

# A Subcell Remapping Method on Staggered Polygonal Grids for Arbitrary-Lagrangian-Eulerian Methods

R.Loubère, M.Shashkov\*

\* T-7, MS B284, Los Alamos National Lab., Los Alamos, NM, USA

*We describe a new remapping algorithm for use in Arbitrary-Lagrangian-Eulerian (ALE) simulations. The new features of this remapper are designed to complement a staggered-mesh Lagrangian phase in which the cells may be general polygons (in two dimensions), and which uses subcell discretizations to control unphysical mesh distortion and hourglassing. Our new remapping algorithm consists of three stages. A gathering stage, in which we interpolate momentum, internal energy, and kinetic energy to the subcells in a conservative way. A subcell remapping stage, in which we conservatively remap mass, momentum, internal, and kinetic energy from the subcells of the Lagrangian mesh to the subcells of the new rezoned mesh. A scattering stage, in which we conservatively recover the primary variables: subcell density, nodal velocity, and cell-centered specific internal energy on the new rezoned mesh. We prove that our new remapping algorithm is conservative, reversible, and satisfies the DeBar consistency condition. We also demonstrate computationally that our new remapping method is robust and accurate for a series of test problems in one and two dimensions.*

## Introduction

In numerical simulations of multidimensional fluid flow, the relationship between the motion of the computational grid and the motion of the fluid is an important issue. Two choices that are typically made represent either a Lagrangian framework, in which the mesh moves with the local fluid velocity, or an Eulerian framework, in which the fluid flows through a grid fixed in space. More generally, however, the motion of the grid can be chosen arbitrarily. The philosophy of the Arbitrary Lagrangian-Eulerian methodology (ALE; cf [12], [1], [2], [17], [13], [20]) is to exploit this degree of freedom to improve both the accuracy and the efficiency of the simulation. The main elements of most ALE algorithms are an explicit Lagrangian phase, a rezone phase in which a new grid is defined, and a remap phase in which the Lagrange solution is transferred to the new grid [17]. Most ALE codes use a grid of fixed connectivity that, in two spatial dimensions, is formed by quadrilaterals or by a mix of quadrilaterals and triangles, the latter being considered as degenerate quadrilaterals. Ultimately, we are interested in the development of ALE methods for meshes whose connectivity may change during the calculation. In such methods, the total number of cells remains fixed, but the number of edges bounding each cell may change with time, leading to the appearance of general polygonal cells.

As a first step toward this goal, here we consider ALE methods on a mesh with fixed connectivity, but allow the mesh to contain general polygonal cells. Extending the ALE methodology to this more general mesh is valuable in of itself as it simplifies the setup process for computational domains with complex geometrical shapes and helps to avoid artificial mesh imprinting due to the restrictions of a purely quadrilateral mesh, [4,5].

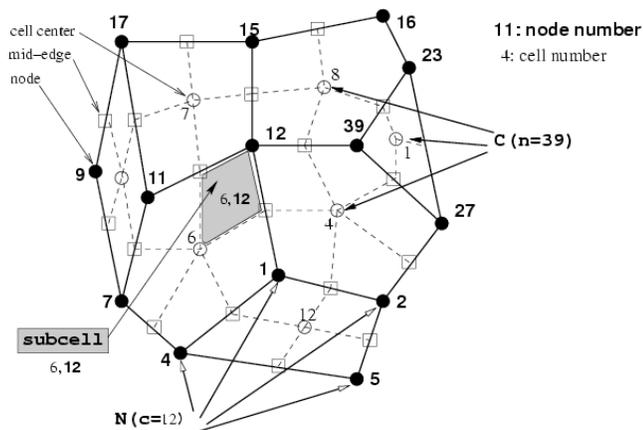
In the rest of this introductory section, we will introduce notation related to a general polygonal staggered mesh, will review algorithms for the Lagrangian phase and rezone phase as presented in [6,7,14,23], and finally will describe the main ideas of our new remap procedure, which is the main topic of this paper.

*Loubère R., Shashkov M.*

**Polygonal Mesh**

We consider a two-dimensional computational domain  $W$  assumed to be a general polygon. We assume we are given a mesh on  $W$  whose cells,  $\{c\}$ , cover the domain without gaps or overlaps. Each cell may be a general polygon, and is assigned an unique index that for simplicity will also be denoted by  $c$ . The set of vertices (nodes) of the polygons is denoted by  $\{n\}$ , where each node has an unique index  $n$ . Then each cell can be defined by an ordered set of vertices. We denote the set of vertices of a particular cell  $c$  by  $N(c)$ . Further, we denote the set of cells that share a particular vertex  $n$  by  $C(n)$ .

Note that each vertex may be shared by an arbitrary number of cells. We will subdivide each cell into a set of quadrilaterals that we will term subcells. A pair of indexes  $c$  and  $n$  uniquely defines a quadrilateral, identified as subcell  $cn$ ; this subcell is constructed by connecting the geometrical center of the cell  $c$  with the middle points of cell faces having the same node  $n$  as one end point and the node itself (see figure 1). Hence each cell can be divided uniquely into quadrilaterals (subcells or corners).



**Figure 1:** Grid and notations. The  $\circ$  are the cell centers, the  $\bullet$  are the nodes (vertices), the  $\square$  are the mid-face (edge) points. The set of vertices for cell  $c=\{12\}$  is  $N(c=12) = \{ 5,2,1,4\}$ , and the set of cells sharing node,  $n=39$  is  $C(n=39) = \{ 1,8,4 \}$ . The gray subcell,  $cn=6,12$  is the quadrilateral defined by connecting the geometrical center of the cell  $c=6$  with middle points of cell faces having the same node  $n=12$  as one end point and the node itself.

We denote the cell and subcell volumes (in 2D Cartesian geometry these are (x,y) areas) by  $V(c)$

and  $V(cn)$ , where by construction  $V(cn) \in V(c)$

A nodal volume can be defined as the sum of the volumes of subcells shared by the node  $n$ ,

i.e.,  $V(n) = \sum_{c \in C(n)} V(cn)$

**Lagrangian Phase**

The equations of Lagrangian gas dynamics can be written as

$$\frac{d}{dt} \int_{V(c)} \rho \mathbf{v} = \int_{V(c)} \rho \mathbf{v} \mathbf{v} \cdot \mathbf{n} dV$$

where  $\rho$  is the density,  $p$  is the pressure,  $\epsilon$  is the specific internal energy, and  $\mathbf{v}$  is the velocity. The pressure is linked to density and specific internal energy via an equation of state:

$\rho = \frac{m}{V}$ . We can define mass as  $m = \rho V$  where  $V$  is the volume, momentum as  $\vec{p} = \rho \vec{v} V$  and total energy as  $E = \rho \left( \frac{1}{2} v^2 + \epsilon \right) V$ . This previous system of equations is going to be solved by the Lagrangian phase.

