximum displacement $\mathcal{O}(\Delta x^2)$ of the propagated old *POINT*s and the propagated *CURVE*s. We call the result an interface conforming grid node displacement.

Hypothesis 1 *The INTERFACE is assumed to be grid based. Each CURVE is topologically equivalent to a line segment with its two end points on the boundary, or*

a circle contained in the interior of Ω*. (Triple points where three or more CURVEs meet at a point are disallowed.) Each CURVE has at least three BONDs and the maximum angle between two adjacent BONDs is* $\mathcal{O}(\Delta x)$*. All interior POINTs of the CURVE must be interior to* Ω*. There is no topological change of the INTERFACE during the time interval of computation.*

To avoid consideration of degenerate cases, we assume that *POINT*s never lie exactly at center of a grid cell edge.

Hypothesis 2 *At most one BOND intersects the interior of a given cell, and if this occurs, the CURVE separates the interior of the cell into two non-trivial domains.*

This Hypothesis implies that at most two edges of a cell intersect the *INTERFACE*.

Hypothesis 3 *No CURVE is totally contained within a square of side* $2\Delta x$ *made up of four cells.*

A grid node has grid distance d to the *INTERFACE* if there is a grid line segment of length d connecting the grid node to the *INTERFACE*. A grid node will have multiple grid distances. We call the smallest one the shortest grid distance. A grid node is called fixed if all its grid distances are greater than $\Delta x/2$.

We call a grid node which is not on the boundary shiftable if one of the following three conditions holds: (See Fig. 1.)

(I1) Exactly one of its grid distances is less than $\Delta x/2$. If this grid distance is on a grid line parallel to the x-axis, we call it x-shiftable, otherwise we call it y-shiftable.

(I2) Exactly two of its grid distances are less than $\Delta x/2$ and they are not on the same grid line. If the shortest grid distance is on a grid line parallel to the x-axis, we call it x-shiftable; if the shortest grid distance is on a grid line parallel to the y-axis we call it y-shiftable. In the degenerate case of equal grid distance, the node is both x-shiftable and y-shiftable.

(I3) Exactly three of its grid distances are less than $\Delta x/2$. In this case, the node is shiftable in the direction of the single grid distance.

We call a grid node which is on the boundary shiftable if it is not at a boundary corner, *i.e.* not at the intersection of two boundary lines, not on the *INTERFACE* and if the following condition holds: (See Fig. 1.)

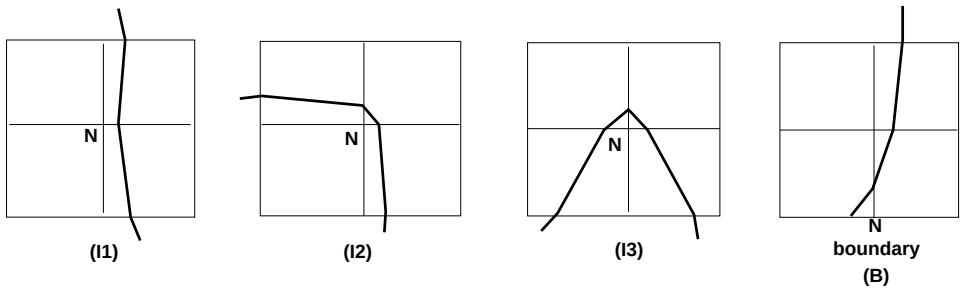

Figure 1: Several cases for a shiftable node N

(B) Exactly one of its grid distances along a boundary grid line is less than $\Delta x/2$. We call the grid node x-shiftable if the boundary grid line is parallel to x-axis; otherwise we call it y-shiftable.

If a node is neither fixed or shiftable, we call it unshiftable.

Hypothesis 4 *At each time level during the computation, every grid node is either fixed or shiftable.*

For each grid node, there is either one (I1, I3, B) or two (I2) interface *POINT*s to which it can be shifted. For each interface *POINT*, there is at most one grid node which can be shifted to it. Boundary grid nodes can be shifted only to boundary interface *POINT*s and interior (non boundary) grid nodes can be shifted to non boundary interface *POINT*s only.

The point shifted algorithm consists of shifting all shiftable nodes to interface *POINT*s with, for example, a choice of the x-direction in case of ambiguity. The *POINT*s to which no grid nodes are shifted are deleted. We state an obvious result concerning the algorithm.

Proposition 1 *Assume Hypotheses 1-4. If a grid node is shifted to the INTERFACE, then it is shifted to a POINT located at an intersection of the INTERFACE with a grid line. The intersection point lies within an open circle of radius* $\Delta x/2$ *centered at the original position of the node.*

Theorem 1 *Assume Hypotheses 1-4. Then the topology of the grid remains unchanged. Each cell area is between* $0.5\Delta x^2$ *and* $2.5\Delta x^2$. *Each POINT of the INTERFACE is a shifted grid cell corner and each BOND is a shifted grid cell edge or diagonal.*

After the point shift algorithm, it is easily seen from Proposition 1 and Theorem 1 that the length of each *BOND* is no more than $\sqrt{5}\Delta x$. Because of the bound on *BOND* angles, the *BOND* is within a $\mathcal{O}(\Delta x^2)$ displacement to the grid based *CURVE* segment it approximates. We have the following corollary.

Corollary 1 *Assume Hypotheses 1-4. If the grid based CURVE is within* $\mathcal{O}(\Delta x^2)$ *displacement to the smooth interface curve, then after point shift algorithm, the CURVE is still a* $\mathcal{O}(\Delta x^2)$ *approximation to it.*

2.2 Construction of the Space-Time Hexahedra

We require a space-time triangulated interface surface joining the two spatial *INTERFACE*s at times t_n and t_{n+1}. This construction is the major task of the present section.

