LA-UR-07-1537

Approved for public release; distribution is unlimited.

Title:

Moment-of-fluid interface reconstruction

Author(s):

Vadim Dyadechko `vdyadechko@lanl.gov`

Mikhail Shashkov `shashkov@lanl.gov`

Mathematical Modeling and Analysis Group, T-7

Los Alamos National Laboratoty

Submitted to: Journal of Computational Physics

Moment-of-fluid interface reconstruction

Vadim Dyadechko[*][†] Mikhail Shashkov[†]

Abstract

We present a new volume-conservative interface reconstruction method, offering several major advantages over the traditional Volume-of-Fluid (VoF) methods. The key feature of the new Moment-of-Fluid (MoF) method is utilization of the cell-wise material centroids for the interface reconstruction. The location of the linear interface in each mixed cell is chosen to preserve the volumes and provide the best possible approximation to the material centroids. The MoF construction of the linear interface in a mixed cell depends only on the moment data from within the cell and not on the data from its neighbors. Therefore, the MoF method is able to resolve interface details as small as the cell itself, which are 2-3 times smaller than conventional VoF methods can resolve. Also, the MoF interface reconstruction can be implemented as a cell-by-cell black-box routine, which is a great technological advantage over the VoF, especially in 3D. The technique proposed is 2nd-order accurate and is shown to be more accurate than similar VoF methods. Since the centroid of any Lagrangian volume of fluid moves very much like a Lagrangian particle, the cell-wise material centroids can be updated in hydro simulations with sufficient accuracy.

1 Introduction

Accurate tracking of material interfaces is an important aspect of multi-material hydrodynamic simulations. Methods that move the mesh with the flow (Lagrangian) automatically maintain interfaces. Unfortunately, the moving mesh may become prohibitively distorted or tangled. Methods that keep mesh fixed and let the material flow through it (Eulerian), on the other hand, can easily

[*]corresponding author; e-mail: `vdyadechko@lanl.gov`
[†]Mathematical Modeling and Analysis Group (T-7) at Los Alamos National Laboratory

handle large deformations. However, the Eulerian methods require an explicit procedure to track material interfaces.

A variety of discrete interface models has been developed for multi-material (multi-phase) Eulerian simulations.

- In *level set* methods [34, 33, 24] the interface is represented implicitly by the zero level set of the discrete signed distance function, which is a constitutive part of the system of governing equations.
- *Front tracking* methods [37, 11, 36] use a supplementary Lagrangian surface grid on the interface between different phases.
- *Interface reconstruction* methods calculate the interface location at each time step from the solution data; the most popular among them are *volume-of-fluid (VoF)* methods [13, 39, 29, 6], which construct the interface from the volume fractions of different phases in each cell.

Each approach has its virtues and drawbacks. What really distinguishes VoF methods from the other techniques is simple treatment of the interface topology changes and rigorous enforcement of mass conservation of each fluid component. For a wide range of applications, the discrete mass conservation is a decisive argument for VoF employment.

Originally VoF was developed for Eulerian simulation of incompressible fluid flows with free boundaries[13]. Excellent reviews of the VoF methods and, in particular, interface reconstruction techniques can be found in the following papers [29, 6, 26, 31, 30]. Here we only present some observations which are important for understanding of our paper and positioning of our new method with respect to other interface reconstruction methods.

On each time step of a hydro simulation, a typical VoF method:

- reconstructs the material interfaces in the mixed cells,
- updates the material content of the cells, usually by computing fluxes through the cell boundaries.

The reconstruction and the update steps are tightly coupled inside the hydrocode: the update step provides the input data for the interface reconstruction algorithm, and the interface reconstruction helps the update routine to calculate the material fluxes with higher accuracy.

To estimate the accuracy and understand the limitations of a stand-alone interface reconstruction algorithm, we would like to take it out of the hydrocode context and try it in the *static reconstruction test*. The test is quite simple: given a true material layout and a computational grid, one has to find the cell-wise material volumes and use this data to construct the interfaces with a VoF algorithm. By comparing the reconstructed material layout to the true one, one can understand how accurate the VoF interface reconstruction algorithm is.

All VoF interface reconstruction algorithms are formulated in terms of two materials; in this paper we refer to them as materials A and B. The volume fractions of two materials sharing a mixed cell are not independent but complement each other to 1. Essentially only the volume fractions of one material (A in our case), explicitly participate in the calculations. The most common interface approximation consists of a single linear interface in each mixed cell. This class of interface reconstructions is commonly called Piecewise-Linear Interface Calculation (PLIC). The primary objective of the reconstruction algorithm is evaluation of the interface normal; once the direction of the normal is known, the location of the interface is uniquely identified by the volume of the cell fraction behind the interface.

There is a number of ways to evaluate the direction of the normal:

- by estimating the gradient of the discrete volume fraction function [38, 39, 3][1)];
- by finding a linear interface, which, being extended outside the mixed cell, gives the best possible approximation to the given volume fractions in the surrounding cells [27, 26];
- by finding a common linear interface for a pair of adjacent mixed cells that satisfies the given volume fractions [8, 18, 35, 19].

Algorithms that calculate the normal as the gradient of the discrete volume fraction function (Youngs' [39], Green-Gauss [3], and Least Square Gradient (LSG) [3] algorithms) allow direct implementation but are only 1st-order accurate. To get a 2nd-order accurate approximation on an unstructured grid, one has to use an iterative technique, like the LVIRA [27, 26] or Mosso-Swartz [35, 19] algorithms.

Neither of the VoF interface reconstruction algorithms can evaluate the direction of the normal without the material volume data from the surrounding cells. As a result, the reconstructed interfaces in adjacent mixed cells are never quite independent. This inherent feature of VoF methods prohibits the resulting approximation from resolving interface details smaller than a characteristic size of the cell cluster involved in the evaluation of the normal. This statement is illustrated by Figure 1, which demonstrates how the quality of the VoF interface reconstruction varies with the scale of interface details.

[1)] Monograph [3] is not about the interface reconstruction. The Green-Gauss and Least Square Gradient algorithms presented there specify the means to calculate the gradient of a cell-centered discrete function on unstructured meshes.

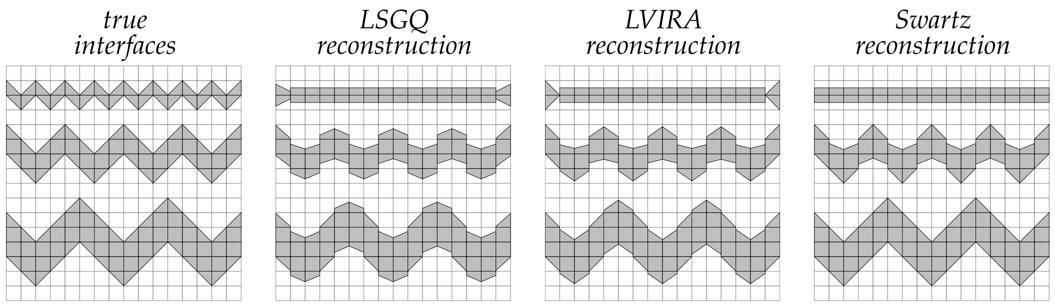


Figure 1. The true material interfaces (on the left) and their reconstructions obtained with various VoF-PLIC methods. This example demonstrates that the VoF interface reconstruction methods tend to oversmooth the interface, which effectively prohibits them from resolving the interface details smaller than the double mesh-step size.

In order to improve the resolution of VoF methods, we propose to use for the interface reconstruction not only the cell-wise material volumes, but also the cell-wise material centroids. The amount of information carried by the volume and centroid or, equivalently, by the first two moments of the cell fraction is sufficient to define a linear interface bounding this cell fraction *even without data from the neighboring cells*. Subject to matching the prescribed volume exactly, we minimize the discrepancy between the centroid of the material behind the interface and the prescribed centroid. This approach, called the *Moment-of-Fluid (MoF)* interface reconstruction, results in a unique, stable, volume-conservative, interface approximation. The technique is 2nd-order accurate and reconstructs the linear interfaces exactly. This is not the same as the exact reconstruction of linear interfaces by the VoF algorithms; to capture the true interface exactly, a 2nd-order accurate VoF method requires the interface to be linear all through the cluster of 3 cells, while the MoF algorithm requires the interface to be linear only inside one cell. In particular, the MoF algorithm can reconstruct the "zigzag" layout (Figure 1) exactly.

The MoF construction of the linear interface in a mixed cell depends only on the moment data from within the cell and not on the data from its neighbors. Therefore, the MoF method is able to resolve interface details as small as the cell itself, which are *2-3 times smaller* than conventional VoF methods can resolve. Due to its autonomous nature, the MoF interface reconstruction can be implemented as a cell-by-cell black-box routine, which is a great technological advantage over the VoF, especially in 3D.

Centroid involvement in interface reconstruction has a clear motivation: the volume fraction indicates only the amount of material in the cell, while the centroid specifies the average location of the material. The centroid of any volume of incompressible fluid moves very much like a Lagrangian particle, which makes it feasible to update in hydro applications.

The idea to utilize the centroid data in interface reconstruction is not new. J. Saltzman and E. Puckett [28] successfully used the *center of mass* algorithm for interface reconstruction; recently we have learned that similar approach is used by D. Bailey e.a. [2]. S. Mosso and S. Clancy [20] used centroid information to derive the material ordering for multi-component interface reconstruction. The similar technique was independently presented by D. Benson [5]. The major difference of the Benson's technique from the others is that he treats the material centroids as independent variables, rather than derives their approximations from the volume fractions.

The rest of the paper is organized as follows. In Section 2 we give a formal statement of the Moment-of-Fluid interface reconstruction problem and describe its numerical solution in 2D. After that, in Section 4, we compare the MoF and VoF algorithms by means of static test; one can find there numerous visual examples and the results of the convergence study. In section 5 we present a feasible way to update the moment data in Eulerian simulation and show how the MoF and VoF algorithms behave in dynamic tests. Section 6 discusses the interface reconstruction aspects of the Arbitrary Lagrangian-Eulerian (ALE) methods.

2 Moment-of-fluid interface reconstruction

Essential definitions. Consider a polygonal mixed cell $\Omega \subset \mathbb{R}^2$ filled with two materials: A and B. Let $\omega \subset \Omega$ be the part of the cell occupied by material A. *The first two moments of* ω are

$$M_0(\omega) \;\equiv \int\limits_{\omega} \mathrm{d}\omega = |\omega| \;\;\in\;\; \mathbb{R}$$

and

$$\mathbf{M}_1(\omega) \;\equiv \int\limits_{\omega} \mathbf{x}\,\mathrm{d}\omega \;\;\in\;\; \mathbb{R}^2;$$

here $|\cdot|$ denotes *volume (area in 2D)*.

An equivalent description of the first two moments of such a subcell is given by the combination of the *volume fraction*

$$\mu(\omega) \;\equiv\; \frac{|\omega|}{|\Omega|} \quad (0 < \mu(\omega) < 1)$$

and the *centroid*

$$\mathbf{x}_c(\omega) \equiv \frac{1}{|\omega|}\,\mathbf{M}_1(\omega) = \frac{1}{|\omega|}\int\limits_{\omega}\mathbf{x}\,\mathrm{d}\omega$$

(since $|\omega|$ is strictly positive, the centroid $\mathbf{x}_c(\omega)$ is always well-defined).

The part of the subcell boundary $\partial\omega$ different from the boundary of the cell $\partial\Omega$ represents the *interface* $\Gamma(\omega)$ between materials A and B:

$$\Gamma(\omega) = \partial\omega \setminus \partial\Omega.$$

A subcell with a flat interface is called *linear*. To emphasize the fact that the subcell is linear, we equip its symbol with "l" subscript: ω_l. The outward unit normal on the interface of linear subcell ω_l is denoted by $\mathbf{n}(\omega_l)$. The *family of all linear subcells* is referred to as $\mathfrak{S}_l$.

2.1 Problem formulation

Suppose $\omega^* \subset \Omega$ is a true subcell occupied by material A. Given the first two moments of ω^* or, equivalently, the volume fraction $\mu^* \equiv \mu(\omega^*)$ and the centroid $\mathbf{x}^* \equiv \mathbf{x}_c(\omega^*)$, one is required to find a linear subcell ω_l^* that provides a good approximation to the true subcell ω^*.

It would be natural to try to find a linear subcell ω_l^* that matches the given moments exactly. Unfortunately, such a problem almost certainly have no solution. Each linear subcell ω_l is uniquely defined by two independent parameters: the polar angle of the interface outward normal $\mathbf{n}(\omega_l)$ and the subcell height measured in the normal direction; i.e. the solution space $\mathfrak{S}_l$ is two-dimensional. And since the space of the input parameters (given volume fraction and two components of the given centroid) is three-dimensional, the problem is overdetermined.

Therefore, one has to use a different strategy. Since the volume conservation is of highest priority to us, our strategy will be the following:

Problem 1 (Moment-of-fluid (MoF) interface reconstruction)**.** *Among all the linear subcells of the given volume fraction*

$$\mathfrak{S}_l^* \equiv \{\, \omega_l \in \mathfrak{S}_l \mid \mu(\omega_l) = \mu^* \,\}$$

find the one, whose centroid is closest to the given centroid:

$$\omega_l^* = \arg\min_{\omega_l \in \mathfrak{S}_l^*} ||\,\mathbf{x}_c(\omega_l) - \mathbf{x}^*||^2, \tag{1}$$

where $||\cdot||$ is Euclidean norm.

Properties of the problem. One special case of this problem is of particular interest. If the true subcell ω^* is linear itself, then it is automatically a member of $\mathfrak{S}_l^*$ family, and for any solution ω_l^* of the MoF problem

$$0 \ \leqslant \ ||\,\mathbf{x}_c(\omega_l^*) - \mathbf{x}^*|| \ \leqslant \ ||\,\mathbf{x}_c(\omega^*) - \mathbf{x}^*|| \ = \ 0,$$

i.e. the solution centroid matches the true centroid:

$$\mathbf{x}_c(\omega_l^*) = \ \mathbf{x}^* \equiv \mathbf{x}_c(\omega^*).$$

There is one trivial solution of the MoF problem in this case $\omega_l^* = \omega^*$, and since *each linear subcell is uniquely identified by its centroid* [10], such a solution is unique. Therefore, *the MoF reconstruction of linear subcell is always exact*, which also suggests that *the MoF reconstruction of any twice-differentiable true interface is 2nd-order accurate*.

A detailed analysis presented in [10] shows that *the MoF interface reconstruction problem is well-posed, i.e. the solution always exists, is unique and stable (with absolute certainty).*

2.2 Implementation of the MoF reconstruction in 2D

If the mixed cell Ω is a triangle or a convex quad, the MoF interface reconstruction problem can be solved analytically. But our objective is to design a method that works for an arbitrary polygonal mixed cell; that is why we solve (1) numerically.

In order to make the numerical optimization feasible, one has to introduce a suitable parameterization of the solution space $\mathfrak{S}_l^*$ and provide efficient algorithms for calculating the objective function and its derivative(s).

Parameterization. Each linear subcell of the given volume is uniquely identified by the polar angle ϕ of the interface outward normal $\mathbf{n}(\omega_l)$ (Figure 2); we use notation $\omega_l(\phi)$ for the member of $\mathfrak{S}_l^*$, whose interface outward normal is given by ϕ, i.e.

$$\mathbf{n}(\omega_l(\phi)) = \ \mathbf{n}_\phi \equiv (\cos\phi, \sin\phi),$$

and notation $\mathbf{x}_l(\phi)$ for the respective centroid:

$$\mathbf{x}_c(\phi) \equiv \mathbf{x}_c(\omega_l(\phi)).$$

The value of the objective function

$$f(\phi) \equiv ||\,\mathbf{x}_l(\phi) - \mathbf{x}^*||^2 \tag{2}$$

for an arbitrary ϕ can be calculated in three steps:

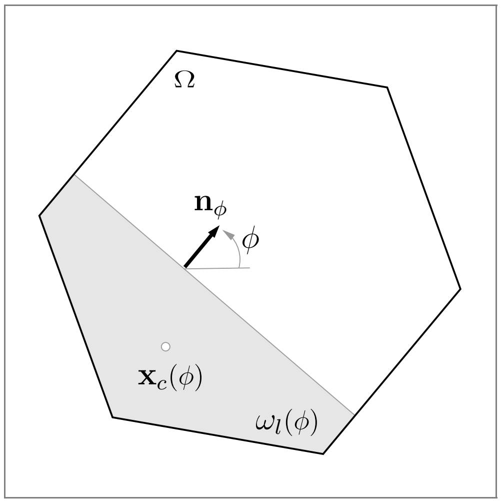


Figure 2. Parameterization of the search space $\mathfrak{S}_l^*$. Ω is a hexagonal mixed cell. Each linear subcell ω_l of the given volume fraction μ^* is uniquely identified by the polar angle ϕ of the interface normal $\mathbf{n}$.

1) explicitly find $\omega_l(\phi)$,
2) calculate the centroid of $\omega_l(\phi)$ using (19),
3) find the squared distance between the calculated and given centroids.

The last two steps are straightforward, but the first one requires a detailed explanation.

Since the direction of the interface normal is fixed ($\mathbf{n}(\omega_l(\phi)) = \mathbf{n}_\phi$), all we have to know to identify $\omega_l(\phi)$ is its height in the normal direction. Let us agree to call the direction defined by vector $\mathbf{n}_\phi$ *vertical*. This convention allows us to use common terms, like *height* and *altitude*, to specify the size and position of geometrical objects along the direction $\mathbf{n}_\phi$. The direction perpendicular to the vertical we will call *horizontal*.

Consider a horizontal line l moving upwards from the bottom to the top of the cell. An instant position of the line is described by its altitude ξ over the cell bottom. The intersection $\Gamma(\xi)$ between the cell Ω and the line $l(\xi)$ is the cross section of the cell at level ξ. As the line moves, the end points of the cross section $\Gamma(\xi)$ slide along the line segments (the cell edges). Therefore, the total length of the cell cross section $|\Gamma(\xi)|$ is a piecewise-linear function of ξ; the linear coefficients may change only when the line $l(\xi)$ crosses a cell vertex.

The volume fraction $\mu(\xi)$ of the region swept by the moving line (the linear subcell bounded from above by level ξ) is given by the integral

$$\mu(\xi) = \frac{1}{|\Omega|} \int\limits_0^{\xi} |\Gamma(\xi)| \, \mathrm{d}\xi.$$

It is a continuous monotone piecewise-quadratic function of ξ; the quadratic coefficients may change only when the line crosses a cell vertex (Figure 3). Or one can say that the second derivative of $\mu(\xi)$ is a piecewise-constant function with potential discontinuity points given by the vertex altitudes.

