A Collation and Analysis of Two-Dimensional Unsplit Conservative Advection Methods for Volume of Fluid at Interfaces

by

Adil Ansari

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved March 2019 by the
Graduate Supervisory Committee:

Marcus Herrmann, Chair
Yulia Peet
Huei-Ping Huang

ARIZONA STATE UNIVERSITY

May 2019

ABSTRACT

The goal of this paper was to do an analysis of two-dimensional unsplit mass and momentum conserving Finite Volume Methods for Advection for Volume of Fluid Fields with interfaces and validating their rates of convergence. Specifically three unsplit transport methods and one split transport method was amalgamated individually with four Piece-wise Linear Reconstruction Schemes (PLIC) i.e. Unsplit Eulerian Advection (UEA) by Owkes and Desjardins (2014), Unsplit Lagrangian Advection (ULA) by Yang *et al.* (2010), Split Lagrangian Advection (SLA) by Scardovelli and Zaleski (2003) and Unsplit Averaged Eulerian Lagrangian Advection (UAELA) with two Finite Difference Methods by Parker and Youngs (1992) and two Error Minimization Methods by Pilliod Jr and Puckett (2004). The observed order of accuracy was first order in all cases except when unsplit methods and error minimization methods were used consecutively in each iteration, which resulted in second order accuracy on the shape error convergence. The Averaged Unsplit Eulerian Lagrangian Advection (AUELA) did produce first order accuracy but that was due to temporal error in the numerical setup. The main unsplit methods, Unsplit Eulerian Advection (UEA) and Unsplit Lagrangian Advection (ULA), preserve mass and momentum and require geometric clipping to solve two-phase fluid flows. The Unsplit Lagrangian Advection (ULA) can allow for small divergence in the velocity field perhaps saving time on the iterative solver of the variable coefficient Poisson System.

ACKNOWLEDGMENTS

I would like to thank the Committee Chair for giving me guidance on this subject since 2017. I would like to acknowledge Dr. Kostelich for introducing me to High Performance Computing and the Research Computing staff. I would like to appreciate ASU faculty members who have helped me gain valuable knowledge including the committee members Dr. Yulia Peet and Dr. Huei-Ping Huang that has helped in the genesis of this thesis.
This paper would not have been possible had it not been for my family and friends who helped me through some tough times especially my sister, Zeba.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

Symbol	Definition
ψ	Volume fraction of volume of fluid field
ϕ	Level-set field
$\vec{u}$ or $[u, v]^\top$	Velocity vector
ρ	Density
p	Pressure
T_σ	Surface tension force
$\hat{n}$ or $[n_x, n_y]^\top$	Interface normal
$\vec{m}$ or $[m_x, m_y]^\top$	Interface normal normalized by sum of component absolute values
α	Smallest distance between cell center and interface
γ	Surface tension coefficient
κ	Curvature of the interface
p	Pressure
ϵ_{shape}	Shape rrror
ϵ_{volume}	Volume rrror
$\vec{p}_k$	Cell corner points
Φ_k	Flux volume
Δt	Temporal iteration time-step
$\Delta x, \Delta y$	Square cell side dimension
$N(\vec{x})$	Interpolating shape function
t_f	End of time domain
TCF	Transported cell fluid polygon

PREFACE

This paper deals with the a fundamental aspect of fluid simulation i.e. transport equation which is a Hyperbolic Partial Differential Equation. The basis for this paper stems from innate passion in me to ascertain efficient methods to solve deterministic physics problems on my decrepit laptop.

The topic of Geometric Fluxing has made satisfactory progress over the years and in this paper we will look at a few methods that can transport fluid volumes in a conservative fashion across cells while preserving an immiscible interface. The methods discussed in this paper can be extended to the third dimension with enough work. However, this paper only deals with the 2-dimensional projection of those methods.

Chapter 1

INTRODUCTION

1.1 Navier-Stokes Equation

Transient incompressible multi-phase flow can be described by a specific simplification of the general Navier-Stokes fluid formulation as described in Equation 1.1.

$$\frac{\partial \rho \vec{u}}{\partial t} + \nabla \cdot (\rho \vec{u}\vec{u}) = -\nabla p + \nabla \cdot (\mu(\nabla \vec{u} + \nabla^T \vec{u})) + \rho \vec{g} + T_\sigma \tag{1.1}$$

The incompressibility constraint is enforced by Equation 1.2 by ensuring a divergence free velocity field.

$$\nabla \cdot \vec{u} = 0 \tag{1.2}$$

The density scalar field of two or more immiscible species transported along the velocity field is described by a conservative formulation of scalar advection i.e. Equation 1.3.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{u}) = 0 \tag{1.3}$$

Expanding Equation 1.3 with product rule of partial derivatives results in Equation 1.4.

$$\frac{\partial \rho}{\partial t} + \vec{u} \cdot \nabla \rho + \rho(\nabla \cdot \vec{u}) = 0 \tag{1.4}$$

Substituting divergence of velocity with zero as per Equation 1.2 results in the final form of the transport equation as described by Equation 1.5, which is one of the most elementary aspect of fluid dynamics: material derivatives.

$$\frac{D\rho}{Dt} \equiv \frac{\partial \rho}{\partial t} + \vec{u} \cdot \nabla \rho = 0 \tag{1.5}$$

Apart from that, given sharp interfaces, molecular configurations near interfaces result in additional forces such as surface tension which can be computed as a function of the interface normal vector using Equation 1.6.

$$T_\sigma = \gamma\kappa\hat{n} = \gamma(\nabla\cdot\hat{n})\hat{n} \tag{1.6}$$

1.2 Literature Review

In two-phase computational fluid dynamics, there is an elementary requirement of tracking the immiscible interface such that the overall phase density field including the interface obeys Equation 1.5. There are many choices of methods that can satisfy said requirement. Finite difference methods are not one of them because their stability is entirely dependent on the addition of numerical diffusion, and retaining interfaces can be challenging if the interface keeps numerically diffusing by a small amount every time step. Level-set methods provide an alternative approach, whereby instead of transporting the phase field, a signed distance scalar function is transported along the velocity field with the isoline of zero representing the interface. However, this method causes a loss or gain in phase area surrounded by the isoline of zero on the level-set field i.e. these class of methods arent conservative as per Sussman *et al.* (1999). In order to keep track of the interface and keep the volume conservative, Geometric fluxing can be used on phase fields called Volume of Fluids with information of interface as seen in Figure 1.1. This information of interface requires interface reconstruction every temporal iteration. For that reason and many more, geometric fluxing can be challenging. In this paper, a very specific framework is set whereby the normal fluxes in each cell add up to zero implying divergence free velocities in every cell which is a fundamental constraint of incompressibility as per Equation 1.2. In this conservative framework, the obvious way of advection would be using flux velocity to compute gradients for transport i.e. using horizontal and

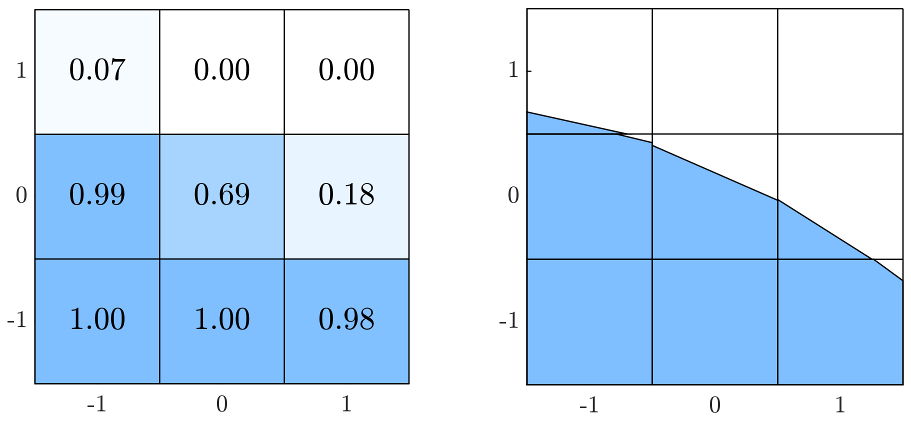

Figure 1.1: Volume of Fluid (VOF) Field and Corresponding Interface on the right

vertical velocities to flux the phase distribution field horizontally and vertically as proposed by Scardovelli and Zaleski (2003). However, overlapping fluxes prevent the ability to do this simultaneously. Depending on the split lagrangian or split eulerian methodology mentioned in said paper, one can either preserve monotonicity or total volume of fluid but not both. In the eulerian framework, monotonicity is not preserved and further correction is required to preserve interface as unphysical interfaces are generated practically everywhere. This might lead to loss or gain in volume of fluid, and worse, an extreme gain in compute times. Similarly the Split Lagrangian Advection (SLA) does preserve monotonicity by stretching its contents based on flux velocities and projecting it on neighbouring cells, but when repeated in the next dimension, the shift from the divergence free cell caused the whole method to not be conservative. A paper by Yang *et al.* (2010) uses flux velocities to create a transformation to project the fluid and interface on neighbouring cells to compute the flux. The method is conservative and monotone, but it fails to obey the velocity fluxes to second order accuracy as it only relies on information from a local cell and hence doesn't conserve momentum. The work done by Owkes and Desjardins (2014) proposes a solution that solves all the aforementioned problems by means of using corner velocities ascertained from averages of fluxes of local and neigbouring cells in addition to corrections to ensure conservativeness. An additional paper by Comminal *et al.* (2015) takes this concept and attempts to apply Runge-Kutta steps to the corner velocities to the point where 'geometrical errors are dominated by Piece-wise Linear Interface Reconstruction (PLIC) errors'.

There are four primary methods of PLIC reconstruction explored in this paper and they work hand in hand with the VOF advection procedures. They are finite difference method (FDM) approach as well as center of mass method (FDM (COMM)) by Parker and Youngs (1992) and Least squares volume-of-fluid interface reconstruction algorithm (LVIRA) as well as efficient Least squares volume-of-fluid interface reconstruction algorithm (ELVIRA) by Pilliod Jr and Puckett (2004).

In this paper, the goal is to compare three unsplit-methods and one directionally split method and analyse the details of their performance and inner-workings. This paper also attempts to combine and average eulerian and lagrangian methods by geometrically averaging them in time.

Chapter 2

METHODS

2.1 Initialization

To generate an accurate volume of fluid (VOF) field, there is a need to find accurate intersections with the initial condition shape for every individual cells. One efficient way of doing that is using recursive quad-tree area resolution of the VOF field by means of the level-set field function. A known analytical level-set field is used to figure out whether any point is greater or less than zero. Greater than zero constitutes and liquid while less than constitutes the gas. As such to resolve the mass fraction within the cell, a quad-tree recursive algorithm is used zoom in further into sub-grid cells that have the interface going through it. In the smallest possible square one can use linear simplifications of interface using level set functions. See Figure 2.1 for visualization. This algorithm is significantly more efficient than a naive grid search within a cell.

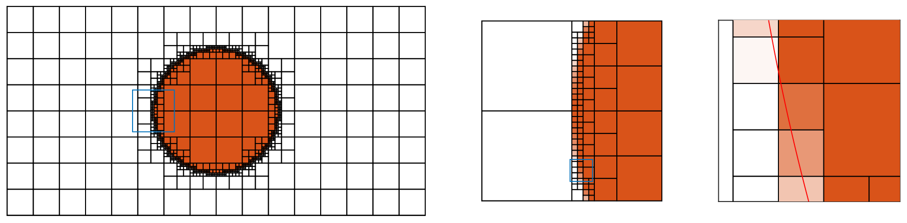

Figure 2.1: Quadtree Recursion

2.2 PLIC Reconstruction

Typically, the PLIC geometry of interface and the side containing the liquid is expressed as Equation 2.1.

$$\hat{n} \cdot \vec{x} \geq \alpha \tag{2.1}$$

This is represented on a normalized cell of unit length one, extending from $-1/2$ to $1/2$ in both dimensions. ψ, which is amount of fraction of the cell that is liquid, can be computed using Equation 2.2 and Equation 2.3.

$$\psi'(\hat{n}, \alpha) = \frac{1}{2} + \frac{\sum_{k=1}^{4} \left(\frac{d_k^3}{|d_k|} \right)}{4 n_x n_y} \tag{2.2}$$

$$d_k = (-1)^k \left(\alpha n_x^2 + \alpha n_y^2 \right) - (-1)^{\left\lfloor \frac{k}{2} \right\rfloor} \left(\frac{n_x}{2} \right) + (-1)^{\left\lceil \frac{k}{2} \right\rceil} \left(\frac{n_x}{2} \right) \tag{2.3}$$

Vice-versa, α can be computed using Algorithm 1 with inputs of α and interface normal.

2.2.1 *Finite Difference Method (FDM)*

At a given interface location, the interface normal can be ascertained as Equation 2.4.

$$\hat{n} = \frac{\nabla \psi}{|\nabla \psi|} \tag{2.4}$$

Taking naive simple central finite differences between neighboring cells could lead to incorrect interface reconstruction as they skip the information from the central cell. Hence the solution is to use every neighboring cell as shown in the Equation 2.5 and Equation 2.6 to allow for a apposite generalization of the interface normal.

$$m_{x i,j} = \sum_{k=-1}^{1} c_k \left(\psi_{i+1,j+k} - \psi_{i-1,j+k} \right) \tag{2.5}$$

$$m_{y i,j} = \sum_{k=-1}^{1} c_k \left(\psi_{i+k,j+1} - \psi_{i+k,j-1} \right) \tag{2.6}$$

Algorithm 1: Ascertaining α from $\hat{n}$ and ψ

```
1: $s \leftarrow \mid n_x \mid + \mid n_y \mid$
2: $m_x \leftarrow n_x/\mathrm{s}$
3: $m_y \leftarrow n_y/\mathrm{s}$
4: if $\psi$ is between 0 and 1 then
5:     $m_1 \leftarrow min(\mid m_y \mid, \mid m_y \mid)$
6:     $\psi_1 \leftarrow m_1/2(1 - m_1)$
7:     $\chi \leftarrow min(\psi, 1 - \psi)$
8:     if $\chi \leq \psi_1$ then
9:         $\alpha \leftarrow \sqrt{(2m_1(1 - m_1)\chi)}$
10:    end if
11:    if $\chi \geq \psi_1$ & $\chi \leq 0.5$ then
12:        $\alpha \leftarrow (1 - m_1)\chi + 0.5m_1$
13:    end if
14:    if $\psi \geq 0.5$ then
15:        $\alpha \leftarrow 1 - \alpha$
16:    end if
17:    $\alpha \leftarrow \alpha + min(0, m_x) + min(0, m_y)$
18:    $\alpha \leftarrow (\alpha - 0.5(m_x + m_y))/\sqrt{m_x^2 + m_y^2}$
19: end if
```

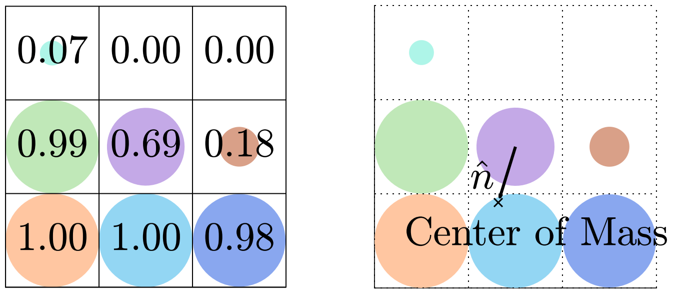

Figure 2.2: Normal computed using Center of Mass of center and neighboring cells.

It should be noted that c_k is 2 for $k = 0$, else it is 1. This methodology was proposed by Parker and Youngs (1992).

2.2.2 Center of Mass Method (FDM (COMM))

This method attempts to find the approximate center mass of 9 cell stencil based on the fractional volume being considered as a mass centered at the respective cell as described in Equation 2.7 and Equation 2.8. The resulting position of center of mass with respect to the origin of 9-cell-stencil is the direction of the interface as seen in figure 2.2.

$$m_{xi,j} = \frac{\sum_{l=-1}^{1}\sum_{k=-1}^{1} k\left(\psi_{i+k,j+l}\right)}{\sum_{l=-1}^{1}\sum_{k=-1}^{1}\left(\psi_{i+k,j+l}\right)} \tag{2.7}$$

$$m_{y_{i,j}} = \frac{\sum_{l=-1}^{1}\sum_{k=-1}^{1} l\left(\psi_{i+k,j+l}\right)}{\sum_{l=-1}^{1}\sum_{k=-1}^{1}\left(\psi_{i+k,j+l}\right)} \tag{2.8}$$

Upon closer analysis it is evident that is merely another finite difference method by Parker and Youngs (1992) without c_k in Equation 2.5 and Equation 2.6. In both cases, these normals have to be normalized to $\hat{n}$ to be compatible with many other routines such as Algorithm 1 and Algorithm 2.

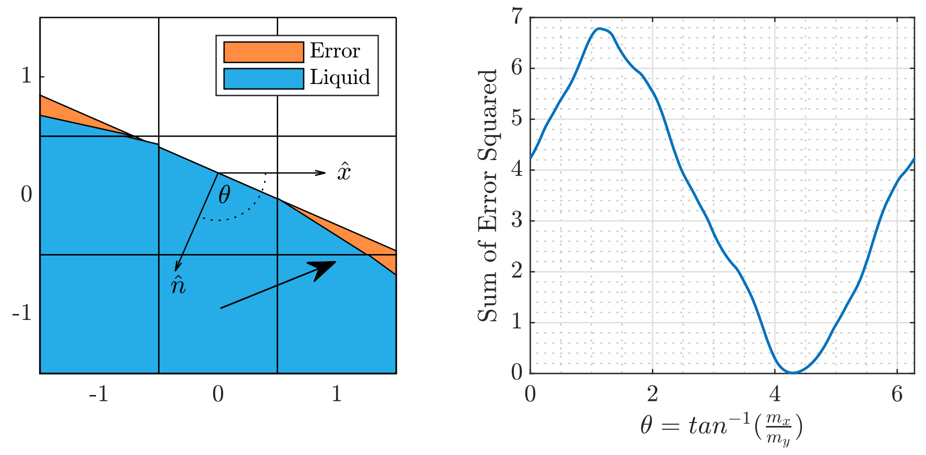

Figure 2.3: LVIRA Error Minimization

2.2.3 Least squares volume-of-fluid interface reconstruction algorithm (LVIRA) Method

Short form for Least Squares volume-of-fluid interface reconstruction algorithm, the motivation for this method by Pilliod Jr and Puckett (2004) is to minimize the squared error between 3 by 3 cell of a given discrete fields specific interface region and sample 3 by 3 cell with straight line interface with the center cell having the same volume fraction in both configurations. In two dimensions, the error is minimized along a single dimension θ.

$$\epsilon(\theta) = \sum_{l=-1}^{1}\sum_{k=-1}^{1}\left(\psi_{i+k,j+l} - \psi'\left(\langle\cos(\theta), \sin(\theta)\rangle, -\alpha_{i,j} + k\sin(\theta) + l\cos(\theta)\right)\right)^2 \quad (2.9)$$

The θ at which the error is the least possible ends up being the PLIC reconstruction normal. The minimization can be done using nelder-mead simplex or parabolic search. Since parabolic search can fail, simplex search was the choice for this paper. Nelder and Mead (1965) At each iteration, the error function needs to be evaluated. Having a good initial guess is crucial to avoiding unnecessary computations. One can use previously mentioned methods for an initial guess for the normal. Since LVIRA is computationally very expensive a set of guesses can be used to compute the normals.