A discretization of the gas dynamic equations for the Lagrangian phase of the ALE method for a mesh consisting of general polygons is described in [6,7], and is based on the philosophy of compatible hydrodynamic discretization [10]. This discretization assumes a staggered grid, where the components of the velocity vector are defined at the nodes (vertices) of the cells,  $\vec{v} = (v_x, v_y)$ , and where the thermodynamic variables density  $\rho$  and internal energy  $\epsilon$  are defined at the cell centers. In addition to nodal and cell-centered quantities, this discretization employs as additional variables the densities of the subcells,  $\rho_{cn}$ . The preservation of subcell mass during the Lagrangian phase of the calculation introduces new forces that prevent artificial grid distortion and hourglass patterns. This enhancement of the Lagrangian algorithm was shown to be effective both for quadrilateral meshes [8], as well as for polygonal meshes [6]. The Lagrangian phase including subcell forces, is conservative; i.e., discrete forms of mass, momentum, and total energy are conserved [10]. The use of subcell masses and corresponding densities places new requirements on the remap phase of an ALE method because these subcell densities have to be remapped in addition to the usual remapping of the primary variables - nodal velocities, cell-centered densities and internal energies. We can define the subcell mass in terms of the primary cell variables as follows:

$$m_{cn} = \rho_{cn} V_{cn}$$

Then the mass of the cell/node are defined:  $m_c = \sum_n m_{cn}$   $m_n = \sum_c m_{cn}$ .

All of these masses are employed in the Lagrangian phase of our ALE method. Since the subcell mass,  $m_{cn}$  is assumed to be Lagrangian and so does not change with time, it follows that

$$\frac{dm_{cn}}{dt} = 0$$

which serves as a definition of the subcell density for a given subcell mass. The masses of the individual cells and nodes are also Lagrangian because they are sums of the masses of the associated subcells. The mass of the cell is used in the equation for the internal energy, while the mass of the node is used in the momentum equation. Finally, by definition we have

$$\frac{dm_c}{dt} = 0 \quad \frac{dm_n}{dt} = 0$$

The total mass,  $M$ , which is conserved in the Lagrangian phase is

$$M = \sum_c m_c = \sum_n m_n$$

On the staggered mesh, momentum is most naturally defined at the nodes

$$\vec{p}_n = \rho_n \vec{v}_n m_n$$

or equivalently

$$\vec{p}_c = \rho_c \vec{v}_c m_c$$

Note that as a result of the remap stage we will have new momenta and masses at the nodes, so to recover velocities we will use the previous equations as the definition of velocities for given momenta and nodal mass. The total momentum components,  $\mu$ , which are individually conserved in the Lagrangian phase, are

$$\mu = \sum_{c \in \mathcal{C}} m(c) v(c)$$

It will be useful to define a cell-centered momenta as:

$$\mu_c = \frac{1}{V(c)} \int_{V(c)} \mu \, dV$$

Using this definition and the definition of nodal mass, the total momentum components  $\mu$  can be expressed as:  $\mu = \sum_{c \in \mathcal{C}} m(c) v_c$ . Kinetic energy is also most naturally

defined at the nodes:  $K = \frac{1}{2} \sum_{c \in \mathcal{C}} m(c) v_c^2$ . The internal energy is naturally defined at the cells as

$E = \sum_{c \in \mathcal{C}} E(c)$ . The previous equation can be used after the remap phase to define  $E(c)$  given  $E(c)$  and  $m(c)$ :  $E(c) = \frac{1}{m(c)} \int_{V(c)} E \, dV$ . The total energy, which is also conserved in the Lagrangian phase is  $E = K + U$ . Later we will require the concept of a cell-centered kinetic energy,

which we define as follows:  $K_c = \frac{1}{2} m(c) v_c^2$ . Using this definition and the definition of nodal mass, the total energy,  $E$  can be expressed as:  $E = \sum_{c \in \mathcal{C}} K_c + U$ . By introducing total internal and kinetic energies as:  $K = \sum_{c \in \mathcal{C}} K_c$  and  $U = \sum_{c \in \mathcal{C}} U(c)$  we finally can express the total energy as:  $E = K + U$ .

**Rezone Phase**

In the rezone phase, we use the reference Jacobian matrix (RJM) strategy described in [14,21]. The RJM rezone algorithm is based on a nonlinear optimization procedure, which requires a valid mesh as an initial guess, and so it may be necessary to untangle the mesh (see, e.g., [22]) prior to rezoning. The RJM rezone strategy ensures the continuing geometric quality of the computational grid, while keeping the "rezoned" grid at each time step as close as possible to the Lagrangian grid. Sets of cells and nodes of rezoned mesh will be denoted by  $c'$  and  $n'$  respectively.

When the rezoned and Lagrangian grids are sufficiently close to each other, it is possible to use a local procedure on the remapping stage, meaning that mass, energy and momentum are exchanged only between neighboring cells. Local rezoning is conceptually simpler and computationally less expensive than global rezoning. For some of the tests presented in Section Numerical Results, we will use the ALE code in the Eulerian framework, so that the rezoned mesh will always coincide with the initial mesh.

**Remap Phase**

To guarantee conservation in the overall ALE simulation, the remapping phase must conservatively interpolate the Lagrange solution onto the rezoned grid. The main purpose of this paper

is to describe a new algorithm for remapping on general, polygonal, staggered grid, including treatment of the density defined in the subcells. Readers interested in the history of remapping methods on staggered meshes are referred to [2,3,19,20,16,18].

To best of our knowledge there is no existing remapping method that addresses all of our requirements - remapping on general polygonal staggered mesh with subcell densities. We have designed a new remapping strategy consisting of the three following stages:

First: *gathering stage*. We define momentum, internal energy, and kinetic energy in the subcells. Recall that mass of subcell is already defined as the multiplication of the subcell density by the subcell volume. Mass, momentum, internal energy and kinetic energy in the subcells are defined in such a way that the corresponding total quantities (defined as the sums over subcells) are the same as at the end of the Lagrangian phase, ensuring that the gathering stage is conservative.

Second: *subcell remapping stage*. We use the algorithm described in [14] to remap mass, momentum, internal, and kinetic energy from the subcells of the Lagrangian mesh to the subcells of the new rezoned mesh. This algorithm is linearity- reserving and computationally efficient. It consists of a piece-wise linear reconstruction and an approximate integration based on the notion of swept regions. The algorithm does not require finding the intersections of the Lagrangian mesh with the rezoned mesh, which contributes to its efficiency. The algorithm is conservative: total mass, momentum, internal and kinetic energy over subcells of the rezoned mesh are the same as mass, momentum, internal and kinetic energy over subcells of Lagrangian mesh. The total energy is also conserved, being the sum of (individually conserved) internal and kinetic energies. We suggest that remapping internal and kinetic energy separately is more accurate than remapping total energy, because we are not merging two quantities that can have very different magnitudes and behavior.