To find the height ξ^* of $\omega_l(\phi)$, one has to solve equation

$$\mu(\xi^*) = \mu^*. \tag{3}$$

The continuity and monotonicity of $\mu(\xi)$ guarantee that the inverse is well-defined, and the piecewise-quadratic property of $\mu(\xi)$ yields an efficient solution strategy for (3) that requires only $O(n)$ operations whenever the cell Ω is convex and $O(n^2)$ operations otherwise (here n is the number of the cell vertices). We call this strategy the *Flood Algorithm*, since it models the process of flooding a vessel, represented by the cell, with fluid.

The Flood Algorithm specifies three steps to identify the "fluid level" ξ^* in the "vessel" Ω, at which the volume of "fluid" matches $\mu^*|\Omega|$:

Algorithm 1 (Flood Algorithm)**.**

1) *sort the cell vertices by their altitudes to have all the discontinuity points of $\mu''(\xi)$ arranged in a non-descending order;*
2) *find the interval of quadraticity (linearity) of $\mu(\xi)$ that includes ξ^*;*
3) *calculate ξ^* by means of polynomial interpolation.*

1) Sorting the cell vertices. Let $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ be the cell vertices enumerated in counter-clockwise order, and let $\xi_1, \xi_2, \ldots, \xi_n$ be their respective altitudes over the cell bottom. One can find a permutation of the vertex indices $(i_1, i_2, \ldots, i_n)$ that puts the vertex altitudes in a non-descending order

$$0 \equiv \xi_{i_1} \leqslant \xi_{i_2} \leqslant \ \ldots \ \leqslant \xi_{i_n} \tag{4}$$

in $O(n)$ operations whenever the cells is convex and in $O(n \log n)$ operations otherwise.

2) Bracketing the solution. Since $\mu(\xi)$ is monotonically increasing, the solution of (3) belongs to the interval $[\,\xi_{i_{k^*}}, \xi_{i_{k^*+1}}]$, $1 \leqslant k^* \leqslant n-1$ that satisfies condition

$$\mu_{k^*} \leqslant \mu^* \leqslant \mu_{k^*+1}, \tag{5}$$

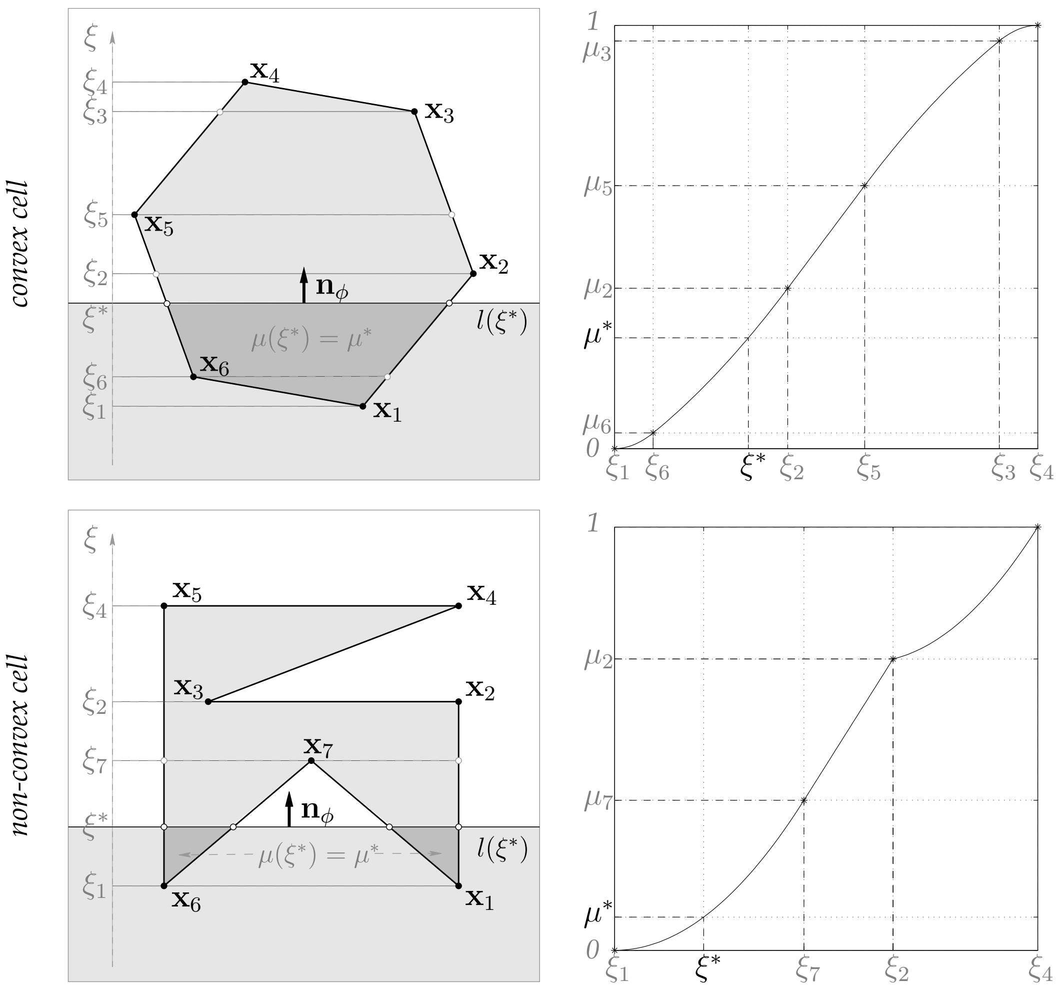


Figure 3. A convex and a non-convex polygonal cell (on the left) and the graphs of the volume fraction $\mu(\xi)$ of their respective linear subcells bounded from above by level ξ (on the right). The volume fraction $\mu(\xi)$ is continuous monotone piecewise-quadratic function of the linear subcell height ξ; the jumps of the quadratic coefficients are given by the vertex altitudes. Function $\mu(\xi)$ is also smooth, unless the cell has an entering angle with a horizontal edge (see the non-convex example).

where $\mu_k \equiv \mu(\xi_{i_k})$ is the volume fraction of the linear subcell bounded from above by level ξ_{i_k}, $k = \overline{1,n}$. Thus, k^* can be found by successive evaluation of all $\mu_k, k = 1, 2, \dots$.

Since $\mu(\xi)$ is at most quadratic on each interval $[\xi_{i_k}, \xi_{i_{k+1}}]$, $k = \overline{1, n\text{-}1}$, the area of Ω, enclosed between levels ξ_{i_k} and $\xi_{i_{k+1}}$, can be calculated by the trapezoid rule:

$$|\Omega|\,(\mu_{k+1} - \mu_k) = \frac{1}{2}\,(\xi_{i_{k+1}} - \xi_{i_k})\,\big(|\Gamma_{k+1}| + |\Gamma_k|\big); \tag{6}$$

here $|\Omega|$ is the area of the cell, and $|\Gamma_k|$ is the total length of the cell cross section $\Gamma_k \equiv \Gamma(\xi_{i_k})$ at level $\xi_{i_k},\ k = \overline{1,n}$.

We use equation (6) to evaluate all $\mu_k,\ k = 2, 3, \ldots$ and find k^*. We start the search at the "bottom"

$$k^* \leftarrow 1, \quad \mu_1 = 0, \quad |\Gamma_1| = 0,$$

and go "upwards"

$$\begin{aligned} &k^* \leftarrow k^* + 1, \\ &\mu_k = \mu_{k-1} + \tfrac{1}{2\,|\Omega|}\,(\xi_{i_k} - \xi_{i_{k-1}})\,\big(|\Gamma_k| + |\Gamma_{k-1}|\big), \quad k = 2, 3, \ldots, \end{aligned}$$

until the given volume fraction is bracketed, i.e. condition (5) is satisfied. To calculate each next μ_k, we have to know the cross-section length at level ξ_k first, thus we must identify the end points of the cross section Γ_k. This takes $O(1)$ operations whenever the cell is convex and $O(n)$ operations otherwise.

3) Finding the solution. Once the given volume fraction is bracketed (5), the solution of (3) can be found by means of polynomial interpolation:

- whenever $|\Gamma_{k^*+1}| = |\Gamma_{k^*}|$, the volume fraction $\mu(\xi)$ is linear on $[\,\xi_{i_{k^*}}, \xi_{i_{k^*+1}}\,]$, and
$$\xi^* = \xi_{i_{k^*}} + \frac{\mu^* - \mu_{k^*}}{\mu_{k^*+1} - \mu_{k^*}}\,(\xi_{i_{k^*+1}} - \xi_{i_{k^*}});$$
- otherwise the cross-section length $|\Gamma(\xi)|$ is linear, the volume fraction $\mu(\xi)$ is quadratic, and
$$\xi^* = \xi_{i_{k^*}} + \frac{|\Gamma^*| - |\Gamma_{k^*}|}{|\Gamma_{k^*+1}| - |\Gamma_{k^*}|}\,(\xi_{i_{k^*+1}} - \xi_{i_{k^*}}),$$
where
$$|\Gamma^*| = \sqrt{|\Gamma_{k^*}|^2 + \frac{\mu^* - \mu_{k^*}}{\mu_{k^*+1} - \mu_{k^*}}\,(|\Gamma_{k^*+1}|^2 - |\Gamma_{k^*}|^2)}$$
is the length of the cell cross-section at the level ξ^*.

Concluding notes on the Flood Algorithm. If the given volume fraction μ^* exceeds *1/2*, the subcell $\omega_l(\phi)$ can be found faster and with higher accuracy, if one will “flood” the mixed cell with material B instead, i.e. will set the vertical direction to $-\mathbf{n}_\phi$ and “pour” fluid B in the amount of $(1-\mu^*)|\Omega|$.

The derivative of the objective function is given by

$$f'(\phi) = \big((\mathbf{x}_l(\phi) - \mathbf{x}^*) \cdot (\mathbf{x}_l(\phi) - \mathbf{x}^*)\big)' = 2\left((\mathbf{x}_l(\phi) - \mathbf{x}^*) \cdot \mathbf{x}_l'(\phi)\right),$$

where $(\cdot\cdot\cdot)$ is *2D dot product*.

To complete the formula above, one needs the expression for the first derivative of $\mathbf{x}_l(\phi)$; the latter can be found by evaluating the ratio

$$\frac{\mathbf{M}_1(\omega_l(\phi+\Delta\phi)) - \mathbf{M}_1(\omega_l(\phi))}{|\Omega|\Delta\phi} = \mathbf{x}_l'(\phi) + o(\Delta\phi)$$

for small $\Delta\phi$:

- whenever the interface $\Gamma(\phi) \equiv \Gamma(\omega_l(\phi))$ consists of a single segment (which is always the case for a convex cell),

$$\mathbf{x}_l'(\phi) = -\frac{1}{12}\frac{|\Gamma(\phi)|^3}{\mu^*|\Omega|}\,\mathbf{t}_\phi, \tag{7}$$

  where $\mathbf{t}_\phi \equiv (-\sin\phi, \cos\phi)$ is counter-clockwise unit tangent on $\Gamma(\phi)$.

- for a non-convex cell the interface $\Gamma(\phi)$ may consist of several separate segments of the interface line (see Figure 3, non-convex cell); in this case

$$\mathbf{x}_l'(\phi) = -\frac{M_2(\Gamma(\phi))}{\mu^*|\Omega|}\,\mathbf{t}_\phi, \tag{8}$$

  where $M_2(\Gamma(\phi))$ is the *second central moment* of interface $\Gamma(\phi)$:

$$M_2(\Gamma) = \int\limits_\Gamma ||\,\mathbf{x} - \mathbf{x}_c(\Gamma)||^2\,\mathrm{d}\Gamma, \quad \mathbf{x}_c(\Gamma) = \frac{1}{|\Gamma|}\int\limits_\Gamma \mathbf{x}\,\mathrm{d}\Gamma.$$

Numerical optimization of objective function (2) can be performed with any smooth optimization routine available. For completeness we included the description of the optimization routine we used in Appendix C.

Initial guess. A safe initial guess ϕ_0 for the optimization routine is provided by the polar angle of vector $\mathbf{x}_c(\Omega) - \mathbf{x}^*$. There is a simple explanation of such a choice: the centroid of a linear subcell is always located behind the interface, therefore, the outward interface normal should point out of the centroid towards the cell center. A typical configuration shown on Figure 4 demonstrates, that such an initial guess is a good approximation to the MoF objective function minimum.

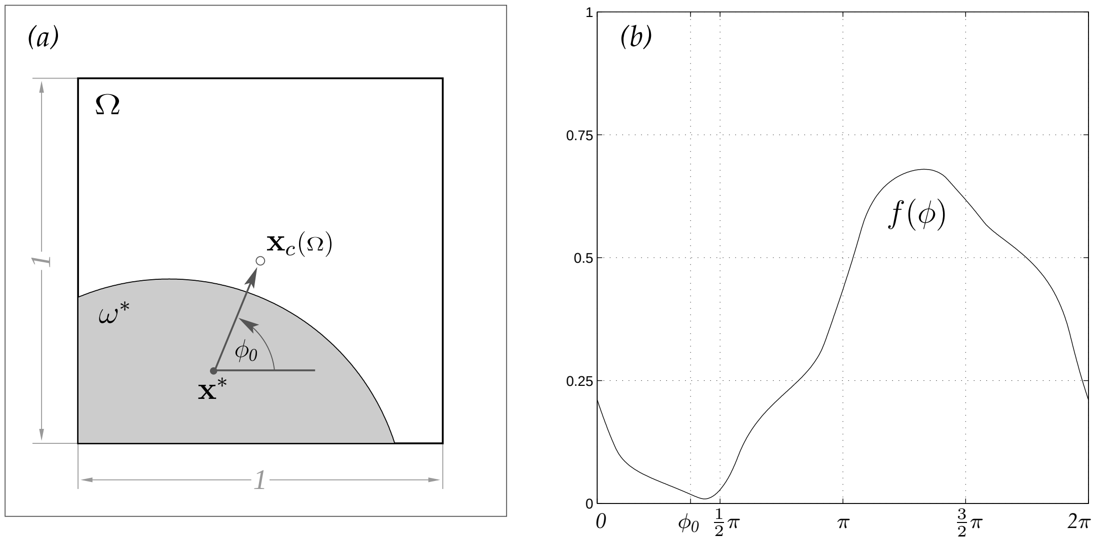


Figure 4. On the left one can see a unit square mixed cell Ω, and a sample subcell ω^* occupied by material A. The vector between the centroid of the true subcell $\mathbf{x}^*$ and the centroid of the cell $\mathbf{x}_c(\Omega)$ defines the initial guess ϕ_0 for the MoF solver. The respective MoF objective function graph on the right shows, that such an initial guess is a good approximation to the function minimum.

3 The VoF-PLIC algorithms challenged.

To compare the MoF algorithm to the existing interface reconstruction methods, we picked three VoF-PLIC algorithms for general polygonal grids:

- the Least Square Gradient [2) algorithm with reciprocal Quadratic distance weights [3], which we refer to as LSGQ,
- the Least square Volume(-of-fluid) Interface Reconstruction Algorithm (LVIRA) [27, 26],
- and the Swartz algorithm [35, 19].

The choice of the last two algorithms is pretty obvious: they are the only 2nd-order accurate VoF-PLIC methods that are suitable for unstructured grids.

Among the 1st-order accurate VoF-PLIC methods we picked the LSGQ algorithm. Frankly, the only real alternative to LSGQ on unstructured grids is the Green-Gauss algorithm [3]. We picked LSGQ over the Green-Gauss algorithm, because it is less demanding with respect to the mesh data structure: unlike the

2) See the notes on the name of this algorithm in Appendix D.

Green-Gauss algorithm, LSGQ does not require the adjacent cells to be ordered (counter-)clockwisely. Both algorithms belong to the class of "gradient" methods, and even though they use different approaches to estimate the gradient of the volume fraction function, the results produced on a uniform rectangular grid are equivalent[3]; moreover, the result is exactly the same as with the Youngs' algorithm [39]. Therefore, any example of the LSGQ reconstruction on a uniform rectangular grid, is also that of the Green-Gauss (Youngs').

Since the behavior of the interface reconstruction algorithms varies widely with the implementation details, we highlighted all essential details about our implementation of these algorithms in Appendix D.

4 Static tests

The accuracy of any interface reconstruction technique can be evaluated through the static reconstruction test: *given a true shape of material A and a computational grid, one has to*

1) *compute the cell-wise material moments (for this, one has to explicitly find the intersection of each cell with the true material shape first; see Appendices B and A on how to find the intersection and calculate its moments);*
2) *based on these data, construct the interfaces in the mixed cells;*
3) *and then compare the reconstructed shape of material A to the true one.*

4.1 Visual comparison of the PLIC reconstructions.

We created six different material layouts to test the PLIC algorithms against:

- the "constellation" (Figure 5, top left),
- the "A" glyph (Figure 5, top center),
- the "cross" (Figure 5, top right),
- the "slice of pie" (Figure 6, top left),
- the "compass rose" (Figure 6, top center),
- and the "circle" (Figure 6, top right).

Beneath each true material layout presented on Figures 5 and 6 we placed the reconstructions obtained with different PLIC methods. Even a brief look at the results reveals the superior quality of the MoF reconstructions over the VoF ones.

[3] Provided that neither of the boundary cells is mixed.