2.2.4 Efficient Least squares volume-of-fluid interface reconstruction algorithm (ELVIRA) Method

This method also uses the squared shape error in a 3x3 cell stencil but instead of finding a minimum using a heuristic approach, this method only requires going through a criteria of 6 possible normal reconstructions on the basis of slope computations. The first procedure is to compute sums of rows and columns as described in Equation 2.10 and Equation 2.11.

$$C_{hk} = \sum_{k=-1}^{1} \psi_{i+k,j} \tag{2.10}$$

$$C_{vk} = \sum_{k=-1}^{1} \psi_{i,j+k} \tag{2.11}$$

It should be noted that these normal candidates also depend on overall distribution of liquid on all quadrants and as such the sign function of the complimentary derivative is used to find the complimentary direction as described in Equation 2.12, Equation 2.13 and Equation 2.14.

$$\begin{aligned}\vec{m}_{candidates} = &\left(s_v, \frac{C_{h1}-C_{h0}}{\Delta x}\right), \left(s_v, \frac{C_{h1}-C_{h-1}}{2\Delta x}\right), \left(s_v, \frac{C_{h0}-C_{h-1}}{\Delta x}\right),\\ &\left(\frac{C_{v1}-C_{v0}}{\Delta y}, s_h\right), \left(\frac{C_{v1}-C_{v-1}}{2\Delta y}, s_h\right), \left(\frac{C_{v0}-C_{v-1}}{\Delta y}, s_h\right)\end{aligned} \tag{2.12}$$

$$s_v = sgn\left(C_{v1} - C_{v-1}\right) \tag{2.13}$$

$$s_h = sgn\left(C_{h1} - C_{h-1}\right) \tag{2.14}$$

2.3 Geometric Advection

PLIC reconstruction enables us to get a profile and approximate distribution of fluid within a cell with the interface parameters. However, a fluxing scheme is needed

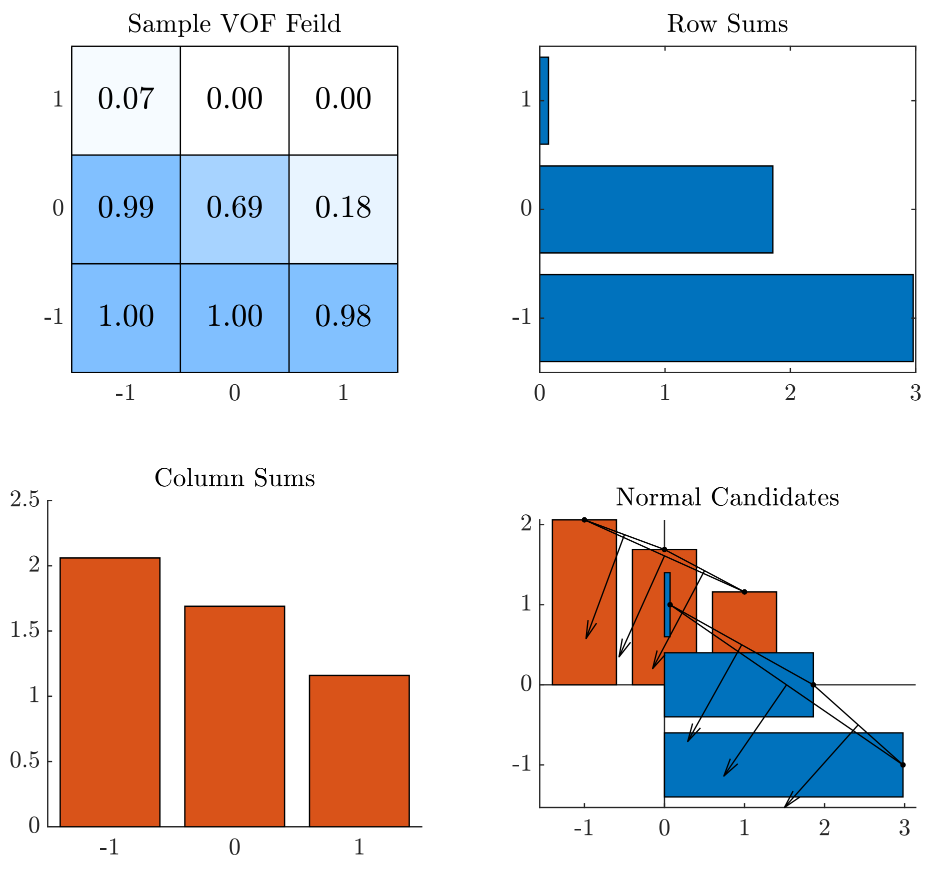

Figure 2.4: ELVIRA candidates

transport fluid material across cells while obeying certain conditions. Before going into the fluxing scheme, one has to consider the layout of the staggered velocity field. On a uniform grid, In-compressible velocity field implies said field is divergence free. Numerically speaking Equation 2.15 is a discretized version of Equation 1.2.

$$\frac{\left(u_{i+\frac{1}{2},j} - u_{i-\frac{1}{2},j}\right)}{\Delta x} + \frac{\left(v_{i,j+\frac{1}{2}} - v_{i,j-\frac{1}{2}}\right)}{\Delta y} = 0 \tag{2.15}$$

This is advantageous as it implies that the sum of volume of liquid and gas over every cell remains constant if these fluxes are obeyed. However, if flux volumes are constructed, the underlying problems of overlapping fluxes become visible. The key

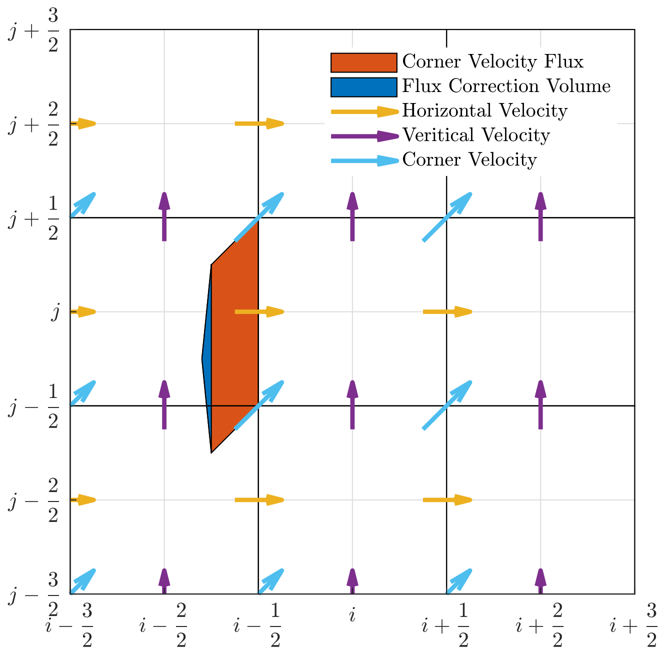

Figure 2.5: Conservative Fluxing on Staggered Grid

is to use corner velocities as they are consistent with all neighboring cells as seen in Figure 2.5.

2.3.1 Unsplit Eulerian Advection (UEA)

The problem of overlapping fluxes problem can be solved by using corner velocities. However, in order to ensure conservative fluxing, one has to add consistent volume on top of flux volume generated by corner velocities to match it with the flux volume generated from wall flux velocities. Owkes and Desjardins (2014). The resulting flux correction volume can be computed by the Equation 2.16

$$\begin{aligned} A_{correction} = u_{i-\frac{1}{2},j}\frac{\Delta x}{\Delta t} - \frac{1}{2}\frac{\Delta x}{\Delta t}(u_{i-\frac{1}{2},j-\frac{1}{2}} + u_{i-\frac{1}{2},j+\frac{1}{2}} \\ -\frac{\Delta x}{\Delta t}u_{i-\frac{1}{2},j-\frac{1}{2}}v_{i-\frac{1}{2},j+\frac{1}{2}} + \frac{\Delta x}{\Delta t}u_{i-\frac{1}{2},j+\frac{1}{2}}v_{i-\frac{1}{2},j-\frac{1}{2}}) \end{aligned} \tag{2.16}$$

One can compute a point approximately between corner projections such that the area of the flux pentagon equals the flux near the wall flux velocity. The point is

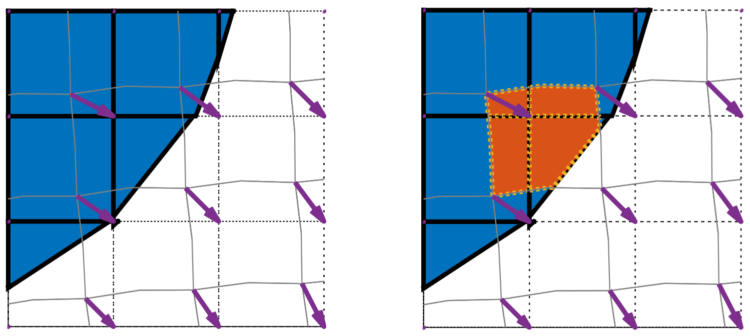

Figure 2.6: Eulerian Model for flux based on Owkes and Desjardins (2014)

normal to the line between corner projections emanating the from the midpoint of the projections. In this paper, the said method was modified in such a way that instead of computing flux volumes across cell walls, an octagon was projected backwards in space and time as per the velocity field that consistently obeys the flux. The intersection area between neighbouring cells and the octagon tells the amount of fluid present in the specific cell after transport. This can be seen in Figure 2.6. This method was used as a starting point in Comminal *et al.* (2015). This method works extremely well, but if the numerical divergence property is not satisfied as per Equation 2.15 for various reasons, a situation with cells away from the interface having values above and below zero and one arises. This could create numerical bubbles and damage the interface tracking procedure and waste valuable time doing interface reconstruction at locations where it isn't really needed.

2.3.2 Split Lagrangian Advection (SLA)

This is a method that simplifies advection to individual dimensions with interface reconstruction after individual dimensional transport. In simple terms, it is stretching and squeezing of the fluid distribution polygon within the cell as per left and right flux velocities and intersecting it with center and neighboring cells as show in Figure

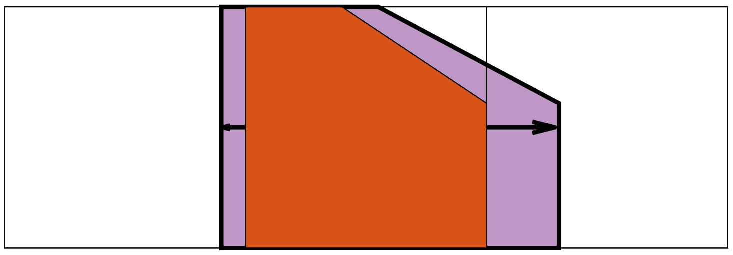

Figure 2.7: Horizontal Split Lagrange Advection

2.7. In said figure, the orange polygon represents the state of liquid clipped by the interface while the purple polygon is an extension computed from horizontal flux velocities shown by black arrows. This method has the advantage of not requiring a clipping algorithm as analytical function from Equation 2.2 is sufficient.

2.3.3 Unsplit Lagrangian Advection (ULA)

The Unsplit Lagrangian Advection (ULA) method by Yang *et al.* (2010) was modified slightly for this section as said method didn't preserve momentum. One can take the idea of Unsplit Eulerian Flux (UEA) and reverse it. Instead of extending the volume backwards and capturing the contents geometrically, one can also take the contents of current cell and distribute it over neighboring cells as described by velocity vectors. The first step is to construct an octagon forward in time with velocity that is a result of transformation due to transport. Starting from bottom-left and numbering the vertices 0 to 7, one can ascertain the even numbered vertices i.e. corner velocity projections using Equation 2.17 and Equation 2.18.

$$\vec{x}_k = \vec{p}_k + \frac{\Delta t}{\Delta h}\vec{u}_{(i,j)+\vec{p}_k} : \forall k \in 0, 2, 4, 6 \tag{2.17}$$

$$\vec{p}_k = 0.5\left[-(-1)^{\left\lfloor \frac{k}{4} \right\rfloor}, (-1)^{\left\lceil \frac{k}{4} \right\rceil}\right]^{\top} \tag{2.18}$$

The odd numbered vertices are equated as a correction of flux using Equation 2.19 and Equation 2.20.

$$Area\left(\vec{p}_{k-1}, \vec{p}_{k+1}, \vec{x}_{k+1}, \vec{x}_k, \vec{x}_{k-1}\right) = \Phi_{\lfloor k/2 \rfloor} : \forall k \in 1,3,5,7 \tag{2.19}$$

$$\Phi_k = \frac{\Delta t}{\Delta x}\left\{v_{i,j-\frac{1}{2}}, u_{i+\frac{1}{2},j}, -v_{i,j+\frac{1}{2}}, -u_{i-\frac{1}{2},j}\right\} \tag{2.20}$$

Since Equation 2.19 is an under-determined system we use the Equation 2.21 to close the equation and solve for the odd numbered vertices. All these points culminate to form the Projected Octagon (PO) as seen in Figure 2.8.

$$(\vec{x}_{k+1} - \vec{x}_{k-1}) \cdot (\vec{x}_{k-1} - 2\vec{x}_k + \vec{x}_{k+1}) = 0 : \ \forall k \in 1,3,5,7 \tag{2.21}$$

When all corner velocities of a cell equal each other, the odd-numbered $\vec{x}_k$ becomes the midpoint of $\vec{x}_{k+1}$ and $\vec{x}_{k-1}$. If the cell volume fraction before advection i.e. $\psi^n_{i,j}$ was one then our projection polygon is the octagon itself. However, if the cell has an interface, one takes the points at which the interface intersects the cell wall using a line clipping algorithm and then computes their appropriate location on the projected octagon (PO) as described in Algorithm 2. This appropriate location on the projection can be computed using an 8-point shape-function described in Equation 2.22 and used in Equation 2.23.

$$N(x,y) = \begin{bmatrix} (|x|-x)(|y|-y) \\ (-2|x|+1)(|y|-y) \\ (|x|+x)(|y|-y) \\ (|x|+x)(-2|y|+1) \\ (|x|+x)(|y|+y) \\ (-2|x|+1)(|y|+y) \\ (|x|-x)(|y|+y) \\ (|x|-x)(-2|y|+1) \end{bmatrix} \tag{2.22}$$

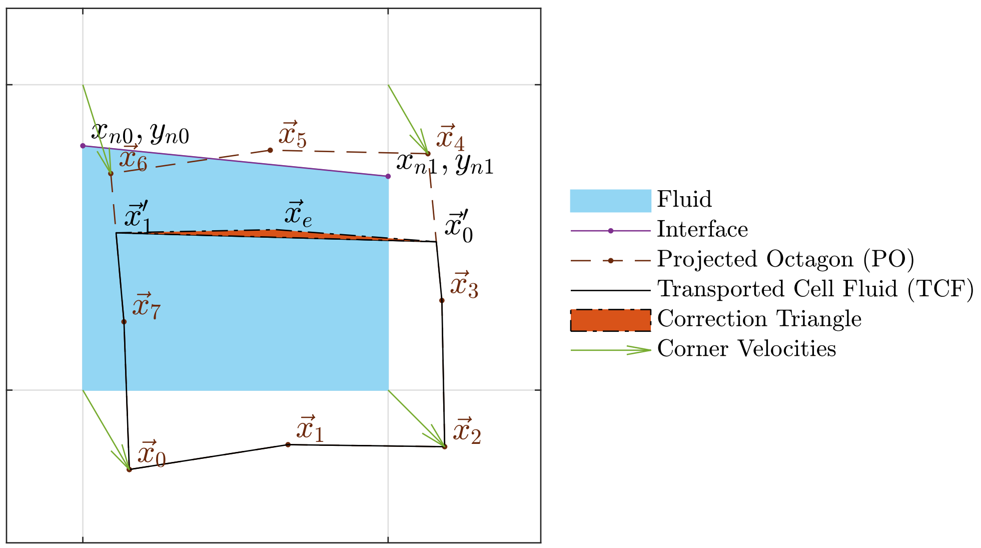

Figure 2.8: Interface Cell Lagrangian Projection

$$\begin{bmatrix} x_n' \\ y_n' \end{bmatrix} = \begin{bmatrix} x_0 & x_1 & \cdots & x_7 \\ y_0 & y_1 & \cdots & y_7 \end{bmatrix} N(x_n, y_n) \tag{2.23}$$

This shape function preserves distance ratio between points upon projection. Now that an approximately reconstructed version of the interface is reconstructed in the octagon, the projected polygon needs to be constructed that has the same area as the volume of fraction of the parent cell. This can be done simply by adding a correction triangle similar to flux correction shown in the eulerian framework in Figure 2.5.

The point $\vec{x}_e$ ensures that the polygon TCF equals $Area(PO)\psi^n_{i,j}$ which would essentially be $\psi^n_{i,j}$ if the net flux is zero in the cell. As per Algorithm 2, there are protocols set in place to ensure prevention of anomalous geometry. Specifically, the point $\vec{x}_e$ is extended in such a way that it is snapped to horizontal, vertical or diagonal rails of the projected octagon as long as both the interface points before projection, ($\vec{x}_n$), are not in the same quadrant.