The point shifted algorithm does not change the rectangular index structure of the mesh. Thus we connect the nodes of a cell D_i^n at time $t = t_n$, to the nodes of its corresponding cell D_i^{n+1} at time $t = t_{n+1}$ to form a space-time hexahedron. We call D_i^{n+1} the top of the hexahedron and D_i^n the bottom. If the both cells have not been affected by the point shifted algorithm we call the hexahedron *regular*, otherwise it is called *irregular*. We call a hexahedron *mixed* if the interface passes through its interior; otherwise it is *pure*. The *mixed* hexahedra are divided into *pure partial* hexahedra, and if necessary, these are combined with neighbors to form the finite volume space-time grid suitable for construction of a conservative difference algorithm in Sec. 2.3

Two hexahedra are adjacent if they share a non-trivial surface which is not on the space time interface. It is easy to see that two adjacent hexahedra must be on the same side of the space time interface. We consider *INTERFACE*s after the point shift algorithm. In this case, each *POINT* is a displaced, or shifted grid node, and all *BOND*s, connecting adjacent *POINT*s are edges or diagonals of displaced or shifted grid cells. We also observe that the *POINT*s P_1 and P_2, defined at a common or adjacent time level, that is both at times t_n or t_{n+1}, or one at time t_n and the other at time t_{n+1}, share a common space time hexahedron if and only if they are identical, adjacent or diagonally adjacent as shifted grid nodes in space. We say that P_1 and P_2 are spatially nearest neighbors in this case. These points are strictly spatially nearest neighbors if they are identical or adjacent grid nodes (diagonal adjacency excluded) in space. Besides all the hypotheses in Section 2.1, we also assume that the CFL number is less than 1/2 throughout this section so that each *POINT* of the *INTERFACE* can be propagated a distance less than $\Delta x/2$.

Hypothesis 5 *The CFL number is less than* 1/2.

We introduce terminology for Proposition 2 and the supporting Lemmas 4-7. (See Appendix 2.)

Let B^n be a BOND connecting adjacent POINTs P_1 and P_2 at the time level t_n. At the time level t_{n+1} B^n has been propagated and point shifted to become the INTERFACE polygonal segment b^{n+1} with a left end point M_1 as the image of P_1 and a right end point M_2 as the image of P_2.

Proposition 2 *Assume Hypotheses 1-5. In the ordering of POINTs along b^{n+1}, those spatially nearest neighbor to P_1 only (if any) lie closest to M_1, followed by POINTs (if any) spatially nearest neighbor to both P_1 and P_2, followed by POINTs (if any) spatially nearest neighbor to P_2 only. The middle set of POINTs necessarily occurs if both of the other two are non empty.*

Proposition 3 *Assume Hypotheses 1-5. Let B_1 be a BOND at the time level t_{n+1} connecting adjacent POINTs P_1 and P_2. Suppose that adjacent time level t_n POINTs are propagated to B_1. Then those spatially nearest neighbor only to P_1 (if any) occur first in the INTERFACE order, followed by POINTs (if any) spatially nearest neighbor to both P_1 and P_2, followed by POINTs (if any) spatially nearest neighbor only to P_2. The middle set of POINTs necessarily occurs if both of the other two are non empty.*

We construct the space time interface as triangles whose edges joint *POINT*s on successive time levels t_n and t_{n+1}, and which are spatially nearest neighbor. Each edge of the triangle is then a surface edge or diagonal, or interior diagonal of a single space time hexahedron. The only further requirement, to avoid tangling of the space time interface, is that the edges preserve order along the interface, as a mapping from t_n *POINT*s onto t_{n+1} *POINT*s.

The correspondence between interface *POINT*s is nearly determined by Propositions 2 and 3. There is an alternation between expansion and contraction events in this pairing, as defined below;

Expansion: The *INTERFACE* segment b^{n+1} defined by propagation and shifting of the *BOND* B contains one or more *POINT*s.

Contraction: A single time level t_{n+1} *BOND* contains the images defined by propagation and shifting of one or more consecutive time level t_n *POINT*s.

For an Expansion event, following the terminology of Proposition 2, we connect (introducing triangle edges) P_1 to all *POINT*s on $[M_1, M_2]$ which are spatially nearest neighbor to P_1 but not P_2. We connect to P_2 all *POINT*s on $[M_1, M_2]$ spatially nearest neighbor to P_2 but not P_1. The *POINT*s, if any, spatially nearest neighbor to both

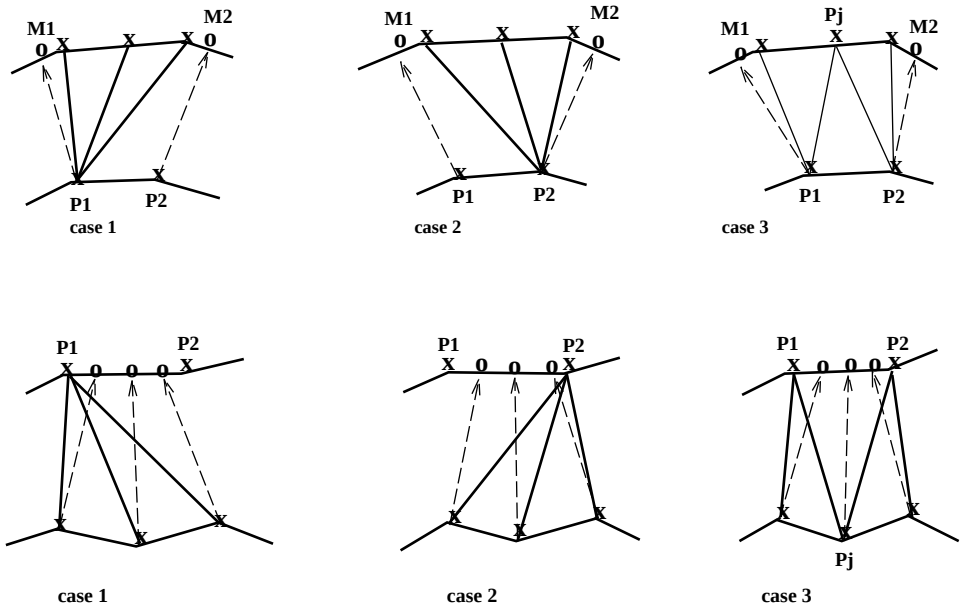

Figure 2: Construction of space time interface joining the time t_n and t_{n+1} interfaces. Upper three frames, expansion case; lower three frames, contraction case. In case 1, all *POINT*s are spatially nearest neighbor to P_1, in case 2 to P_2, and case 3 includes at least one mixed *POINT*. "x": interface *POINT*; "o": image of propagation.

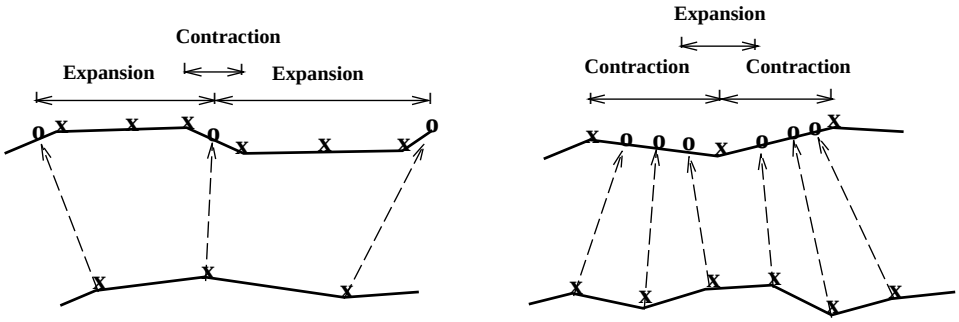

Figure 3: Adjacent events are necessarily of opposite type. "x": interface *POINT*; "o": image of propagation.