Third: *scattering stage*. We recover the primary variables - subcell density, nodal velocity, and cell-centered specific internal energy - on the new rezoned mesh.

Subcell density is recovered by dividing the remapped mass by the remapped volume of subcell from the rezoned mesh. The subcell masses and the corresponding densities are then adjusted using a conservative repair procedure [14] to enforce local-bounds, which may be violated during the subcell remapping stage. This produces the final subcell density and the corresponding subcell mass that will be used in next time step. The new nodal masses and the cell-centered masses are defined using the summation convention thanks to subcell values. Next, we define the remapped nodal momenta using the remapped subcell momenta, in such a way that total momenta is conserved. New velocity components are defined according to their equations (see previous paragraph). Then nodal velocity is repaired, resulting in the final velocity that will be used to move the point during the Lagrangian phase in the next computational cycle.

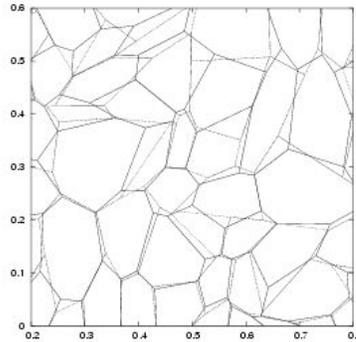
To enforce the conservation of total energy, the discrepancy between the remapped kinetic energy in the cell and the kinetic energy that is computed from the remapped subcell masses and the final nodal velocities is contributed to the remapped internal energy in the cell. The new internal energy is recovered. Finally, the internal energy and the corresponding specific internal energy is conservatively repaired.

The outline of the rest of this paper is as follows. In next section we will give a precise statement of our goals for remapping on the staggered mesh and will list the properties of the remapping algorithm. Numerical results that demonstrate the accuracy and convergence of the remapping algorithm are presented in Section Numerical Results. Finally, we conclude the paper in the last section.

## **Statement of the Remapping**

As a result of the Lagrangian phase of a computational cycle, we have a mesh consisting of cells  $\{c\}$ , and nodes  $\{n\}$ . We will call this the Lagrangian or old mesh. We have values of density,  $\rho(cn)$  in

subcells, values of specific internal energy,  $\epsilon(c)$ , in cells, and values of the components of velocity,  $u(n)$ ,  $v(n)$ , at the nodes of the old mesh. As a result of the rezone phase we have the rezoned or new mesh consisting of cells  $\{c'\}$ , and nodes  $\{n'\}$ . An example of old and new meshes is given in figure 2.



**Figure 2:** Fragment of the Lagrangian (dotted lines) and the rezoned (solid lines) grids.

The goal of the remapping phase is to find an accurate approximation to  $r(cn')$ ,  $e(c')$ ,  $u(n')$ ,  $v(n')$  on the new mesh. Using the primary variables we can define the total mass  $M$ , the momentum vector  $\mu$ , the internal energy  $e$ , the kinetic energy  $K$ , and the total energy  $E$  on the old mesh (see equations in the previous section).

Our remapping algorithm satisfy the following requirements:

*Conservation.* The total mass, momenta and energy of the new mesh must be the same as that of the old mesh  $= \mu \mu =$

This property, combined with the same conservation properties of the Lagrangian phase, guarantees the conservation of the overall ALE method.

*Bound-preservation.* The remapped density, velocity components and internal energy have to be contained within physically justified bounds, which are determined from the corresponding fields in the Lagrangian solution. For example, density and internal energy have to be positive. Moreover, because we assume that the new mesh is obtained by small displacement of the old mesh, one can require that the new value lie between bounds determined by the values of its neighbors on the old mesh, [14].

*Accuracy.* It is straightforward to define accuracy in the remap of density; we will require that the remap of density is linearity-preserving. That is, if the values on the old mesh are obtained from a global linear function, then the values on the new mesh have to coincide with the values of the same linear function on the new mesh. For the remap of velocity, there are several different notions related to accuracy. For example, one widely used test of consistency is the so-called DeBar condition (see for example [2,3]) which can be stated as follows: if a body has a uniform velocity and spatially varying density, then the remapping process should exactly reproduce a uniform velocity. For internal energy, the situation is more complicated. We will demonstrate the accuracy of our new algorithm through the practical expedient of well-chosen test problems.

*Reversibility.* If the new and old meshes are identical, then the remapped primary variables should show no change. This property is closely related to the notion of being free of inversion error, see [2,3], where it is stated that if the new and old grids coincide, then the remapped velocity on new mesh should coincide with the velocity on the old mesh.

## Numerical Results

In this section we will investigate numerically the performance of our new method. All problems are solved in Cartesian coordinates. Our remapping method is unique in the sense that it is intended for a staggered mesh of general polygons using a subcell discretization of the density. As previously

mentioned we are not aware of any other method that can treat such a remapping problem. However, we are still interested in comparing our new remapping method with other known methods for remapping on a staggered mesh.

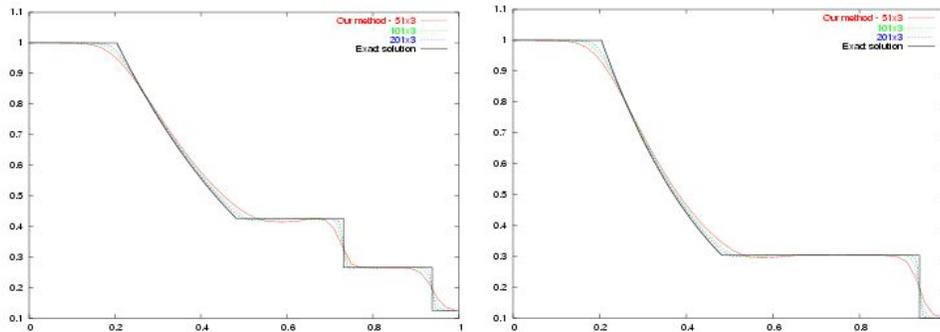
**One dimensional tests**

For all 1D test problems we use our new method implemented in a 2D code, but run in an initially square mesh with only two cells in  $y$  direction. The length of the computational domain in the  $y$  direction,  $y_{\max}$  depends on number of cells in  $x$  direction (in the  $x$  direction initial mesh is always uniform). In our description of the test problems we will specify only the length of computational domain and the number of cells in the  $x$  direction.

***Sod Problem.***

The Sod problem is a Riemann shock tube with a relatively small discontinuity, and so is very mild test. Its solution consists of a left rarefaction, a contact discontinuity and a right shock and the exact solution is illustrated in Figure 3 by the solid line.