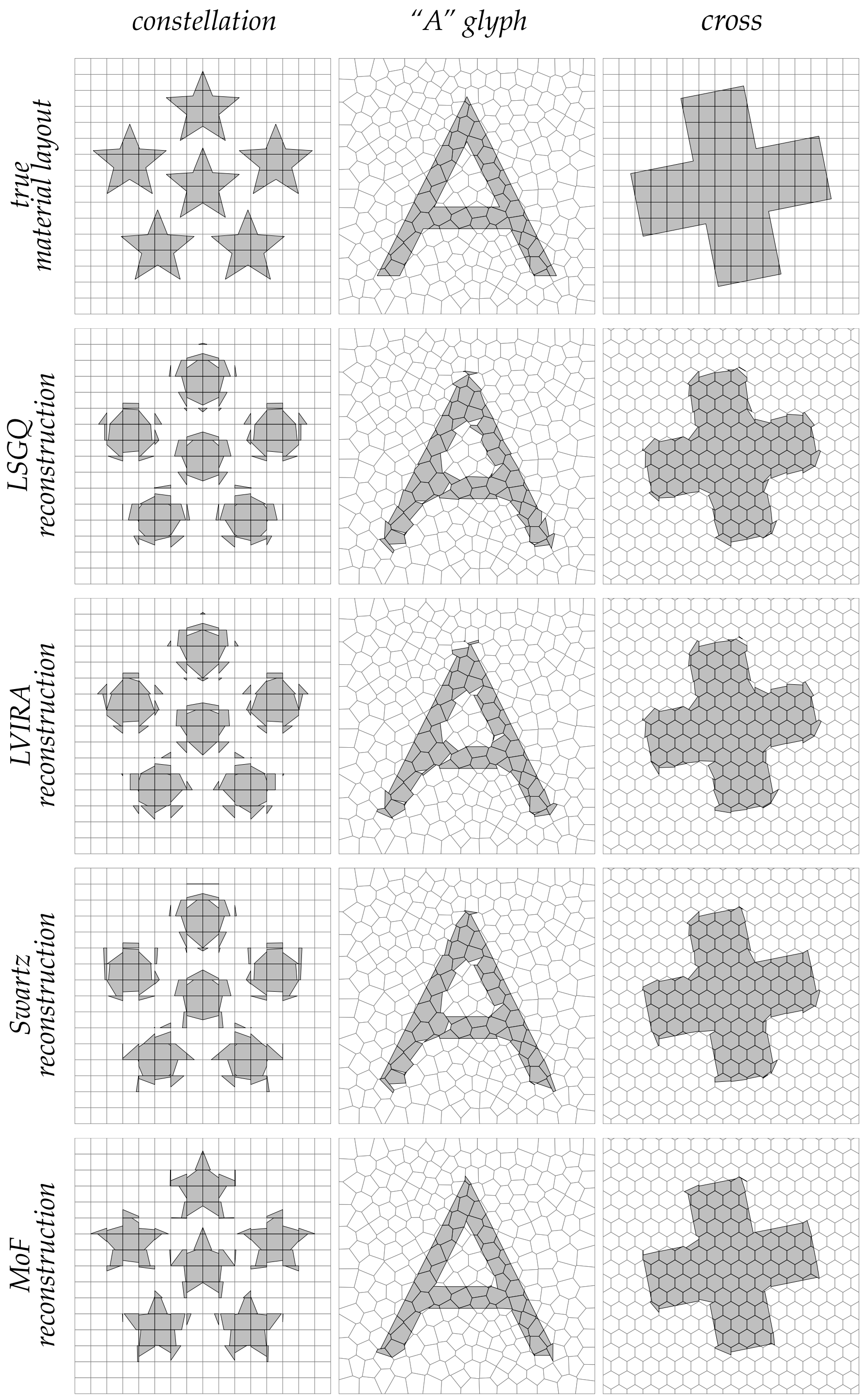


Figure 5. The true material layouts (top row) and their reconstructions obtained with different PLIC methods. The MoF reconstructions (bottom row) are the most accurate and sharp.

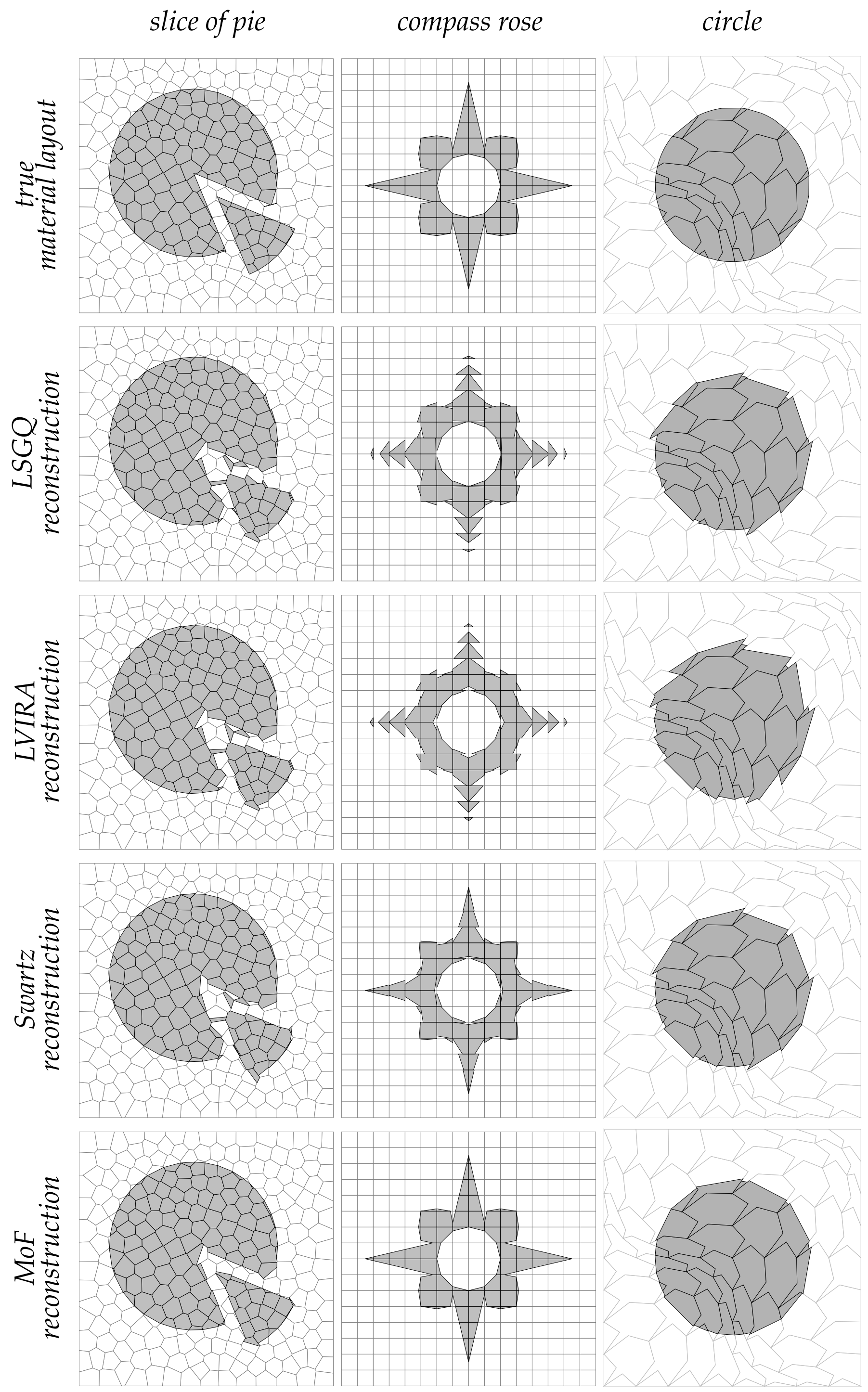


Figure 6. The true material layout (top row) and their reconstructions obtained with different PLIC methods. The MoF reconstructions (bottom row) are the most accurate and sharp.

"Constellation". The "constellation" shape consists of several "stars"; their size and the distance between them is comparable to the size of the mesh cells, which confuses the VoF algorithms a lot. As a result, all the VoF methods diffuse the interface so mush, that no one can recognize the true stars behind their VoF reconstructions. Since the MoF algorithm uses no material moment data from the adjacent cells, it demonstrates much better results than VoF algorithms. Taking into account extremely small size of the interface details, the quality of the MoF reconstruction, is very impressive.

"A" glyph. The thickness of the glyph strokes is comparable to the cell size, which creates inevitable interference between the VoF-PLIC reconstructions of the parallel edges of the strokes.

All the VoF algorithms successfully capture the external edges of the lateral strokes (the LVIRA and Swartz algorithm reconstruct these line segments exactly), but experience difficulties resolving the internal contour of the glyph, where the volume data interference is stronger.

The glyph poses no problem for the MoF algorithm: the only parts of the glyph contour that are not reconstructed exactly are the vertices.

"Cross". The "cross" layout is even less challenging than the "A" glyph. The interference between the reconstructed interfaces is a noticeable only around the vertices of the polygonal interface.

The strongest numerical diffusion is demonstrated by the LSGQ algorithm; the LVIRA reconstruction is less diffused, and the Swartz reconstruction is the sharpest among all the VoF ones.

Once again, the MoF algorithm reconstructs all the line segments of the interface exactly, failing only at the vertices. Clearly, one can make a similar statement about the 2nd-order accurate VoF algorithms, but there is a quantitative difference: the MoF algorithm fails to reconstruct the interfaces only in the mixed cells that host the vertices, while the LVIRA and Swartz algorithms also fail to capture the interface in all the adjacent cells.

"Slice of pie". The round edge of the "pie" is a well isolated interface with low curvature, therefore all the PLIC algorithms tested had no problem reconstructing it properly (one can notice that the LSGQ reconstruction of the north-east edge is somewhat ragged).

The small gap between the "pie" and the "slice" parts of the figure tricks the VoF algorithms into creation of artificial "bridges" between the parts, but can not confuse the MoF algorithm.

"Compass rose". The "compass rose" is yet another example of the polygon shape with small details. Only this time all the interface vertices are located exactly on the grid lines, such that the restriction of the true interface to each mixed cell is a single line segment.

The MoF algorithm takes advantage of the special properties of the figure and reconstructs the whole "compass rose" exactly. Neither of the VoF methods can achieve such a result. The LSQG and the LVIRA reconstructions are too diffused to be of any use, but the Swartz algorithm, due to the *discrimination strategy* described in Appendix D, demonstrates a good ability to fight the volume data interference.

"Circle". The circular shape seems to be an easy target for the VoF-PLIC methods: the interface has low curvature, and there is no potential source of severe interference between the mixed cells. The tough component of this static reconstruction problem is the grid: it is highly non-uniform, and all the cells are non-convex. These unpleasant properties of the grid are responsible for raggedness of the VoF reconstructions; the LVIRA algorithm turned out to be the most sensitive to the quality of the grid. Clearly, the MoF algorithm is the most resistant to the low quality of the grid.

Concluding remarks on visual comparison. One can hardly notice the difference between the the MoF and VoF reconstruction of an isolated interface with low curvature. The difference becomes more and more evident, as the size of the interface details or the distance between the interfaces approaches the size of the mesh cells. In this case, the numerical diffusion introduced by the VoF methods significantly degrades the quality of reconstruction. The most diffusive among the VoF-PLIC methods is LSGQ, the LVIRA algorithm is somewhat better in resolving polygonal interfaces, and the best results are due to the Swartz method. The MoF algorithm, on the other side, demonstrates an excellent ability to capture the interface details on the edge of the grid resolution.

4.2 Convergence study

Cell-wise reconstruction error. Let us consider three different interface reconstruction errors (Figure 7):

- the *defect of the first moment:*

$$\triangle M_1 \equiv ||\,\mathbf{M}_1(\omega_l^*) - \mathbf{M}_1(\omega^*)|| = |\Omega|\,\mu^*||\,\mathbf{x}_l^* - \mathbf{x}^*||; \tag{9}$$

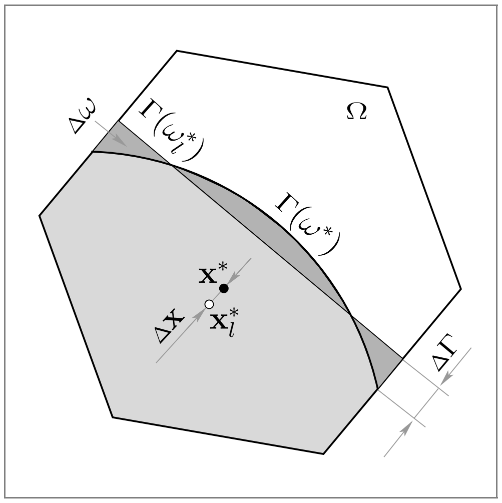


Figure 7. The reconstruction errors illustrated. Ω is a hexagonal mixed cell, $\Gamma(\omega^*)$ and $\Gamma(\omega_l^*)$ are the true and reconstructed material interfaces respectively. $\triangle\Gamma$ is *the maximum deviation* of the true interface $\Gamma(\omega^*)$ from the line given by $\Gamma(\omega_l^*)$; $\triangle\omega$ is *the area of the symmetric difference* between the true and reconstructed subcells (in dark grey); *the defect of the first moment* $\triangle M_1$ is proportional to the distance $\triangle\mathbf{x}$ between the true and reconstructed centroids ($\mathbf{x}^*$ and $\mathbf{x}_l^*$ respectively).

- the *area of the symmetric difference* between the reconstructed and true subcells:
$$\triangle\omega \equiv |\,\omega_l^* \triangle\, \omega^*| = |\,\omega_l^* \setminus \omega^*| + |\,\omega^* \setminus \omega_l^*|;$$
- the *maximum deviation* of the true interface $\Gamma(\omega^*)$ from the *interface line* l^* defining $\Gamma(\omega_l^*) = \Omega \cap l^*$:
$$\triangle\Gamma \equiv \mathrm{dist}(\Gamma(\omega^*), l^*) = \max_{\mathbf{x}\,\in\,\Gamma(\omega^*)} \; \min_{\mathbf{y}\,\in\,l^*} ||\mathbf{x} - \mathbf{y}||.$$

First of all, all three errors above have a nice property of being independent of the choice of material (A or B) represented by ω^* and ω_l^*. Among them, $\triangle M_1$ is the weakest, and $\triangle\Gamma$ is the strongest one (convergence in $\triangle\Gamma$ implies convergence in $\triangle\omega$, and convergence in $\triangle\omega$ implies convergence in $\triangle M_1$). By default, the reconstruction error is measured in terms of $\triangle\Gamma$. Thus, *a reconstruction is known to be k-th-order accurate, if it results in $\triangle\Gamma = O(d^k)$ for all sufficiently small d (the diameter of the cell).* Since both the true and approximate interfaces are confined within the cell, *any interface reconstruction is at least 1st-order accurate.*

As long as the original interface $\Gamma(\omega^*)$ is known to be twice-differentiable with the curvature radius bounded from below by a positive constant R, the following estimates hold true:

$$\vartriangle M_1 = O(d^5/R^2), \quad \vartriangle\omega = O(d^3/R), \quad \vartriangle\Gamma = O(d^2/R), \tag{10}$$

where $d = \mathrm{diam}(\Omega)$ is the cell diameter. The fact that $\vartriangle\Gamma$ scales quadratically with the diameter of the cell confirms that *the MoF interface reconstruction is 2nd-order accurate* [10].

For a non-smooth true interface all the errors above reach their respective pessimistic upper bounds:

$$\vartriangle M_1 = O(d^3), \quad \vartriangle\omega = O(d^2), \quad \vartriangle\Gamma = O(d). \tag{11}$$

The defect of the first moments $\vartriangle M_1$ is interesting mostly from the theoretical perspective: since the MoF interface reconstruction is achieved by minimizing the centroid discrepancy $||\,\mathbf{x}_l^* - \mathbf{x}^*||$, then according to (9) it should result in the minimal defect of the first moment attainable with a volume-conservative PLIC approximation. In this sense *the MoF interface reconstruction is optimal in the class of the volume-conservative PLIC methods*.

Although $\vartriangle\Gamma$ is the default choice of the reconstruction error, it may be ill-defined in dynamic tests (Section 5). That is why we are going to use the area of the symmetric difference $\vartriangle\omega$ as a primary error indicator. J. E. Pilliod and E. G. Puckett [26] were first to use the $\vartriangle\omega$ indicator to measure the reconstruction error in static tests, although they used an alternative definition of $\vartriangle\omega$ as the L_1 norm of the difference between the characteristic functions of ω_l^* and ω^*.

Note that $\vartriangle\omega$ and $\vartriangle\Gamma$ errors are not that much different. The symmetric difference area divided by the length of the true interface gives the *average deviation*

$$\overline{\vartriangle\Gamma} = \vartriangle\omega / |\Gamma(\omega^*)|$$

of the reconstructed interface from the true one. Under pretty non-restrictive conditions[4)] the average $\overline{\vartriangle\Gamma}$ and the maximum $\vartriangle\Gamma$ deviations are equivalent, i.e. there exist positive constants $0 < c_1 < c_2$, such that

$$c_1\, \overline{\vartriangle\Gamma} \;\leqslant\; \vartriangle\Gamma \;\leqslant\; c_2\, \overline{\vartriangle\Gamma};$$

therefore, $\vartriangle\omega(\sim \overline{\vartriangle\Gamma})$ and $\vartriangle\Gamma$ errors should have the same order of accuracy.

How to measure the cell-wise $\vartriangle\omega$ error. Since the PLIC algorithms we examine are volume-conservative, then in static interface reconstructions tests

$$\vartriangle\omega = |\,\omega^* \triangle\, \omega_l^*| = |\,\omega_l^* \setminus \omega^*| + |\,\omega^* \setminus \omega_l^*| = 2|\,\omega_l^* \setminus \omega^*| = 2\big(|\omega^*| - |\omega^* \cap \omega_l^*|\big). \tag{12}$$

4) The angular size of the cell corners must be bounded away from $0 + \varepsilon_0$ and $2\pi - \varepsilon_0$ for some fixed $\varepsilon_0 > 0$; the true interface must be piecewise-smooth and satisfy the cone condition [**?**].

Therefore, in order to find $\Delta\omega$ for a mixed cell, it is sufficient to measure the area of the intersection between the true and reconstructed subcells (see Appendices B and A on the numerical intersection of plane figures and the area calculation respectively).

The global errors. To explain how the cell-wise $\Delta\omega$ errors can be conveniently aggregated in global error indicators, we need to change the notations. From now on $\Omega \subset \mathbb{R}^2$ represents the entire computational domain, quasi-uniformly partitioned into N polygonal cells $\{\Omega_i\}_{i=1}^{N}$; $\omega^* \subset \Omega$ and $\omega_l^* \subset \Omega$ represent the true and the reconstructed shapes of material A; the restriction of ω^* and ω_l^* to a single cell $\Omega_i,\ i = \overline{1, N}$ will be referred to as ω_i^* and $\omega_{l,i}^*$ respectively. Without loss of generality, we can assume that all the mixed cells are enumerated from 1 to $M \leqslant N$.

First of all, we would like to know the maximum of the cell-wise $\overline{\Delta\Gamma}$ errors, which shows the worst local error attainable:

$$\Delta\Gamma_{max} = \max_{i=\overline{1,M}} \frac{|\,\omega_i^* \triangle\, \omega_{l,i}^*|}{|\Gamma(\omega_{l,i}^*)|};$$

we call it the *maximum deviation*.

Another useful global error is the average distance between the reconstructed and true interfaces:

$$\Delta\Gamma_{avg} = \frac{|\,\omega^* \triangle\, \omega_l^*|}{|\partial\omega^*|} = \frac{1}{|\partial\omega^*|}\sum_{i=1}^{M} |\,\omega_i^* \triangle\, \omega_{l,i}^*|;$$

it is referred to as the *average deviation*.

Direct measurements of the reconstruction errors. To figure out how these errors scale with mesh refinement, we conducted a series of direct error measurements on a sequence of uniform rectangular grids ($\Omega =]0,1[\times]0,1[$) with the mesh spacing h varying from $1/2$ to $1/2^{10}$ ($h_k = 1/2^k,\ k = \overline{1,10}$).