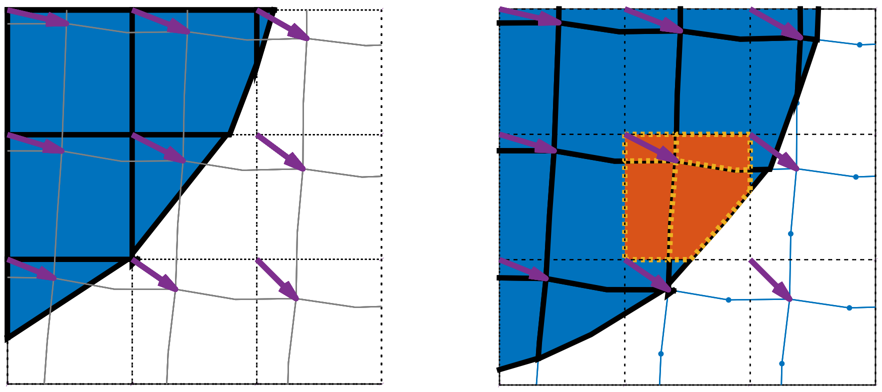
Figure 2.9: Unsplit Lagrangian Advection (ULA)

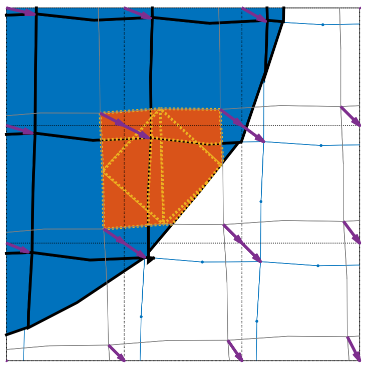
Figure 2.10: Unsplit Averaged Eulerian Lagrangian Advection (UAELA)

Once the projected polygon has been constructed, one can intersect it with all neighboring cells including itself and accumulate the area of the intersected polygon on the respective cell of $\psi_{i,j}^{n+1}$. With information of the bounding box of said polygon one can do 3 to 8 intersections instead of 9 with majority of cases being 3 especially when all corner velocities point in the same quadrant.

2.3.4 Unsplit Average of Eulerian and Lagrangian Advection (UAELA)

This method is a mere geometric mean of Unsplit Eulerian Advection (UEA) Method and Unsplit Lagrangian Advection (ULA) Method. Taking the aforementioned Unsplit Lagrangian Advection (ULA) but projecting it only as far as $\Delta t/2$ on deformed cells advected backwards $\Delta t/2$ gives the geometrically averaged version as shown in Figure 2.10. This method can work with a single velocity in time, $\vec{u}^n$, but giving it two velocities, i.e. $\vec{u}^{n+1}$ and $\vec{u}^n$ or $\vec{u}^{n+\frac{1}{2}}$ can give it second order accuracy.

2.4 Clipping

In this paper, the most commonly used clipping algorithm is Sutherland-Hodgeman clipping algorithm if Equation 2.2 was not used. This algorithm can clip convex and concave polygons as long as the clipping polygon is convex as described by Sutherland and Hodgman (1974). Another algorithm used here is line clipping algorithm for ascertaining the intersections of walls and interface in an efficient fashion. Specifically, Sutherland-Cohen line clipping algorithm was used although, there have been many other algorithms that have been significantly more optimized over the years to do the same task. In theory, a generic line-clipping algorithm with a rectangular clipper could be used in Unsplit Lagrangian Advection (ULA) method as long as the grid is cartesian likely speeding up the routines.

Algorithm 2: Ascertaining Polygon TCF from α, $\hat{n}$ and ψ

```
1: $\vec{x}_{n_k} \leftarrow \alpha\vec{n} + (-1)^k[n_y, -n_x]^\top \forall k = 0, 1$
2: $\vec{x}_{n_{0,1}} \leftarrow CohenSutherlandLineClip(\vec{x}_{n_{0,1}}, -.5, .5, -.5, .5)$
3: Append $\begin{bmatrix}\vec{x}_0 \cdots \vec{x}_7\end{bmatrix} N(\vec{x}_{n_0})$ to TCF
4: Append all $\vec{x}_n$ that are in fluid in counter-clockwise order of pre-projection
   state to TCF.
5: Append $\begin{bmatrix}\vec{x}_0 \cdots \vec{x}_7\end{bmatrix} N(\vec{x}_{n_1})$ to TCF
6: $n_{case} \leftarrow 2n_xn_y$
7: if $n_{case} > 0.9$ or $\vec{x}_{n_{0,1}}$ are on the same quadrants then
8:    $x_{r_0} \leftarrow x_0$, $y_{r_0} \leftarrow y_0$, $x_{r_1} \leftarrow x_4$, $y_{r_1} \leftarrow y_4$
9: else if $n_{case} < -0.9$ or $\vec{x}_{n_{0,1}}$ are on different quadrants then
10:    $x_{r_0} \leftarrow x_2$, $y_{r_0} \leftarrow y_2$, $x_{r_1} \leftarrow x_6$, $y_{r_1} \leftarrow y_6$
11: else
12:    if $|n_x| > |n_y|$ then
13:       $x_{r_0} \leftarrow x_7$, $y_{r_0} \leftarrow y_7$, $x_{r_1} \leftarrow x_3$, $y_{r_1} \leftarrow y_3$
14:    else
15:       $x_{r_0} \leftarrow x_1$, $y_{r_0} \leftarrow y_1$, $x_{r_1} \leftarrow x_5$, $y_{r_1} \leftarrow y_5$
16:    end if
17: end if
18: $A \leftarrow \begin{bmatrix} y'_{n0} - y'_{n1} & x'_{n0} - x'_{n1} \\ y_{r0} - y_{r1} & x_{r0} - x_{r1} \end{bmatrix}$
19: $\vec{b} \leftarrow \begin{bmatrix} x'_{n1}y'_{n0} - x'_{n0} + 2\left(\psi_{i,j}Area(PO) - Area(TCF)\right) \\ -\left(x_{r0} - x_{r1}\right)y_{r0} - \left(-y_{r0} - y_{r1}\right)x_{r0} \end{bmatrix}$
20: $\vec{x}_e \leftarrow A^{-1}b$.
21: Append $\vec{x}_e$ to TCF
```

Chapter 3

RESULTS

3.1 Convergence of Shape Error

The shape Error is described as the absolute value of difference in Volume of Fluid (VOF) fields as described in Equation 3.1.

$$\epsilon_{shape} = \Delta x \Delta y \sum_{j=0}^{N} \sum_{i=0}^{M} |\psi_{i,j}^{t=0} - \psi_{i,j}^{t=t_f}| \tag{3.1}$$

It is expected that post transport, the final state of the VOF field will produce shape errors convergent to zero in two test cases evaluated ahead.

3.1.1 Zalesak's Disk on a Circular Velocity Field

In this benchmark, a specific VOF field known as zalesak's disk is advected. Said field results from a level-set field shown in Listing 3.1.

```
double zalesak(double x, double y){
  double R = sqrt(pow(x-0.5,2)+pow(y-0.75,2));
  double phi = 0.15-R;
  double bottom = 0.75-0.15*cos(asin(0.025/0.15));
  double dfc = phi;
  if ((y > 0.85) && (R < 0.15))
    phi = min(dfc,y-0.85);
  if ((y < 0.85) && (x < 0.475) && (R < 0.15))
    phi = min(dfc,0.475-x);
  if ((y > 0.85) && (x < 0.475) && (R < 0.15))
```

```
    phi = min(dfc, sqrt(pow(0.475-x,2)+pow(y-0.85,2)));
  if ((y < 0.85) && (x > 0.525) && (R < 0.15))
    phi = min(dfc,x-0.525);
  if ((y > 0.85) && (x > 0.525) && (R < 0.15))
    phi = min(dfc, sqrt(pow(0.525-x,2)+pow(y-0.85,2)));
  if ((y < 0.85) && (x > 0.525) && (R < 0.15))
    phi = min(dfc,x-0.525);
  if ((x > 0.475) && (x < 0.525) && (y < 0.85) && (y > bottom
   ))
    phi = min(0.85-y,min(0.525-x,x-0.475));
  if ((x > 0.475) && (x < 0.525) && (y < bottom))
    phi = min(norm(bottom-y,x-0.475),norm(bottom-y,x-0.525));
  double sgn = (R<0.15);
  if ((x>0.475) && (x<0.525) && (y<0.85))
    sgn = 0;
  sgn = 2.0*sgn-1.0;
  phi = fabs(phi)*sgn;
  return phi;
}
```
Listing 3.1: Level-Set description of Zalesak's Disk.

The field is advected on a velocity field as described by Equation 3.2 on domain extending between 0 and 1 on both dimensions and 0 and 2π in the temporal dimension. This test shows the effect of one whole rotation on zalesak's disk.

$$\vec{u} = \begin{bmatrix} y - 0.5 \\ 0.5 - x \end{bmatrix} \tag{3.2}$$

For the sake of consistency, a Δt of $\frac{3\Delta x}{4\pi}$ was used every iteration to have a stable and

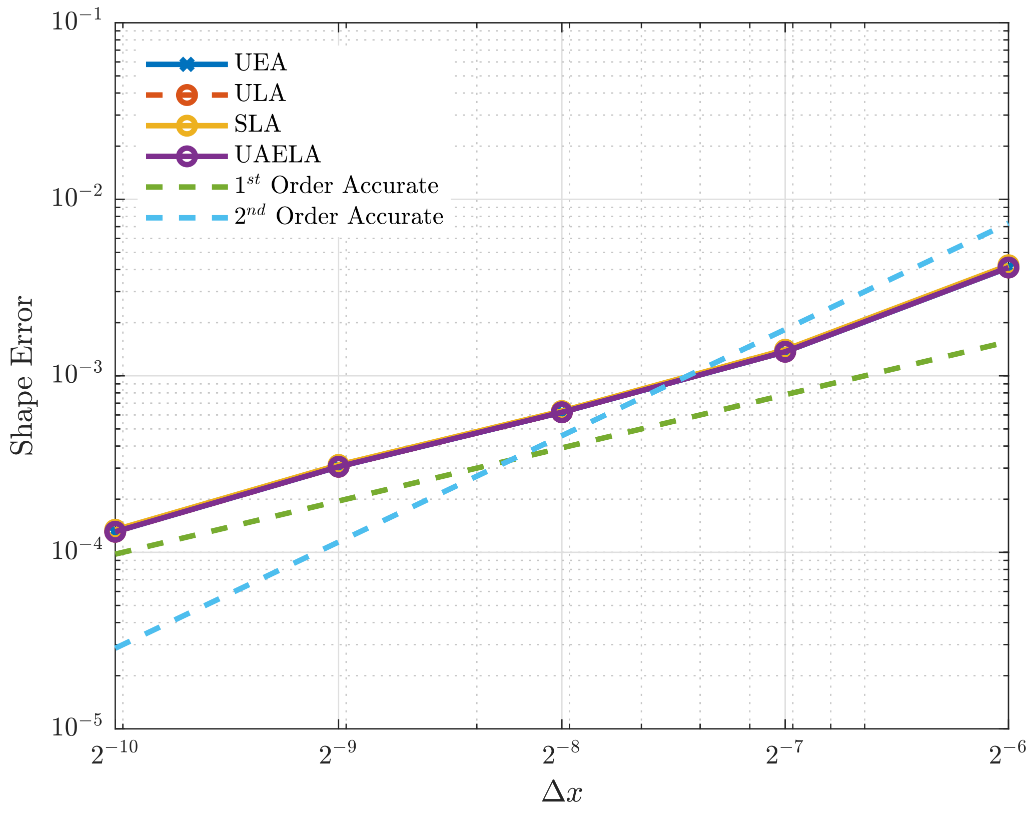

Figure 3.1: Convergence for FDM

irrational time step until the $t = 2\pi$ whereby an appropriate Δt was computed to stop at precisely $t = 2\pi$.

Finite Difference Method (FDM)

All advection methods give more or less the same results which make sense since the velocity field components don't change along their respective dimension. The unsplit methods effectively and practically end up reproducing the Split Largrangian Advection (SLA). The shape error reduces with only first order accuracy as seen in Table 3.1 and Figure 3.1. This is partly due to sharp interface bends on the initial condition and the fact that PLIC reconstructions of the cells are not restricted to the center of the cell while the central difference is restricted to that location, incapable

Table 3.1:
Shape Error for FDM

Δx	**UEA**		**ULA**		**SLA**		**UAELA**	
$^1/_{64}$	4.10e-03	Ratio	4.11e-03	Ratio	4.23e-03	Ratio	4.11e-03	Ratio
$^1/_{128}$	1.37e-03	3.00	1.37e-03	3.00	1.41e-03	3.01	1.37e-03	3.00
$^1/_{256}$	6.21e-04	2.20	6.22e-04	2.20	6.32e-04	2.22	6.22e-04	2.20
$^1/_{512}$	3.05e-04	2.03	3.05e-04	2.04	3.13e-04	2.02	3.05e-04	2.04
$^1/_{1024}$	1.31e-04	2.34	1.31e-04	2.34	1.34e-04	2.33	1.31e-04	2.34

Table 3.2:
Shape Error for FDM (COMM)

Δx	**UEA**		**ULA**		**SLA**		**UAELA**	
$^1/_{64}$	5.45e-03	Ratio	5.46e-03	Ratio	5.66e-03	Ratio	5.46e-03	Ratio
$^1/_{128}$	2.35e-03	2.32	2.35e-03	2.32	2.43e-03	2.33	2.35e-03	2.32
$^1/_{256}$	1.13e-03	2.09	1.13e-03	2.09	1.17e-03	2.07	1.13e-03	2.09
$^1/_{512}$	6.57e-04	1.72	6.57e-04	1.72	6.79e-04	1.73	6.57e-04	1.72
$^1/_{1024}$	3.79e-04	1.73	3.79e-04	1.73	4.00e-04	1.70	3.79e-04	1.73

of achieving second order accuracy. Figure 3.2 shows the end result of advection with various transport methods combined with the optimal finite difference reconstruction for multiple resolutions.

Center of Mass Method (FDM (COMM))

All advection methods give more or less the same results just like the previous PLIC method. The shape errors are systematically higher as seen in Table 3.2 and Figure 3.3 which is symptomatic of the method itself. There are visible oscillations on interface clearly implying this reconstruction method is not ideal. Nevertheless, it is a viable option since a first order convergence is seen. Figure 3.4 shows the end

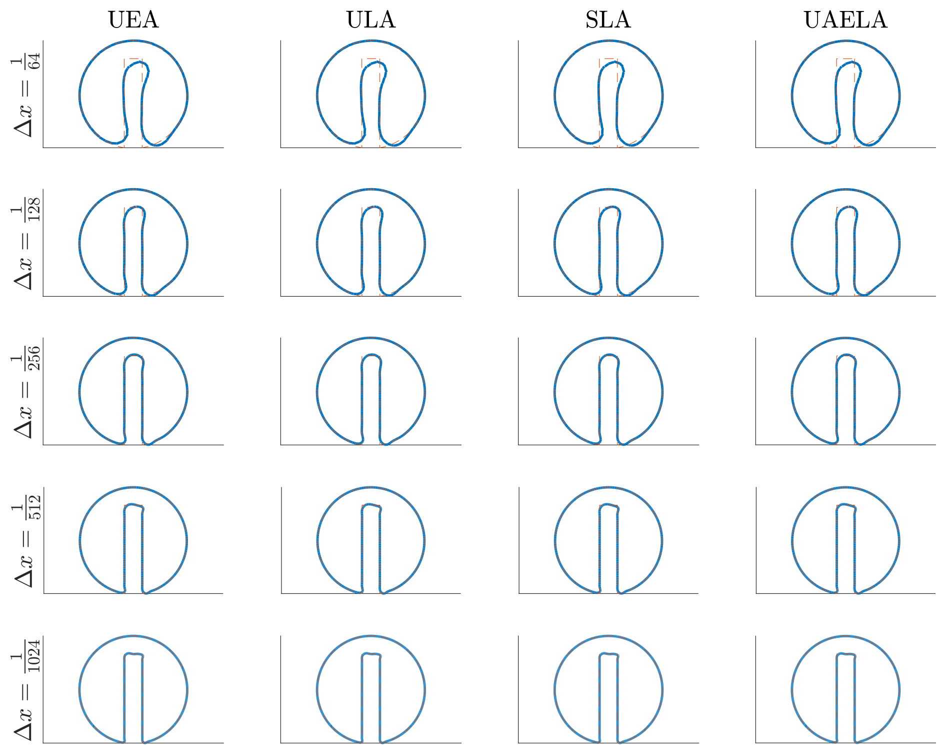

Figure 3.2: Results at t = 2π for FDM

result of advection with various transport methods combined with center of mass reconstruction for multiple resolutions.

ELVIRA

Again, in this case all advection methods give more or less the same results. With ELVIRA method, a marginal improvement is seen in comparison to the first FDM method as shown in Table 3.3 and Figure 3.5. Figure 3.6 shows the end result of advection with various transport methods and ELVIRA for multiple resolutions.