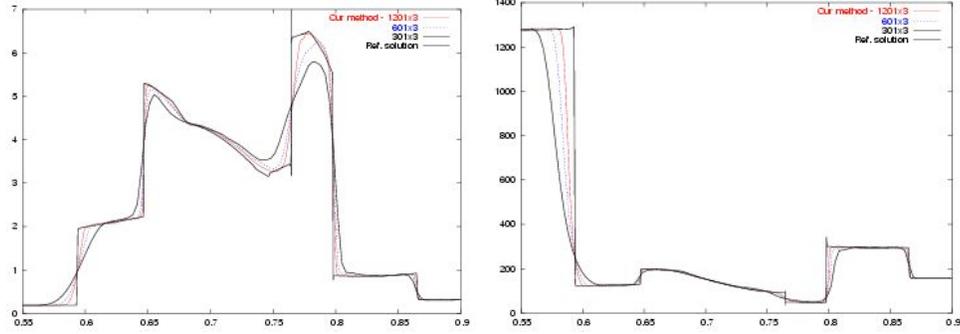
In our numerical experiments, the computational domain is  $[0;1]$ . The discontinuity is initially at  $0.5$ . The domain is filled with an ideal gas with  $\gamma=1.4$ . The density/pressure values on the left side of the discontinuity are  $1.0/1.0$ , while those on the right side are  $0.125/0.1$ . In Figure 3, we present numerical results for the density and the pressure at the final time  $t=0.25$  for a run with  $N=50, 100, 200$  computational cells in  $x$  direction.



**Figure 3:** Convergence for Sod problem: density and pressure at  $t=0.25$  for 50, 100, 200 points in  $x$  direction.

***Woodward-Colella Blast Wave Problem.***

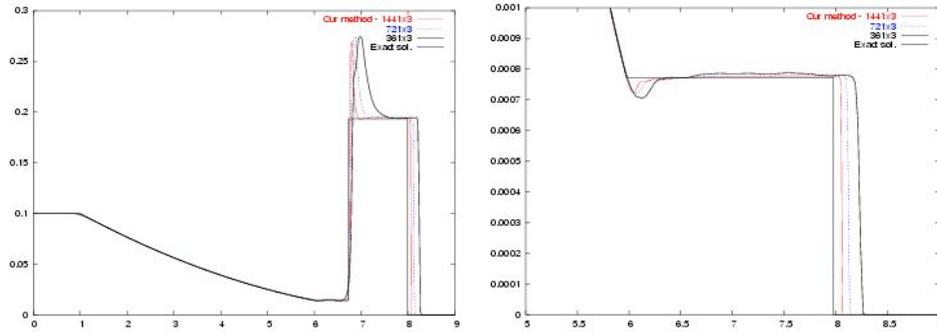
The computational domain for this problem has length one, with reflecting walls at the both ends. The gas is an ideal gas with  $\gamma=1.4$ . At  $t=0$ , the gas is at rest with an uniform density equal to  $1.0$ . The initial pressure is  $1000.0$  in the leftmost tenth of the domain,  $100.0$  in the rightmost tenth, and  $0.01$  everywhere else. The final problem time is  $t=0.038$ . Initially, two shocks and two contacts develop at the initial discontinuities and propagate toward one another, while two rarefactions develop, propagate toward the walls, and reflect off them. As time progresses, these six initial waves interact and create additional contact discontinuities. There is no analytical solution for this problem and typically a solution obtained by purely Lagrangian method with very high resolution ( $N=3600$  cells in our case) is considered as the reference "truth" (the solid line in Fig.4). As has been mentioned in [19], the Lagrangian solution has a flaw, a spurious overshoot at  $x \sim 0.765$ . In Fig.4 we present numerical results obtained by our new method for  $N=300, 600, 1200$  points in  $x$  direction.



**Figure 4:** Convergence for Woodward-Colella Blast Wave Problem. Density (zoom) --- left, and specific internal energy (zoom) --- right at  $t=0.038$ , for 300, 600, 1200 points in  $x$  direction

**LeBlanc Shock Tube Problem.**

In this extreme shock tube problem the initial discontinuity separates a region of very high energy and density from one of low energy and density. This is a much more severe test than the Sod Problem. The computational domain is  $[0;9]$  and is filled with an ideal gas with  $\gamma=5/3$ . The gas is initially at rest. The initial discontinuity is at  $x=0.3$ :  $(\rho,\epsilon)=(1,0.1)$  for  $x < 3$  and  $(0.001, 10^7)$  for  $x > 3$ . The solution consists of a rarefaction moving to the left, and a contact discontinuity and a strong shock moving to the right - solid line in Fig.5. At the final time of  $t=6.0$ , the shock wave is located at  $x=7.975$ . In Fig.5 we present numerical results obtained by our new method for  $N=361, 721, 1441$  points in  $x$  direction.



**Figure 5:** Convergence for Le Blanc shock tube problem. Specific internal energy (Left: entire domain, Right: zoom) at  $t=0.038$ , for 300, 600, 1200 points in  $x$  direction.

**Two-dimensional tests**

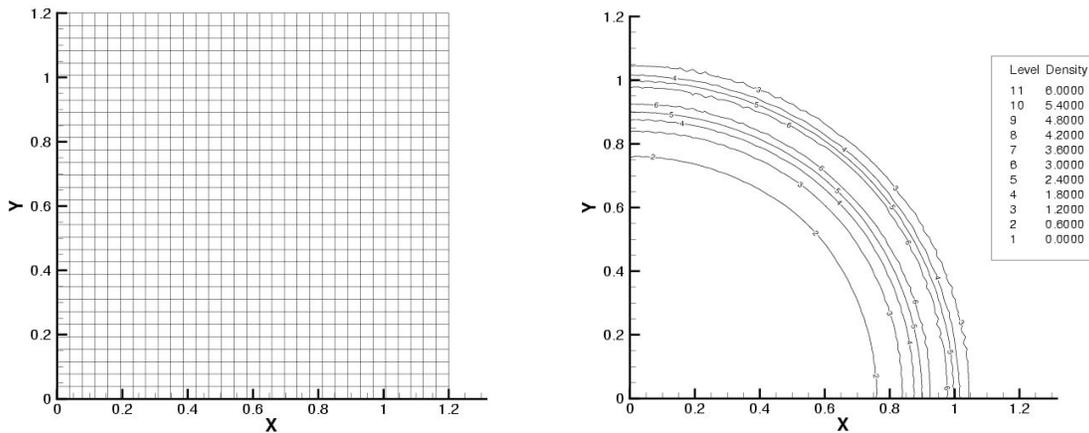
In this subsection we present numerical results for the Sedov blast wave problem, [25], which describes the evolution of a blast wave in a point symmetric explosion; it is an example of a diverging shock wave.