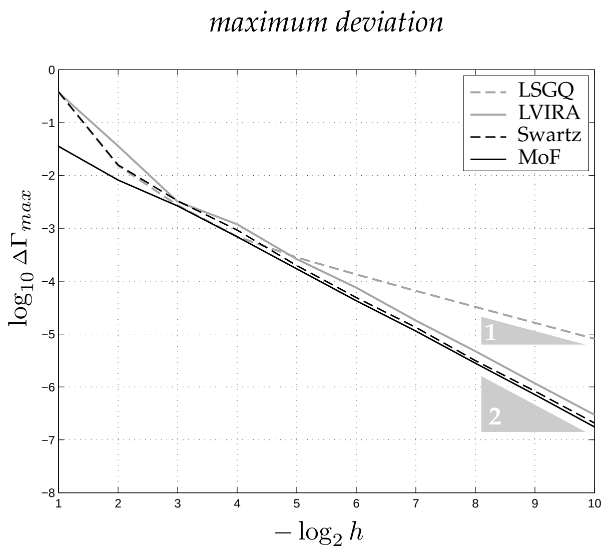
Each PLIC algorithm was tested against two shapes:

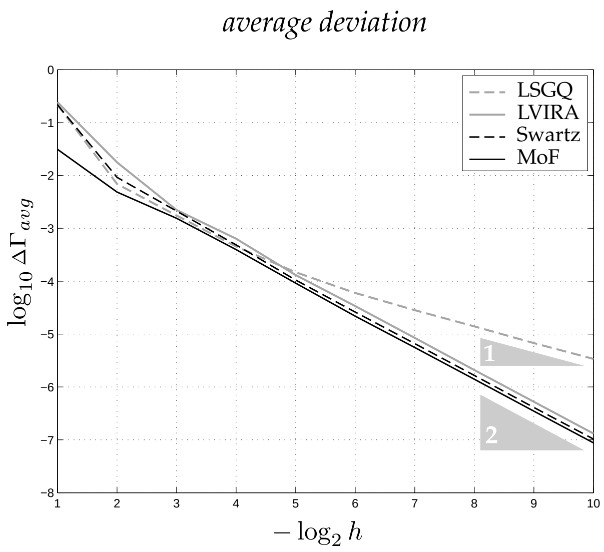
1) the circle of radius $R = 0.25$ centered at *(0.5 + 1/17, 0.5 + 1/41)*; $|\partial\omega^*| = \pi/2$;

2) the $L \times L = 0.5 \times 0.5$ square centered at *(0.5 + 1/17, 0.5 + 1/41)* and rotated counter-clockwise by $\pi/3$ radians; $|\partial\omega^*| = 2$.

The location and orientation of the true shapes were chosen to eliminate the mesh imprint in error readings.

The errors measured are collected in tables and presented on graphs; Figure 8 shows the results for the circle, and Figure 9 shows the results for the square shape. Note that the graph axes are logarithmic.



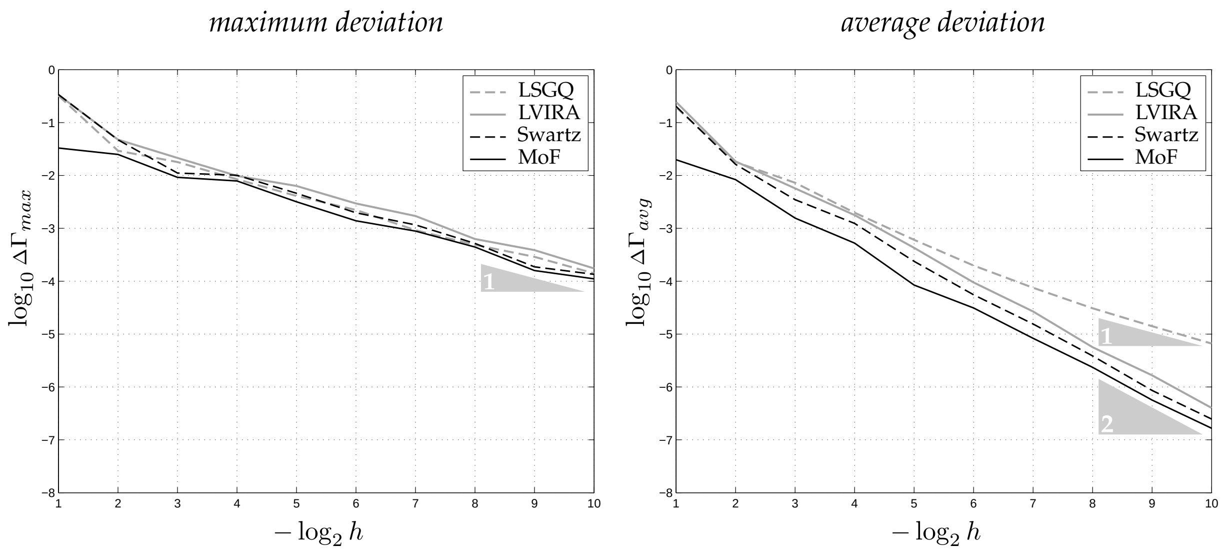


$-\log_2 h$	$1/h$	LSGQ	LVIRA	Swartz	MoF
1	2	3.81e-1	3.71e-1	3.81e-1	3.56e-2
2	4	1.53e-2	3.60e-2	1.58e-2	8.18e-3
3	8	2.73e-3	3.21e-3	3.31e-3	2.65e-3
4	16	6.97e-4	1.19e-3	9.27e-4	6.91e-4
5	32	2.81e-4	2.61e-4	1.98e-4	1.71e-4
6	64	1.34e-4	7.55e-5	4.91e-5	4.26e-5
7	128	6.56e-5	1.80e-5	1.34e-5	1.14e-5
8	256	3.27e-5	4.75e-6	3.15e-6	2.81e-6
9	512	1.63e-5	1.17e-6	8.34e-7	7.14e-7
10	1024	8.15e-6	2.96e-7	2.07e-7	1.75e-7

$-\log_2 h$	$1/h$	LSGQ	LVIRA	Swartz	MoF
1	2	2.17e-1	2.43e-1	2.17e-1	3.12e-2
2	4	6.90e-3	1.77e-2	9.21e-3	4.86e-3
3	8	1.75e-3	2.22e-3	2.12e-3	1.55e-3
4	16	4.52e-4	6.36e-4	4.86e-4	3.96e-4
5	32	1.48e-4	1.31e-4	1.05e-4	9.18e-5
6	64	5.98e-5	3.39e-5	2.61e-5	2.18e-5
7	128	2.83e-5	8.48e-6	6.60e-6	5.56e-6
8	256	1.40e-5	2.09e-6	1.65e-6	1.40e-6
9	512	6.76e-6	5.24e-7	4.10e-7	3.48e-7
10	1024	3.39e-6	1.31e-7	1.03e-7	8.73e-8

Figure 8. The maximum (on the left) and the average (on the right) deviations of the reconstructed interface from the true one as functions of the mesh spacing h. The true interface is a circle of radius R =*0.25*. Due to the uniform curvature of the true interface, the respective maximum and average deviations are comparable and exhibit similar asymptotic behavior. The LSQG algorithm shows to be 1st-order accurate, the rest (LVIRA, Swartz, and MoF algorithms) are 2nd-order accurate. Among all the methods tested, the Moment-of-Fluid method is the most accurate.

Circle. Due to the uniform curvature of the true interface, the maximum and average deviations behave similarly. As the asymptotics show, the LVIRA, Swartz



$-\log_2 h$	$1/h$	LSGQ	LVIRA	Swartz	MoF
1	2	3.23e-1	3.34e-1	3.38e-1	3.30e-2
2	4	2.94e-2	4.76e-2	4.73e-2	2.50e-2
3	8	1.79e-2	2.14e-2	1.11e-2	9.21e-3
4	16	8.49e-3	9.82e-3	1.01e-2	7.86e-3
5	32	4.09e-3	6.33e-3	4.60e-3	3.18e-3
6	64	2.22e-3	2.95e-3	1.97e-3	1.39e-3
7	128	9.22e-4	1.71e-3	1.16e-3	8.87e-4
8	256	4.88e-4	6.27e-4	5.18e-4	4.44e-4
9	512	2.89e-4	3.87e-4	1.87e-4	1.59e-4
10	1024	1.41e-4	1.74e-4	1.35e-4	1.11e-4

$-\log_2 h$	$1/h$	LSGQ	LVIRA	Swartz	MoF
1	2	1.95e-1	2.45e-1	2.02e-1	1.98e-2
2	4	1.79e-2	1.84e-2	1.62e-2	8.31e-3
3	8	7.28e-3	5.70e-3	3.48e-3	1.56e-3
4	16	1.98e-3	1.78e-3	1.24e-3	5.25e-4
5	32	6.07e-4	4.27e-4	2.38e-4	8.47e-5
6	64	1.98e-4	9.47e-5	5.55e-5	3.11e-5
7	128	7.53e-5	2.68e-5	1.55e-5	8.30e-6
8	256	3.07e-5	5.64e-6	3.87e-6	2.35e-6
9	512	1.41e-5	1.66e-6	8.57e-7	5.64e-7
10	1024	6.61e-6	4.01e-7	2.47e-7	1.65e-7

Figure 9. The maximum (on the left) and the average (on the right) deviations of the reconstructed interface from the true one as functions of the mesh spacing h. The true interface is a *0.5*×*0.5* square. Since the true interface is not smooth, the maximum deviations measured are 1st-order accurate for all the methods. The average deviations, due to their aggregate nature, are less volatile than the maximum ones. Surprisingly, the asymptotic order of convergence of the average deviations is the same, as if the true interface were twice-differentiable (explained in the text). The average deviation of any of the VoF reconstructions is at least *50*% higher than the average deviation of the Moment-of-Fluid reconstruction.

and MoF algorithms approximations of the circular interface are 2nd-order accurate, while the LSGQ approximation is only 1st-order accurate.

Among all the methods tested, the MoF algorithm is the most accurate over the full range of grid resolutions. In asymptotic regime ($h \ll R$), the average deviation of the LVIRA and Swartz reconstructions are respectively *18%* and *50%* higher the average deviation of the MoF reconstruction.

Even though the LSGQ algorithm is asymptotically less accurate than the LVIRA and Swartz ones, on the coarse grid ($R/8 \leqslant h$) the 1st-order-accurate VoF algorithm results in significantly smaller average deviation than the 2nd-order accurate VoF algorithms (just *6%* above the MoF error). The LSGQ algorithm starts to loose the advantage over the LVIRA and Swartz algorithms only when the cell size drops below *1/8* of the interface curvature radius.

Square. Because the interface is non-smooth, all four PLIC methods result in the maximum deviations of order $O(h)$.

The asymptotic average deviation due to the LSGQ algorithm is of order $O(h)$, as it should be for the 1st-order accurate algorithm. Surprisingly, the 2nd-order-accurate algorithms in this test demonstrate the average deviations of order $O(h^2)$, as if the interface were twice-differentiable. This behavior has an explanation. All 2nd-order accurate methods are able to capture the line segments of the true interface exactly; the error is due only to the mixed cells, surrounding the vertices of the square. With $O(1)$ mixed cells contributing to the average error, the global $\Delta\omega$ is only $O(h^2)$ (see (11)), which results in the average deviation $\Delta\Gamma_{avg} = \Delta\omega/(4L) = O(h^2)$.

As with the circle, the MoF algorithm shows to be the most accurate one, only this time the superiority of the MoF method over the VoF competitors is more pronouncing. The LSGQ algorithm shows the worst average deviation of all the PLIC algorithms tested: it is unable to reconstruct the line segments of the true interface exactly, as all the 2nd-order methods do, and over-smoothes the interface around the vertices. The average deviation due to the Swartz algorithm and LVIRA are respectively *50%* and *100%* higher than the respective MoF error. Such a big error difference between the MoF and 2nd-order accurate VoF algorithms is explained by the fact that there is only one mixed cell per square vertex that contributes to the average MoF error, while there are two to four mixed cells per square vertex that contribute to the average VoF error (see Figure 10). We also want to emphasize that the Swartz method resolves the corners better than LVIRA; this is due to the novel *discrimination strategy* used in the Swartz algorithm (described in Appendix D).

Concluding notes on the reconstruction error. The numerical experiments confirm that the MoF interface reconstruction is 2nd-order accurate and also show

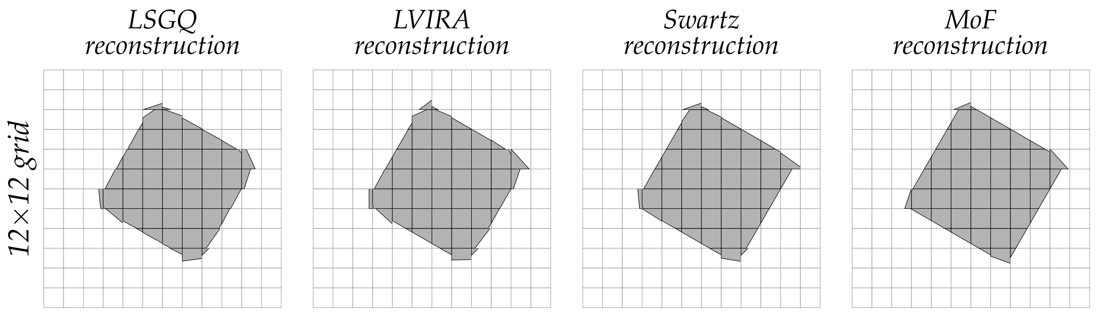


Figure 10. The reconstructions of the square shape obtained with various PLIC algorithms. All 2nd-order accurate methods (LVIRA, Swartz, MoF) reconstruct the line segments of the interface exactly, introducing some error only around the vertices. The MoF reconstruction is inexact only in 4 mixed cells that overlay the vertices. The LVIRA and Swartz reconstructions are inexact not only in the cells that overlay the vertices, but also in their direct neighbors. This explains why the MoF error is significantly lower error than the respective VoF errors. We also want to emphasize that the Swartz method resolves the corners better than LVIRA; this is due to the novel discrimination strategy used in the Swartz algorithm (described in Appendix D).

that it results in lower absolute error than the conventional VoF methods.

One can see that symmetric difference area $\Delta\omega$ can be successfully used instead of conventional $\Delta\Gamma$ error; both errors have similar behavior and converge at the same rate. Between the maximum and average deviations, the latter seems to be more convenient, since it is less volatile and reveals the correct order of accuracy of the reconstruction algorithms even when the interface is only piecewise-smooth.

A word on performance. Among all the PLIC methods tested, LSGQ is the fastest one. It is about five times faster than MoF and Swartz methods and seven times faster than LVIRA. One have to understand that these numbers are approximate and may vary. To make the performance comparison unbiased, we shared the common software components among the reconstruction routines. Thus, the implementation of all four PLIC algorithms used the same routine for cutting off subcells of the given volume; the implementations of the LVIRA and MoF algorithms shared the optimization routine; all the iterative methods (LVIRA, Swartz, and MoF) were subject to termination under the same condition, namely, when the polar angle of the interface normal stabilizes within 10^{-6} radians.

5 How to update the cell-wise material moments

The MoF interface reconstruction algorithm can not be used in a fluid flow simulation without the consistent scheme for updating the material moments. Below we present an example of the material transport algorithm that can be used to update the volume and centroid data in incompressible fluid flow simulations. The scheme is referred to as *Lagrangian remap*.

The solenoidal velocity field $\mathbf{v}(\mathbf{x}, t)$ is assumed to be given analytically, therefore no explicit constraint on the Courant number $CFL = v \vartriangle t/h$ is imposed (here $\vartriangle t$ is the time step, h is the local mesh spacing, and v is the local flow speed).

5.1 Updating the volumes

Given the material distribution at $t_{k-1} = \vartriangle t\ (k-1)$, one can evaluate the content of the cells at $t_k = t_{k-1} + \vartriangle t$ as follows:

```
for each cell $\Omega_i,\ i = \overline{1, N}$ do
    track $\Omega_i$ back in time to $t = t_{k-1}$ to identify the Lagrangian preimage $\tilde{\Omega}_{i,k-1}$
                                                                              of the cell;
    find the volume $\tilde{v}_{i,k-1}$ of material A enclosed in Lagrangian
                                                                     preimage $\tilde{\Omega}_{i,k-1}$;
    put the volume $v_{i,k}$ of the material A in $\Omega_i$ equal to $\tilde{v}_{i,k-1}$.
end do
```

The vertices of each polygonal cell $\Omega_i,\ i = \overline{1, N}$ are tracked back along the streamlines by means of the 4-th order Runge-Kutta scheme and then connected in proper order by the line segments. This results in a polygon that we consider to be a *discrete Lagrangian preimage* $\tilde{\Omega}_{i,k-1}$ of cell Ω_i (see Figure 11a).

Using the polygon intersection routine (Appendix B) we find the intersections of $\tilde{\Omega}_{i,k-1}$ with all pure-material (sub)cells at $t = t_{k-1}$ overlayed by the preimage. One can significantly accelerate the search of the cells overlayed with the bucket sort: partition the bounding box of the entire computational domain in 2D array of rectangular buckets and pre-sort all the mesh cells among these buckets, based on the cell centroid location. If $CFL \leqslant 1$, it is sufficient to intersect the preimage only with the pure-material subcells of Ω_i and its direct neighbors. Moreover, if $CFL \leqslant 1$ and Ω_i, along with all its neighbors, at $t = t_{k-1}$ are filled with the same material, then the content of Ω_i will not change by $t = t_k$, and there is no need to perform any polygon intersections in this case at all.

Area defect. In a linear velocity field, which is known to preserve straight lines, the true Lagrangian preimage of a polygonal cell is always a polygon, and the ac-

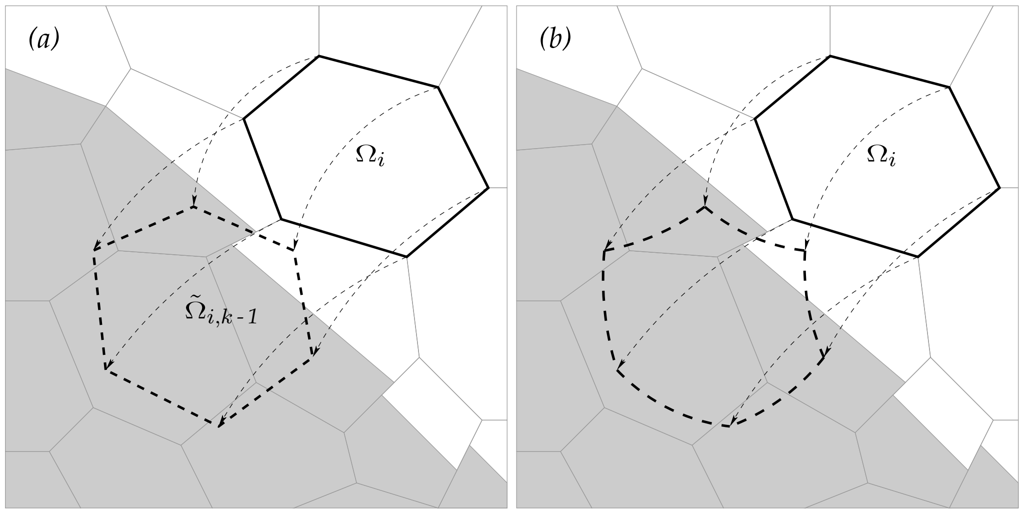


Figure 11. The discrete (a) and true (b) Lagrangian preimages of the cell.

curacy of the remapping is limited only by the accuracy the integration scheme. For a p-th-order accurate scheme the *area defect* $|\tilde{\Omega}_{i,k\text{-}1}| - |\Omega_i|$ is estimated as $O(h\,\Delta t^{p+1})$.