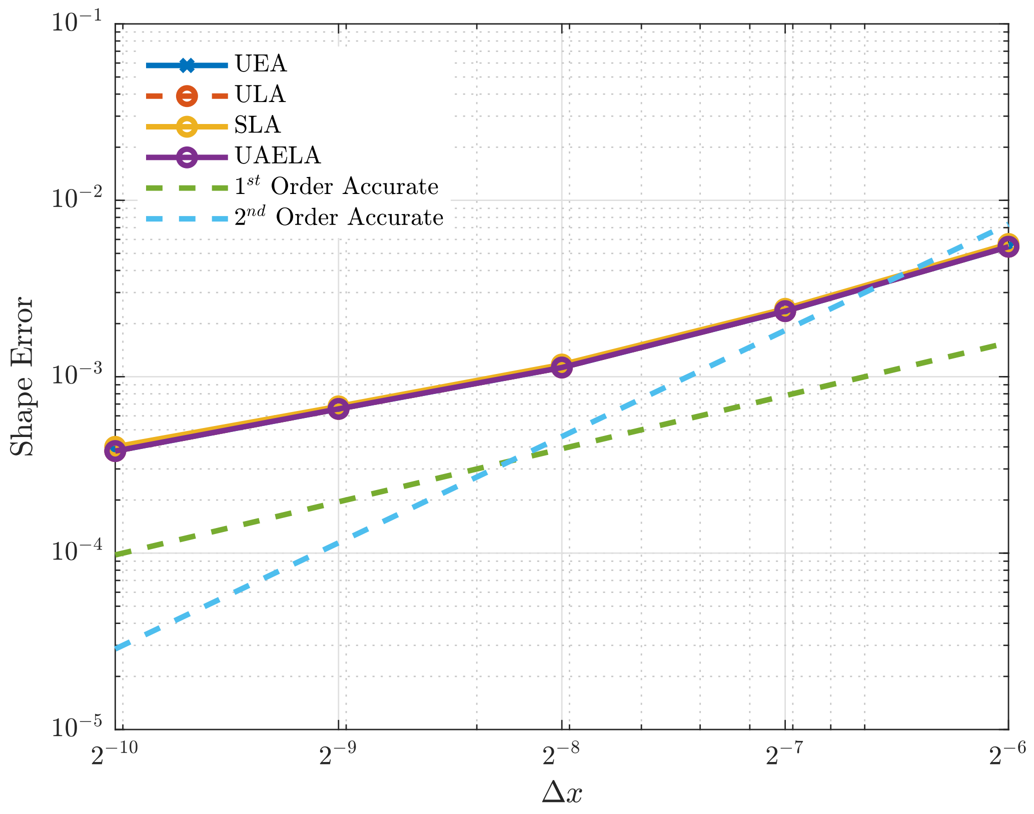

Figure 3.3: Convergence for FDM (COMM)

Table 3.3:
Shape Error for ELVIRA

Δx	**UEA**		**ULA**		**SLA**		**UAELA**	
$^1/_{64}$	4.01e-03	**Ratio**	4.02e-03	**Ratio**	4.12e-03	**Ratio**	4.01e-03	**Ratio**
$^1/_{128}$	1.29e-03	3.11	1.29e-03	3.11	1.32e-03	3.12	1.29e-03	3.11
$^1/_{256}$	5.47e-04	2.35	5.48e-04	2.36	5.55e-04	2.38	5.48e-04	2.36
$^1/_{512}$	2.51e-04	2.18	2.51e-04	2.18	2.56e-04	2.17	2.51e-04	2.18
$^1/_{1024}$	9.86e-05	2.54	9.86e-05	2.54	1.01e-04	2.54	9.86e-05	2.54

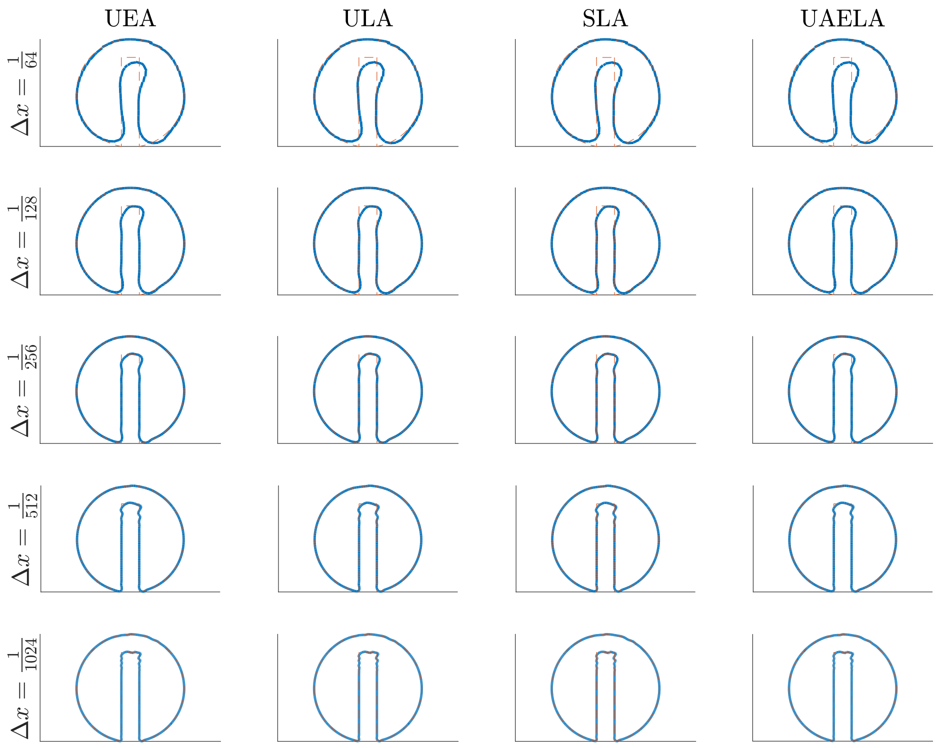

Figure 3.4: Results at t $= 2\pi$ for FDM (COMM)

Table 3.4:
Shape Error for LVIRA

Δx	**UEA**		**ULA**		**SLA**		**UAELA**	
$^1/_{64}$	4.15e-03	**Ratio**	4.16e-03	**Ratio**	4.25e-03	**Ratio**	4.15e-03	**Ratio**
$^1/_{128}$	1.31e-03	3.16	1.32e-03	3.16	1.35e-03	3.16	1.31e-03	3.16
$^1/_{256}$	5.59e-04	2.35	5.59e-04	2.35	5.67e-04	2.38	5.59e-04	2.35
$^1/_{512}$	2.57e-04	2.17	2.57e-04	2.17	2.62e-04	2.16	2.57e-04	2.17
$^1/_{1024}$	1.01e-04	2.55	1.01e-04	2.55	1.03e-04	2.55	1.01e-04	2.55

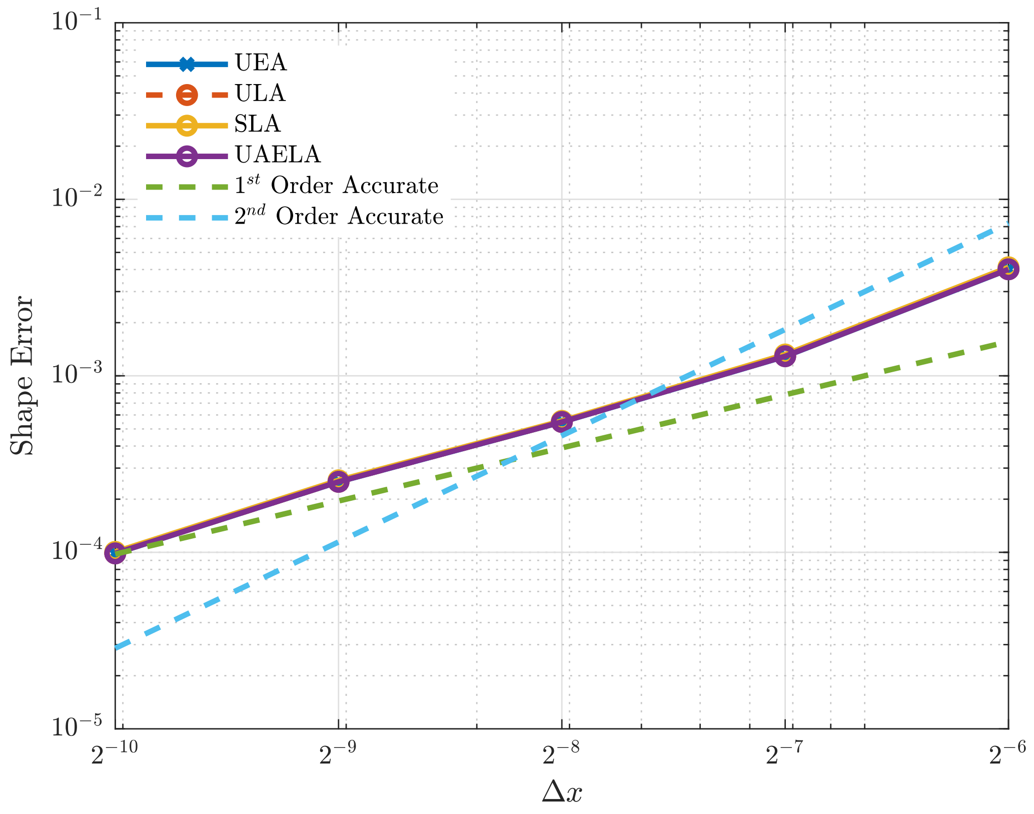

Figure 3.5: Convergence for ELVIRA

LVIRA

Just like the previous method, all advection methods give more or less the same results as seen Table 3.4 and Figure 3.7. With LVIRA method there is no visible improvement over ELVIRA. Figure 3.8 shows the end result of advection with various transport methods combined with LVIRA reconstruction for multiple resolutions.

3.1.2 Deformation Feild

In this benchmark, a specific VOF field that represents a circular concentration of fluid with a radius of 0.15 at location x = 0.5 and y = 0.75 is transported. The field is transported on a velocity field of Equation 3.3 on domain extending from 0 to 1 on

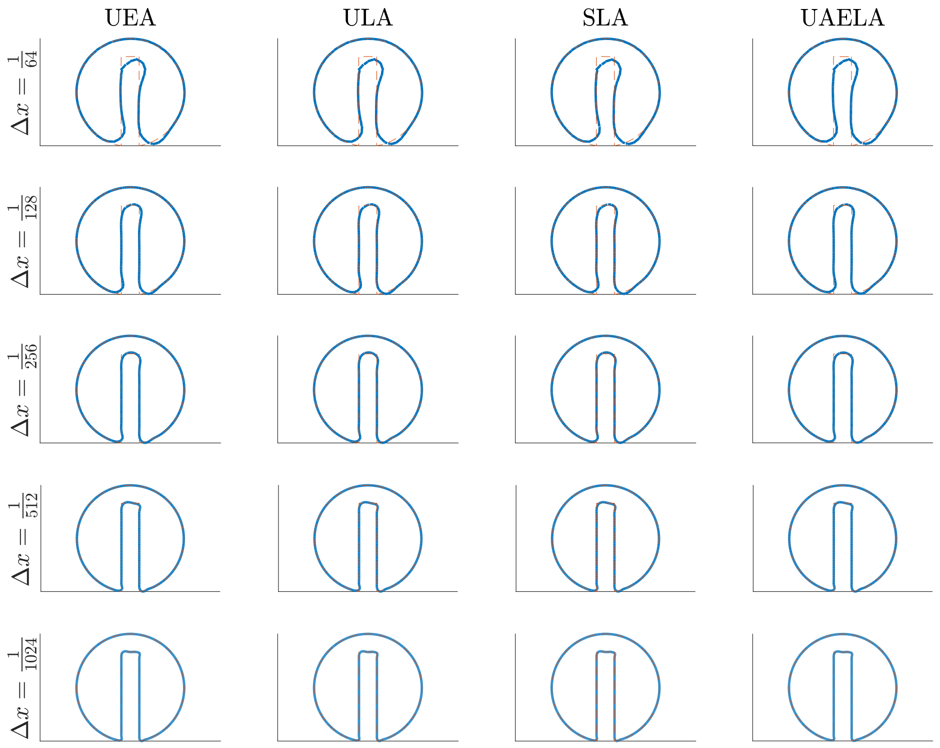

Figure 3.6: Results at t = 2π for ELVIRA

both spatial dimensions and $t = 0$ and $t = 8$ in the temporal dimension. This test shows the effect of a sheared velocity field on a circular disk. The analytical solution of this transport is identical to the initial condition.

$$\vec{u}(x, y, t) = \begin{bmatrix} -\sin^2(\pi x)\, sin\,(2\pi y) \cos\left(\frac{\pi t}{8}\right) \\ +sin\,(2\pi x) \sin^2(2\pi y) \cos\left(\frac{\pi t}{8}\right) \end{bmatrix} \tag{3.3}$$

For the sake of consistency, a Δt of $\frac{3\Delta x}{4\pi}$ was used every iteration to have a stable and irrational time step until the $t = 8$ whereby an appropriate Δt was computed to stop at precisely $t = 8$. The velocity used at each time step is $\vec{u}(x, y, t + \frac{\Delta t}{2})$ which is one of the requirements to get second order accuracy. In the absence of an analytical

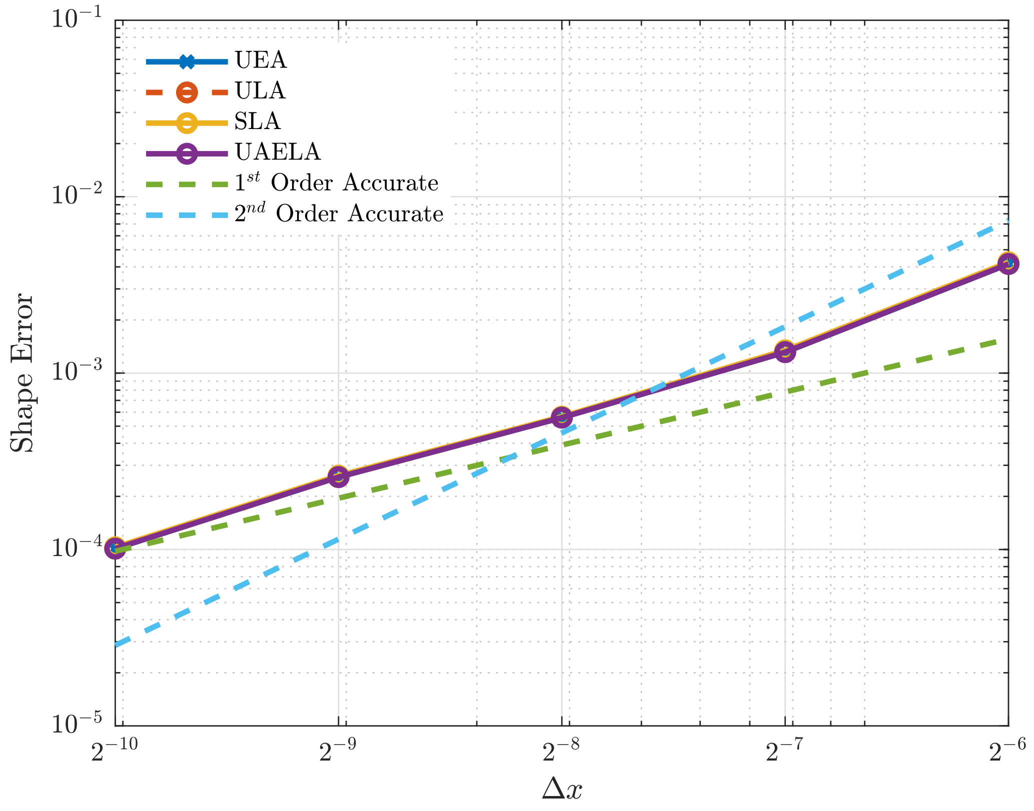

Figure 3.7: Convergence for LVIRA

velocity field, one would approximate it with Equation 3.4.

$$\vec{u}(x, y, t + \frac{\Delta t}{2}) \approx \frac{4}{3}\vec{u}(x, y, t) - \frac{1}{3}\vec{u}(x, y, t - \Delta t) + O(\Delta t^2) \tag{3.4}$$

It should be noted that for the Unsplit Averaged Eulerian Lagrangian Advection (UAELA) method, the velocity used at each time-step used is $\vec{u}(x, y, t)$.

Finite Difference Method (FDM)

In the case of deformation field, it was observed that Split Lagrangian Advection (SLA) produces the most error as seen in Figure 3.10. Also, the Unsplit Lagrangian Advection (ULA) and Unsplit Eulerian Advection (UEA) methods converge with practically the same results as seen in Table 3.5 and Figure 3.9. Compared to SLA,

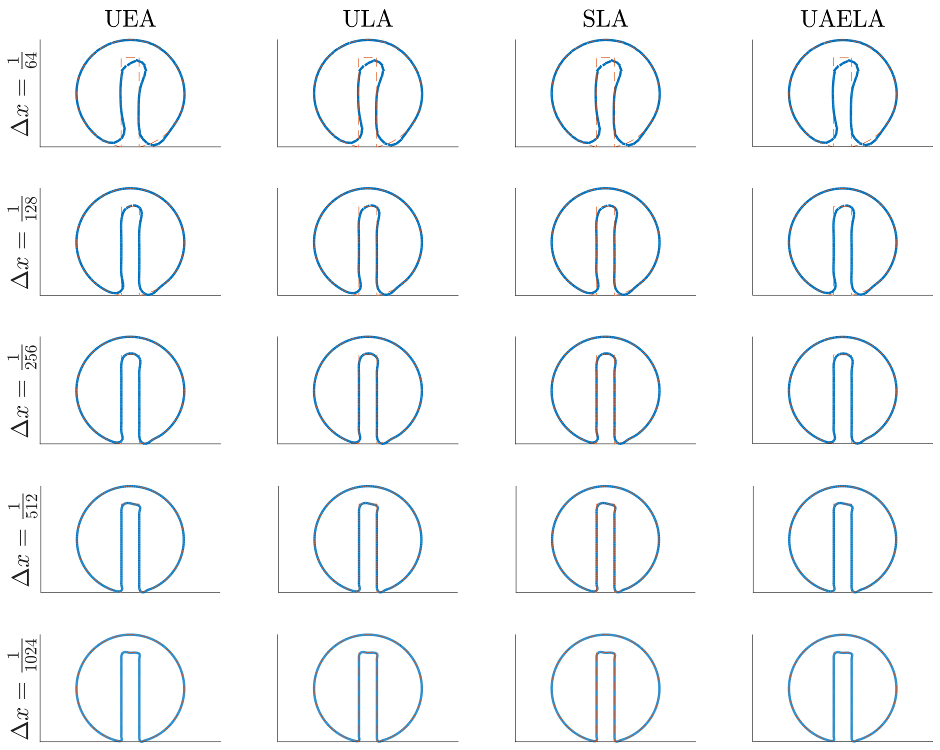

Figure 3.8: Results at t = 2π for LVIRA

Table 3.5:
Shape Error for FDM

Δx	**UEA**		**ULA**		**SLA**		**UAELA**	
$^1/_{64}$	1.49e-02	**Ratio**	1.47e-02	**Ratio**	1.99e-02	**Ratio**	1.51e-02	**Ratio**
$^1/_{128}$	2.34e-03	6.37	2.35e-03	6.25	1.01e-02	1.97	2.94e-03	5.15
$^1/_{256}$	6.76e-04	3.46	6.75e-04	3.47	4.80e-03	2.11	9.44e-04	3.11
$^1/_{512}$	2.26e-04	2.99	2.26e-04	2.99	2.37e-03	2.03	3.38e-04	2.79
$^1/_{1024}$	9.45e-05	2.39	9.45e-05	2.39	1.18e-03	2.00	1.50e-04	2.26

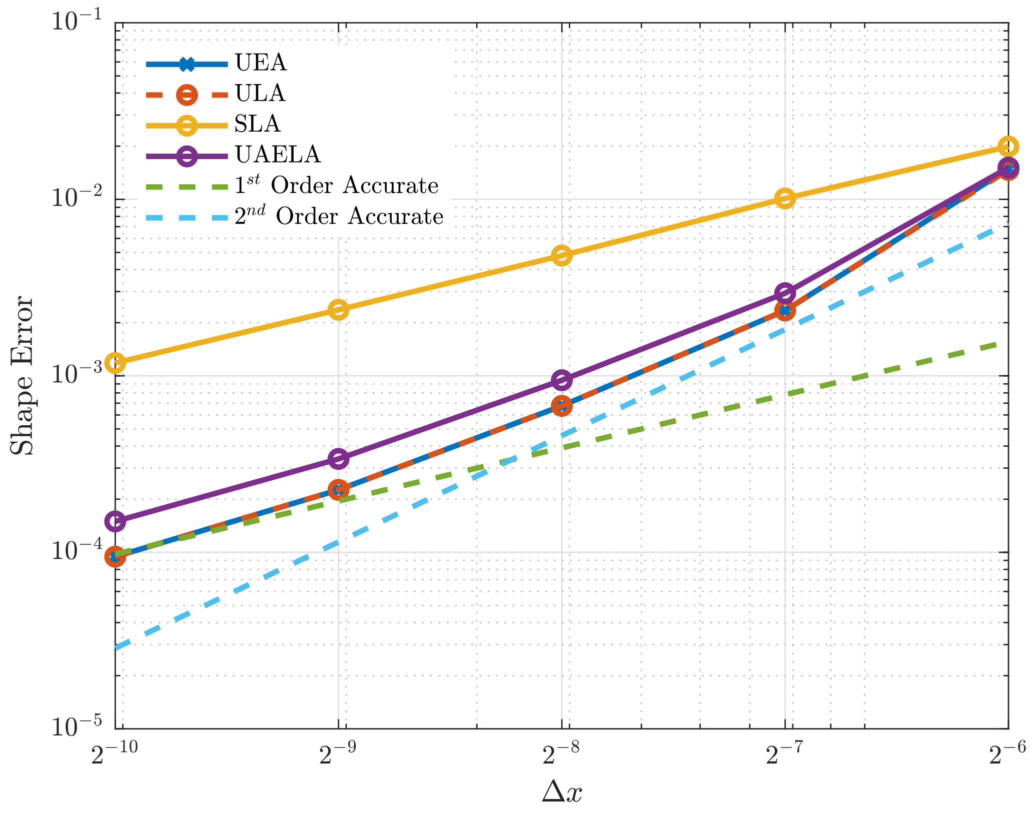

Figure 3.9: Convergence for FDM

they have approximately 10 times less error. The convergence order is second at lower resolutions but settles down to first as the resolution is increased. The Unsplit Averaged Eulerian Lagrangian Advection (UAELA) methods in theory would have produced the same results as the Unsplit Lagrangian and Eulerian Advection methods if the 2 velocities used for each iteration was at $\vec{v}(x, y, t + \frac{\Delta t}{2})$ or averaged to that which it wasn't.