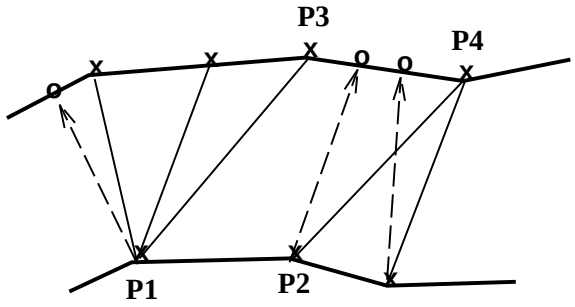

Figure 4: Either P_1 connects to P_4 or P_2 connects to P_3. "x": interface *POINT*; "o": image of propagation.

P_1 and P_2 are split; those before and including a reference *POINT* P_j are connected to P_1, while those including P_j and after are connected to P_2. See Fig. 2, upper frames.

For a Contraction event, following the terminology of Proposition 3, we connect P_1 to the points propagating onto $[P_1, P_2]$, which are spatially nearest neighbor to P_1 but not P_2. Similarly we connect to P_2 those spatially nearest neighbor to P_2 but not P_1. The remainder, spatially nearest neighbor to both P_1 and P_2 are split as above, with a separating point P_j connected to both P_1 and P_2. See Fig. 2, lower frames.

Two events of the same type can not be adjacent to each other. See Fig. 3. Therefore it remains to show that the triangles from the Expansion and Contraction events join, and complete the space time interface. Assume that the Expansion is to the left of the Contraction. There is a gap between the two sets of triangles only if all Expansion triangles join to the left (case 1) and the contraction triangles join to the right (case 2) as we now assume. The gap is a quadrilateral and one of its diagonals must be added to complete the triangulation. According to Lemma 6 (see Appendix 2), either the right most time t_{n+1} *POINT* of the Expansion event is spatially nearest neighbor to the left most time t_n *POINT* of the Contraction event, or the right most time t_n *POINT* of the Expansion event is spatially nearest neighbor to the left most time t_{n+1} *POINT* of the Contraction event, so that we can connect one diagonal pair of *POINT*s of the quadrilateral to triangulate the gap. See Fig. 4. The case of a Contraction to the left of an adjacent Expansion is similar. In the case that a spatial interface *CURVE* is topologically equivalent to a circle, each space-time interface event is adjacent to a space-time interface event on the left, and another one on the right. Therefore the above process fully triangularizes the space time surface joining the two *CURVE*s at the two time levels. If a *CURVE* has its two end points on the boundary, each of the end points (which becomes a *POINT* after the point shift

algorithm) must be in the same space hexahedron at time levels t_n and t_{n+1} so that we can connect the end point at time levels t_n to the end point at time level t_{n+1} to complete the triangulation. This requires that end point should be propagated less than a distance $\Delta x/2$ along the boundary grid line, which depends on the states near the end point, the angle at which the *INTERFACE* intersects the boundary and the time step size (or the CFL number) we choose. We formulate this requirement as the following hypothesis:

Hypothesis 6 *The CFL number chosen during each time step must ensure that each intersection point of the CURVE with the boundary is propagated less than a distance $\Delta x/2$ along the boundary grid line.*

We summarize the above discussion with the following theorem:

Theorem 2 *Assume Hypothesis 1-6. After the above triangulation process, each triangle on the space time interface will be on the face or interior of a point shifted space time hexahedron, with its base being an edge or diagonal of the top (or bottom) cell of the hexahedron and its other vertex being a grid node of the bottom (or top) cell of the hexahedron.*

Proof: Each resulting triangle on the space-time interface has an edge, say the base, which corresponds to an interface *BOND*, *i.e.*, an edge or a diagonal of a cell on t_n (or t_{n+1}), while the other vertex of triangle corresponds to an interface *POINT*, *i.e.*, a point shifted grid node on t_{n+1} (or t_n). Since the vertex is spatially closest to the other two vertices of the triangle, it is easy to see that all of its three vertices will share a common space-time hexahedron. The proof is complete.

The *mixed* hexahedron is separated into two parts, each of which lies on one side of the space time interface. These parts are called *pure partial* hexahedra. We can similarly define a cell to be *regular*, *irregular*, *pure*, *mixed* or *partial*. Any *partial* hexahedra with a trivial top will be merged with an adjacent *pure* hexahedron or *partial* hexahedra having a nontrivial top. Recall that two adjacent hexahedra are on the same side of the interface. The merging process can be accomplished as follows:

Merge every pure or partial hexahedron having a nontrivial top with adjacent partial hexahedra having trivial tops which have not been merged elsewhere. Denote the resulting polyhedra the intermediate hexahedra. Merge every intermediate hexahedron repeatively with adjacent partial hexahedron having a POINT top which have not been merged elsewhere. Denote the resulting polyhedra the big hexahedra.

After the merging process, we also call the remaining *pure* and *partial* hexahedra *big* hexahedra for equivalence in the next computation. The following lemma ensures the success of the above algorithm.

Lemma 1 *Assume Hypothesis 1-6. If a polyhedron is constructed by merging any number of adjacent partial hexahedra with trivial tops, then the polyhedron will be adjacent to a pure or partial hexahedron.*

Proof: At least one non trivial piece (a triangle) of the side surface of the polyhedron is not on the boundary or the space time interface, otherwise the topological structure of the *INTERFACE* changes during this time step and Hypothesis 1 is violated. The proof is complete.

Theorem 3 *Assume Hypothesis 1-6. After the above merging process,*

(1) *every partial hexahedron with a trivial top will be merged into a big hexahedron with non trivial top;*

(2) *every big hexahedron has a non trivial top which is a pure cell or a partial cell.*

Proof: Let H be a *pure* or *partial* hexahedron with trivial top, then it is adjacent to a *pure* or *partial* hexahedron from Lemma 1. We can separate it into several cases.

(a) The top of H contains a *BOND* of the t_{n+1} *INTERFACE*. Then the *BOND* must be the edge of a *pure* or *partial* cell (say C) on the same side of the space time interface since the *BOND* can not be on the boundary according to Hypothesis 1. Therefore H must be adjacent to the *pure* or *partial* hexahedron with C as its top and the algorithm will merge all such partial hexahedra into the *intermediate* hexahedra.

