Journal of Computational Physics 275 (2014) 53–64

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

An interface reconstruction method based on analytical formulae for 2D planar and axisymmetric arbitrary convex cells

S. Diot [a,*], M.M. François [a], E.D. Dendy [b]

[a] Fluid Dynamics and Solid Mechanics (T-3), Los Alamos National Laboratory, Los Alamos, NM 87545, USA
[b] Computational Physics and Methods (CCS-2), Los Alamos National Laboratory, Los Alamos, NM 87545, USA

ARTICLE INFO

Article history:
Received 26 November 2013
Received in revised form 25 June 2014
Accepted 30 June 2014
Available online 8 July 2014

Keywords:
Interface reconstruction
Non-iterative
PLIC
Volume-of-fluid (VOF)
Polygonal cells

ABSTRACT

In this paper, we propose a non-iterative interface reconstruction method for 2D planar and axisymmetric geometries that is valid for arbitrary convex cells and intended to be used in multi-material simulation codes with sharp interface treatment for instance. Assuming that the normal vector to the interface is known, we focus on the computation of the line constant so that the polygon resulting from the cell-interface intersection has the requested volume. To this end, we first decompose the cell in trapezoidal elements and then propose a new approach to derive an exact formula for the trapezoids volumes. This formula, derived for both the planar and axisymmetric cases, is used to first bracket and then find the line constant that exactly matches the prescribed volume. The computational efficiency of the proposed method is demonstrated over a large number of reproducible conditions and against two existing methods.

0. Introduction

Volume-tracking methods for multi-material flow simulations have been introduced in the early seventies with three different approaches in [8,11,10]. Their common points are 1) the different materials are represented by their volume fractions, 2) these volume fractions are advected with the flow velocity to follow the materials evolution, 3) a cell-wise interface is reconstructed at each time step from the volume fractions to prevent its numerical diffusion. Many extensions and improvements have been proposed to these methods in the last decades, and nowadays the most widespread volume-tracking method uses a PLIC interface reconstruction technique (Piecewise Linear Interface Calculation) [17,13]. In this method, the material volume fractions equation is geometrically solved by making use of the cell-wise reconstructed linear interface in order to keep the interface sharp during simulation.

In this work, we are interested in the reconstruction process of a cell-wise linear interface that is represented by the line equation

$$n_x x + n_y y - c = 0$$

where $\vec{n} = (n_x, n_y)$ is the unit normal vector to the interface and c its line constant. The reconstruction step classically consists of two steps, firstly the computation of $\vec{n}$ and secondly the computation of the line constant c such that the

[*] Corresponding author.
E-mail address: diot@lanl.gov (S. Diot).

http://dx.doi.org/10.1016/j.jcp.2014.06.060
0021-9991/© 2014 Elsevier Inc. All rights reserved.

interface splits the cell in two sub-cells whose volumes correspond to the material volume fractions. In this work, we only focus on the latter step that we call *volume-matching* step by assuming that the normal vector is known. We remind that $\vec{n}$ is usually computed as the volume fraction gradient by any existing gradient computation technique as instance a least-squares method (*e.g.* (E)LVIRA in [12] or k-exact in [2]) or a height function method [6]. A good review of these techniques can be found in [14].

The purpose of this paper is to propose an original non-iterative technique to the volume-matching problem for any convex polygonal cell. Let us first point out that several methods already exist for particular cell shapes, see [16] for triangles/tetrahedra and [15,7] for squares/hexahedra, and are widely used in simulation codes. However, up to our knowledge, only the two methods presented in [4,9] are available for arbitrary cells in planar geometry while in the axisymmetric geometry only a very recent extension of [4] can be found in [1]. This recentness is most likely the reason why for complex polygonal meshes (*e.g.* Voronoi meshes), the volume-matching technique proposed by Rider and Kothe in [13] is still mainly used in industrial or academic codes. Therefore we will use it as a reference to evaluate the cost of our method in the numerical results section. This volume-matching technique [13] is made of two sub-steps, the first one consists in computing the line constants and the resulting material volumes when lines parallel to the interface pass through all nodes of the cell in order to bracket the interface line constant. Then, once the two closest bounds are found, a Brent's iterative method [3] is used to find the interface line constant that matches the requested volume fraction. This iterative procedure may be very costly since the intersection between the cell and the interface, and the resulting volumes must be computed at each iteration. In the sequel, we use *Brent's method* to refer to the method used by Rider and Kothe in [13].

Let us now point out the differences between the approach we propose and the non-iterative ones of [4,1] and [9]. In [9], an analytical formula is deduced from the polygonal volume formula that uses the cell nodes coordinates. We did not choose this approach as we believe the reconstruction could be done more efficiently. Instead we use the same trapezoidal decomposition of the cell as the one adopted in [4,1] that allows an incremental bracketing of the solution (see Section 1). However our philosophy is to derive an analytical formula that can be used for both bracketing and finding the interface line constant. This is a major difference compared to [4,1], where they use a classical formula for the trapezoidal volumes during the bracketing step and then use an interpolation technique to find the interface line constant. More specific differences will be highlighted in the text when relevant. We claim that our approach leads to a clearer and more robust technique in addition to be less computationally expensive and more accurate volume-matching as demonstrated in the last section of this paper.

Finally let us add that the interface reconstruction technique we propose in this note is devised for two materials and that it could be adapted for more than two materials per cell by assuming a certain configuration (*e.g.* onion-skin or nested dissection [5]) as done with other classical two-material techniques.