Center of Mass Method (FDM (COMM))

Just like Zalesak's Disk, numerical oscillatory artifacts are seen, which again is symptomatic of the unweighted finite difference method itself. However like the previous finite difference method, Unsplit Eulerian Advection (UEA) and Unsplit Lagrangian

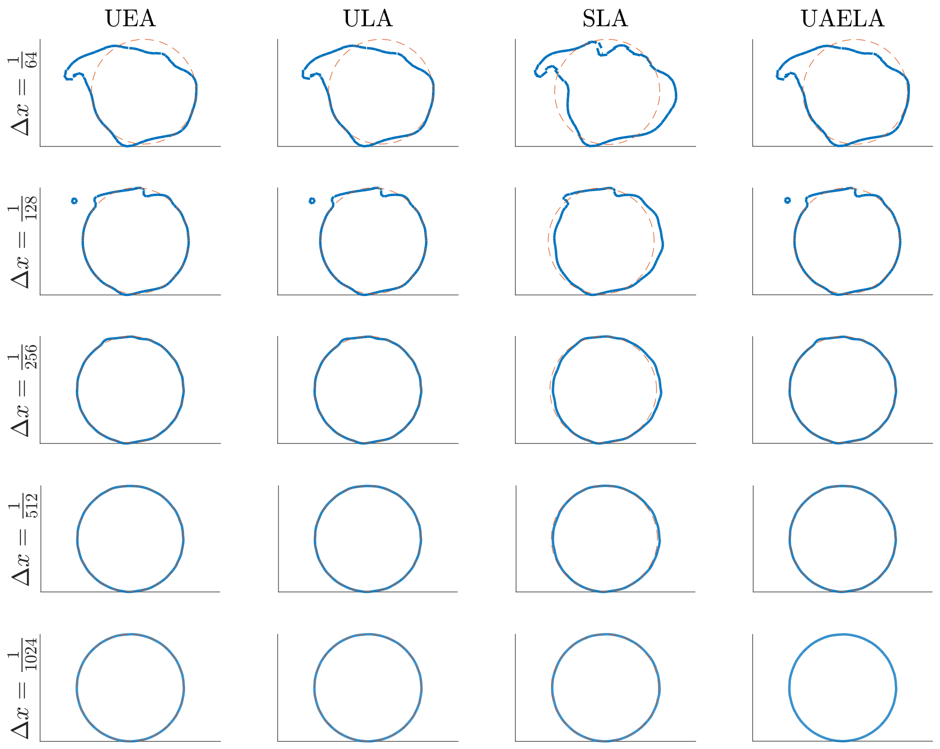

Figure 3.10: Results at t = 8 for FDM

Table 3.6:
Shape Error for FDM (COMM)

Δx	**UEA**		**ULA**		**SLA**		**UAELA**	
1/64	2.14e-02	**Ratio**	2.11e-02	**Ratio**	2.10e-02	**Ratio**	2.17e-02	**Ratio**
1/128	3.40e-03	6.29	3.40e-03	6.21	1.10e-02	1.92	3.87e-03	5.60
1/256	1.33e-03	2.55	1.33e-03	2.55	4.96e-03	2.21	1.46e-03	2.64
1/512	7.25e-04	1.84	7.25e-04	1.84	2.49e-03	2.00	7.56e-04	1.94
1/1024	3.68e-04	1.97	3.68e-04	1.97	1.27e-03	1.96	3.84e-04	1.97

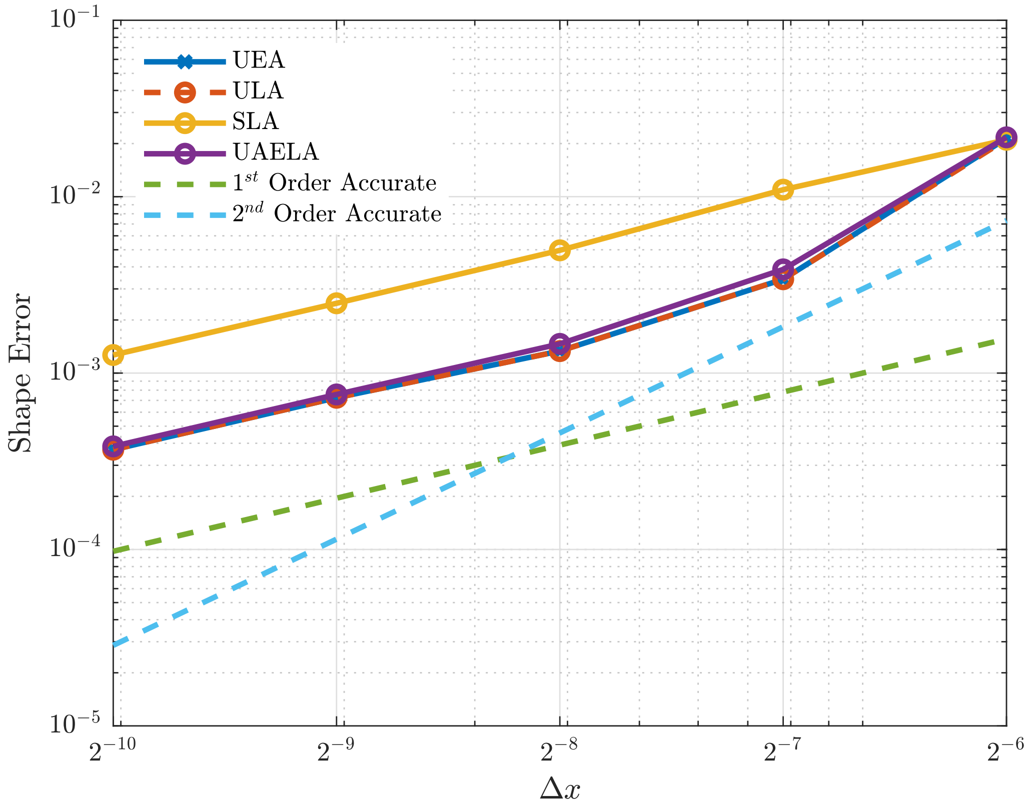

Figure 3.11: Convergence for FDM (COMM)

Advection (ULA) approaches produce practically the same result as seen in Table 3.6, Figure 3.11 and Figure 3.12.

ELVIRA

With ELVIRA, an order of accuracy of second is observed for the unsplit methods as seen in Table 3.7 and Figure 3.13. The Unsplit Lagrangian Advection (ULA) method again replicates the Unsplit Eulerian Advection (UEA) method. The Unsplit Averaged Eulerian Lagrangian Adevtion (UAELA) method lags in shape error reduction due to temporal location in the timestep. The Split Lagrangian Advection (SLA) retains first order accuracy. Figure 3.14 shows the final result of deformation complying to the initial condition as a function of resolution and geometric transport method

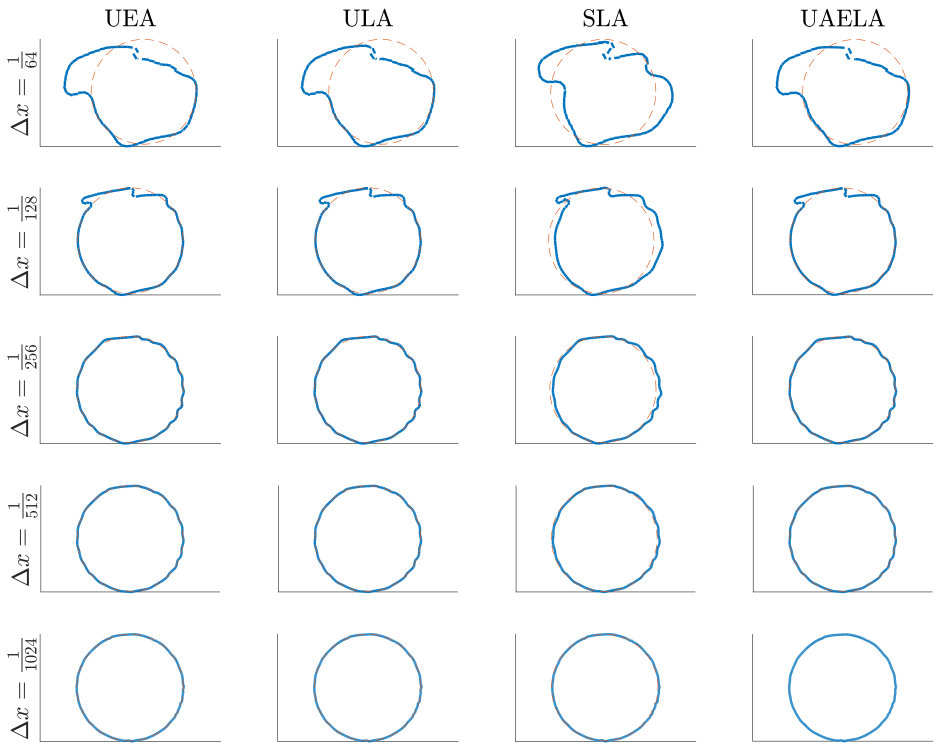

Figure 3.12: Results at t = 8 for FDM (COMM)

Table 3.7:
Shape Error for ELVIRA

Δx	**UEA**		**ULA**		**SLA**		**UAELA**	
$^1/_{64}$	1.08e-02	Ratio	1.09e-02	Ratio	1.98e-02	Ratio	1.28e-02	Ratio
$^1/_{128}$	2.12e-03	5.12	2.13e-03	5.11	9.84e-03	2.01	2.76e-03	4.64
$^1/_{256}$	4.16e-04	5.09	4.19e-04	5.08	4.78e-03	2.06	8.26e-04	3.34
$^1/_{512}$	8.35e-05	4.98	8.36e-05	5.01	2.38e-03	2.01	3.05e-04	2.70
$^1/_{1024}$	2.18e-05	3.82	2.19e-05	3.82	1.18e-03	2.02	1.37e-04	2.23

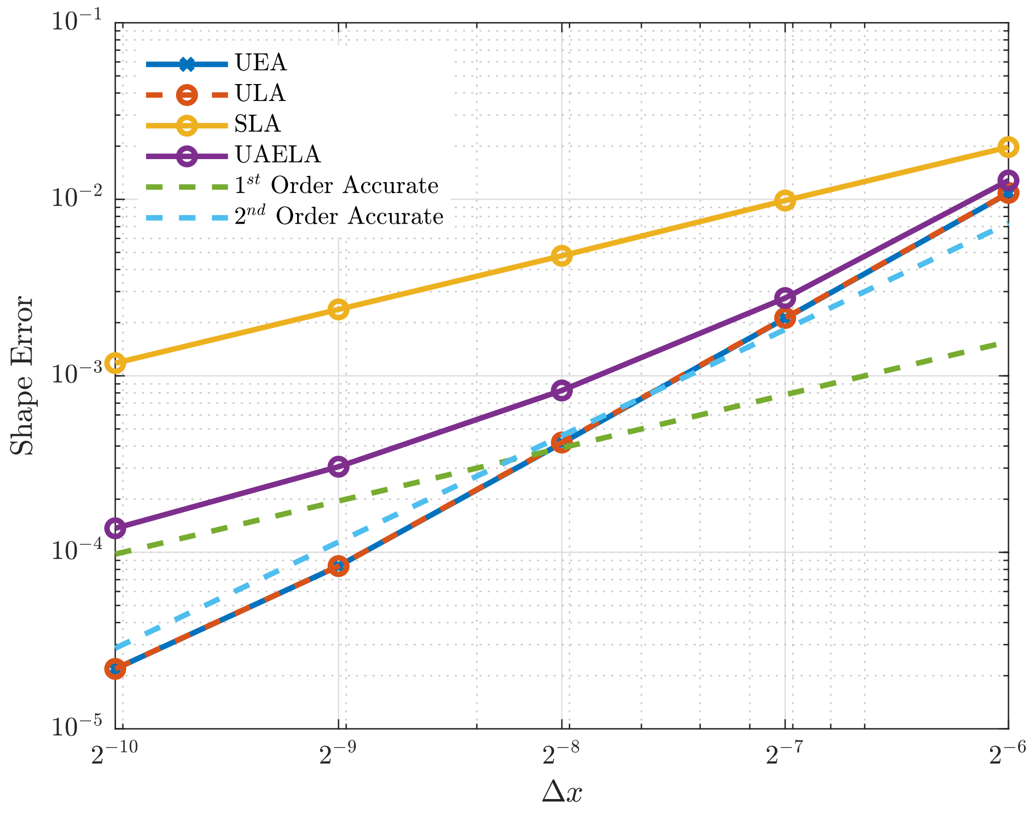

Figure 3.13: Convergence for ELVIRA

for PLIC method of ELVIRA.

LVIRA

The LVIRA method is marginally better at producing more or less the same results as ELVIRA for all advection methods as seen in Table 3.8 and Figure 3.15. Figure 3.16 shows the final result of transport mimicking the initial condition as a function of resolution and geometric transport method for PLIC method of LVIRA.

When velocity chosen for advecting $\psi(t)$ to $\psi(t + \Delta t)$ is at $\vec{u}(t + \frac{\Delta t}{2})$ instead of $\vec{u}(t)$ for Unsplit Averaged Eulerian Lagrangian Advection (UAELA), the rate of convergence of shape error ends up increasing to an order of two as shown in Figure 3.17 as opposed to one as shown in Figure 3.13.

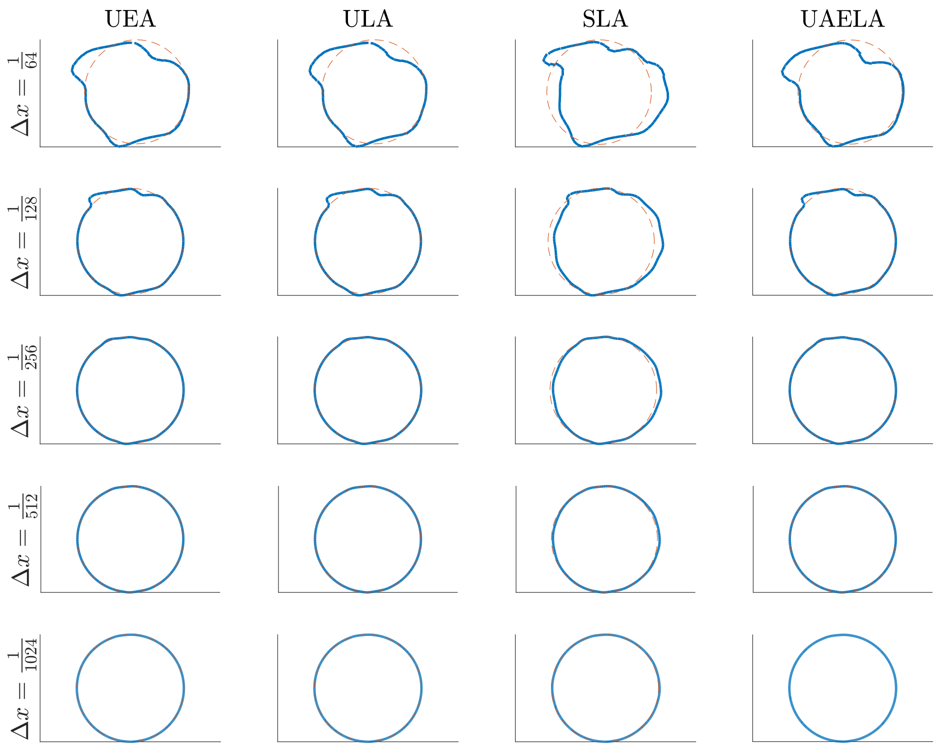

Figure 3.14: Results at t = 8 for ELVIRA

Table 3.8:
Shape Error for LVIRA

Δx	**UEA**		**ULA**		**SLA**		**UAELA**	
1/64	1.06e-02	**Ratio**	1.07e-02	**Ratio**	2.20e-02	**Ratio**	1.14e-02	**Ratio**
1/128	1.81e-03	5.85	1.81e-03	5.89	1.02e-02	2.16	2.53e-03	4.51
1/256	3.05e-04	5.95	3.05e-04	5.94	4.81e-03	2.12	7.18e-04	3.53
1/512	8.43e-05	3.62	8.44e-05	3.62	2.38e-03	2.02	3.06e-04	2.34
1/1024	2.22e-05	3.80	2.22e-05	3.80	1.18e-03	2.01	1.37e-04	2.24

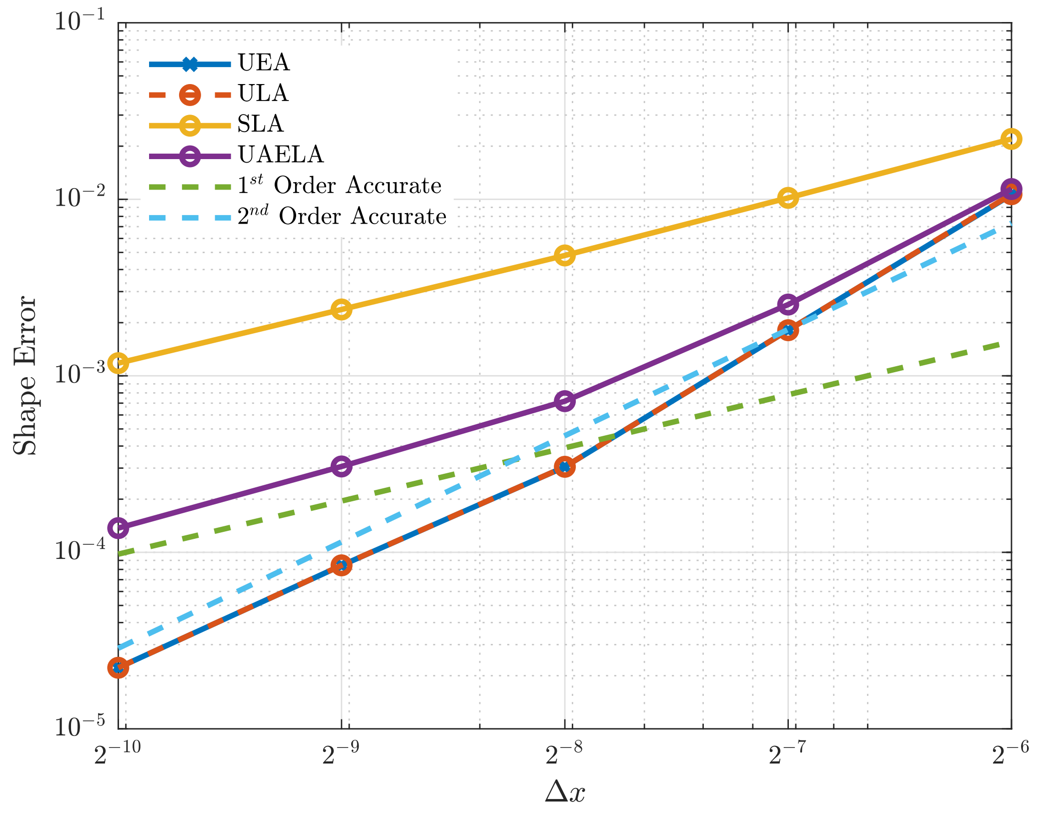

Figure 3.15: Convergence for LVIRA

3.2 Volume Error

The volume Error can be described as the loss or gain in the overall fluid volume and the complimentary gas volume as described in Equation 3.5.

$$\epsilon_{volume} = \Delta x \Delta y \left(\sum_{j=0}^{N} \sum_{i=0}^{M} \psi_{i,j}^{t=0} - \sum_{j=0}^{N} \sum_{i=0}^{M} \psi_{i,j}^{t=t_f} \right) \tag{3.5}$$

It is safe to assume that the magnitude of volume error would always be less than or equal to shape error.