(b) The top of H is a *POINT* and H is adjacent to a *partial* hexahedron with a trivial top which consists of *BOND*s, or to a *pure* or *partial* hexahedron with a non trivial top. Then it will be merged either with an *intermediate* hexahedron (due to (a)) or with a *pure* or *partial* hexahedron with a non trivial top.

(c) The top of H is a *POINT* P and H is adjacent only to *partial* hexahedra with a *POINT* top P. Let M_1 be the polyhedron resulting from merging all the adjacent *partial* hexahedra with a *POINT* top P. Then M_1 consists of at most four *partial* hexahedra with the *POINT* top P since P can belong to at most four *pure* or *partial* hexahedra on the same side of the space time interface. From Lemma 1 M_1 is adjacent to a *pure* or *partial* hexahedron with non trivial top or to a *partial* hexahedron with a trivial top which consists of *BOND*s. In the first case M_1 will be merged with the *pure* or *partial* hexahedron with non trivial top. The second case is similar to the previous case (b).

The first statement of the Theorem is proved. The second statement is from the observation that a *pure* or *partial* hexahedron with a non trivial top (a *pure* cell or a partial cell) will merge only with hexahedra with trivial tops. The proof is complete.

Note that the top of a *big* hexahedron (including the trivial parts) is within the union of the top cell (or *partial* cell) and its spatially closest neighbors, therefore the total number of *pure* or *partial* hexahedra in the *big* hexahedron is bounded. Actually in most cases of the computation the merging process yields the *big* hexahedron consisting of two *pure* or *partial* hexahedra. The number of *pure* or *partial* hexahedra in the *big* hexahedron could become larger if the radius of curvature of the moving *CURVE* is closer to the mesh size.

Note that the merging process does not increase the computational complexity since the net outflux of a *big* hexahedron is equal to the sum of the net outflux of each *pure* or *partial* hexahedron contained in it.

Since after the merging process the *big* hexahedron will have either a cell or a *partial* cell as its top, this construction does not change the reconstruction of state functions at the next time level discussed in the next section.

2.3 The Reconstruction, Limiter and the Numerical Scheme

Suppose at the time level $t = t_n$ we know the approximate state averages on each cell, *regular*, *irregular* or *partial*. We want to reconstruct a piecewise linear state function on these cells with 2nd order accuracy. The reconstruction of the piece wise linear state function on *irregular* cells follows [1], with modifications to the limiter and some simplification. Let D_i^n be a *pure* cell, *regular*, *irregular*, or *partial* with approximate state average $\mathcal{U}_i$ and cell center (centroid) Y_i, surrounded by any of these types of cells $D_j^n, D_k^n, D_l^n, D_m^n$ with approximate state averages $\mathcal{U}_j^n, \mathcal{U}_k^n, \mathcal{U}_l^n, \mathcal{U}_m^n$ and cell centers Y_j, Y_k, Y_l, Y_m respectively, on the same side of the *INTERFACE* . Let $\tilde{\mathcal{U}}_i = \mathcal{U}_i + (a, b) \cdot (X - Y_i)$ be the 2nd order accurate linear state function on D_i^n, where a, b are two constants. Choose any two surrounding cells, say D_j^n, D_k^n so that Y_i, Y_j, Y_k are not colinear. We can determine a, b by solving the following equation:

$$\begin{aligned} \tilde{\mathcal{U}}_i(Y_j) &= \mathcal{U}_j^n, \\ \tilde{\mathcal{U}}_i(Y_k) &= \mathcal{U}_k^n. \end{aligned} \tag{2}$$

Further, for the solution of the above equation to be well conditioned, we require the angle θ formed by line segment $\overline{Y_iY_j}$ and $\overline{Y_iY_k}$ to satisfy $0 < \theta_1 < \theta < \theta_2 < \pi$ where θ_1, θ_2 are two constants. We repeat the above procedure until we find all possible solutions, say, a_i, b_i, for all $0 \le i \le I$ where $I \le 4$. Then we set $a = \text{minmod}\{a_1, \cdots, a_I\}$ and $b = \text{minmod}\{b_1, \cdots, b_I\}$. When there are not enough surrounding cells on the same side of the *INTERFACE*, we choose $a, b = 0$ so that the reconstruction becomes first order.

When D_i^n is a *regular* cell surrounded by *regular* cells, the reconstruction process is simpler. Let the cell center of D_i^n be $(i_1\Delta x, i_2\Delta y)$ with neighboring cell centers

$\{((i_1+k_1)\Delta x,(i_2+k_2)\Delta y)|k_1,k_2=-1,0,1\}$. Let

$$\begin{aligned}\text{xslope}_\text{i} &= \text{minmod}\{[\mathcal{U}((i_1+k_1)\Delta x,(i_2+k_2)\Delta y)- \\ &\quad \mathcal{U}((i_1+k_1-1)\Delta x,(i_2+k_2)\Delta y)]/\Delta x \mid \\ &\quad k_1=0,1;k_2=-1,0,1\},\end{aligned} \tag{3}$$

and

$$\begin{aligned}\text{yslope}_\text{i} &= \text{minmod}\{[\mathcal{U}((i_1+k_1)\Delta x,(i_2+k_2)\Delta y)- \\ &\quad \mathcal{U}((i_1+k_1)\Delta x,(i_2+k_2-1)\Delta y)]/\Delta y \mid \\ &\quad k_1=-1,0,1;k_2=0,1\},\end{aligned} \tag{4}$$

and define

$$\tilde{\mathcal{U}}_i = \mathcal{U}_i + \text{xslope}_\text{i}\cdot(x-i_1\Delta x) + \text{yslope}_\text{i}\cdot(y-i_2\Delta y).$$

This is clearly a second order reconstruction which is better suited in multiple dimensional problem than operator splitting single line reconstruction (or limiter) for a uniform rectangular grid, because for example an untracked discontinuity in 2D may be in the form of a strip of width between $2\Delta x$ and $3\Delta x$. When the strip is almost parallel to and fully covers the line in which the single line reconstruction occurs, one cannot expect the limiter to choose any smooth solutions nearby.