In a nonlinear velocity field straight lines are not preserved, and the true Lagrangian preimage of a polygonal cell is not a polygon. Therefore, by ignoring the curvature of the preimage edges, we introduce additional $O(h^3\Delta t)$ area defect.

Even a small area defect can eventually cause the algorithm to halt. Indeed, if the volume of material A $\tilde{v}_{i,k\text{-}1}$ enclosed in the preimage $\tilde{\Omega}_{i,k\text{-}1}$ exceeds the capacity of the cell $|\Omega_i|$, we are in trouble. Another, less critical, situation occurs when the discrete preimage, being completely filled with material A, happens to have the volume $|\tilde{\Omega}_{i,k\text{-}1}|$ smaller than $|\Omega_i|$. In this case the cell Ω_i turns into mixed, even though its preimage contained only one material.

Volume repair procedure. In order to fix these flaws, we use a post-remapping *volume repair* procedure. For every cell $\Omega_i,\ i = \overline{1, N}$ we specify the lower $\underline{v}_{i,k}$ and the upper $\overline{v}_{i,k}$ bounds of the material A volume $v_{i,k}$ allowed:

$$(\underline{v}_{i,k},\ \overline{v}_{i,k}) = \begin{cases} (\ \ 0,\ \ \ 0), & \text{if the preimage is empty,} \\ (|\Omega_i|, |\Omega_i|), & \text{if the preimage is full,} \\ (\ \ 0, |\Omega_i|) & \text{otherwise,} \end{cases} \tag{13}$$

and then force each volume $v_{i,k},\ i = \overline{1, N}$ to fit into these bounds:

```
for each cell $\Omega_i,\ i = \overline{1, N}$ do
```

```
    if $\Omega_i$ is *overfilled* ($\overline{v}_{i,k} < v_{i,k}$) then
        try to redistribute the excess between the *non-overfilled* neighbors
        while there is still some excess remained do
            redistribute them among the next layer of the surrounding cells
        end do
    else if $\Omega_i$ is *underfilled* ($v_{i,k} < \underline{v}_{i,k}$) then
        try to compensate the shortfall by borrowing from the *non-underfilled* neighbors
        while there is still some shortage of material do
            borrow it from the next layer of the surrounding cells
        end do
    end if
end do
```

Due to the local nature of the area defect, the redistribution usually involves only direct neighbors of the cell. Therefore, the complexity of the whole volume repair step is estimated as $O(N)$.

5.2 Updating the centroids

Minimal modernization of the volume-tracking technique above allows to update the cell-wise material centroids as well. Whenever a non-empty intersection of the discrete Lagrangian preimage $\tilde{\Omega}_{i,k-1}$ with any subcell from the previous time step is found, one has to calculate not only its volume but also the first moment. After all the pieces of material A overlayed by the preimage are found, one should calculate the centroid of material A enclosed in $\tilde{\Omega}_{i,k-1}$ as the moment to volume ratio. This centroid, tracked forth along the streamlines, determines the position of the material-A centroid in cell Ω_i. There is no need to track the centroid, if the preimage contains only one material: the volume repair step guarantees that the cell will stay pure as well.

Centroid error. Whenever the velocity field is linear in space, the actual centroid velocity of any volume of fluid ω is the same as the field velocity at the centroid location:

$$\frac{d}{dt}\mathbf{x}_c(\omega) = \mathbf{v}(\mathbf{x}_c(\omega)). \tag{14}$$

Therefore, in this case, the trajectory of the true centroid follows the streamline and the centroid error is due only to approximate integration ($O(\Delta t^{p+1})$ for a p-th order scheme).

For a nonlinear velocity field the identity (14) is void. As long as $\mathbf{v}(\mathbf{x}, t)$ is twice-differentiable in $\mathbf{x}$, the following estimate, given by the Taylor expansion,

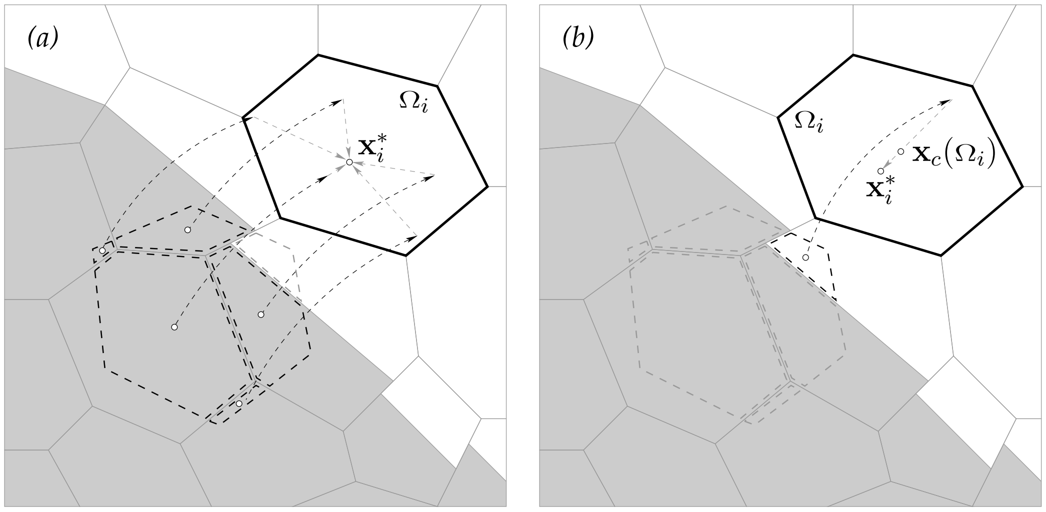


Figure 12. Two ways to improve the accuracy of the centroid advection: (a) advect the material in pieces, (b) advect the centroid of the smaller fraction.

holds:

$$\frac{d}{dt}\mathbf{x}_c(\omega) = \mathbf{v}(\mathbf{x}_c(\omega)) + O(d^2). \tag{15}$$

Here $d = \operatorname{diam}(\omega)$ is the diameter of the volume advected. The final centroid error in this case is $O(d^2 \Delta t)$.

Since the centroid error is determined by the diameter of the advected volume, some accuracy improvement can be gained by advecting the fluid in smaller pieces. By the virtue of the method, the Lagrangian preimage is "assembled" of smaller polygonal pieces, the intersections $\tilde{\Omega}_{i,k\text{-}1}$ with the pure-material subcells from the previous time step. Therefore, instead of advecting the aggregated centroid of material A, one can advect the centroids of these constitutive parts independently, and proceed with their aggregation only after the advection step (Figure 12a).

Whenever the volume of material A in the preimage exceeds the volume of material B, it is likely that the material-B centroid can be advected more accurately than the material-A centroid. Since the choice of material, whose moments are explicitly used in the interface reconstruction, is just a matter of convention, one should stick with material B in this situation, i.e. advect the material-B centroid and then use it to separate material B from the mixed cell (the MoF interface reconstruction itself is more stable for smaller volume fractions).

5.3 Numerical examples

To demonstrate the Lagrangian remap at work, we prepared two examples:

- the solid rotation of the "A" glyph and
- the evolution of the round "blot" the reversible shear deformation vortex.

Solid rotation of the "A" glyph. The "A" glyph is defined by two closed polylines; the vertices of the outer contour (in counter-clockwise order) are:

$$\begin{array}{l}
(0.25\quad\;\;,\ 0.25\quad),\\
(0.3125\;\;,\ 0.25\quad),\\
(0.378125,\ 0.38125),\\
(0.621875,\ 0.38125),\\
(0.6875\;\;,\ 0.25\quad),\\
(0.75\quad\;\;,\ 0.25\quad),\\
(0.5\quad\;\;\;,\ 0.75\quad);
\end{array}$$

the vertices of the inner contour are:

$$\begin{array}{l}
(0.40859375,\ 0.4421875),\\
(0.59140625,\ 0.4421875),\\
(0.5\qquad\quad,\ 0.625\quad\;\;).
\end{array}$$

The velocity field:

$$\mathbf{v}(x,y,t) = \begin{bmatrix} y_0 - y \\ x - x_0 \end{bmatrix}^T, \quad (x_0,\ y_0) = (0.5,\ 0.5). \tag{16}$$

Other parameters of the numerical experiment:

computational domain	$]0,\ 1[\times]0,\ 1[$,
simulation time	$T = 2\pi$,
computational grid	1024 polygonal cells,
number of the time steps	$N_T = 144$.

The snapshots of the "A" glyph rotation performed with different PLIC methods are presented on Figure 13. Ideally, after one period ($t = T$), the glyph is supposed to return to the initial state, but it doesn't happen because of the discretization errors. Since the solid rotation velocity field is linear, the cell-wise material moments are updated very accurately, and the interface reconstruction error prevails over the advection error. One can see that all the VoF methods diffuse the interface to much higher degree than the MoF method.

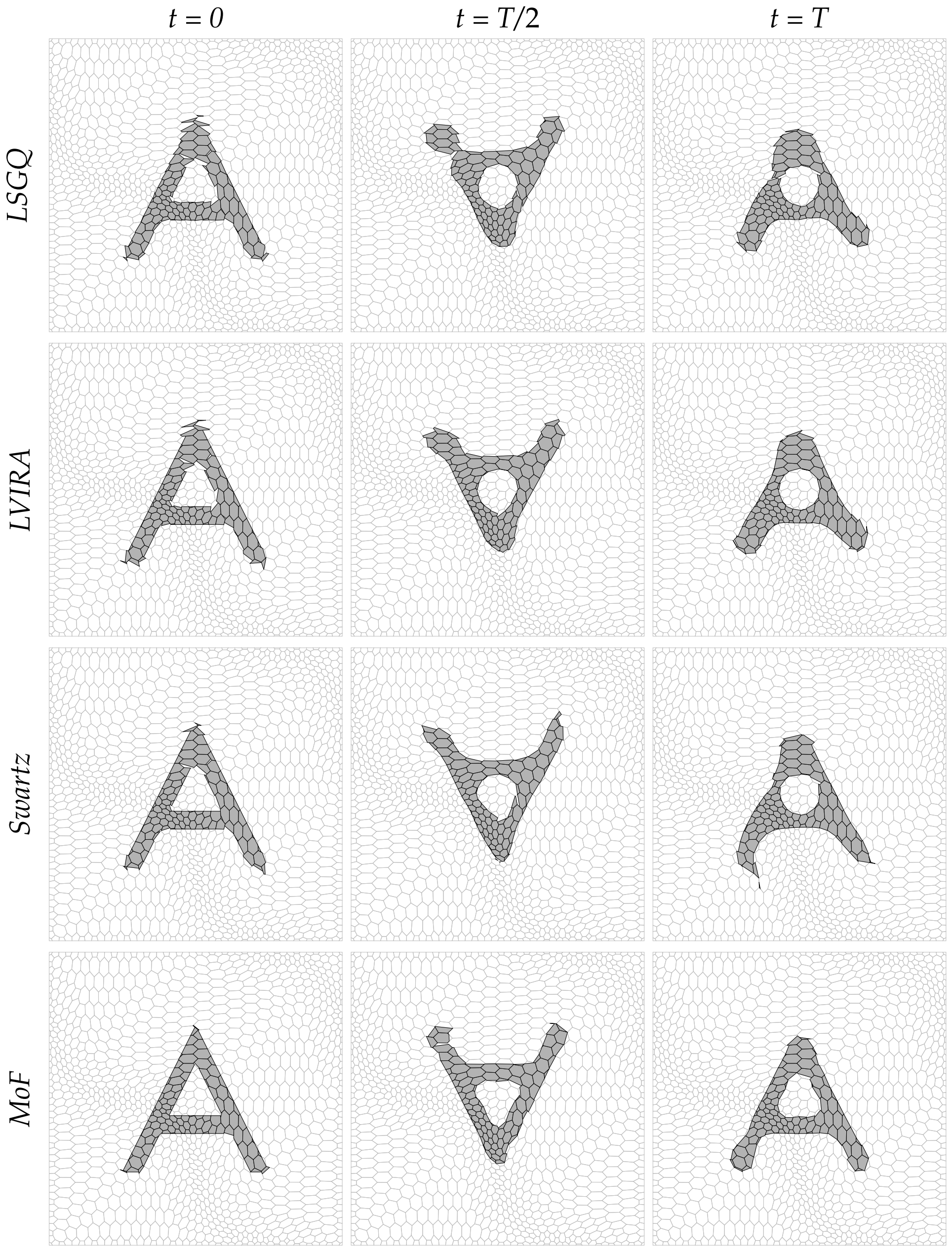


Figure 13. The snapshots of the "A" glyph in the solid-rotation velocity field (16) at $t = 0$ (left column), $t = T/2$ (central column), and $t = T$ (right column). Each row corresponds to one of the four PLIC algorithms: LSGQ, LVIRA, Swartz, and MoF. The least diffused results (bottom row) are due to the MoF algorithm.

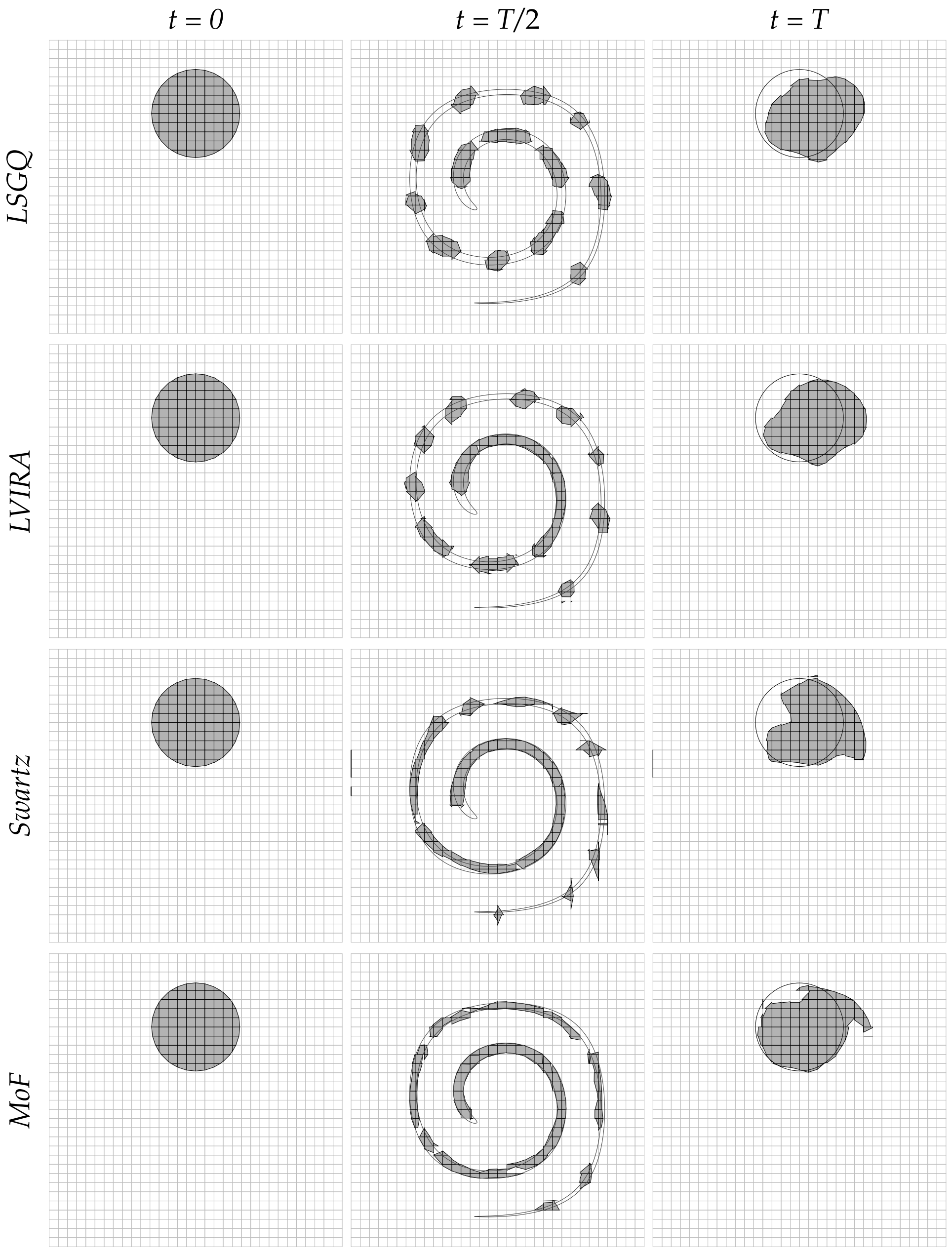


Figure 14. The snapshots of the "blot" in the reversible vortex (17) at $t = 0$ (left column), $t = T/2$ (central column), and $t = T$ (right column). Each row corresponds to one of the four PLIC algorithms: LSGQ, LVIRA, Swartz, and MoF. For easier judgment, we also showed the true material interfaces at $t = T/2$ and $t = T$. The most accurate results (bottom row) are due to the MoF algorithm.

Shear deformation test. The second test case [29] simulates the evolution of the round "blot"

$$\{\, \mathbf{x} \in \mathbb{R}^2 \;|\; ||\,\mathbf{x} - \mathbf{x}_0|| \leqslant R \,\}, \quad \mathbf{x}_0 = (0.5,\ 0.75),\ R = 0.15$$

in the shear deformation vortex field:

$$\mathbf{v}(x, y, t) = \begin{bmatrix} +\sin^2(\pi x)\sin(2\pi y) \\ -\sin^2(\pi y)\sin(2\pi x) \end{bmatrix}^T \cos(\pi t/T). \tag{17}$$

The cosine multiplier gradually slows down the vortex until the complete stop ($\mathbf{v}(\mathbf{x}, t) \equiv \mathbf{0}$) at $t = T/2$ and then starts to "rewind" it. Since

$$\mathbf{v}(\mathbf{x}, t) = -\mathbf{v}(\mathbf{x}, T - t), \quad 0 < t < T,$$

i.e. the velocity field is reversible, the exact integration of the fluid motion from $t = 0$ to $t = T$ should result in zero changes. Therefore, by comparing the final configuration against the initial setup, one can get the idea of accuracy of the numerical technique used for the interface tracking.