3.2.1 Zalesak's Disk

For the all the unsplit methods tested, a volume error of zero is seen with only precision errors. The Split Lagrangian Advection (SLA) method does produce near-

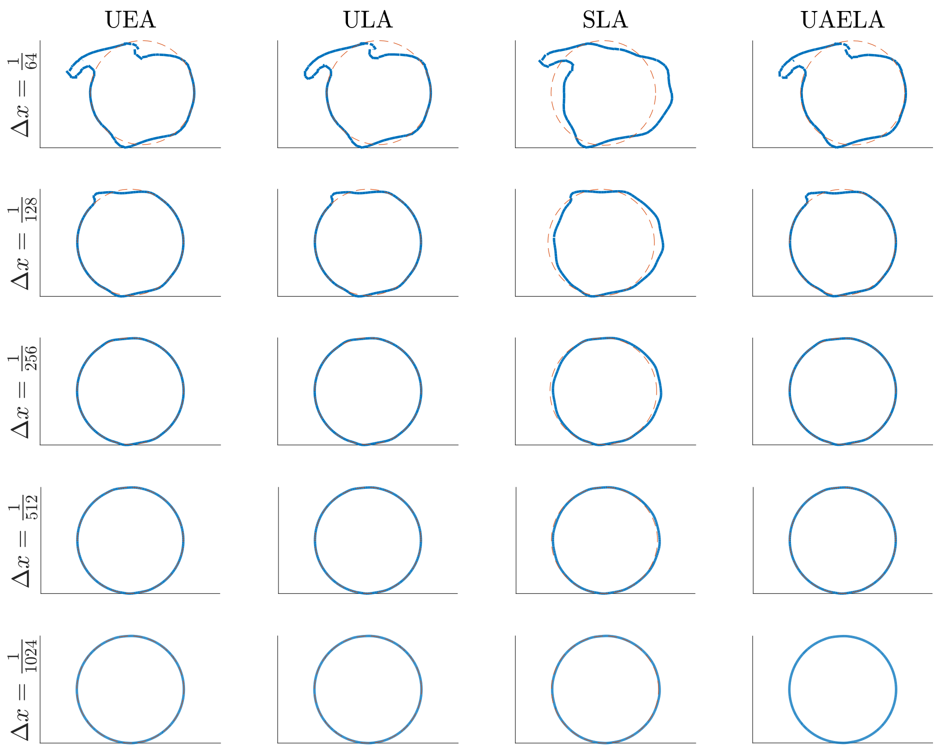

Figure 3.16: Results at t = 8 for LVIRA

precision errors as well in the case of Zalesak's Disk as seen in Figure 3.18. Although the nature of discrepancy across reconstruction methods is most likely due to Equation 2.2 failing when the normal is extremely close to a cardinal direction.

3.2.2 Deformation Field

In the deformation feild, again the unsplit methods produce precision error zeros as the shape error. The SLA method produces first order convergent Volume Error which is nearly equal to shape error itself. Figure 3.19 shows the volume error for all methods and resolution for the case of circular disk on a deformation field.

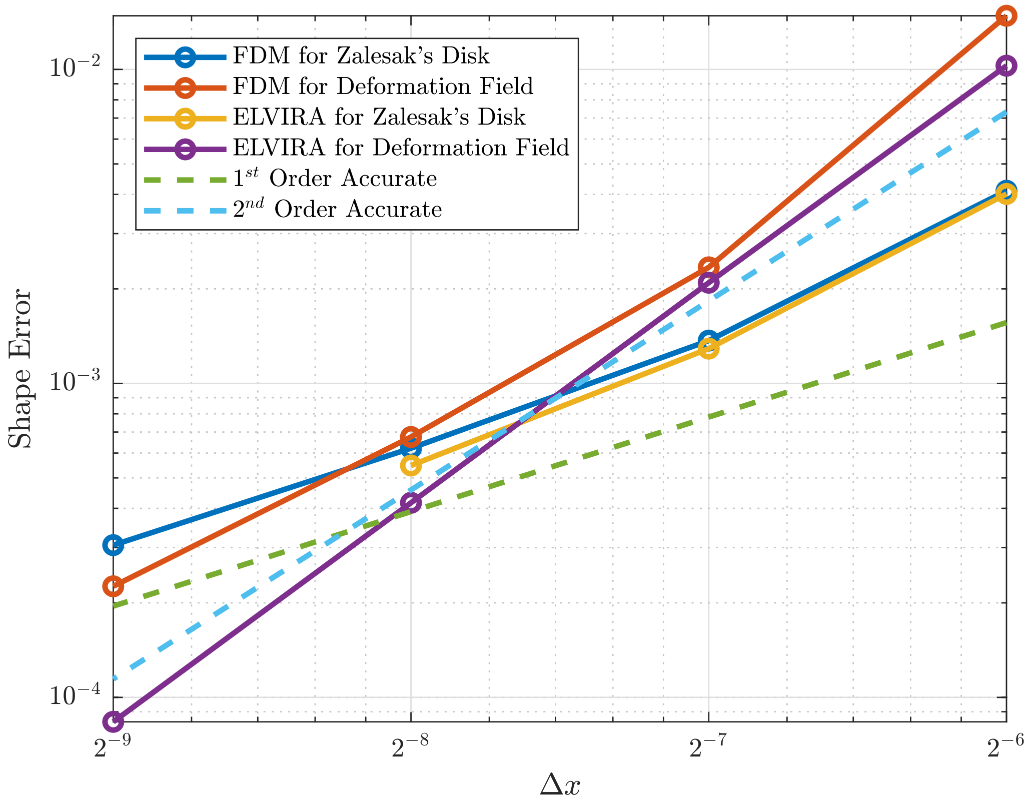

Figure 3.17: Shape Error Convergence for Unsplit Averaged Eulerian Lagrangian Advection (UAELA) with time-step centered velocity

3.2.3 Fluid Simulation

A simple setup was established whereby a circular drop akin to the one used in the deformation field example is used as the initial condition. The domain is scaled to a one centimeter scale. The density (ρ) of the liquid and gas is 1000 $\frac{kg}{m^3}$ and 100 $\frac{kg}{m^3}$. The viscosity of said phases are 8.9e-4 $\frac{N \cdot s}{m^2}$ and 1.81e-5 $\frac{N \cdot s}{m^2}$. A surface tension Coefficient of 0.07 $\frac{N}{m}$ was used. The gravity is established as 1 $\frac{m}{s^2}$. The PLIC method used for interface reconstruction was ELVIRA. The multi-phase's field's transient evolution was done using velocity ascertained from incompressible 2-phase Navier Stokes Partial Differential Equation coupled with the phase field advection as described in Harlow and Welch (1965) on a staggered grid. The variable density pressure poisson system

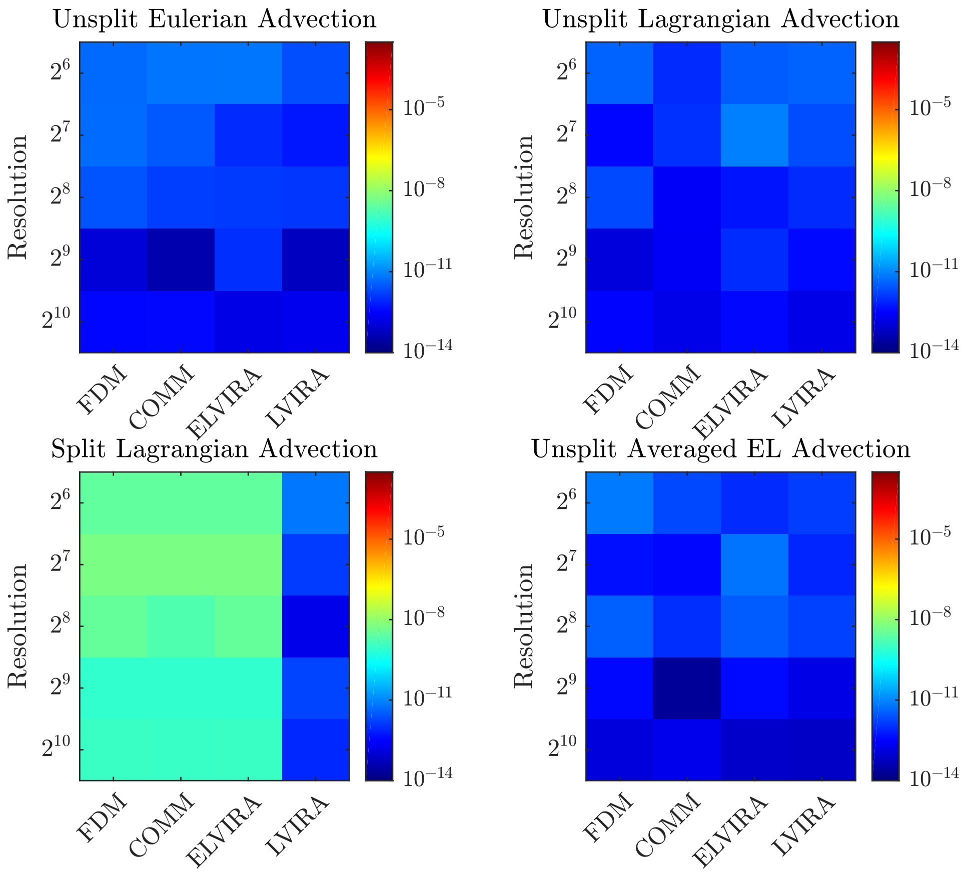

Figure 3.18: Volume Error at $t = 2\pi$ for Case Zalesak's Disk on Circular Velocity Field

was solved using a direct QR solver. The method used to compute curvature as part surface tension forces computation was computed by using height functions as described in a paper by Cummins *et al.* (2005). Two simulations were done with resolution of 64x64 and 128x128 cells respectively as seen in Figure 3.20. In Figure 3.20 the coarser mesh is above the finer mesh.

It was seen that the Split Lagrangian Advection (SLA) method does not conserve the total amount of fluid and hence the volume error is not constant. It is not constant for the split methods either, but the volume error changes by extremely small values which are at the edge of precision limit. There are tiny jumps in the volume error most likely due to cells that end up having so little volume fraction of

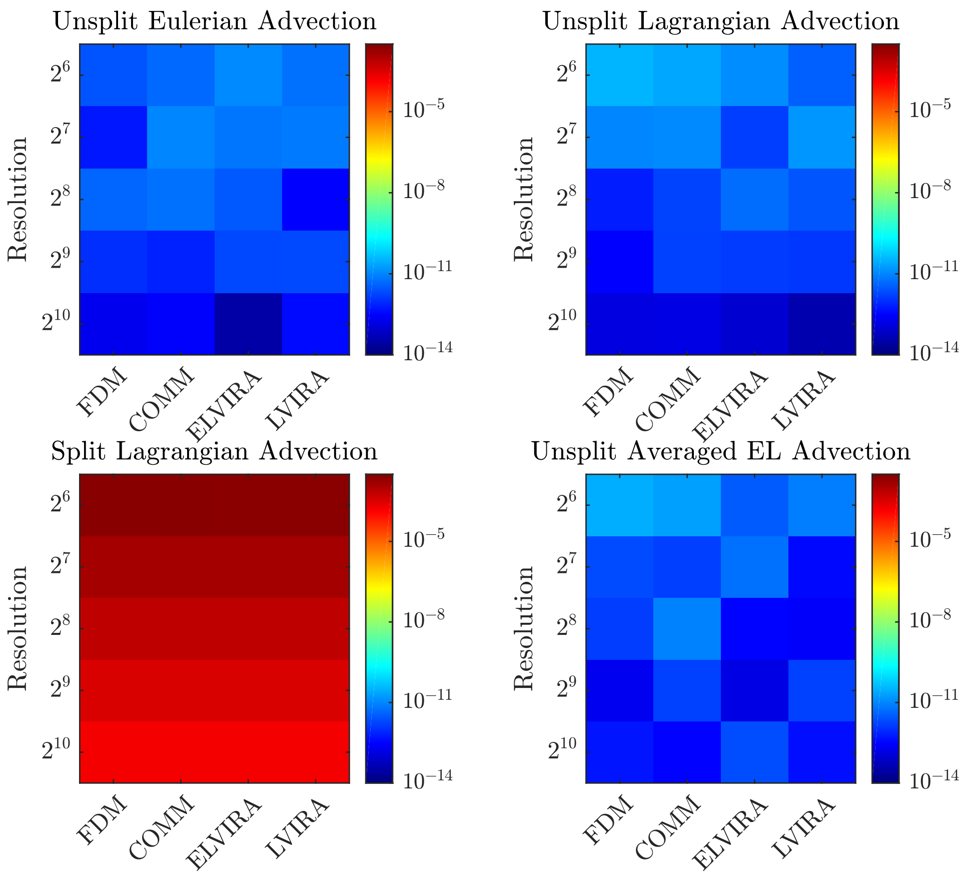

Figure 3.19: Volume Error at $t = 8$

liquid after advection that they get disregarded due to the small positive non-zero threshold required to establish and flag the presence of liquid in any specific cell.

3.3 Compute Times

The compute times for each advection method is timed for the Zalesak's Disk on a circular field and circular disk on a deformation field as shown in Figure 3.21. This timing was ascertained independently of the previous shape and volume errors. Specifically, for the first 10 times steps, the advection times only were summed up. As such we can see that deformation field case computes slightly faster than the Zalesak's disk case which makes sense since its perimeter is smaller. The main thing to see here

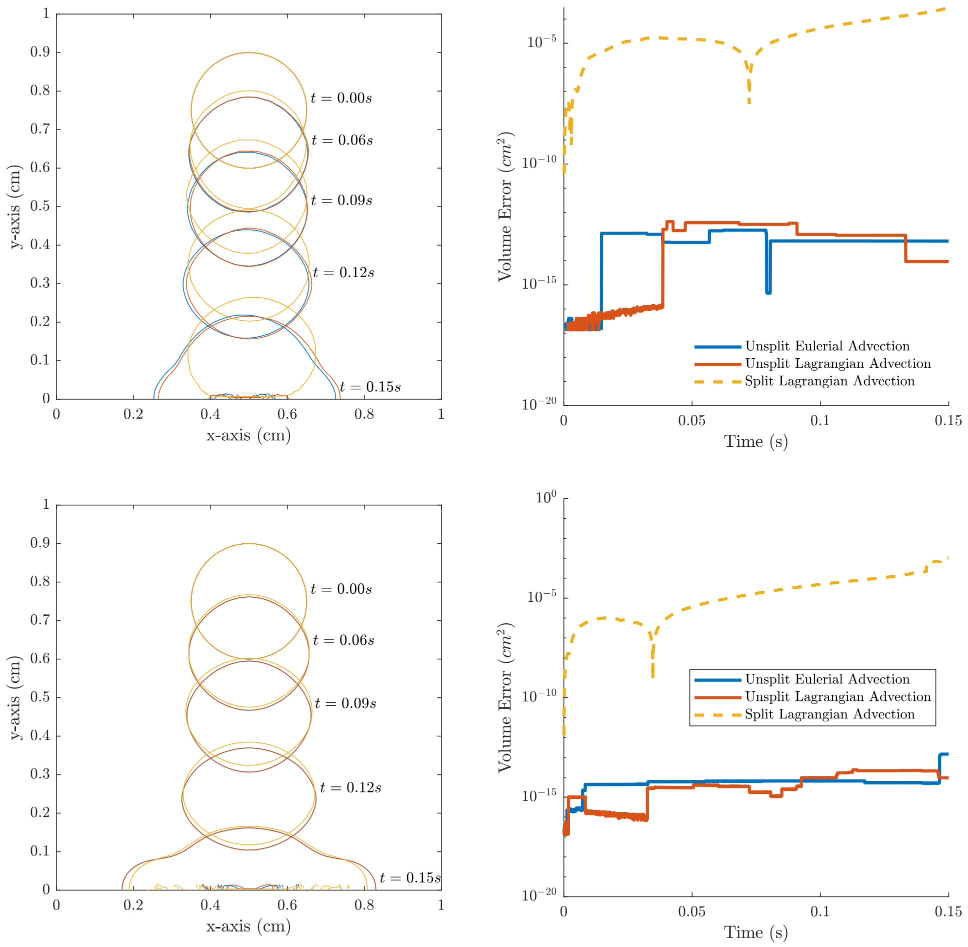

Figure 3.20: Volume Error on a 2D Fluid Simulation with M = 64 and 128 in order

is that the Unsplit Eulerian Advection (UEA) method performs slightly slower than the Unsplit Lagrangian Advection (ULA) Method. However, this does not mean anything partially because both approaches aren't optimized and hence it hard to say which one would be faster if optimized to its fullest extent. The initial method proposed by Owkes and Desjardins (2014) is fairly optimized and original version of Unsplit Eulerian Advection (UEA) method coded for this paper. Another factor that might impact computation time is the clipping method itself. The Sutherland-Hodgeman Clipping algorithm used is a general approach but it is not necessarily an optimal approach. Especially, in the Unsplit Lagrangian Advection (ULA) method,