Next we apply the technique in Section 2.2 to generate space-time hexahedra between time levels t^n and t^{n+1}. Let H be a big hexahedron with top D^{n+1} and bottom D^n, and triangle sides $\{S_i\}$ with a unit outer normal n_i and centroid Z_i. Notice that some elements of the $\{S_i\}$ may be on the approximate space time interface. Integrating (1) over H, we obtain

$$|D^{n+1}|\ U^{n+1} = |D^n|\ U^n - \sum_i \int_{S_i} (u,f,g)\cdot n_i ds. \tag{5}$$

Here $|D^n|$ represents the area of D^n and similarly $|S_i|$ is the area of S_i. The numerical scheme can be written as

$$|D^{n+1}|\mathcal{U}^{n+1} = |D^n|\mathcal{U}^n - \sum_i |S_i|(\tilde{\mathcal{U}}_{i,m}, f(\tilde{\mathcal{U}}_{i,m}), g(\tilde{\mathcal{U}}_{i,m}))\cdot n_i, \tag{6}$$

where $\tilde{\mathcal{U}}_{i,m}$ can be calculated as follows: First use a Cauchy-Kowalewski procedure on the reconstructed state function on each side of S_i to get 2nd order approximate states at Z_i on the respective side of S_i, say $\mathcal{U}_{i,l}$ and $\mathcal{U}_{i,r}$. If S_i is not on the tracked space time interface, we can simply use a Riemann solver, say R, to get the middle state on S_i, *i.e.*

$$\tilde{\mathcal{U}}_{i,m} = R(\mathcal{U}_{i,l},\ \mathcal{U}_{i,r}).$$

If S_i is on the tracked space time interface, we use the Riemann solver to get the left and the right side states $\tilde{\mathcal{U}}_{i,l}$ and $\tilde{\mathcal{U}}_{i,r}$ on the wave we are supposed to track, and

the wave speed ν_i. Then $\tilde{\mathcal{U}}_{i,m}$ in (6) can be replaced by either $\tilde{\mathcal{U}}_{i,l}$ or $\tilde{\mathcal{U}}_{i,r}$, depending on whether l or r is located within H or not. Also the n_i in (6) should be replaced by $\tilde{n}_i/|\tilde{n}_i|$, where $\tilde{n}_i = (-\theta_i \nu_i \sqrt{n_{ix}^2 + n_{iy}^2}, n_{ix}, n_{iy})$, $n_i = (n_{it}, n_{ix}, n_{iy})$ and θ_i is a sign function which is 1 if the tracked wave from the Riemann solver is in the direction of (n_{ix}, n_{iy}), -1 otherwise. Note that $\tilde{n}_i$ is normal direction of the tracked space time wave from the Riemann solver, therefore this modification ensures that the Rankine-Hugoniot condition is satisfied.

The finite volume difference algorithm constitutes a flux through each boundary of the full, *partial* and *big* hexahedron. Since the flux through a boundary face of the hexahedron is identical when viewed from either side of the face, we have

Theorem 4 $\sum_{cells} |D^n|\mathcal{U}^n$ *in the finite volume difference scheme is conserved so that its increment over any time interval is equal to the net influx at the boundary.*

Away from the *INTERFACE* the scheme is clearly a second order scheme. For the cells along the *INTERFACE*, its local truncation error is one order lower than in the 1D case since we use a piece wise linear *INTERFACE* and its local displacement error is $\mathcal{O}(\Delta x^2)$. The scheme is one order better than untracked schemes, which typically have $\mathcal{O}(1)$ local truncation error at the untracked fronts.

Theorem 5 *Suppose the exact space time interface and the solution on either side of it are smooth. Then the L_∞ local truncation error is $\mathcal{O}(\Delta x)$ for cells adjacent to the INTERFACE.*

Proof: Let the *INTERFACE* at t_n be the interpolation of the exact interface and let H be a big hexahedron adjacent to the approximate space time interface. We apply the finite volume scheme to obtain the approximate state average $\mathcal{U}_i^{n+1}$ at the time level t_{n+1}, with top T and bottom B and side boundaries $\{S_i\}$, where S_i is a triangle. The *INTERFACE* at time t_{n+1} has an $\mathcal{O}(\Delta x^2)$ displacement from the exact interface according to Corollary 1. The exact space time interface will cut H into two pieces. Let H_1 be the piece on the same side of the interface with H. Let T_1, B_1, and S^1 be the top, bottom and side boundaries of H_1 respectively. Let $U_{T_1}^{n+1}, U_{B_1}^n$ be the exact state averages over T_1 and B_1 respectively. Choosing $\mathcal{U}_B^n = U_{B_1}^n$, we want to show that $U_{T_1}^{n+1} - \mathcal{U}_T^{n+1} = \mathcal{O}(\Delta x)$. In fact from (6),

$$|T|\mathcal{U}_T^{n+1} = |B|\mathcal{U}_B^n - \sum_i |S_i|(\tilde{\mathcal{U}}_{i,m}, f(\tilde{\mathcal{U}}_{i,m}), g(\tilde{\mathcal{U}}_{i,m})) \cdot n_i. \tag{7}$$

The exact solution satisfies

$$|T_1|U_{T_1}^{n+1} = |B_1|U_{B_1}^n - \int_{S^1} (u, f(u), g(u)) \cdot n ds. \tag{8}$$

Note that $|B|\mathcal{U}_B^n - |B_1|U_{B_1}^n = \mathcal{O}(\Delta x^3)$ due to the interpolation error from the *INTERFACE* at time t_n. Also the numerical flux in (7) approximates the exact flux in (8) to at least $\mathcal{O}(\Delta x^3)$. In fact when S_i is not on the approximate space time interface this is easily seen since $\int_{S_i}(u,f,g)\cdot n_i ds = |S_i|(u,f,g)(Z_i)\cdot n_i + \mathcal{O}(\Delta x^4)$. When S_i is on the approximate space time interface. Because the approximate space time interface has an $\mathcal{O}(\Delta x^2)$ displacement error relative to the exact one, the difference between their respective areas is of $\mathcal{O}(\Delta x^3)$ due to the smoothness assumption of the exact space time interface and the area of $\bigcup S_i$ being $\mathcal{O}(\Delta x^2)$. Also the choices of $\tilde{\mathcal{U}}_{i,m}$ and n_i in (7) ensure that $(\tilde{\mathcal{U}}_{i,m}, f(\tilde{\mathcal{U}}_{i,m}), g(\tilde{\mathcal{U}}_{i,m}))\cdot n_i$ in (7) is a first order approximation to the integrand in (8) at any point within an $\mathcal{O}(\Delta x)$ distance from the centroid Z_i of S_i. Therefore we have

$$\begin{aligned} U_{T_1}^{n+1} - \mathcal{U}_T^{n+1} &= (|T_1|U_{T_1}^{n+1} - |T|\mathcal{U}_T^{n+1})/|T| + U_{T_1}^{n+1}((|T| - |T_1|)/|T|), \\ &= \mathcal{O}(\Delta x). \end{aligned} \tag{9}$$

The proof is complete.