Other parameters of the experiment are:

computational domain	$]0,\ 1[\times]0,\ 1[$,
simulation time	$T = 8$,
computational grid	uniform 32×32 cells,
number of the time steps	$N_T = 256$.

The snapshots of the "blot" tracked with different PLIC methods are presented on Figure 14. For easier judgment, we also showed the true material interfaces at $t = T/2$ (central column) and $t = T$ (right column). The former one was obtained by the front tracking technique described in [1].

All the PLIC reconstructions of the of the initial state (right column) are of decent quality: the material interface is isolated an has a relatively low curvature.

The final ($t = T$) configuration due to the MoF method has a better overlap with the initial shape, than the final configurations due to the VoF competitors. From this point of view, the MoF method tracks the interface more accurately. Also note that the MoF method misplaces only the piece of material A that corresponds to the end of the tail of the stretched body. Such an error may be excused: the thickness of the tail is way below the grid resolution. The VoF errors are less localized. This is because all VoF methods, due to the aggregate nature of the interface normal estimate, have a tendency to oversmooth the interface, which diffuses the misplaced material along the interface.

On the dynamic level, this property of the VoF methods manifests itself in excessive artificial surface tension. This feature becomes determinative when a VoF method is forced to work on the edge of the resolution limit. Thus, by the moment of maximum stretch ($t = T/2$) the material-A shape becomes so thin, that the VoF interface reconstruction breaks it in a series of separate blobs. The MoF method is significantly less prone to that kind of behavior and gives much better approximation to the true interface at $t = T/2$.

5.4 Convergence study

To quantify the accuracy of the Lagrangian remap, we use the following technique. *Given a true initial shape ω^* of material A at moment $t = 0$, a computational grid, a static velocity field, and a total time T, we*

1) *calculate the cell-wise material moments of the initial shape of material A and then reconstruct it statically;*
2) *track the discrete material interfaces in the given velocity field from $t = 0$ to $t = T/2$, reverse the field, and then continue to track the interfaces in the reversed velocity field until $t = T$; the final reconstructed shape of material A is denoted by ω_l^*;*
3) *calculate the area of the symmetric difference between ω^* and ω_l^* to find the average deviation of the final interface from the true one*

$$\Delta\Gamma_{avg} = \frac{|\,\omega_l^* \triangle \omega^*|}{|\,\partial\omega^*|} = \frac{1}{|\,\partial\omega^*|}\sum_{i=1}^{N}\left(|\,\omega_i^*| + |\,\omega_{l,i}^*| - 2|\,\omega_i^* \cap\, \omega_{l,i}^*|\right);$$

here $|\,\partial\omega^*|$ is the perimeter of the true interface, ω_i^* and $\omega_{l,i}^*$ are the restrictions of ω^* and ω_l^* to the i-th cell, $i = \overline{1, N}$.

Common parameters. In all the experiments conducted, the computational domain Ω is the unit square $]0, 1[\times]0, 1[$. The initial true shape of material A is given by the ellipse with axes *0.3 · 1.1 = 3.3* and *0.3/1.1 ≈ 0.272727*; the polar angle of the major axis is *(1/2 + 1/16)*π; the perimeter of such as ellipse $|\,\partial\omega^*| \approx$ *9.48900*. The eccentricity of our ellipse is pretty moderate $e \approx$ *0.552771*. We do not use the circle this time, because in real computations the curvature of the interface is highly unlikely to be uniform. The ellipse is deliberately not co-aligned with the coordinate axes to decrease the potential mesh imprint in error readings. Each interface tracking technique was tested at four different mesh resolutions $h \in \{$*1/16, 1/32, 1/64, 1/128, 1/256*$\}$. The time step Δt in each experiment is determined by the is the maximum velocity of material A: $\Delta t = h/v_{max}$, i.e. the *CFL* is uniformly bounded by *1*. The total time $T = \Delta t/h = 1/v_{max}$.

We tested the Lagrangian remap against three different velocity fields:

- a diagonal translation,
- a solid rotation,
- and a shear deformation.

Diagonal translation. The velocity field:

$$\mathbf{v}(x,y) = (1/\sqrt{2},\ 1/\sqrt{2})$$

with $v_{max} = 1$. The initial shape of material A is given by the ellipse described above, centered at $(0.5 + 1/41,\ 0.75 + 1/101)$. The corresponding error graphs are presented on Figures 15.

Solid rotation. The velocity field:

$$\mathbf{v}(x,y) = \frac{1}{2\pi R_{max}}\big(y_0 - y,\ x - x_0\big), \qquad (x_0,\ y_0) = (0.5,\ 0.5), R_{max} = 0.4$$

with maximum velocity $v_{max} = 1$ (assuming that $||\,(x,\ y) - (x_0,\ y_0)|| \leqslant R_{max}$). The initial shape of material A is given by the ellipse described above, centered at $(0.5{+}1/41,\ 0.75{+}1/101)$. The corresponding error graphs are presented on Figure 16.

Shear deformation. The velocity field:

$$\mathbf{v}(x,y) = \big(+\sin^2(\pi x)\sin(2\pi y),\ -\sin^2(\pi y)\sin(2\pi x)\big)$$

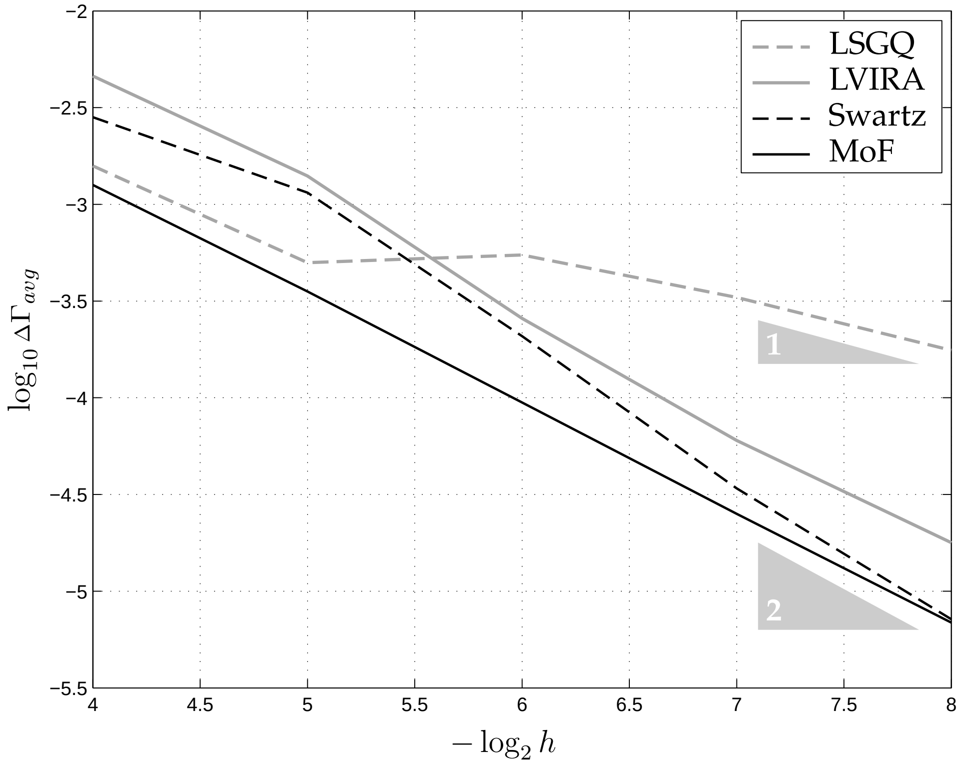
with maximum velocity $v_{max} = 1$. The initial material distribution is the same as for the solid rotation test. The error graphs are presented on Figure 17.

The summary of the numerical results.

As one can see from the data presented, the convergence order is completely determined by the accuracy order of the interface reconstruction algorithm: the 1st-order algorithm (LSGQ) results in $O(h)$ error; the 2nd-order accurate algorithms (LVIRA, Swartz, and MoF) result in $O(h^2)$ error. The MoF algorithm shows the best results in all three experiments.

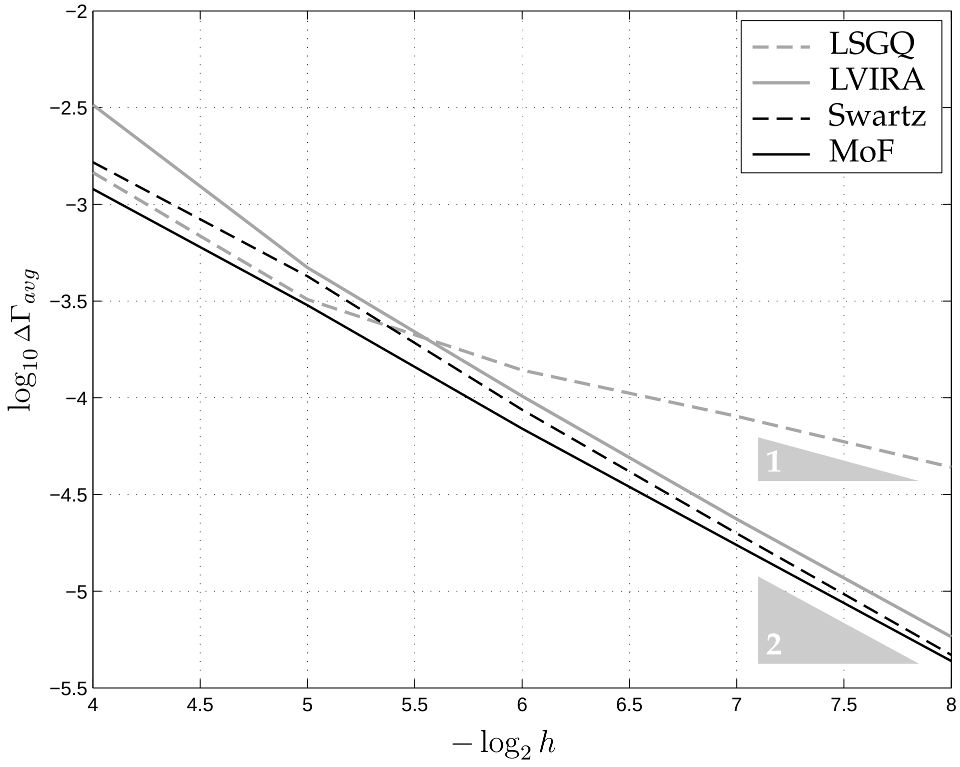
The first two velocity fields are linear in space, therefore the Lagrangian remap updates the cell-wise material moment data exactly, and the errors are due to the approximate reconstruction only. The translation and solid rotation are also special in that they preserve the distance between the fluid points, i.e. they do not

Figure 15. The average deviation error accumulated in the diagonal translation velocity field. The most accurate results are due to the MoF algorithm. The Swartz is the only VoF algorithm that can keep pace with the MoF algorithm on the fine grid.



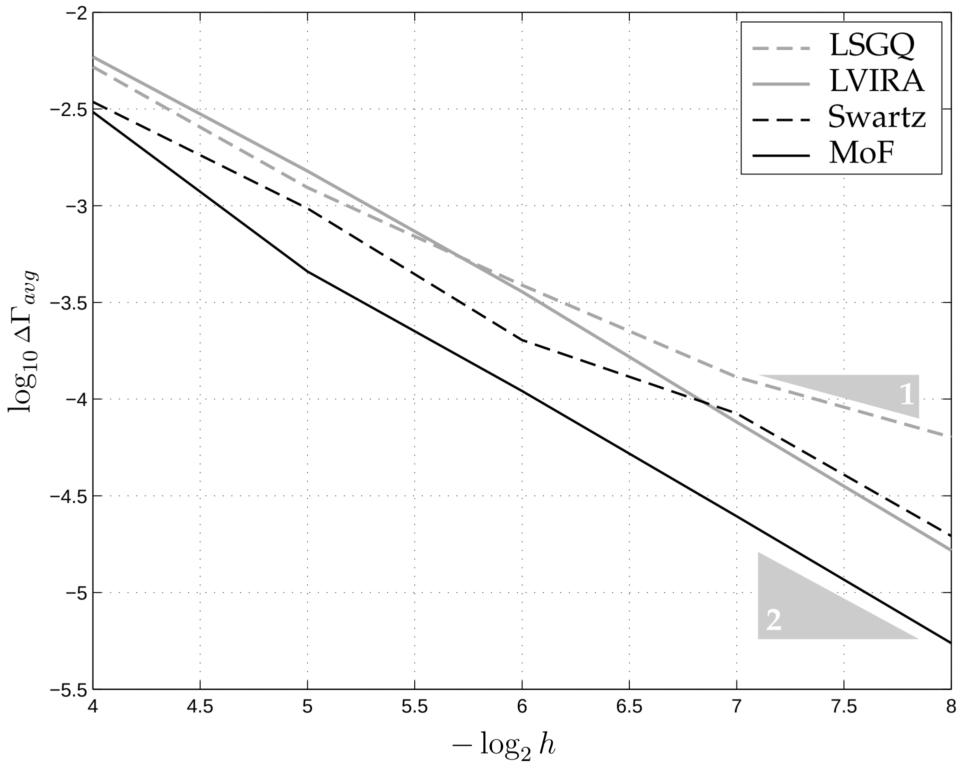
-log$_2 h$	1/h	LSGQ	LVIRA	Swartz	MoF
4	16	1.58e-3	4.61e-3	2.82e-3	1.26e-3
5	32	4.99e-4	1.40e-3	1.15e-3	3.54e-4
6	64	5.47e-4	2.57e-4	2.08e-4	9.45e-5
7	128	3.30e-4	6.01e-5	3.40e-5	2.51e-5
8	256	1.76e-4	1.78e-5	7.16e-6	6.88e-6

Figure 16. The average deviation error accumulated in the solid rotation velocity field. Since the velocity field is linear and preserves the distance between the fluid points, the error graphs look very similar to the static errors graphs (Figure 8). The most accurate results are due to the MoF algorithm.



-$\log_2 h$	1/h	LSGQ	LVIRA	Swartz	MoF
4	16	1.46e-3	3.27e-3	1.65e-3	1.20e-3
5	32	3.22e-4	4.71e-4	4.25e-4	3.01e-4
6	64	1.39e-4	1.02e-4	8.68e-5	6.93e-5
7	128	8.02e-5	2.36e-5	1.99e-5	1.73e-5
8	256	4.37e-5	5.81e-6	4.68e-6	4.35e-6

Figure 17. The average deviation error accumulated in the shear deformation velocity field. The velocity field is not linear, therefore the Lagrangian remap is only approximate. Also, such a velocity field amplifies the interface reconstruction errors in the direction of shear. In this situation the superiority of the MoF technique over the VoF competitors becomes more obvious: the MoF error is at least two times lower than any of the VoF errors; in asymptotic regime it is three times lower.



$-\log_2 h$	1/h	LSGQ	LVIRA	Swartz	MoF
4	16	5.22e-3	5.88e-3	3.44e-3	3.05e-3
5	32	1.24e-3	1.51e-3	9.68e-4	4.57e-4
6	64	3.88e-4	3.59e-4	2.02e-4	1.10e-4
7	128	1.30e-4	7.61e-5	8.43e-5	2.48e-5
8	256	6.35e-5	1.65e-5	1.96e-5	5.47e-6

amplify the reconstruction errors. That is why the rotation error graphs look very much like the static error graphs (see Figure 8). The translation error graphs exhibit less similarities with the static results, probably, because of the mesh imprint. Diagonal translation seems to be a tough test for all VoF methods. The Swartz is the only VoF algorithm that can keep pace with the MoF algorithm on the fine grid.

The shear deformation velocity field is not linear, therefore the Lagrangian remap is only approximate. Also, such a velocity field amplifies the interface reconstruction errors in the direction of shear. In this situation the superiority of the MoF technique over the VoF competitors becomes more obvious: the MoF error is at least two times lower than any of the VoF errors; in asymptotic regime it is three times lower.

6 Interface reconstruction in ALE simulations

In a pioneering paper [12] A. Hirt et al. developed the formalism for a grid, whose motion is independent from the fluid motion, and showed that this framework could be used to combine the best properties of Lagrangian and Eulerian methods. This class of methods was called Arbitrary Lagrangian-Eulerian (ALE). Many authors have described ALE strategies to optimize accuracy, robustness, and computational efficiency [4, 17, 14, 15, 25, 16].

It is possible to formulate the ALE scheme as a single algorithm [9] by solving the governing equations in a moving coordinate frame. The simulation cycle of typical ALE scheme usually consists of three separate stages. These are: 1) a Lagrangian stage, in which the solution and grid are updated; 2) a rezoning stage, in which the nodes of the computational grid are moved to a more optimal position; and 3) a remapping stage, in which the Lagrangian solution is interpolated onto the rezoned grid.

The initial mesh in multi-material flows is usually aligned with material interface, that is, each mesh cell contains only one material. For simple flows, it is possible to keep interface aligned with the mesh, that is do not move the interface nodes at all or move them along the interface. With a strong shear deformation, it is not possible to keep the material interfaces Lagrangian, and, therefore, the material interfaces require special treatment. In ALE methods for multi-material compressible flows interface reconstruction is needed in rezone/remap stage.

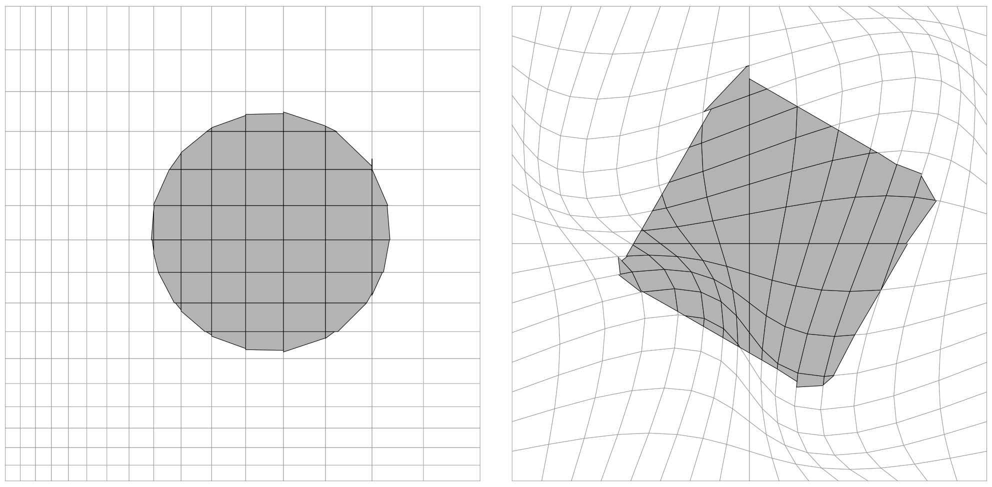


Figure 18. The examples of the logically-rectangular grids obtained with map I (left) and map II (right). One can also see the respective MoF reconstructions of the circular and square interfaces.