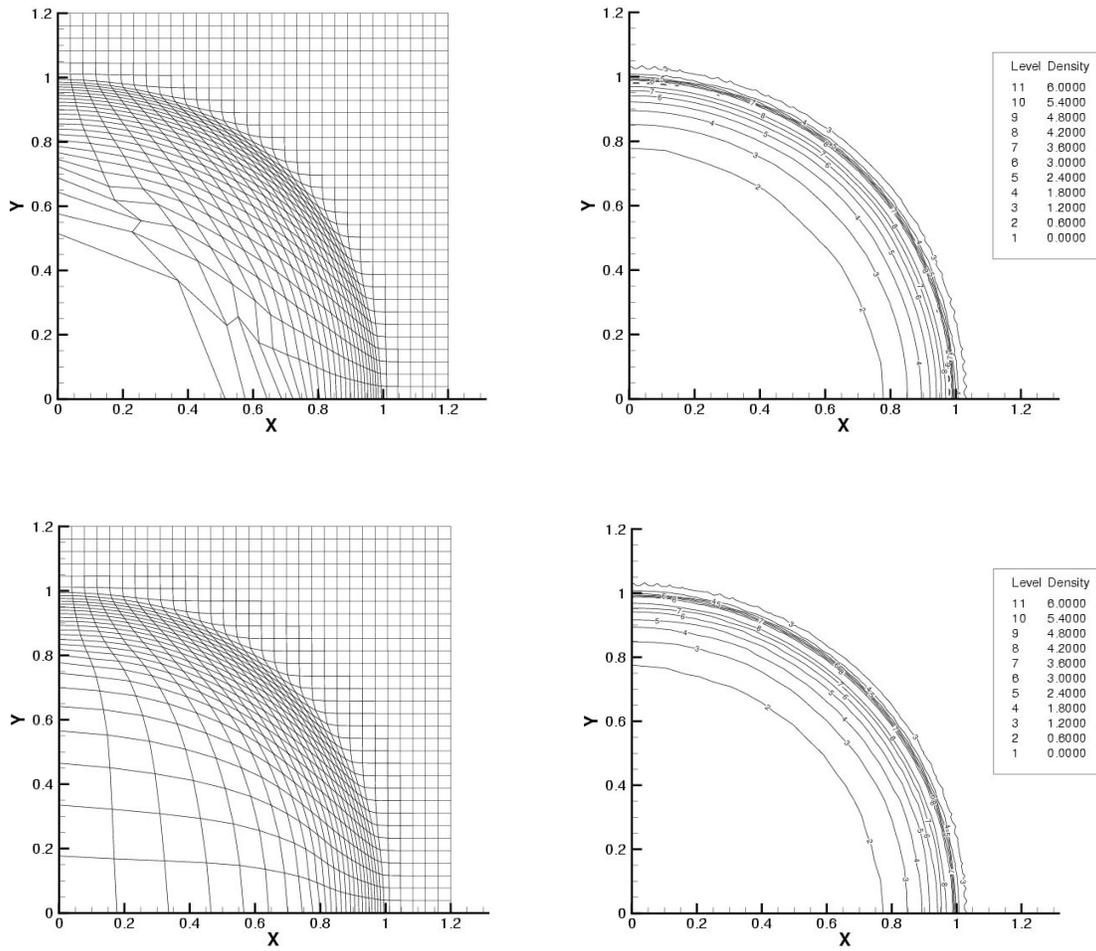
We consider the cylindrically symmetric Sedov problem, in Cartesian coordinates  $(x,y)$ . The total energy of the explosion is concentrated at the origin and has magnitude  $E=0.244816$  (similar to [7]). The material is an ideal gas with  $\gamma=1.4$  and initially is at rest with an initial density equal to 1. At time  $t=1.0$  the exact solution is a cylindrically symmetric diverging shock whose front is at radius,  $r=1$  and with a peak density of 6.0 (the solid line in Fig.7). In our numerical experiments  $E$  is concentrated in one cell located at the origin (that is, containing the vertex  $(x,y)=(0,0)$ ). The specific internal energy of this cell,  $c$  is defined as  $\epsilon(c)= E/V(c)$ . Therefore the initial pressure is  $p=(\gamma-1)\rho\epsilon=0.4 E/V(c)$ . For this problem we compare results obtained by our 2D code in three different frameworks: a purely Eulerian, a purely Lagrangian, and an ALE framework. Simulations in all three frameworks have been carried

out on both quadrilateral and polygonal meshes. In the ALE calculation, the rezoning/remapping is performed once every 10 Lagrangian steps. The CFL number is chosen to be equal to 0.25 for all simulations. For each simulation we show both the initial and the final mesh, with 11 density isolines equally distributed in magnitude between 0.0 and 6.0. (Figs.6 and 8). Each isoline has a label that refers to a density value in the legend scale. Also we show a 1D plot of density as a function of the radius,  $r$ , and a corresponding plot of the exact solution (Figs.7 and 9). The 1D plots demonstrate how well the numerical solution preserves cylindrical symmetry.