3 Appendix 1: Proof of Theorem 1 (§2.1)

Lemma 2 *Assume Hypotheses 1-4. If a cell edge connecting grid nodes N_1 and N_2 intersects a CURVE at a point P, then either N_1 or N_2 will be shifted to P, or to an intersection point of the CURVE with another edge adjacent to P along the CURVE.*

Proof: Without loss of generality, suppose the cell edge l connecting N_1 and N_2 is parallel to the x-axis. At least one of the nodes must have grid distance less than or equal to $\Delta x/2$, say node N_1 which is shiftable. We first suppose that N_1 is not on the boundary. If N_1 belongs to (I1) then N_1 is x-shiftable and therefore will be shifted to P.

If (I2) or (I3) is true for N_1, we consider two cases:
Case 1 N_1 is x-shiftable, the result is the same as in (I1) during Step 1;
Case 2 N_1 is only y-shiftable, then N_1 will be shifted to the intersection point adjacent to P (along the *CURVE*).

If N_1 is on the boundary it will be shifted along the boundary to P or to the *POINT* adjacent to P along the *CURVE*. The proof is complete.

Lemma 3 *Assume Hypotheses 1-4. Let Q be the union of closed mesh cells with connected interior. Suppose a CURVE enters Q at $P_1 \in \partial Q$ and leaves Q at $P_2 \in \partial Q$ with the CURVE segment $[P_1, P_2] \subset Q$. Let P_0 be the intersection point between the CURVE and a cell edge just prior to P_1 if it exists; otherwise let $P_0 = P_1$. Similarly let P_3 be the intersection point between the CURVE and a cell edge just after P_2 if it exists; otherwise let $P_3 = P_2$. Then*

(1) *After the point shift algorithm, only the nodes originally in Q are on the CURVE segment $[P_1, P_2]$.*

(2) *At least one such node will be on the CURVE segment $[P_0, P_3]$.*

(3) *If a grid node originally in the interior of Q is shifted to the segment (P_1, P_2), then after the point shift algorithm it will be adjacent (along the CURVE) only to the grid nodes originally in Q.*

Proof: From Proposition 1 and Lemma 2 it follows that after the point shift algorithm, at least one grid node originally in Q will be on the *CURVE* segment $[P_0, P_3]$ to prove statement (2). Also from Proposition 1 we know that only nodes originally in Q will be on the *CURVE* segment $[P_1, P_2]$ proving statement (1). Statement (3) is obvious.

The proof is complete.

Proof of Theorem 1 By drawing an open circle of radius $\Delta x/2$ centered at the original position of each grid node, according to Proposition 1 we find all the possible positions of each node after point shift algorithm. Note that these circles are disjoint. Because the original interface is grid based, and thus consists of straight line segments between its crossings with grid lines, the new grid does not introduce any new intersection between edges and thus the topology of the grid remains unchanged.

Proposition 1 combined with the fact that the shift is along grid lines only gives the upper bound of the cell area. Hypotheses 2 implies that at most two nodes of a cell can be shifted to the *INTERFACE* within its cell boundary by the point shift algorithm (in other words, at least two nodes of the cell will remain fixed or be shifted outside the cell), which gives the lower bound of the cell area.

The *INTERFACE* passes through displaced grid cell corners only, by direct construction. We further assert that if a point shifted interface *BOND* connects two nodes, then the two nodes must share a common cell. In fact, if an interior node N is on a *CURVE* after the point shift algorithm, from the Hypothesis 3 the *CURVE* must intersect the boundary of the square of side $2\Delta x$ centered at the original position of N. Therefore from Lemma 3, after the point shift algorithm, N must be adjacent only to the nodes originally from the square, each of which share a common cell with N. If N is a boundary node that is on a *CURVE* after point shift algorithm, then it follows from Hypothesis 1 that N will be adjacent to an interior node along the *CURVE*, which returns to the previous case. The proof is complete.

4 Appendix 2: Proof of Propositions 2 and 3 (§2.2)

We introduce Lemmas 4-7 following the terminology of Proposition 2.

Lemma 4 *All POINTs on b^{n+1} are strictly spatially nearest neighbors with either P_1 or P_2.*

Proof: The CFL number is less than $\frac{1}{2}$. Thus any point on B^n can propagate at most a distance less than $\Delta x/2$. Shifting at each of the time levels t_n and t_{n+1} moves the grid nodes at most $\Delta x/2$, and only along grid lines. Therefore only grid nodes with a zero or unit lattice displacement from P_1 or P_2 in mesh index space can be shifted to lie on b^{n+1}. The lemma is proved.

Lemma 5 *Let M_1 belong to a BOND connecting adjacent Points P_3 and P_4. Then either P_1 and P_3 are spatially nearest neighbors or P_1 and P_4 are.*

Proof: Let D be a closed square of side $2\Delta x$ centered at the unshifted original position of grid node which is shifted at time t_n to be the *POINT* P_1. Then M_1 is in the interior of D. According to the first two statements of Lemma 3, M_1 must be adjacent (along the *INTERFACE*) to at least one of the grid nodes originally contained in D before being shifted. In other words, at least one of P_3 and P_4 is originally in D before being shifted. The grid nodes originally in D are all spatially nearest neighbor to P_1. Thus the proof is complete.

Lemma 6 *Assume that at least one of two adjacent POINTs P_3, P_4 at time level t_{n+1} lie on b^{n+1}. Suppose $[P_1, P_2]$ and $[P_3, P_4]$ have the same orientation relative to the INTERFACE, and that the pairs P_1, P_3 and P_2, P_4 are spatially nearest neighbors. Then either P_1, P_4 or P_2, P_3 are spatially nearest neighbors.*

Proof: The points P_1, P_2, P_4, P_3 in cyclic order form a four sided loop of spatially nearest neighbor grid nodes, when projected to a common time, and viewed in grid index space. See Fig. 5. Moreover, the points P_3 and P_4 belonging to b^{n+1} are strictly spatially nearest neighbor to either P_1 or P_2 by Lemma 4. This fact either completes the proof directly, or it forces one side of the loop to be a unit lattice distance (not a diagonal) at most. Each side of the loop is a single point, a unit lattice line or a unit lattice diagonal. The loop thus consists of a single line (multiply covered), a unit triangle (one side of the loop reduces to a point), a pair of adjacent unit triangles, a unit cell or a unit parallelogram (displaced by one lattice site from being a cell). The doubly displaced parallelogram of Fig. 6 is excluded. We are to prove that one of the diagonals of this loop is also spatially nearest neighbor. For the unit parallelogram, the shorter diagonal is a unit lattice diagonal, and its end points are thus spatially nearest neighbors. The other cases are elementary. The proof is complete.