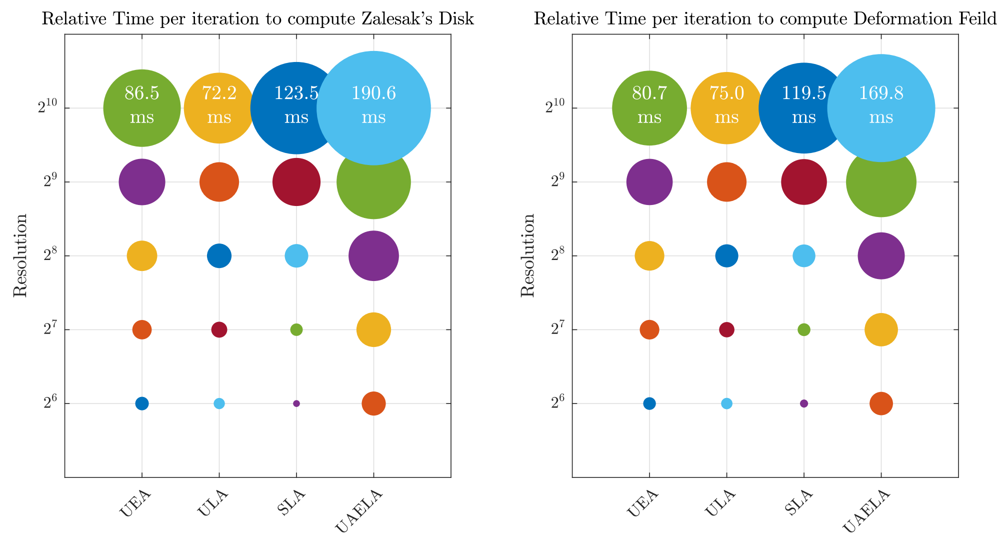

Figure 3.21: Compute Times for Advection Method at all Resolutions

one could use line-clipping algorithms which are optimized for rectanglular clippers to attain superior performance. At the same time, computing the Polygon TCF also takes a significant amount of time and that might count against this approach. The Unsplit Averaged Eulerian Lagrangian Adevtion (UAELA) method ends up being the slowest since it has to do a large number of intersections, about six times as many clippings as the Unsplit Lagrangian Advection (ULA) method. Despite that, it is only 3 times slower implying the construction of polygon TCF has a descent overhead. Typically the computational time of a transient solution of a hyperbolic PDE on Cartesian grids scale up by a factor of 8 when the resolution is doubled. However a factor of 5 was noticed in all unsplit methods. This is because, the code only operates near the interface, specifically interface cells and neighbours and neighbours of neighbours of interface cells. This would theoretically scale by a factor of 4 but due to unknown overheads it scales by a factor of 5.

Chapter 4

CONCLUSION

It is evident from the testing that all PLIC methods and geometric fluxing methods converge with varying orders of accuracy. Specifically, the order of convergence is first order for all cases except when the fluxing methods are unsplit and the reconstruction method is a variant of the sum of error squared minimization variety in which case a superior second order rate of convergence is achieved. It was established that the Unsplit Lagrangian Advection (ULA) method described in this paper is practically analogous to any conservative Unsplit Eulerian Advection (UEA) method. It was also demonstrated the ULA method can be combined with UEA method; the computational cost did render said geometrically averaged method impractical. The computational cost analysis of the fluxing methods is not conclusive due to lack of optimization. The Unsplit Lagrangian Advection (ULA) can allow for small divergence in the velocity field which would would typically come from incomplete solving of the variable density Poisson system to machine precision. Ultimately, that choice of advection method would be trade-off between accuracy and compute times.

REFERENCES

Comminal, R., J. Spangenberg and J. H. Hattel, “Cellwise conservative unsplit advection for the volume of fluid method”, Journal of Computational Physics **283**, 582 – 608, URL `http://www.sciencedirect.com/science/article/pii/S0021999114008109` (2015).

Cummins, S. J., M. M. Francois and D. B. Kothe, “Estimating curvature from volume fractions”, Computers and Structures **83**, 6, 425 – 434, URL `http://www.sciencedirect.com/science/article/pii/S0045794904004110`, frontier of Multi-Phase Flow Analysis and Fluid-Structure (2005).

Harlow, F. H. and J. E. Welch, “Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface”, The physics of fluids **8**, 12, 2182–2189 (1965).

Kernighan, B. W. and D. M. Ritchie, *The C Programming Language* (Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1978).

MATLAB, *version 9.4.0.813654 (R2018a)* (The MathWorks Inc., Natick, Massachusetts, 2018).

Nelder, J. A. and R. Mead, “A Simplex Method for Function Minimization”, The Computer Journal **7**, 4, 308–313, URL `https://dx.doi.org/10.1093/comjnl/7.4.308` (1965).

Owkes, M. and O. Desjardins, “A computational framework for conservative, three-dimensional, unsplit, geometric transport with application to the volume-of-fluid (vof) method”, Journal of Computational Physics **270**, 587–612 (2014).

Parker, B. and D. Youngs, *Two and three dimensional Eulerian simulation of fluid flow with material interfaces* (Atomic Weapons Establishment, 1992).

Pilliod Jr, J. E. and E. G. Puckett, “Second-order accurate volume-of-fluid algorithms for tracking material interfaces”, Journal of Computational Physics **199**, 2, 465–502 (2004).

Scardovelli, R. and S. Zaleski, “Interface reconstruction with least-square fit and split eulerianlagrangian advection”, International Journal for Numerical Methods in Fluids **41**, 3, 251–274, URL `https://www.onlinelibrary.wiley.com/doi/abs/10.1002/fld.431` (2003).

Sussman, M., A. S. Almgren, J. B. Bell, P. Colella, L. H. Howell and M. L. Welcome, “An adaptive level set approach for incompressible two-phase flows”, Journal of Computational Physics **148**, 1, 81–124 (1999).

Sutherland, I. E. and G. W. Hodgman, “Reentrant polygon clipping”, Commun. ACM **17**, 1, 32–42, URL `http://doi.acm.org/10.1145/360767.360802` (1974).

Yang, W., S. hong Liu and Y. lin Wu, "An unsplit lagrangian advection scheme for volume of fluid method", Journal of Hydrodynamics, Ser. B **22**, 1, 73 – 80, URL `http://www.sciencedirect.com/science/article/pii/S1001605809600305` (2010).

APPENDIX A

RAW DATA

Table A.1 contains all the data ascertained from 2 cases, 4 PLIC Methods for each case, 4 Transport Methods for PLIC Method and 5 Resolutions for each Transport Method.

Table A.1: Shape Error and Volume Error Raw Data

Case	PLIC Method	Transport Method	Size	Shape Error	Volume Error
Zalesak's Disk	FDM	UEA	64	4.099E-03	4.262E-12
Zalesak's Disk	FDM	UEA	128	1.368E-03	4.359E-12
Zalesak's Disk	FDM	UEA	256	6.215E-04	2.466E-12
Zalesak's Disk	FDM	UEA	512	3.054E-04	1.006E-13
Zalesak's Disk	FDM	UEA	1024	1.308E-04	3.159E-13
Zalesak's Disk	FDM	UEA	64	5.453E-03	5.771E-12
Zalesak's Disk	FDM	UEA	128	2.353E-03	2.752E-12
Zalesak's Disk	FDM	UEA	256	1.127E-03	1.371E-12
Zalesak's Disk	FDM	UEA	512	6.570E-04	3.404E-14
Zalesak's Disk	FDM	UEA	1024	3.789E-04	3.205E-13
Zalesak's Disk	FDM	UEA	64	4.005E-03	5.812E-12
Zalesak's Disk	FDM	UEA	128	1.289E-03	8.116E-13
Zalesak's Disk	FDM	UEA	256	5.473E-04	1.220E-12
Zalesak's Disk	FDM	UEA	512	2.509E-04	9.027E-13
Zalesak's Disk	FDM	UEA	1024	9.860E-05	1.398E-13
Zalesak's Disk	FDM	UEA	64	4.147E-03	2.069E-12
Zalesak's Disk	FDM	UEA	128	1.313E-03	4.916E-13
Zalesak's Disk	FDM	UEA	256	5.588E-04	1.118E-12
Zalesak's Disk	FDM	UEA	512	2.572E-04	5.356E-14
Zalesak's Disk	FDM	UEA	1024	1.011E-04	1.768E-13
Zalesak's Disk	FDM (COMM)	ULA	64	4.113E-03	3.510E-12
Zalesak's Disk	FDM (COMM)	ULA	128	1.370E-03	3.129E-13
Zalesak's Disk	FDM (COMM)	ULA	256	6.216E-04	1.849E-12
Zalesak's Disk	FDM (COMM)	ULA	512	3.054E-04	1.128E-13
Zalesak's Disk	FDM (COMM)	ULA	1024	1.308E-04	3.084E-13
Zalesak's Disk	FDM (COMM)	ULA	64	5.458E-03	8.151E-13
Zalesak's Disk	FDM (COMM)	ULA	128	2.354E-03	9.497E-13
Zalesak's Disk	FDM (COMM)	ULA	256	1.128E-03	2.246E-13
Zalesak's Disk	FDM (COMM)	ULA	512	6.570E-04	2.116E-13
Zalesak's Disk	FDM (COMM)	ULA	1024	3.789E-04	1.541E-13
Zalesak's Disk	FDM (COMM)	ULA	64	4.017E-03	2.993E-12
Zalesak's Disk	FDM (COMM)	ULA	128	1.292E-03	7.415E-12
Zalesak's Disk	FDM (COMM)	ULA	256	5.479E-04	4.500E-13
Zalesak's Disk	FDM (COMM)	ULA	512	2.510E-04	8.472E-13
Zalesak's Disk	FDM (COMM)	ULA	1024	9.864E-05	3.273E-13
Zalesak's Disk	FDM (COMM)	ULA	64	4.157E-03	3.455E-12
Zalesak's Disk	FDM (COMM)	ULA	128	1.316E-03	2.024E-12
Zalesak's Disk	FDM (COMM)	ULA	256	5.594E-04	8.164E-13
Zalesak's Disk	FDM (COMM)	ULA	512	2.573E-04	3.368E-13
Zalesak's Disk	FDM (COMM)	ULA	1024	1.011E-04	1.497E-13

Zalesak's Disk	ELVIRA	SLA	64	4.231E-03	2.568E-09
Zalesak's Disk	ELVIRA	SLA	128	1.405E-03	4.844E-09
Zalesak's Disk	ELVIRA	SLA	256	6.319E-04	2.791E-09
Zalesak's Disk	ELVIRA	SLA	512	3.128E-04	7.321E-10
Zalesak's Disk	ELVIRA	SLA	1024	1.342E-04	9.142E-10
Zalesak's Disk	ELVIRA	SLA	64	5.656E-03	2.568E-09
Zalesak's Disk	ELVIRA	SLA	128	2.426E-03	4.841E-09
Zalesak's Disk	ELVIRA	SLA	256	1.172E-03	1.547E-09
Zalesak's Disk	ELVIRA	SLA	512	6.793E-04	7.341E-10
Zalesak's Disk	ELVIRA	SLA	1024	4.004E-04	8.778E-10
Zalesak's Disk	ELVIRA	SLA	64	4.119E-03	2.580E-09
Zalesak's Disk	ELVIRA	SLA	128	1.320E-03	4.842E-09
Zalesak's Disk	ELVIRA	SLA	256	5.555E-04	2.792E-09
Zalesak's Disk	ELVIRA	SLA	512	2.559E-04	7.331E-10
Zalesak's Disk	ELVIRA	SLA	1024	1.006E-04	9.148E-10
Zalesak's Disk	ELVIRA	SLA	64	4.249E-03	6.064E-12
Zalesak's Disk	ELVIRA	SLA	128	1.346E-03	1.260E-12
Zalesak's Disk	ELVIRA	SLA	256	5.665E-04	1.553E-13
Zalesak's Disk	ELVIRA	SLA	512	2.622E-04	1.598E-12
Zalesak's Disk	ELVIRA	SLA	1024	1.030E-04	7.521E-13
Zalesak's Disk	LVIRA	UAELA	64	4.107E-03	6.600E-12
Zalesak's Disk	LVIRA	UAELA	128	1.369E-03	3.971E-13
Zalesak's Disk	LVIRA	UAELA	256	6.215E-04	3.279E-12
Zalesak's Disk	LVIRA	UAELA	512	3.054E-04	3.308E-13
Zalesak's Disk	LVIRA	UAELA	1024	1.308E-04	1.125E-13
Zalesak's Disk	LVIRA	UAELA	64	5.456E-03	1.769E-12
Zalesak's Disk	LVIRA	UAELA	128	2.354E-03	3.198E-13
Zalesak's Disk	LVIRA	UAELA	256	1.128E-03	9.143E-13
Zalesak's Disk	LVIRA	UAELA	512	6.570E-04	1.874E-14
Zalesak's Disk	LVIRA	UAELA	1024	3.789E-04	1.691E-13
Zalesak's Disk	LVIRA	UAELA	64	4.013E-03	8.164E-13
Zalesak's Disk	LVIRA	UAELA	128	1.291E-03	5.501E-12
Zalesak's Disk	LVIRA	UAELA	256	5.477E-04	2.967E-12
Zalesak's Disk	LVIRA	UAELA	512	2.509E-04	3.504E-13
Zalesak's Disk	LVIRA	UAELA	1024	9.862E-05	6.949E-14
Zalesak's Disk	LVIRA	UAELA	64	4.154E-03	1.355E-12
Zalesak's Disk	LVIRA	UAELA	128	1.315E-03	7.203E-13
Zalesak's Disk	LVIRA	UAELA	256	5.592E-04	1.445E-12
Zalesak's Disk	LVIRA	UAELA	512	2.573E-04	1.448E-13
Zalesak's Disk	LVIRA	UAELA	1024	1.011E-04	6.409E-14
Deformation Field	FDM	UEA	64	1.490E-02	2.421E-12
Deformation Field	FDM	UEA	128	2.340E-03	4.848E-13
Deformation Field	FDM	UEA	256	6.756E-04	3.882E-12
Deformation Field	FDM	UEA	512	2.258E-04	9.146E-13
Deformation Field	FDM	UEA	1024	9.449E-05	1.738E-13
Deformation Field	FDM	UEA	64	2.140E-02	4.139E-12
Deformation Field	FDM	UEA	128	3.402E-03	8.854E-12

Deformation Field	FDM	UEA	256	1.334E-03	5.130E-12
Deformation Field	FDM	UEA	512	7.247E-04	6.526E-13
Deformation Field	FDM	UEA	1024	3.679E-04	2.543E-13
Deformation Field	FDM	UEA	64	1.084E-02	9.948E-12
Deformation Field	FDM	UEA	128	2.117E-03	5.928E-12
Deformation Field	FDM	UEA	256	4.160E-04	2.713E-12
Deformation Field	FDM	UEA	512	8.354E-05	1.787E-12
Deformation Field	FDM	UEA	1024	2.184E-05	2.698E-14
Deformation Field	FDM	UEA	64	1.061E-02	5.134E-12
Deformation Field	FDM	UEA	128	1.815E-03	6.636E-12
Deformation Field	FDM	UEA	256	3.049E-04	2.794E-13
Deformation Field	FDM	UEA	512	8.429E-05	1.844E-12
Deformation Field	FDM	UEA	1024	2.218E-05	3.509E-13
Deformation Field	FDM (COMM)	ULA	64	1.467E-02	3.001E-11
Deformation Field	FDM (COMM)	ULA	128	2.346E-03	8.864E-12
Deformation Field	FDM (COMM)	ULA	256	6.754E-04	5.788E-13
Deformation Field	FDM (COMM)	ULA	512	2.258E-04	2.784E-13
Deformation Field	FDM (COMM)	ULA	1024	9.449E-05	1.211E-13
Deformation Field	FDM (COMM)	ULA	64	2.114E-02	2.103E-11
Deformation Field	FDM (COMM)	ULA	128	3.404E-03	9.846E-12
Deformation Field	FDM (COMM)	ULA	256	1.332E-03	1.573E-12
Deformation Field	FDM (COMM)	ULA	512	7.245E-04	1.491E-12
Deformation Field	FDM (COMM)	ULA	1024	3.678E-04	1.388E-13
Deformation Field	FDM (COMM)	ULA	64	1.087E-02	1.074E-11
Deformation Field	FDM (COMM)	ULA	128	2.126E-03	1.361E-12
Deformation Field	FDM (COMM)	ULA	256	4.189E-04	4.671E-12
Deformation Field	FDM (COMM)	ULA	512	8.358E-05	1.278E-12
Deformation Field	FDM (COMM)	ULA	1024	2.186E-05	8.220E-14
Deformation Field	FDM (COMM)	ULA	64	1.068E-02	3.226E-12
Deformation Field	FDM (COMM)	ULA	128	1.813E-03	1.357E-11
Deformation Field	FDM (COMM)	ULA	256	3.052E-04	2.507E-12
Deformation Field	FDM (COMM)	ULA	512	8.438E-05	1.157E-12
Deformation Field	FDM (COMM)	ULA	1024	2.218E-05	3.415E-14
Deformation Field	ELVIRA	SLA	64	1.990E-02	2.425E-03
Deformation Field	ELVIRA	SLA	128	1.011E-02	1.189E-03
Deformation Field	ELVIRA	SLA	256	4.800E-03	6.016E-04
Deformation Field	ELVIRA	SLA	512	2.367E-03	3.029E-04
Deformation Field	ELVIRA	SLA	1024	1.182E-03	1.520E-04
Deformation Field	ELVIRA	SLA	64	2.101E-02	2.446E-03
Deformation Field	ELVIRA	SLA	128	1.095E-02	1.191E-03
Deformation Field	ELVIRA	SLA	256	4.964E-03	6.017E-04
Deformation Field	ELVIRA	SLA	512	2.487E-03	3.029E-04
Deformation Field	ELVIRA	SLA	1024	1.266E-03	1.520E-04
Deformation Field	ELVIRA	SLA	64	1.976E-02	2.364E-03
Deformation Field	ELVIRA	SLA	128	9.838E-03	1.189E-03
Deformation Field	ELVIRA	SLA	256	4.784E-03	6.016E-04
Deformation Field	ELVIRA	SLA	512	2.378E-03	3.029E-04