6.1 Convergence study

To simulate an ALE environment, we consider a sequence of computational grids, obtained by deformation of initial uniform rectangular grid. Namely, given a uniform rectangular grid Ω_h in the unit-square domain $\Omega =]0,\ 1[\times]0,\ 1[$, and a uniparametric family $\{F_t\}_{0 \leqslant t \leqslant 1}$ of plane transformations with positive Jacobian

$$F_t : \Omega \longrightarrow \Omega, \qquad J(F_t) > 0, \qquad 0 \leqslant t \leqslant 1;$$

the n+1 logically rectangular meshes generated are

$$\Omega_{h,k} = F_{kh}(\Omega_h), \qquad k = \overline{0, n},$$

where $h = 1/n$ is the mesh spacing of the uniform grid Ω_h. We successively remap the material moments from grid $\Omega_{h,k-1}$ to grid $\Omega_{h,k},\ k = \overline{1, n}$, performing the interface reconstruction after each remap. To estimate the quality of the successive reconstructions, we measure the average deviations of the final material interface from the initial one.

We conducted the experiments with the same two test shapes used the in static tests: the circle and the square. The two types of the plane transformations used are:

$$F_t^{(I)}(x,\ y) = \big((1 + \alpha_t(x^2 - 1))x,\ (1 + \alpha_t(y - 1))y\big),$$

where

$$\alpha_t = \sin{(2\pi t)}/2;$$

and

$$F_t^{(II)}(x,\ y) = \big(x + \alpha_t \sin(2\pi x) \sin(2\pi y),\ y + \alpha_t \sin(2\pi x) \sin(2\pi y)\big),$$

where

$$\alpha_t = \begin{cases} t/5, & t \leqslant 1/2, \\ (1-t)/5, & t > 1/2. \end{cases}$$

Figure 18 shows the deformation patterns due to such maps.

The average deviations measured for different mesh resolutions are collected in tables and put on the graphs. The results for the circular shape are presented on Figure 19, and the results for the square shape are presented on Figure 20.

As one can see, the circular-shape results are similar to those measured in the respective static tests. The MoF algorithm shows the best results, except for the sequence of sufficiently fine grids under deformation II, where it looses to the Swartz algorithm. We attribute this fact to the uniform curvature of the true interface, which creates certain preferences to the VoF algorithms; at the same time the interface diffusion introduced by the remap step alleviates the advantage of the MoF algorithm.

The results of the square-shape test show definite superiority of the MoF algorithm. Note that the 2nd-order algorithms are unable to demonstrate 2nd-order convergence with a non-smooth interface; the reason for this is the inevitable diffusion of the sharp interface features due to successive remapping.

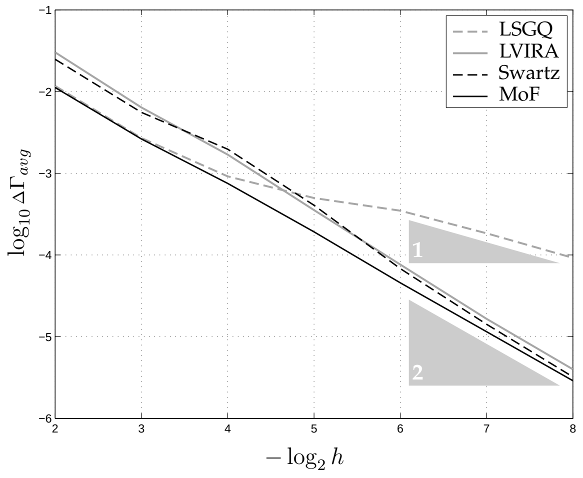
7 Conclusions

We developed a new Moment-of-Fluid interface reconstruction method, which is a generalization of the Volume-of-Fluid technique, presented a detailed description of the MoF algorithm in 2D, and conducted an extensive numerical testing of the method. Compared to traditional VoF methods, the MoF reconstruction shows higher accuracy and resolution of the interface details. Beyond being superior in terms of accuracy, the MoF approach to the interface reconstruction has a number of indisputable technological advances:

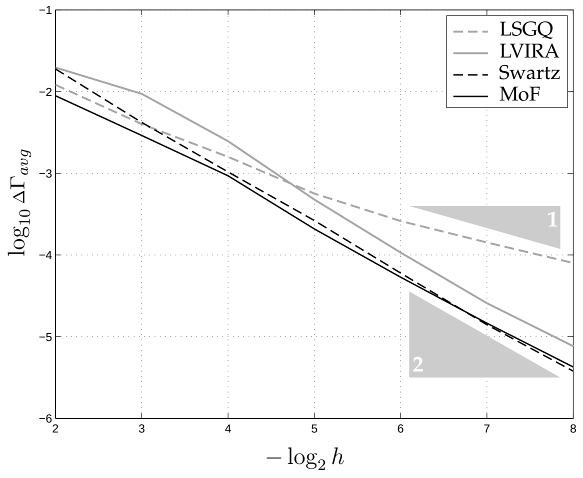
- its variational formulation is dimension-independent, therefore a 3D implementation is just a technical issue;
- it is completely autonomous, i.e. does not require any information from the adjacent cells, which means that
    - the algorithm can be efficiently implemented as a black-box routine;
    - internal and boundary mixed cells can be processed uniformly.

Figure 19. The average deviations between the true interface and its final reconstruction measured in a series of remap-reconstruction steps. The true interface is given by a circle of radius R =*0.25*. The graph and table on the left correspond to the sequence of the logically rectangular meshes generated with map I, those on the right correspond to the sequence obtained with map II. As one can see, the MoF algorithm shows the best results, except for the sequence of sufficiently fine grids under deformation II, where it looses to the Swartz algorithm. We attribute this fact to the uniform curvature of the true interface, which creates certain preferences to the VoF algorithms; the interface diffusion introduced by the remap step alleviates the advantage of the MoF algorithm.

map I



map II

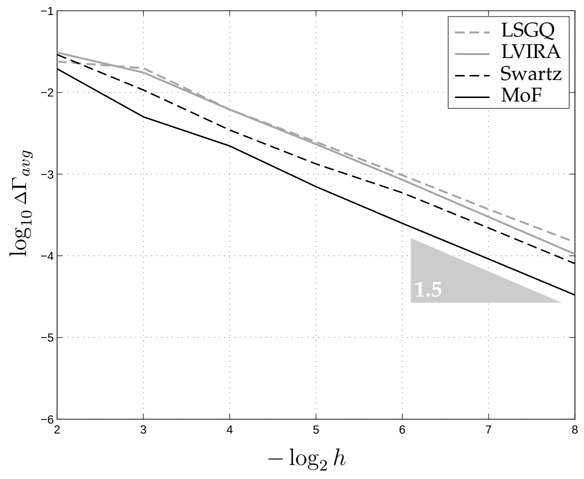


-log$_2h$	1/h	LSGQ	LVIRA	Swartz	MoF
2	4	1.17e-2	3.02e-2	2.48e-2	1.12e-2
3	8	2.70e-3	6.40e-3	5.57e-3	2.63e-3
4	16	9.17e-4	1.69e-3	1.97e-3	7.51e-4
5	32	4.98e-4	3.54e-4	4.06e-4	1.92e-4
6	64	3.48e-4	7.63e-5	6.82e-5	4.56e-5
7	128	1.84e-4	1.65e-5	1.42e-5	1.15e-5
8	256	9.21e-5	3.98e-6	3.22e-6	2.89e-6

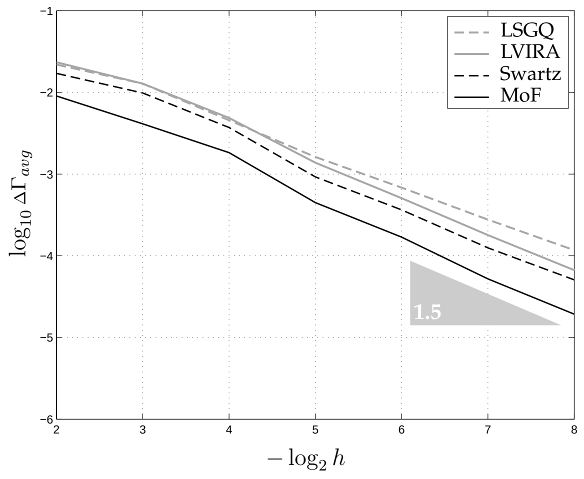
-log$_2h$	1/h	LSGQ	LVIRA	Swartz	MoF
2	4	1.21e-2	1.97e-2	1.89e-2	8.88e-3
3	8	4.00e-3	9.38e-3	4.19e-3	2.90e-3
4	16	1.59e-3	2.47e-3	1.04e-3	9.32e-4
5	32	5.64e-4	4.77e-4	2.64e-4	2.08e-4
6	64	2.60e-4	1.07e-4	6.01e-5	5.35e-5
7	128	1.42e-4	2.57e-5	1.40e-5	1.46e-5
8	256	7.98e-5	7.65e-6	3.78e-6	4.26e-6

Figure 20. The average deviations between the true interface and its final reconstruction measured in a series of remap-reconstruction steps. The true interface is given by a *0.5* $\times$ *0.5* square. The graph and table on the left correspond to the sequence of the logically rectangular meshes generated with map I, those on the right correspond to the sequence obtained with map II. The results of the square-shape test show definite superiority of the MoF algorithm. Note that the 2nd-order algorithms are unable to demonstrate 2nd-order convergence with a non-smooth interface; the reason for this is the inevitable diffusion of the sharp interface features due to successive remapping.

map I



map II



-$\log_2 h$	1/h	LSGQ	LVIRA	Swartz	MoF
2	4	2.38e-2	3.07e-2	2.90e-2	1.94e-2
3	8	1.98e-2	1.75e-2	1.07e-2	5.02e-3
4	16	6.17e-3	6.17e-3	3.46e-3	2.21e-3
5	32	2.48e-3	2.32e-3	1.33e-3	7.02e-4
6	64	9.70e-4	8.53e-4	5.91e-4	2.50e-4
7	128	3.71e-4	2.99e-4	2.19e-4	9.15e-5
8	256	1.47e-4	1.05e-4	8.02e-5	3.31e-5

-$\log_2 h$	1/h	LSGQ	LVIRA	Swartz	MoF
2	4	2.19e-2	2.34e-2	1.71e-2	9.03e-3
3	8	1.28e-2	1.28e-2	9.84e-3	4.12e-3
4	16	4.56e-3	4.87e-3	3.72e-3	1.83e-3
5	32	1.62e-3	1.38e-3	9.23e-4	4.48e-4
6	64	6.83e-4	5.08e-4	3.65e-4	1.69e-4
7	128	2.77e-4	1.79e-4	1.25e-4	5.21e-5
8	256	1.17e-4	6.66e-5	5.07e-5	1.93e-5

In terms of performance, MoF method is at least as fast as a typical 2nd-order accurate VoF method.

We also presented a feasible way to update the cell-wise material moment data in Eulerian simulations by means of the Lagrangian remap. Experiments with analytical solenoidal velocity fields confirm the high quality of the interface tracking attained with the MoF method.

8 Acknowledgment

This work was carried out under the auspices of the National Nuclear Security Administration of the U.S. Department of Energy at the Los Alamos National Laboratory under contracts W-7405-ENG-36 and DE-AC52-06NA25396.

Authors would like to thank Blair Swartz, Rao Garimella, Konstantin Lipnikov, Yuri Vassilevskiy, James Mac Hyman, Martin Staley, William Rider, Robert Lowrie, Douglas Kothe, Marianne Francois, Stewart Mosso, Tariq Aslam, and David Benson for useful discussions and valuable comments.

Authors also acknowledge the support of the Advanced Simulation and Computing (ASC) and the Laboratory Directed Research & Development (LDRD) programs at the Los Alamos National Laboratory.

References

[1] E. Aulisa, S. Manservisi, and R. Scardovelli. A surface marker algorithm coupled to an area-preserving marker redistribution method for three-dimensional interface tracking. *Journal of Computational Physics*, 197(2):555–584, Jul 2004.

[2] D. Bailey, S. Brown, and G. Zimmerman. Interface reconstruction and subzone physics models. Technical Report UCRL-CONF-214875, Lawrence Livermore National Laboratory, Livermore, CA, 2005.

[3] Timothy J. Barth. Numerical methods for gasdynamic systems on unstructured meshes. In D. Kröner, M Ohlberger, and C. Rohde, editors, *An Introduction to Recent Developments in Theory and Numerics for Conservation Laws. Proceedings of the International School on Theory and Numerics for Conservation Laws, Freiburg/Littenweiler, October 20–24, 1997*, Lecture Notes in Computational Science and Engineering, pages 271–273. Springer-Verlag, Berlin Heidelberg, 1999.

[4] David J. Benson. Computational methods in lagrangian and eulerian hydrocodes. *Computer Methods in Applied Mechanics and Engineering*, 99:335–394, 1992.

[5] David J. Benson. Eulerian finite element methods for micromechanics of heterogeneous materials: Dynamic prioritization of material interfaces. *Computer Methods in Applied Mechanics and Engineering*, 151(3–4):343–360, Jan 1998.

[6] David J. Benson. Volume of fluid interface reconstruction methods for multimaterial problems. *Applied Mechanics Reviews*, 55(2):151–165, Mar 2002.

[7] J. Frédéric Bonnans, J. Charles Gilbert, Claude Lemaréchal, and Claudia A. Sagastizábal. *Numerical Optimization. Theoretical and Practical Aspects.* Springer-Verlag, Berlin Heidelberg, 2003.

[8] R. DeBar. Fundamentals of the KRAKEN code. Technical Report UCID-17366, Lawrence Livermore National Laboratory, Livermore, CA, 1974.

[9] J. Donea, S. Giuliani, and J.P. Halleux. An arbitrary lagrangian-eulerian finite element method for transient dynamic fluid-structure interactions,. *Computer Methods in Applied Mechanics and Engineering*, 33:689–723, 1982.

[10] Vadim Dyadechko. Construction of the linear interface from the moment data (tentative). In preparation.

[11] J. Glimm, J. W. Grove, X. L. Li, K. M. Shyue, Y. N. Zeng, and Q. Zhang. Three-dimensional front tracking. *SIAM Journal on Scientific Computing*, 19(3):703–27, May 1998.

[12] C. Hirt, A. Amsden, and J. Cook. An arbitrary lagrangian-eulerian computing method for all flow speeds. *Journal of Computational Physics*, 14:227–253, 1974.

[13] C. W. Hirt and B. D. Nichols. Volume of fluid (VOF) method for the dynamics of free boundaries. *Journal of Computational Physics*, 39(1):201–25, Jan 1981.

[14] D.S. Kershaw, M.K. Prasad, M.J. Shaw, and Milovich J.L. 3d unstructured mesh ale hydrodynamics with the upwind discontinuous finite element method. *Computer Methods in Applied Mechanics and Engineering*, 158:81–116, 1998.

[15] P. Kjellgren and J. Hyvarien. An arbitrary lagrangian-eulerian finite element method. *Computational Mechanics*, 21(2):81–90, 1998.

[16] H.U. Mair. Review: Hydrocodes for structural response to underwater explosions. 6(2):81–96, 1999.

[17] L. Margolin. Introduction to ”an arbitrary lagrangian-eulerian computing method for all flow speeds”. *Journal of Computational Physics*, 135:198–202, 1997.

[18] W. H. McMaster and E.Y. Gong. User’s manual for pele-ic: A computer code for eulerian hydrodynamics. Technical Report UCRL-52609, Lawrence Livermore National Laboratory, Livermore, CA, 1979.

[19] S. J. Mosso, B. K. Swartz, D. B. Kothe, and R. C. Ferrell. A parallel, volume-tracking algorithm for unstructured meshes. In P. Schiano, A. Ecer, J. Periaux, and N. Satofuka, editors, *Parallel Computational Fluid Dynamics: Algorithms and Results Using Advanced Computers*, pages 368–375. Elsevier Science, 1997.

[20] Stewart Mosso and Sean Clancy. A geometrically derived priority system for Young’s interface reconstruction. Technical Report LA-CP-95-0081, Los Alamos National Laboratory, Los Alamos, NM, 1995.

[21] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer-Verlag, New York, 1999.

[22] J. O’Rourke. *Computational Geometry in C*. Cambridge University Press, second edition, 1988.

[23] J. O’Rourke, C.-B. Chien, T. Olson, and D. Naddor. A new linear algorithm for intersecting convex polygons. *Compute. Graph. Image Process.*, 19:384–391, 1982.

[24] S. Osher and R. P. Fedkiw. Level set methods: An overview and some recent results. *Journal of Computational Physics*, 169(2):463–502, May 2001.

[25] J.S. Peery and D.E. Carrol. Multi-material ale methods in unstructured grids. *Computer Methods in Applied Mechanics and Engineering*, 187:591–619, 2000.

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

[27] E. G. Puckett. A volume-of-fluid interface tracking algorithm with applications to computing shock wave refraction. In H. Dwyer, editor, *Proceedings of the 4th International Symposium on Computational Fluid Dynamics*, pages 933–938, Davis, CA, 1991.

[28] E. G. Puckett and J. S. Saltzman. A 3D adaptive mesh refinement algorithm for multimaterial gas dynamics. *Physica D*, 60(1–4):84–93, Nov 1992.

[29] William J. Rider and Douglas B. Kothe. Reconstructing volume tracking. *Journal of Computational Physics*, 121(2):112–152, Apr 1998.

[30] M. Rudman. Volume tracking methods for interfacial flow calculations. *ijnmf*, 24:671–691, 1997.

[31] R. Scardovelli and S. Zaleski. Direct numerical simulation of free-surface and interfacial flow. *Annual Review of Fluid Mechanics*, 31:567–603, 1999.

[32] Martin Staley. CORE: Conservative remapper. Technical Report LA-UR-04-8104, Los Alamos National Laboratory, Los Alamos, NM, 2004.

[33] M. Sussman, 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*, 148:81–124, 1999.

[34] M. Sussman, E. Fatemi, P. Smereka, and S. Osher. Improved level set method for incompressible two-phase flows. *Computers & Fluids*, 27(5–6):663–680, Jun–Jul 1998.

[35] Blair Swartz. The second-order sharpening of blurred smooth borders. *Mathematics of Computation*, 52(186):675–714, Apr 1989.