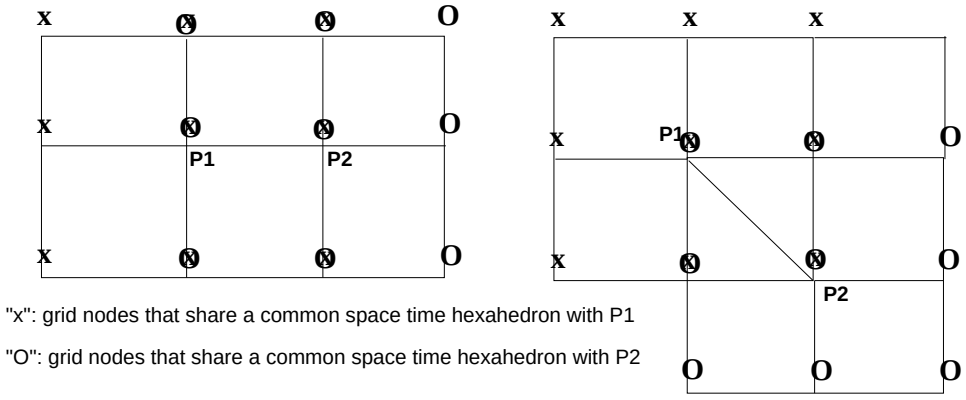

Figure 5: Grid nodes spatially nearest neighbor to a pair of nearest neighbor nodes.

Lemma 7 *Assume that b^{n+1} is contained in a BOND connecting adjacent POINTs P_3 and P_4, Suppose $[P_1, P_2]$ and $[P_3, P_4]$ have the same orientation relative to the INTERFACE, and that the pairs P_1, P_3 and P_2, P_4 are both spatially nearest neighbors. Then either P_1, P_4 are spatially nearest neighbors or P_2, P_3 are.*

Proof: As in the proof of Lemma 6, P_1, P_2, P_4, P_3 form a loop of spatially nearest neighbor grid nodes. If the *BOND* $[P_1, P_2]$ is the edge of a cell then one side of the loop is strictly spatially nearest neighbor, and the proof follows that of Lemma 6. If the *BOND* $[P_1, P_2]$ is the diagonal of a cell, see the right frame of Fig. 5. Fig. 6 shows the only case in which neither P_1, P_4 nor P_2, P_3 are spatially nearest neighbors.

The two circles contain the possible ranges of P_1 and P_2 respectively after the point-shift algorithm at the time t_n and propagation only at the time t_{n+1}. A, B, C, D, E, F, G, H are the midpoints on the cell edges starting at P_3 and P_4. In order for $[P_3, P_4]$ to be an interface *BOND* as described in the proposition at t_{n+1} after the point-shift algorithm, an *INTERFACE* segment γ at time t_{n+1} before the point-shift algorithm has to start from $\overline{AB} \bigcup \overline{CD}$, pass the circle centered at P_1, then pass the circle centered at P_2, and end up in $\overline{EF} \bigcup \overline{GH}$. Therefore γ has to intersect the edges of some cells and cause the grid nodes of the cells to be shifted on γ between *POINT*s P_3 and P_4. (See Lemma 2.) This fact violates the assumption that P_3 and P_4 are adjacent on the *INTERFACE*. The proof is complete.

Proof of Proposition 2 The *INTERFACE*, in mesh index space, traces a polygonal path joining nearest neighbor mesh nodes, and b^{n+1}, as a segment of this *INTERFACE*, does the same. The two extreme sets, of *POINT*s spatially nearest neighbor to P_1 but not to P_2 and the reverse set, while being spatially nearest neighbor to one of P_1 or P_2 are not adjacent. See Fig. 5. Thus b^{n+1} cannot pass from the first

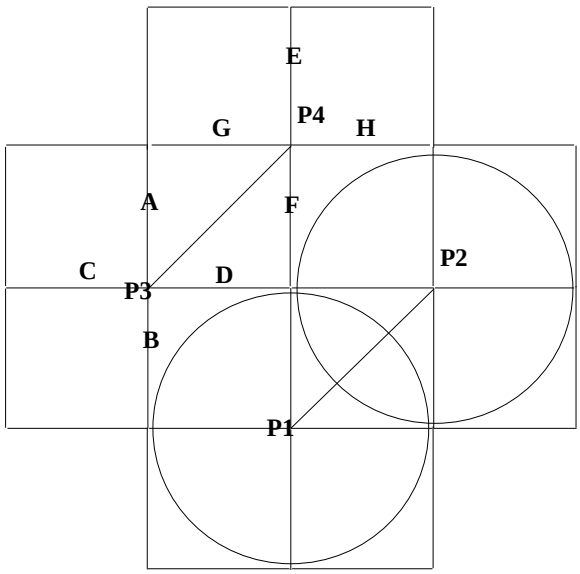

Figure 6: A configuration in which P_1, P_4 and P_2, P_3 are not spatially nearest neighbors.

set to the last without passing through the intermediate set. This proves the final statement of the Proposition.

The rest of the Proposition is an direct consequence of Lemma 4 provided that we can eliminate the following twisted nearest neighbor possibility: there are two *POINT*s P_3, P_4 on b^{n+1}, $[P_3, P_4]$ having the same orientation with $[P_1, P_2]$, so that P_3 is a spatially nearest neighbor to P_2 but not with P_1 and P_4 is a spatially nearest neighbor to P_1 but not with P_2. In fact if it is true, according to Lemma 4, P_3 must be a strictly spatially nearest neighbor to P_2 and P_4 be a strictly spatially nearest neighbor to P_1. Let us consider the case that B is a diagonal of a cell, the other case being similar and simpler. Drawing a circle of radius $(1-\epsilon)\Delta x$ $(0 < \epsilon << 1)$ centered at the original positions of P_1 and P_2, say O_1, O_2 respectively, we find all the possible positions of M_1, M_2 respectively. See Fig. 7.

Let N_1, N_2 denote the nodes which are strictly spatially nearest neighbors to P_1 but not with P_2 and N_3, N_4 denote the nodes which are strictly spatially nearest neighbors to P_2 but not with P_1. Note that before the point shift algorithm at t_{n+1}, the propagation image of the *BOND* B is a straight line which has to start from area O_1, visit the $\Delta x/2$ neighborhood of N_3 or N_4 (in order for it to be shifted to b^{n+1}), then visit the $\Delta x/2$ neighborhood of N_1 or N_2, and finally end up in O_2, which is impossible. The proof is complete.