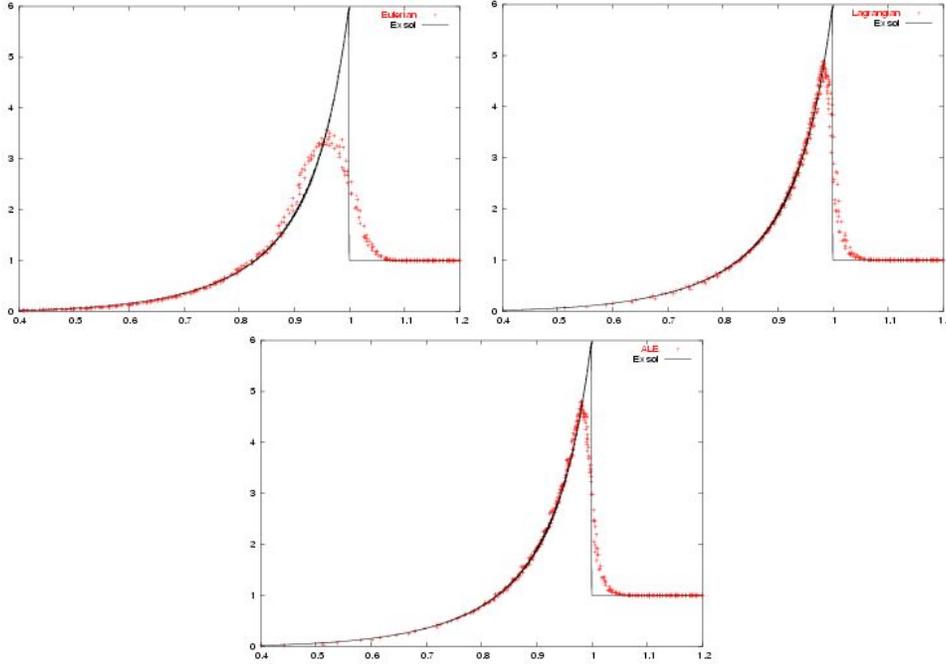
***Quadrilateral Meshes.***

For this set of simulations, the computational domain is a square  $[0:1.2] \times [0:1.2]$  whose initial mesh consists of  $31 \times 31$  square cells (top-left mesh in Fig.6). The two top panels in Fig.6 and the left panel in Fig.7 show the results of purely Eulerian computations. The symmetry of the solution is preserved quite well but the density peak is diminished ( $\rho=3.55$  instead of 6) and the shock wave is spread over several cells. The two panels in the middle of Fig.6 and the central panel in Fig.7 show the results of purely Lagrangian computations. The peak density magnitude, 4.9, is much closer to the correct value than is the Eulerian computational value. Also, the symmetry is better preserved in the Lagrangian calculation, especially near the peak. However, the Lagrangian mesh has a very low geometrical quality near the axis. The two bottom panels in Fig.6 and right panel in Fig.7 show results of the ALE computations. The symmetry of the solution is even better than was found in the Lagrangian calculations and the peak density is 4.75 which is little bit smaller than in the Lagrangian calculations. The geometrical quality of the mesh is significantly improved in comparison with the Lagrangian case.





**Figure 6:** Sedov Problem – Quadrilateral Mesh - Mesh (left), and density isolines (right) at  $t=1.0$  - Eulerian regime (top), Lagrangian regime (middle), ALE regime (bottom)



**Figure 7:** Sedov Problem – Quadrilateral Mesh - Density at  $t=1.0$  as a function of the radius (solid line is an exact solution) - Eulerian regime (left), Lagrangian regime (right), ALE regime (bottom)

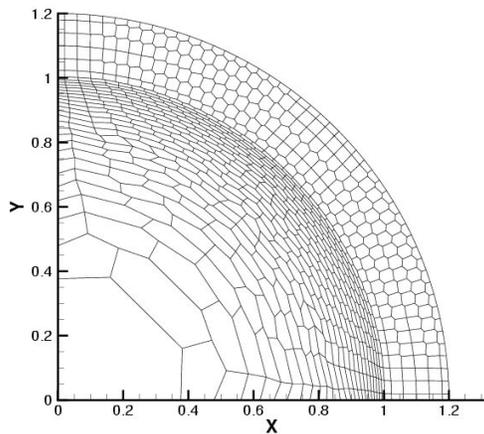
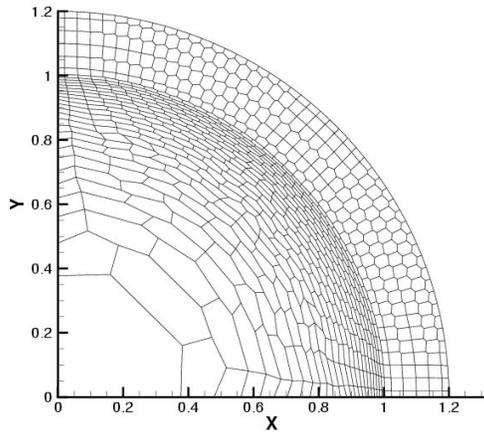
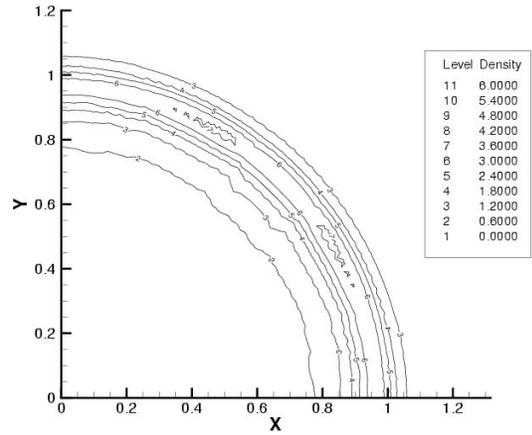
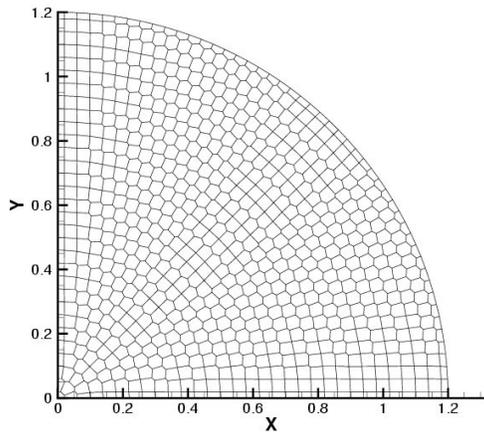
In the top part of Table 1 we present the peak density values and also the number of time steps needed to reach the final time of  $t=1.0$  for the Eulerian, Lagrangian and ALE computations. It is interesting to note that the ALE computation takes the least number of time steps. The ratio between the CPU time spent for the Eulerian regime versus the Lagrangian regime is  $\sim 10$ , between the ALE regime and the Lagrangian one is  $\sim 2$ . Remark that these timing comparisons are strongly dependent on the details of implementation and are presented to the reader as a "rough" approximation.

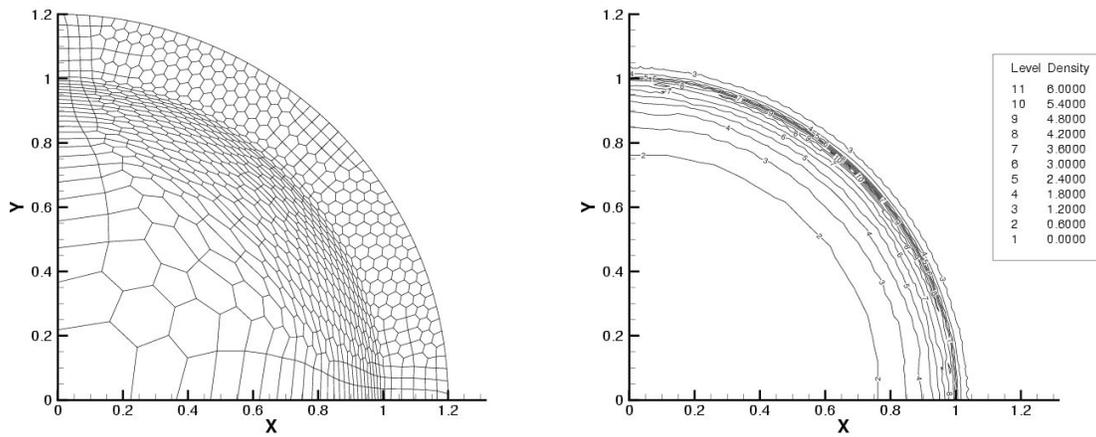
**Polygonal Meshes.**

The computational domain is one quarter of a circular disk with radius of  $r_{max}=1.2$ . A polygonal mesh is constructed in the computational domain using a Voronoi diagrams for the set of point defined as follows:  $x_{i,j} = r_j \cos(\theta_{i,j})$ ,  $y_{i,j} = r_j \sin(\theta_{i,j})$ , for  $j = 1, \dots, J$   $i = 1, \dots, I(j)$ , where