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

[37] S. O. Unverdi and G. Tryggvason. A front-tracking method for viscous, incompressible, multi-fluid flows. *Journal of Computational Physics*, 100(1):25–37, May 1992.

[38] D. L. Youngs. Time-dependent multi-material flow with large fluid distortion. In K. W. Morton and M. J. Baines, editors, *Numerical Methods for Fluid Dynamics*, pages 273–285. Academic Press, 1982.

[39] D. L. Youngs. An interface tracking method for a 3D Eulerian hydrodynamics code. Technical Report 44/92/35, AWRE, 1984.

A Calculating the polygon moments

The area and centroid of the polygon P_n with vertices $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathbb{R}^2$ (in counter-clockwise order) can be evaluated by formulas

$$|P_n| = \frac{1}{2}\sum_{i=1}^{n}[\,(\mathbf{x}_i - \mathbf{x}_0)\times(\mathbf{x}_{i+1} - \mathbf{x}_0)\,],$$

$$\mathbf{x}_c(P_n) = \mathbf{x}_0 \;+\; \frac{1}{6\,|P_n|}\sum_{i=1}^{n}[\,(\mathbf{x}_i - \mathbf{x}_0)\times(\mathbf{x}_{i+1} - \mathbf{x}_0)\,]\big((\mathbf{x}_i - \mathbf{x}_0) + (\mathbf{x}_{i+1} - \mathbf{x}_0)\big),$$

where $[\,\cdot\times\cdot\,]$ is *2D vector product*, $\mathbf{x}_{n+1} \equiv \mathbf{x}_1$, and $\mathbf{x}_0 \in \mathbb{R}^2$ is some point of the plane explained below.

The presence of parameter $\mathbf{x}_0 \in \mathbb{R}^2$ in the formulas above may seem redundant, since in exact arithmetic the formulas are true for an arbitrary $\mathbf{x}_0$. The sole purpose of introducing $\mathbf{x}_0$ is to improve the accuracy of the floating point calculations; i.e. the right choice of $\mathbf{x}_0$ helps to minimize the round-off errors. It is unwise to set $\mathbf{x}_0 = (0,0)$, since it leads to high round-off errors for remote polygons. For stable results $\mathbf{x}_0$ should be local to the polygon; if the polygon P_n is of regular shape (convex or moderately non-convex and also not highly anisotropic), it is safe to set $\mathbf{x}_0$ to one of the vertices, say $\mathbf{x}_1$:

$$|P_n| = \frac{1}{2}\sum_{i=2}^{n-1}[\,(\mathbf{x}_i - \mathbf{x}_1)\times(\mathbf{x}_{i+1} - \mathbf{x}_1)\,], \tag{18}$$

$$\mathbf{x}_c(P_n) = \mathbf{x}_1 \;+\; \frac{1}{6\,|P_n|}\sum_{i=2}^{n-1}[\,(\mathbf{x}_i - \mathbf{x}_1)\times(\mathbf{x}_{i+1} - \mathbf{x}_1)\,]\big((\mathbf{x}_i - \mathbf{x}_1) + (\mathbf{x}_{i+1} - \mathbf{x}_1)\big). \tag{19}$$

B Numerical intersection of plane figures

The most common class of the plane figures, we have to deal with in the context of the VoF/MoF context, are convex polygons. Indeed, if a mesh cells are convex (which is very typical), its linear subcells are also convex, and its Lagrangian preimage for a small Δt is highly likely to be convex too.

Each convex polygon is the intersection of a finite number of half-planes. Most straightforward way to intersect two polygons is successive clipping of one of them against all the constitutive half-planes of the other. This strategy has quadratic complexity in the number of the polygon vertices.

It is possible to find the intersection of two convex polygons in linear time with the Chasing Algorithm [23, 22]. Although it is asymptotically faster than the straightforward Clipping Algorithm, the Chasing Algorithm works only with integer coordinates (the floating point coordinates has to be affine-transformed and casted to integers first) and is not competitive for polygons with a small number of vertices. We find that a properly implemented Clipping Algorithm is more suitable for our purposes. In our numerical experiments we used the convex-polygon-intersection routine from the COnservative REmapper (CORE) library by M. Staley [32].

Occasionally we have to intersect the figures that are not convex polygons. Since

- each plane figure can be approximated with arbitrary accuracy by a polygon, and
- a non-convex polygon can always be represented as a union of non-overlapping convex polygons (be triangulated, for example),

then, using a convex-polygon-intersection routine, one can always find a set of non-overlapping convex polygons, union of which constitute the intersection of the figures. For us, the sole purpose of finding the intersection is evaluation of its moments, therefore the strategy above works just fine.

C Smooth univariate optimization routine

The smooth univariate optimization procedure we use is based on the recommendations from [21] and [7] and is a cut version of the line search presented in [**?**]. It takes four input parameters:

1) an *initial guess* ϕ_0,
2) a *trial step increment* $\triangle\phi > 0$,
3) an *argument tolerance* $tol_\phi \geqslant 0$,
4) a *first derivative tolerance* $tol_{f'} \geqslant 0$;

and returns the iterate ϕ_i that is considered to be a good approximation to the local minimum ϕ^* of the objective function $f(\phi)$, namely, the first ϕ_i that satisfies either of two conditions:

$$|f'(\phi_i)| \leqslant tol_{f'} \quad \text{or} \quad |\phi_i - \phi^*| \leqslant tol_\phi. \tag{20}$$

Clearly, if $|f'(\phi_0)| < tol_{f'}$, then
no search is required, and ϕ_0 is returned;

otherwise,

we proceed with the following logical steps:

1) First, we identify the descending direction of $f(\phi)$.
   If $f'(\phi_0) > 0,$ then
   the positive direction of the polar angle is altered from counter-clockwise to clockwise to ensure that $f(\phi)$ is decreasing at $\phi = \phi_0$.

2) Second, we bracket the local minimum.
   For this we generate a monotonically increasing arithmetic progression

$$\phi_j = \phi_0 + \Delta\phi \cdot j, \quad j = \overline{1, i},$$

until either a good approximation to the local minimum is found:

$$|f'(\phi_i)| \leqslant tol_{f'},$$

or the objective function growth is detected:

$$f(\phi_i) \geqslant f(\phi_{i\text{-}1}) \quad \text{or} \quad f'(\phi_i) \geqslant 0.$$

If the former condition terminates the loop, then
the search is over, and ϕ_i is returned.

The latter conditions guarantees that the last interval $[\phi_{i\text{-}1}, \phi_i]$ contains a local minimum.

3a) Finally, we find the local minimum inside $[\phi_{i\text{-}1}, \phi_i]$. The minimum is localized with recursive subdivisions, effectively conducted by the Zoom Algorithm [21]. The Zoom Algorithm introduces a new iterate inside the given interval; if the new iterate provides a good approximation to the local minimum, the search is over; otherwise the algorithm is recursively applied to the subinterval known to contain a local minimum.

It is convenient to order the end points of the local minimum interval by the value of the objective function; they are labeled ϕ_{lo} and ϕ_{hi} such that

$$f(\phi_{lo}) \leqslant f(\phi_{hi}) \quad \text{and} \quad f'(\phi_{lo})(\phi_{hi} - \phi_{lo}) < 0; \tag{21}$$

for the start, ϕ_{lo} and ϕ_{hi} are set as follows:

$$(\phi_{lo},\ \phi_{hi}) \leftarrow \begin{cases} (\phi_{i\text{-}1},\ \phi_i), & \text{if } f(\phi_{i\text{-}1}) \leqslant f(\phi_i), \\ (\phi_i,\ \phi_{i\text{-}1}), & \text{otherwise.} \end{cases}$$

3b) If $|\phi_{hi} - \phi_{lo}| < tol_\phi$, then the second termination condition (20) is satisfied, the search is over, and ϕ_{lo} is returned.

A new iterate ϕ_{i+1} is defined as a local minimum of the cubic polynomial interpolating the values and the first derivatives of the objective function at the end points. As long as conditions (21) hold true, such a local minimum exists and is located between ϕ_{lo} and ϕ_{hi}:

$$\phi_{i+1} = \phi_{hi} - (\phi_{hi} - \phi_{lo}) \frac{f'(\phi_{hi}) + d_2 - d_1}{f(\phi_{hi}) - f(\phi_{lo}) + 2d_2},$$

where

$$d_1 = f'(\phi_{lo}) + f'(\phi_{hi}) - 3\frac{f(\phi_{lo}) - f(\phi_{hi})}{\phi_{lo} - \phi_{hi}},$$

$$d_2 = \operatorname{sign}(\phi_{hi} - \phi_{lo})\sqrt{d_1^2 - f'(\phi_{lo})f'(\phi_{hi})}.$$

If the new iterate ϕ_{i+1} is too close to ϕ_{lo}, there is a potential danger for the iterations to fall into stagnation loop [7]; to avoid this, we use the corrected iterate

$$\phi_{i+1} \leftarrow \phi_{lo} + (\phi_{hi} - \phi_{lo})\, tol_{\phi_{lo}}$$

whenever

$$(\phi_{i+1} - \phi_{lo})/(\phi_{hi} - \phi_{lo}) < tol_{\phi_{lo}};$$

we find that the value of $tol_{\phi_{lo}} = 0.05$ serves the purpose quite well.

If $|f'(\phi_{i+1})| \leqslant tol_{f'}$, then
the search is over, and ϕ_{i+1} is returned.

The new iterate ϕ_{i+1} partitions the interval in two subintervals. Which one should we pick to continue the recursion? We have to guarantee that the next subinterval contains a local minimum, i.e. satisfies invariants (21), and also ensure that the new ϕ_{lo}, among all the iterates generated so far, gives the smallest value to the objective function:

$$f(\phi_{lo}) \leqslant f(\phi_j), \quad j = \overline{0, i+1}.$$

Therefore, if $f(\phi_{lo}) < f(\phi_{i+1})$, then ϕ_{i+1} replaces ϕ_{hi}:

$$(\phi_{lo},\ \phi_{hi}) \leftarrow (\phi_{lo},\ \phi_{i+1});$$

else if $f'(\phi_{i+1})(\phi_{hi} - \phi_{lo}) < 0$, then ϕ_{i+1} replaces ϕ_{lo}:

$$(\phi_{lo},\ \phi_{hi}) \leftarrow (\phi_{i+1},\ \phi_{hi});$$

otherwise we put

$$(\phi_{lo},\ \phi_{hi}) \leftarrow (\phi_{i+1},\ \phi_{lo}).$$

Update the iterate counter $i \leftarrow i + 1$ and repeat the step 3b.

The default input parameters of the optimization routine are

2) $\triangle\phi = \pi/(2n)$, where n is the number of the mixed cell vertices,

3) $tol_\phi = \mathit{10}^{\mathit{-6}}$, and

4) $tol_{f'} = \mathit{0}$.

We deliberately set $tol_{f'} = \mathit{0}$ to ensure that the iterations stop only when the polar angle of the interface normal stabilizes within $tol_\phi = \mathit{10}^{\mathit{-6}}$, so that the MoF and LVIRA algorithms, relying on this optimization routine, terminate under the same condition as the Swartz iterations.

D Implementation details of the VoF-PLIC interface reconstruction algorithms

It is true to say that the behavior of the interface reconstruction algorithms varies widely with the implementation details, many of which are left behind the general descriptions. In order to make our results meaningful and reproducible, we decided to highlight the important details about our implementations of the VoF-PLIC algorithms.

All VoF-PLIC algorithms need the material volume data from the adjacent cells to evaluate the interface normal. We consider two cells to be adjacent if they share either an edge or a vertex.

LSGQ algorithm. In [3] T. J. Barth presents the Linear Least Square algorithm for evaluating the gradient of discrete cell-centered function. This algorithm is transparently usable for evaluating the mixed-cell-interface normals from the material volume fractions. Since the original name of the algorithm (Linear Least Squares) is too general, and there is yet another VoF-PLIC algorithm based on the least square technique (LVIRA), we took the liberty to rename the Barth's algorithm to the Least Square Gradient (LSG) algorithm to avoid confusion. The LSGQ is just a variation of the LSG algorithm that uses reciprocal Quadratic distance weights. See [3] for details.

LVIRA algorithm. Our implementation of LVIRA algorithm follows the description from [26]. Let Ω_i be a mixed cell of interest surrounded by K neighbors $\Omega_{i_k},\ k = \overline{\mathit{1}, K}$; the volume fractions of material A inside these cells are μ_i^* and $\mu_{i_k}^*,\ k = \overline{\mathit{1}, K}$ respectively. By $\omega_i(\phi)$ we denote the linear subcell of Ω_i, whose volume matches the true one:

$$|\,\omega_i(\phi)| \equiv \mu_i^* |\,\Omega_i|,$$

and whose interface normal is given by the polar angle ϕ:

$$\mathbf{n}(\omega_i(\phi)) \equiv (\cos\phi, \sin\phi).$$

The normal of LVIRA interface reconstruction in Ω_i is given by the minimizer of the objective function

$$f(\phi) = \sum_{k=1}^{K} \big(|\omega_{i_k}(\phi)| - \mu^*_{i_k}|\Omega_{i_k}|\big)^2, \tag{22}$$

where $\omega_{i_k}(\phi),\ k = \overline{1, K}$ are the linear subcells cut off from the neighbors by the linear extension of the interface $\Gamma(\omega_i(\phi))$.

The first derivative of LVIRA objective function (22) is given by:

$$f'(\phi) = \sum_{k=1}^{K} (|\omega_{i_k}(\phi)| - \mu^*_{i,k}|\Omega_{i_k}|)|\Gamma_{i_k}(\phi)|\big(\mathbf{t}_\phi \cdot (\mathbf{x}_i(\phi) - \mathbf{x}_{i_k}(\phi))\big), \tag{23}$$

where $\Gamma_{i_k}(\phi) = \Gamma(\omega_{i_k}(\phi))$ is the interface of $\omega_{i_k}(\phi)$, $|\Gamma_{i_k}(\phi)|$ is its length, and $\mathbf{x}_{i_k}(\phi)$ is its centroid; $\mathbf{x}_i(\phi)$ is the centroids of $\Gamma_i(\phi) \equiv \Gamma(\omega_i(\phi))$, and $\mathbf{t}_\phi = (-\sin\phi, \cos\phi)$ is counter-clockwise unit tangent on $\Gamma_i(\phi)$.

To get an unbiased performance comparison, we minimized the LVIRA objective function (22) with the same smooth univariate optimization routine we used for the MoF algorithm (see Appendix C); for the initial guess we used the polar angle of the normal given by the LSGQ reconstruction.

Swartz algorithm. If two cells are separable by a line (which is always true for the non-overlapping convex cells), then for any given pair of volume fractions there exists a *common linear interface* that cuts off the subcells of the specified volume fractions from the respective cells. Whenever the volume fractions are given by a twice-differentiable true interface, such a common linear interface is 2nd-order accurate [35], which makes it a very useful tool in material interface reconstruction.

If the computational grid is rectangular, a common linear interface for a pair of adjacent mixed cell can be found analytically [8]; even if the grid is not rectangular, a common linear interface can be identified by means of the quadratically convergent iterative procedure [35]:

Algorithm 2 (Swartz iterations). *Given initial VoF-PLIC interface approximations in two mixed cells,*

1) *connect the centroids of the interfaces with a line segment* [5]*; the normal to this "bridge" segment will serve as a new iterate of the common linear interface normal;*

[5] If the linear interface consists of a single segment, which is always the case for a convex cell, the interface centroid is given by its median.

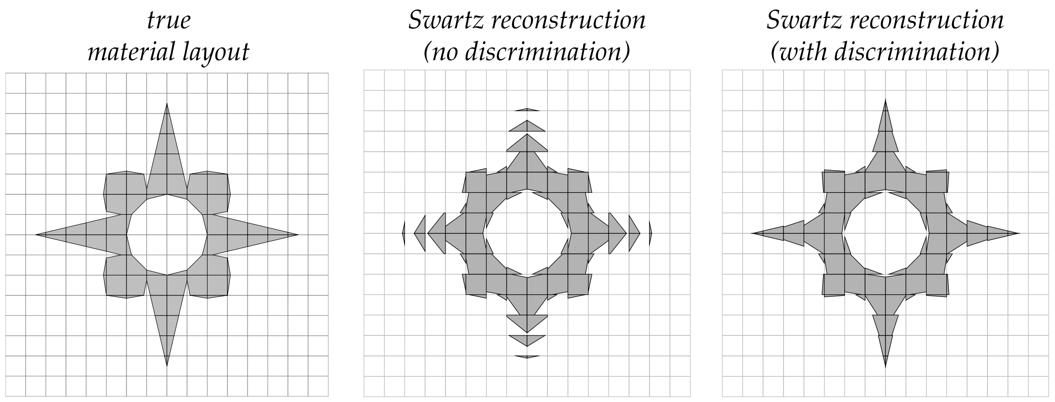


Figure 21. *The effect of the discrimination strategy on the quality of Swartz reconstruction.*

2) *build the new VoF-PLIC interfaces in both cells using the "bridge" normal;*

3) *repeat steps 1) and 2) until the "bridge" normal stabilizes within some small tolerance (we used $tol_\phi = 10^{-6}$).*

The Swartz interface reconstruction routine we implemented, consists of three steps:

1) First we perform the LSGQ reconstruction to get the initial VoF-PLIC approximation.

2) Next we use the Swartz iterations to find the common linear interface for each pair of adjacent mixed cells. There is an important technical detail about the implementation of the Swartz iterations we would like to point out here.

   With two choices of the normal on the "bridge" segment, we use the one that has a positive projection on the previous iterate; for the very first iteration we use the one that has positive projections on both initial normals.

   Clearly, if the initial normals differ significantly, then the true interface is unlikely to have low curvature in this region, and the common linear interface is unlikely to approximate the true interface better than the initial ones; so there is no need to go for it. Therefore, we do not try to find the common linear interface for a pair of adjacent mixed cells at all, unless the initial outward normals differ by less than *45* degrees. This way the common linear interface is determined only for those pairs of mixed cells that cover the low-curvature segments of the true interface. Such a *discrimination strategy* significantly improves the ability of the algorithm to resolve small interface details (see the example on Figure 21).

3) Finally, we update the mixed cell interfaces. The new interface normal in each mixed cell is defined as the arithmetic average of the unit normals due to the all direct mixed neighbors.

The algorithm above is different from the Mosso-Swartz algorithm presented in [19]. The original Mosso-Swartz algorithm prescribes to loop through the list of mixed cells, updating the interface normals by the average of the "bridge" unit normals due to all mixed neighbors. The loop is terminated, when all the interface normals are stabilized within a given tolerance. By means of numerical experiment we established that the Mosso-Swartz algorithm tends to oversmooth the material interfaces and lacks the sharpness of the Swartz algorithm.