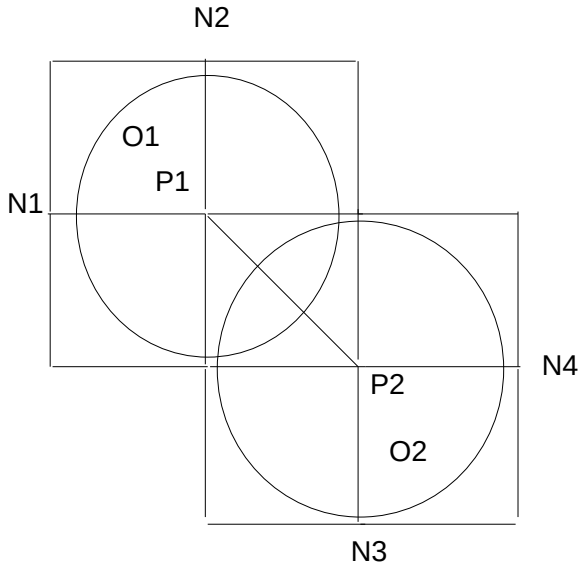

Figure 7: Nodes strictly spatially nearest neighbor to P_1 but not P_2 and the converse.

Lemma 8 *Let P_1, P_2 be two (not necessarily adjacent) time level t_n POINTs which are propagated and shifted to an interface BOND defined by adjacent time level t_{n+1} POINTs P_3 and P_4. Assume that $[P_1, P_2]$ and $[P_3, P_4]$ have the same orientation relative to the INTERFACE. Then if P_3 is spatially nearest neighbor with P_2 but not P_1, P_4 can not have the reverse property of being spatially nearest neighbor with P_1 but not P_2.*

Proof: We only consider the case that $[P_3, P_4]$ is the diagonal of cell, say C, after point-shifted algorithm. The other case that $[P_3, P_4]$ is the edge of a cell is similar and simpler. There are 5 grid nodes which are spatially nearest neighbor with P_3 but not with P_4. Also there are 5 nodes which are spatially nearest neighbor with P_4 but not with P_3. See Fig. 8. By drawing a circle of radius $(1-\epsilon)\Delta x$ $(0 < \epsilon << 1)$ centered at each of these 10 nodes we find all the possible positions of each these nodes after point-shift algorithm at time level t_n and after propagation at time level t_{n+1}. Let the union of the first 5 circles be O_1 and the union of the second 5 circles be O_2. A, B, C, D, E, F, G, H are the midpoints on the respective cell edges which limits the allowed time level t_{n+1} shifting of P_3 and P_4. In order for the *INTERFACE* at the time t_{n+1} before the point-shift algorithm to violate the above proposition the *INTERFACE* has to start from $\overline{AB} \bigcup \overline{CD}$, visit O_2, then O_1 and finally end up in $\overline{EF} \bigcup \overline{GH}$. The t_{n+1} propagated *INTERFACE* will inevitablely intersect edges and cause grid nodes other than P_3 and P_4 to be shifted to the *INTERFACE* between

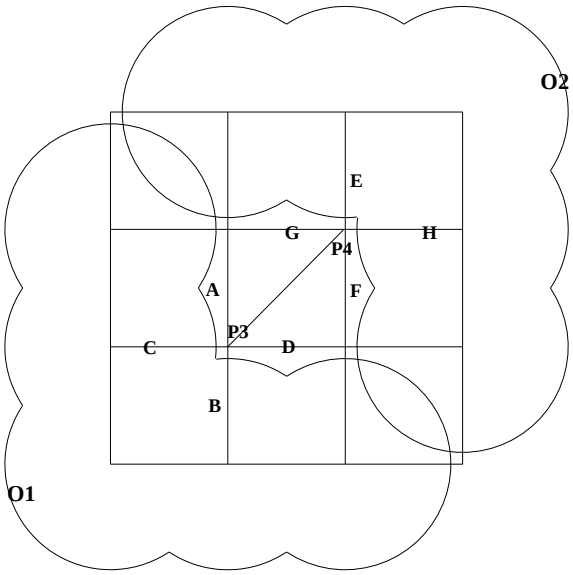

Figure 8: Nodes spatially nearest neighbor with one of P_3, P_4 but not the other.

them after the point-shift algorithm. (See Lemma 2.) This is a contradictory to the assumption that P_3 and P_4 are adjacent on the *INTERFACE*. The proof is complete.

Proof of Proposition 3 This is a corollary of Lemmas 5, 7 and 8.

References

[1] R. Abgrall. On essentially non-oscillatory schemes on unstructured meshes: Analysis and implementation. *J. Comput. Phys.*, 114:45–58, 1994.

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

[3] 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. Comp. Phys.*, 152:457–492, 1999.

[4] J. Glimm, J. W. Grove, X. L. Li, W. Oh, and D. H. Sharp. A critical analysis of Rayleigh-Taylor growth rates. *J. Comp. Phys.*, 169:652–677, 2001. LANL report No. LA-UR-99-5582.

[5] J. Glimm, J. W. Grove, X.-L. Li, K.-M. Shyue, Q. Zhang, and Y. Zeng. Three dimensional front tracking. *SIAM J. Sci. Comp.*, 19:703–727, 1998.

[6] J. Glimm, J. W. Grove, X.-L. Li, and D. C. Tan. Robust computational algorithms for dynamic interface tracking in three dimensions. *SIAM J. Sci. Comp.*, 21, 2000.

[7] J. Glimm, J. W. Grove, X.-L. Li, and N. Zhao. Simple front tracking. In G.-Q. Chen and E. DiBebedetto, editors, *Contemporary Mathematics*, volume 238, pages 133–149. Amer. Math. Soc., Providence, RI, 1999.

[8] J. Glimm, X.-L. Li, and Y.-J. Liu. Conservative front tracking in one space dimension. *to appear*, 2001.

[9] J. Glimm, D. Marchesin, and O. McBryan. Subgrid resolution of fluid discontinuities II. *J. Comput. Phys.*, 37:336–354, 1980.

[10] J. Glimm and O. McBryan. A computational model for interfaces. *Adv. Appl. Math.*, 6:422–435, 1985.

[11] A. Harten. High resolution scheme for hyperbolic conservation laws. *J. Comp. Phys.*, 49:357, 1983.

[12] O. McBryan. Elliptic and hyperbolic interface refinement in two phase flow. In J. Miller, editor, *Boundary and Interior Layers – Computational and Asymptotic Methods*. Boole Press, Dublin, 1980.

[13] B. Swartz and B. Wendroff. Aztec: A front tracking code based on Godunov's method. *Appl. Num. Math.*, 2:385–397, 1986.