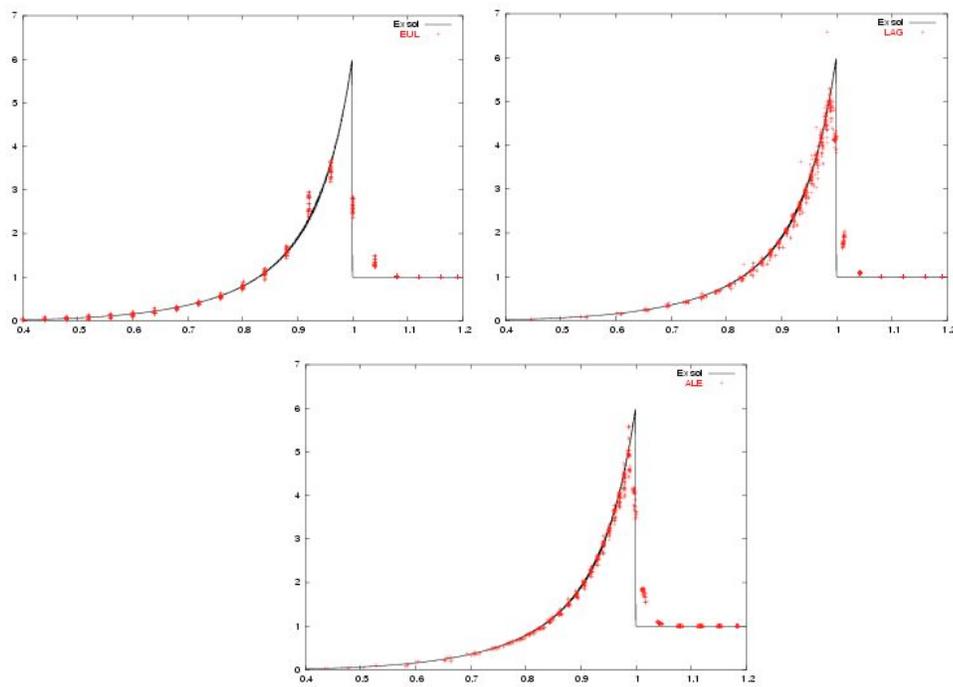
$$r_j = r_{max} \frac{j-1}{J}, \quad I(j) = \text{round}\left(j-1, \frac{\pi}{2}\right), \quad \theta_{i,j} = \frac{i-1}{I(j)} \frac{\pi}{2},$$

and  $J=31$ . Moreover function  $\text{round}(x)$  returns the closest integer to  $x$ . According to these formulas, on each circle of radius  $r_j$  points are distributed so that the distance between adjacent points along the circle is approximately equal to  $\Delta r = r_{j-1}/(J-1)$ . The total number of points is 775. There is exactly one Voronoi cell corresponding to each point. The mesh consists of a mixture of convex polygons: quadrilaterals, pentagons and hexagons, with a total number of vertices is 1325; the mesh is shown in Fig. 8 (top-left panel). The resulting polygonal mesh has approximately the same resolution as the quadrilateral mesh presented in Fig.6. Numerical results for the initially polygonal mesh are arranged in a similar way as was done for our study of the quadrilateral meshes and are presented in Fig.8, and 9, and bottom of Table 1.





**Figure 8:** Sedov problem --- Polygonal Mesh. Mesh (left), and density isolines (right) at  $t=1.0$  --- Eulerian regime (top), Lagrangian regime (middle), ALE regime (bottom).



**Figure 9:** Sedov problem --- Polygonal Mesh. Density at  $t=1.0$  as a function of the radius (solid line exact solution) --- Eulerian regime (left), Lagrangian regime (middle), ALE regime (right).

Mesh Type	Hydro Regime	# Time steps	Peak density
Quad	Eulerian	477	3.55
Quad	Lagrangian	375	4.90
Quad	ALE-10	338	4.75

Poly	Eulerian	1567	3.69
Poly	Lagrangian	603	6.20
Poly	ALE-10	408	5.70

**Table 1:** Sedov Problem. Number of time steps needed to reach final time  $t=1.0$  and peak density values.

Qualitatively, the relative performance of purely Eulerian, purely Lagrangian, and ALE methods on polygonal meshes is the same as for quadrilateral meshes. The results of the purely Eulerian and purely Lagrangian calculations on the polygonal mesh exhibit less symmetry than the corresponding calculations on the quadrilateral meshes. However, the polygonal mesh behaves better near the axes even for purely Lagrangian calculations. The ratio between the CPU time spent for the Eulerian versus the Lagrangian regimes is  $\sim 20$  and between the ALE and Lagrangian regimes  $\sim 2$ .

## Conclusions

In this paper we have constructed a full ALE method for use on a staggered polygonal mesh. The method combines and generalizes previous work on the Lagrangian and rezoning phases, and includes a new remapping algorithm.

In the Lagrangian phase of the ALE method we use compatible methods to derive the discretizations [6], [7]. We assume a staggered grid where velocity is defined at the nodes, and where density and internal energy are defined at cell centers. In addition to nodal and cell-centered quantities, our discretization employs subcell masses that serve to introduce special forces that prevent artificial grid distortion and hourglass-type motions, [8]. This adds an additional requirement to the remap phase - that the subcell densities (corresponding to subcell masses) have to be conservatively interpolated in addition to nodal velocities and cell-centered densities and internal energy.

In the remap phase, we assume that rezone algorithm produces mesh that is "close" to Lagrangian mesh so that a local remapping algorithm (i.e, where mass and other conserved quantities is only exchanged between neighboring cells) can be used.

Our new remapping algorithm consists of three stages.

- *A gathering stage*, where we define momentum, internal energy, and kinetic energy in the subcells in a conservative way such that the corresponding total quantities in the cell are the same as at the end of the Lagrangian phase.
- *A subcell remapping stage*, where we conservatively remap mass, momentum, internal, and kinetic energy from the subcells of the Lagrangian mesh to the subcells of the new rezoned mesh.
- *A scattering stage*, where we conservatively recover the primary variables: subcell density, nodal velocity, and cell-centered specific internal energy on the new rezoned mesh.