Deformation Field	ELVIRA	SLA	1024	1.180E-03	1.520E-04
Deformation Field	ELVIRA	SLA	64	2.198E-02	2.357E-03
Deformation Field	ELVIRA	SLA	128	1.019E-02	1.188E-03
Deformation Field	ELVIRA	SLA	256	4.806E-03	6.015E-04
Deformation Field	ELVIRA	SLA	512	2.376E-03	3.029E-04
Deformation Field	ELVIRA	SLA	1024	1.179E-03	1.520E-04
Deformation Field	LVIRA	UAELA	64	1.512E-02	2.499E-11
Deformation Field	LVIRA	UAELA	128	2.938E-03	1.932E-12
Deformation Field	LVIRA	UAELA	256	9.439E-04	1.332E-12
Deformation Field	LVIRA	UAELA	512	3.382E-04	1.861E-13
Deformation Field	LVIRA	UAELA	1024	1.496E-04	4.663E-13
Deformation Field	LVIRA	UAELA	64	2.170E-02	1.755E-11
Deformation Field	LVIRA	UAELA	128	3.871E-03	1.422E-12
Deformation Field	LVIRA	UAELA	256	1.464E-03	8.065E-12
Deformation Field	LVIRA	UAELA	512	7.564E-04	1.489E-12
Deformation Field	LVIRA	UAELA	1024	3.841E-04	2.879E-13
Deformation Field	LVIRA	UAELA	64	1.279E-02	2.924E-12
Deformation Field	LVIRA	UAELA	128	2.757E-03	5.043E-12
Deformation Field	LVIRA	UAELA	256	8.258E-04	2.929E-13
Deformation Field	LVIRA	UAELA	512	3.055E-04	1.329E-13
Deformation Field	LVIRA	UAELA	1024	1.368E-04	2.034E-12
Deformation Field	LVIRA	UAELA	64	1.143E-02	7.302E-12
Deformation Field	LVIRA	UAELA	128	2.533E-03	3.302E-13
Deformation Field	LVIRA	UAELA	256	7.183E-04	2.410E-13
Deformation Field	LVIRA	UAELA	512	3.065E-04	1.476E-12
Deformation Field	LVIRA	UAELA	1024	1.371E-04	3.856E-13

Note: This simulation was coded on MATLAB (2018) as an entry point but most of the methods discussed in this thesis were subroutined to a machine level using C by Kernighan and Ritchie (1978). The code is available on Github at https://github.com/aansari2/Master-s-Thesis.git.

APPENDIX B

MAXIMUM DEFORMATION PLOTS

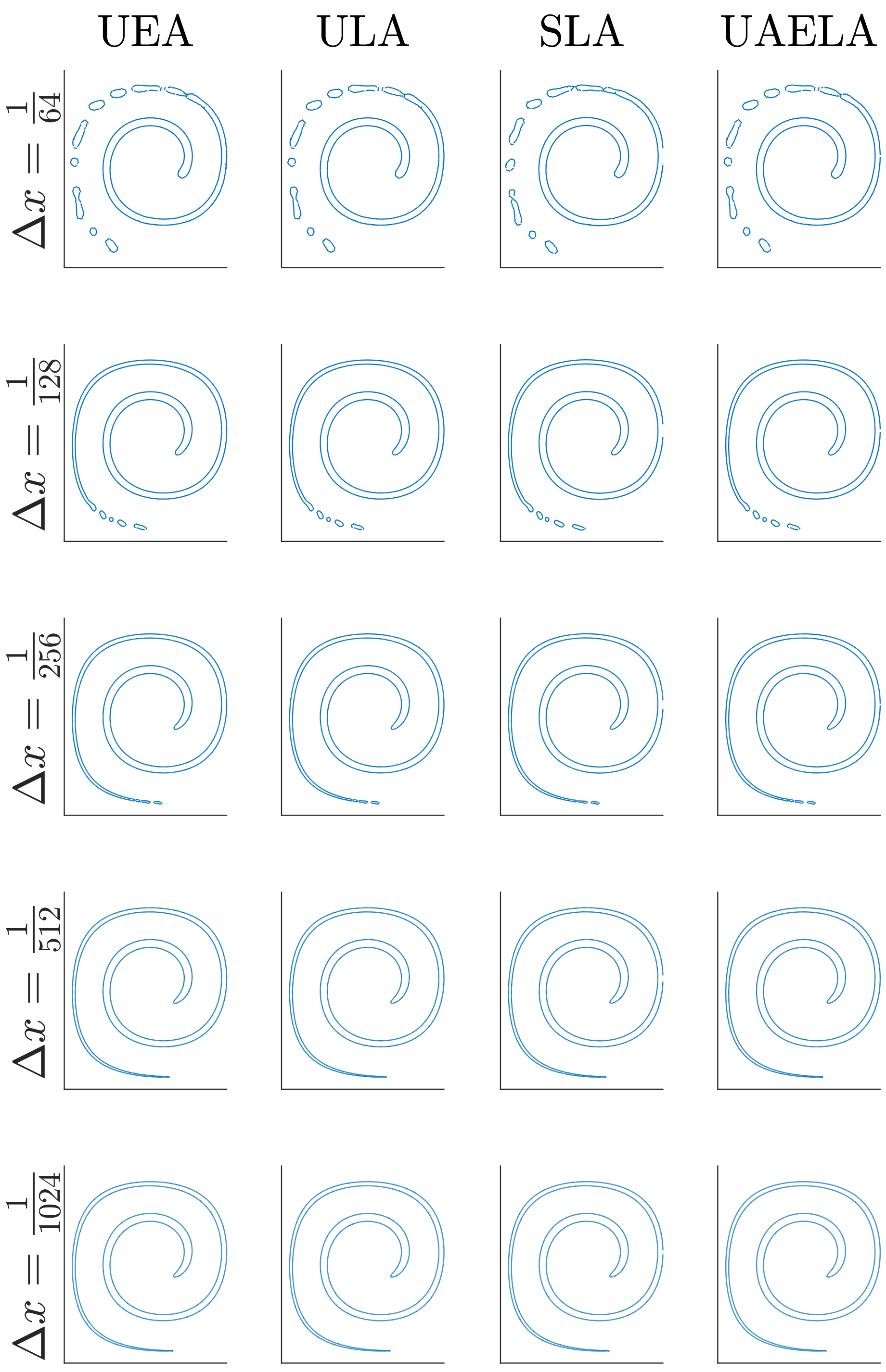

Figure B.1: Maximum Deformation for Deformation Field for PLIC Method FDM

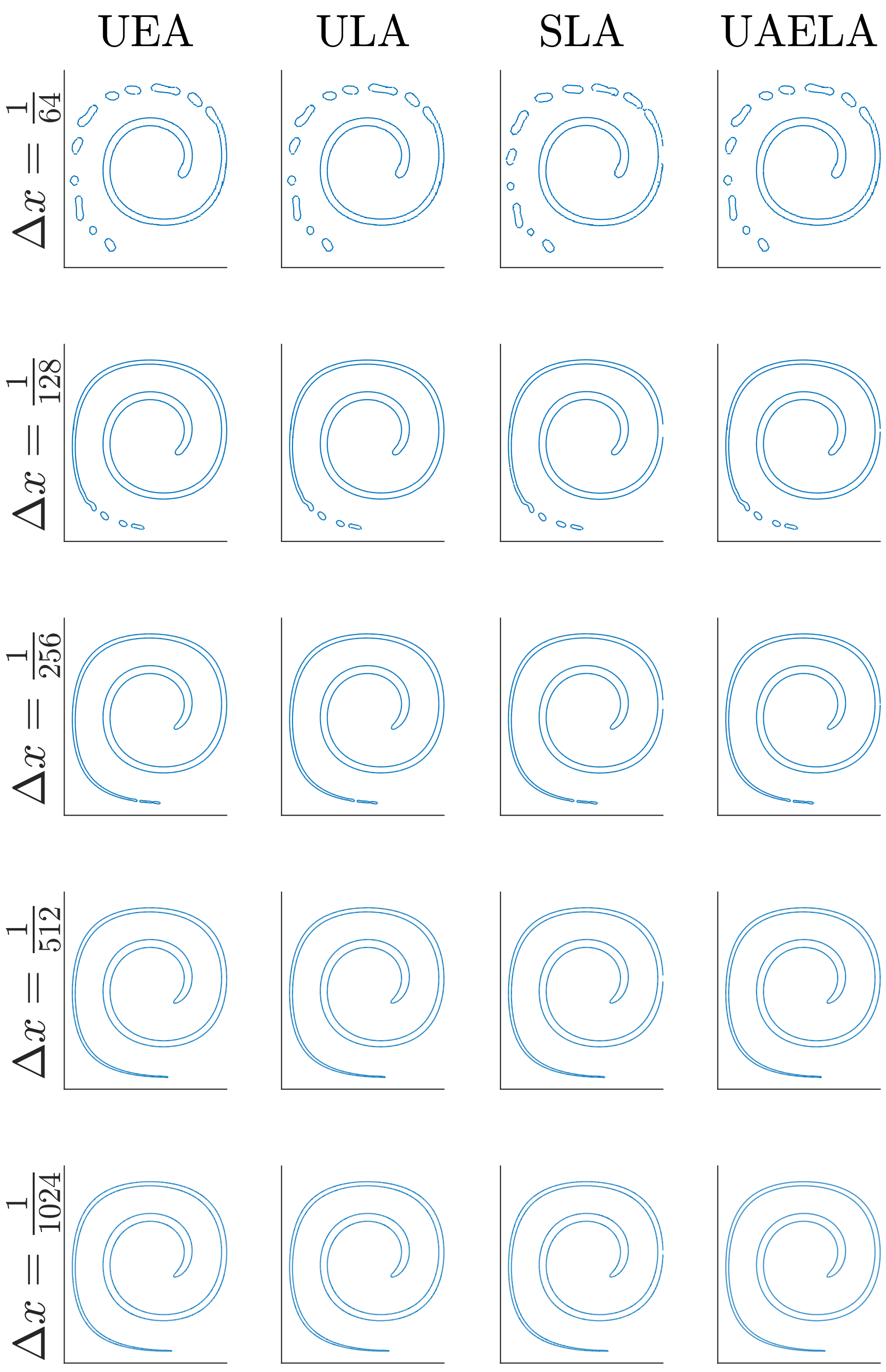

Figure B.2: Maximum Deformation for Deformation Field for PLIC Method FDM (COMM)

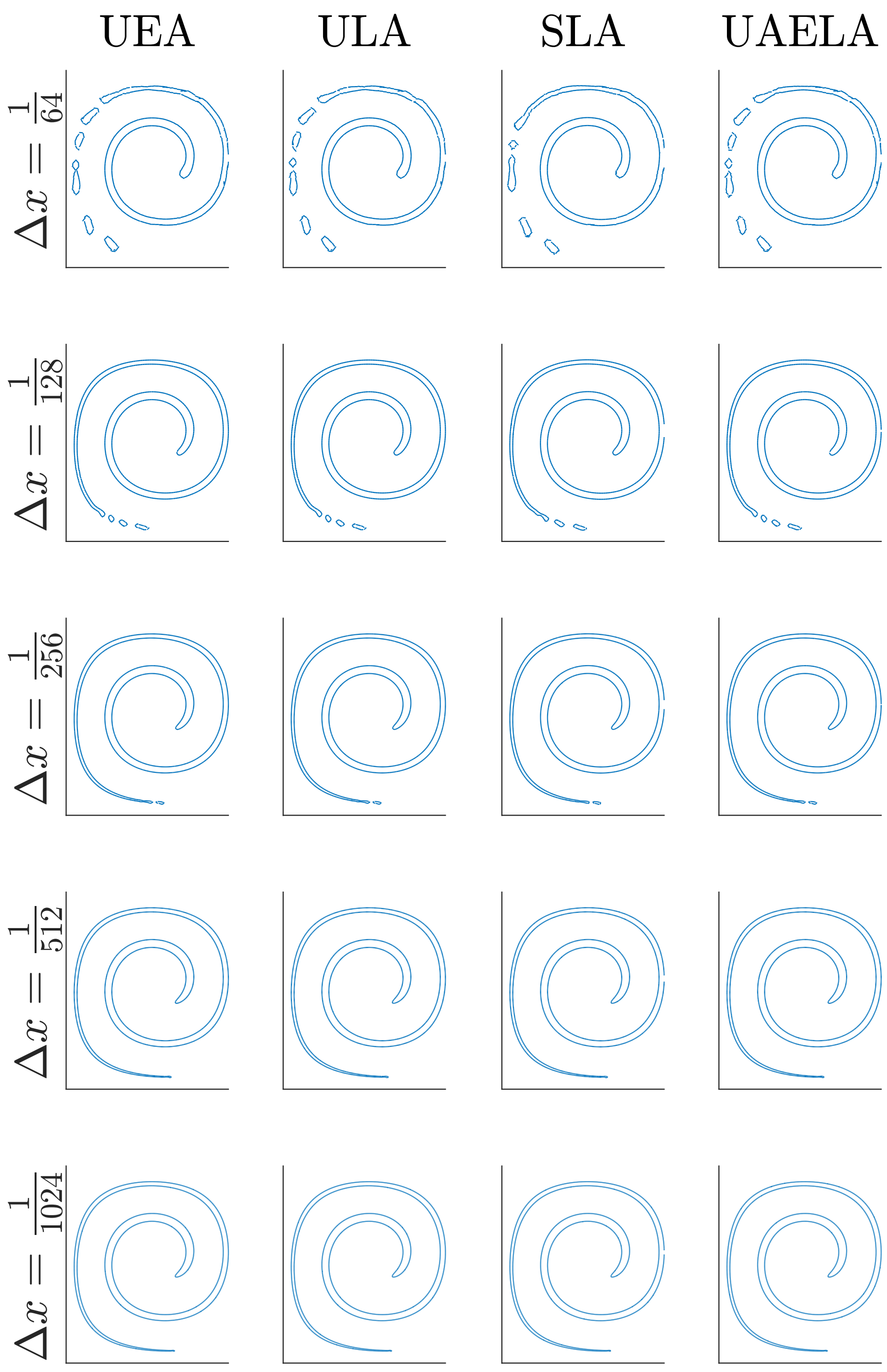

Figure B.3: Maximum Deformation for Deformation Field for PLIC Method ELVIRA

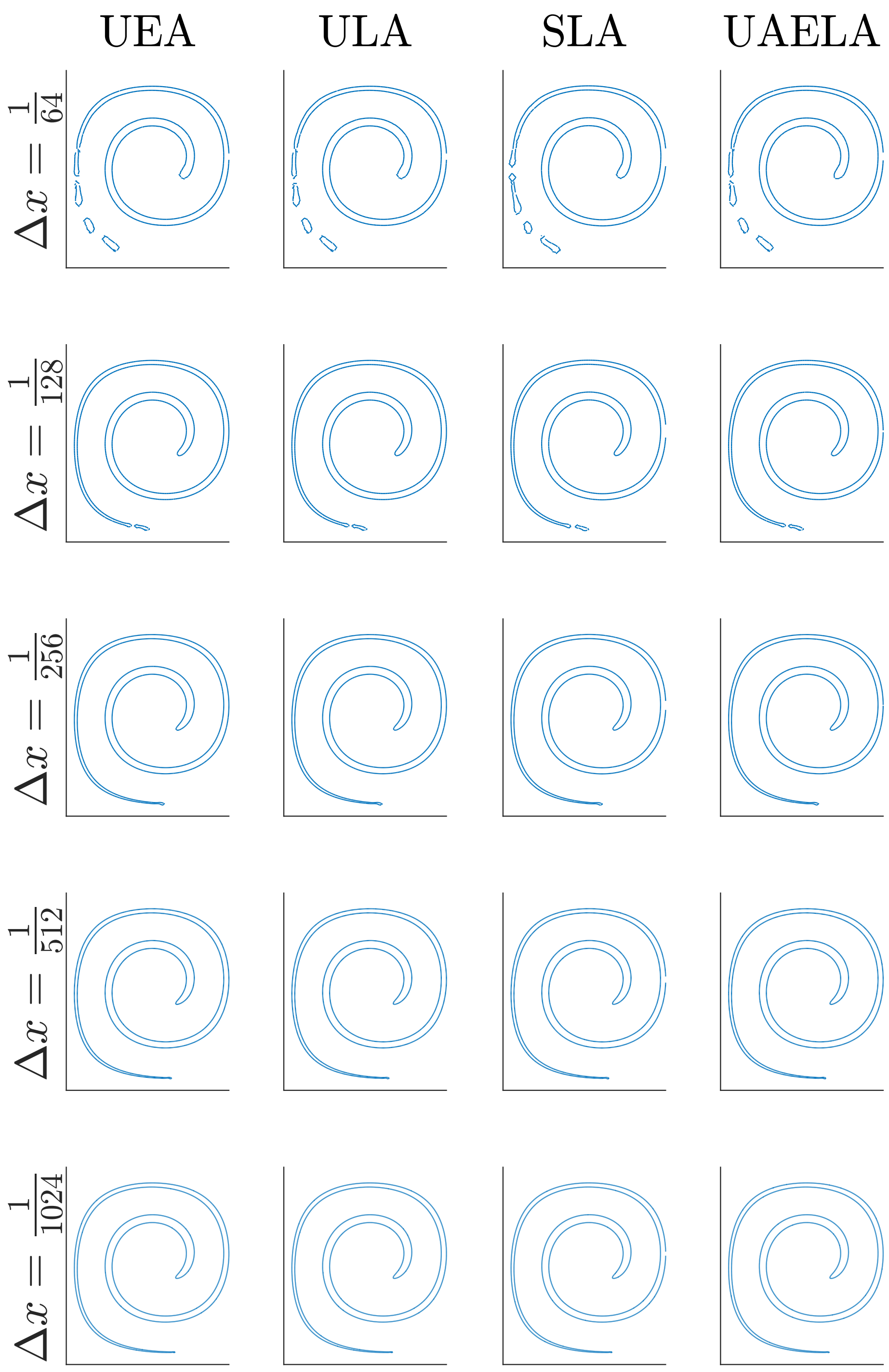

Figure B.4: Maximum Deformation for Deformation Field for PLIC Method LVIRA