We have proved that our new remapping algorithm is conservative, reversible, and satisfies the DeBar consistency condition. We have also demonstrated computationally that our new remapping method is robust and accurate for a series of test problems in one and two dimensions.

## Acknowledgements

The authors thank L. Margolin, B. Rider, S. Li, B. Wendroff, R. Anderson, R. Pember, D. Benson, and T.Dey for fruitful discussions. This work was performed under the auspices of the US Department of Energy at Los Alamos National Laboratory, under contract W-7405-ENG-36. The authors acknowledge the partial support of the DOE/ASCR Program in the Applied Mathematical Sciences and the Laboratory Directed Research and Development program (LDRD). The authors also acknowledge the partial support of DOE's Advanced Simulation and Computing (ASC) program.

## References

- [1] D. J. Benson, *An Efficient, Accurate, Simple ALE Method for Nonlinear Finite Element Programs*, Computer Methods in Applied Mechanics and Engineering, 72 (1989), pp. 305--350.
- [2] D. J. Benson, *Computational Methods in Lagrangian and Eulerian Hydrocodes*, Computer Methods in Applied Mechanics and Engineering, 99 (1992), pp. 235-394.
- [3] D. J. Benson, *Momentum Advection on a Staggered Grid*, J. Comp. Phys. 100 (1992) pp. 143-162.
- [4] D. E. Burton, "Multidimensional Discretization of Conservation Laws for Unstructured Polyhedral Grids", Report UCRL-JC-118306, Lawrence Livermore National Laboratory, 1994.
- [5] D. E. Burton, "Consistent Finite-Volume Discretization of Hydrodynamics Conservation Laws for Unstructured Grids" Report UCRL-JC-118788, Lawrence Livermore National Laboratory.
- [6] J. Campbell and M. Shashkov, *A Compatible Lagrangian Hydrodynamics Algorithm for Unstructured Grids*, Selcuk J. of Applied Mathematics, 4 (2003) pp. 53—70; report at <http://cnls.lanl.gov/~shashkov>.
- [7] J. Campbell, M. Shashkov, *A tensor artificial viscosity using a mimetic finite difference algorithm*, J. of Computational Physics, 172 (2001), pp. 739--765.
- [8] E.J. Caramana and M.J. Shashkov, *Elimination of Artificial Grid Distortion and Hourglass-Type Motions by means of Lagrangian Subzonal Masses and Pressures*, J. of Comp. Physics, 142 (1998), pp. 521--561.
- [9] E.J. Caramana, M.J. Shashkov and P.P. Whalen, *Formulations of Artificial Viscosity for Multi-Dimensional Shock Wave Computations*, J. of Computational Physics, 144 (1998), pp. 70--97.
- [10] E.J. Caramana, D.E. Burton, M.J. Shashkov and P.P. Whalen, *The Construction of Compatible Hydro Algorithms Utilizing Conservation of Total Energy* J. Comp. Phys., 146 (1998), pp. 227-262.
- [11] R.B. Demuth, L.G. Margolin, B.D. Nichols, T. F. Adams and B.W. Smith, "*SHALE : a Computer Program for Solid Dynamics*", Report Los Alamos National Laboratory Report LA-10236, 1985.
- [12] C. W. Hirt, A. A. Amsden and J. Cook, "*An Arbitrary Lagrangian-Eulerian Computing Method for All Flow Speeds*", J. Comp. Phys. 14 pp. 227--253 (1974), and reprinted in 135 (1997) pp. 203--216.
- [13] D.S. Kershaw *et al* *3D Unstructured Mesh ALE Hydrodynamics with the Upwind Discontinuous Finite Element Method*, Comp Meth in Appl Mech and Engin, 158 (1998), pp. 81--116.
- [14] P. Knupp, L.G. Margolin, M. Shashkov, *Reference Jacobian Optimization-Based Rezone Strategies for Arbitrary Lagrangian-Eulerian Methods*, J. of Computational Physics, 176 (2002), pp. 93--12.
- [15] M. Kucharik, M. Shashkov, and B. Wendroff, *An Efficient Linearity-and Bound-Preserving Remapping Method*, J. of Computational Physics, 188 (2003), pp. 462--471.
- [16] L.G. Margolin and C.W. Beason, *Remapping on the Staggered Mesh*, Report UCRL-99682 of Lawrence Livermore National Laboratory (1988).  
<http://www.llnl.gov/tid/lof/documents/pdf/208550.pdf>
- [17] L.G. Margolin, Introduction to "*An Arbitrary Lagrangian-Eulerian Computing Method for All Flow Speeds*", J. Comp. Phys. 135 (1997) pp. 198--202.
- [18] L. G. Margolin and M. Shashkov. *Remapping, Recovery and Repair on Staggered Grid*, LA-UR-03-2230, Computer Methods in Applied Mechanics and Engineering, to appear.
- [19] R. B. Pember and R. W. Anderson, *A Comparison of Staggered-Mesh Lagrange Plus Remap and Cell-Centered Direct Eulerian Godunov Schemes for Eulerian Shock Hydrodynamics*, Preprint UCRL-JC-139820, Lawrence Livermore National Laboratory, 2000.  
<http://www.llnl.gov/tid/lof/documents/pdf/238843.pdf>
- [20] J. S. Peery and D. E. Carroll, *Multi-Material ALE Methods in Unstructured Grids*, Computer Methods in Applied Mechanics and Engineering, 187, (2000), pp. 591--619.

*Proceedings from the 5LC 2005*

- [21] M. Shashkov and P. Knupp, "*Optimization-Based Reference-Matrix Rezone Strategies for Arbitrary Lagrangian-Eulerian Methods on Unstructured Grids*", *Selcuk J. Appl. Math.* 3 (2002) pp. 81--99.
- [22] P. Vachal, R. Garimella, M.J. Shashkov, R. Loubere, *Untangling of meshes in ALE simulations*, Proceedings of 7th US Nat. Congress on Computational Mechanics, Albuquerque, July 2003, <http://cnls.lanl.gov/~shashkov>.
- [23] P. Vachal, R. Garimella, and M. Shashkov, *Untangling of 2D meshes in ALE simulations*, *J. of Computational Physics*, 196 (2004), pp. 627--644.
- [24] P. R. Woodward and P. Colella, *The Numerical Simulation of Two-Dimensional Fluid Flow with Strong Shocks*, *J. of Computational Physics*, 54 (1984), pp. 115--173.
- [25] L. I. Sedov, *Similarity and Dimensional Methods in Mechanics*, Academic Press, New York, 1959.