The paper is split in three parts. The first part describes how the cell is decomposed. The next one presents the derivation of the analytical formulae for the planar and axisymmetric geometries and to their uses while the last one gathers the numerical results that demonstrate the improved efficiency of our method.

1. Cell decomposition and line constant bracketing

Let us consider an arbitrary convex polygonal cell with N nodes of coordinates $X_l = (x_l, y_l)$, $l = 1, \dots, N$ with l the nodes indexes. We denote by V_{cell} its volume, and by $\alpha \in [0, 1]$ and $\bar{V} = \alpha V_{cell}$ the material volume fraction and associated material volume respectively. We recall that the unit normal vector to the interface $\vec{n} = (n_x, n_y)$ is known and assume that it points from the material under consideration to the other one. We also remind that the volume-matching problem consists in finding the line constant c of the interface that splits the cell according to the given volume $\bar{V}$.

In the sequel, for any given line constant c, we call c-line the line parallel to the interface defined by the equation $n_x x + n_y y - c = 0$. Moreover we call c-volume and denote by V_c the volume of the material polygon resulting from intersection of the c-line with the cell. At last, we assume that for all N faces f_l of the cell, the outward unit normal vector $\vec{n}_{f_l} = (n_{f_l,x}, n_{f_l,y})$ is known and so is the tangential one $\vec{t}_{f_l} = (-n_{f_l,y}, n_{f_l,x})$.

The starting point of our procedure is to decompose the cell. To this end, we follow the same idea as in [4] and start by computing and sorting in increasing order the line constants $\tilde{c}_l = n_x x_l + n_y y_l$ of the lines parallel to the interface and passing through all nodes X_l. The sorted line constants are denoted c_k in the following and are such that $c_1 < c_2 < \dots < c_N$. These N c_k-lines define a natural partition of the cell made of (possibly degenerated) trapezoids, that we further denote T_k, formed by the portions of the cell lying between the c_k- and c_{k+1}-lines. It is important to remark that the volumes V_{c_k} are monotonically increasing with the constant lines c_k, thus ensuring a unique solution to the volume-matching problem. We provide an illustration of the result of this procedure in Fig. 1 in which we have rotated the polygonal cell in order to plot the c_k-lines horizontally.

As in existing methods [13,4], the next step is to bracket the solution line constant which consists of finding the closest lower and upper bounding values c_{lo} and c_{up} in $\{c_1, \dots, c_N\}$ such that $V_{c_{lo}} \leq \bar{V} \leq V_{c_{up}}$. To this end, we compute the increasing c_k-volumes until $\bar{V}$ is bounded. This step is made easier by using the formula derived in next section to compute the volumes $(V_{c_{k+1}} - V_{c_k})$ of the trapezoids T_k and summing them to get the current $V_{c_{k+1}}$. Let us remark that as suggested in [4,9], if α is close to 1, it may be more efficient to apply this procedure to $(1 - \alpha)$ using the opposite of the normal vector.

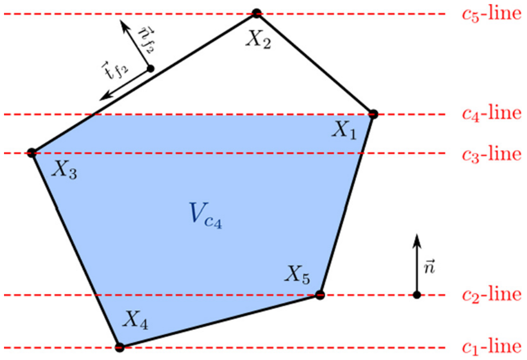

Fig. 1. Illustration of the cell partition defined by the lines parallel to the interface (whose normal vector is $\vec{n}$) and passing through the cell nodes $X_1, \ldots, X_5$. As an example, the outward normal vector $\vec{n}_{f_2}$ to the face f_2 and the tangential one $\vec{t}_{f_2}$ are drawn and the c_4-volume V_{c_4} generated by the c_4-line is also depicted. The cell has been rotated so that the c_k-lines are horizontal.

Once the bounds, c_{lo} and c_{up}, are found the problem reduces to

$$\textbf{Find } c \textbf{ such that } c_{lo} \le c \le c_{up} \textbf{ and } V_c = \bar{V}.$$

Using the volume V_h defined as $(V_c - V_{c_{lo}})$, we rewrite the problem as

$$\textbf{Find } h \textbf{ such that } 0 \le h \le (c_{up} - c_{lo}),\ c = c_{lo} + h \textbf{ and } V_h = (\bar{V} - V_{c_{lo}}).$$

Therefore the problem amounts to finding the unknown h such that the c-line, with $c = (c_{lo} + h)$, is the interface that matches the given volume.

As mentioned in introduction, this approach is similar to the one adopted in [4]. However the formula (derived in next section) that we use to compute the volume of any trapezoid T_k is different from [4] and does not request the computation of the cell cross section by the c_{k+1}-line as pointed out in Section 2.3.

2. Analytical formula for the trapezoidal volumes

We start this section by describing the particular trapezoidal configuration obtained once the cell is decomposed and the bounds found.

The trapezoid is formed by the intersection of the parallel c_{lo}- and c_{up}-lines with exactly two faces of the polygonal cell, denoted f_L and f_R, the subscripts standing for Left and Right, respectively. The side included in the c_{lo}-line is referred to as the basis of the trapezoid. Moreover in order to derive the analytical formula we need to know the coordinates, $X_L = (x_L, y_L)$ and $X_R = (x_R, y_R)$, of the intersection points between the basis and the faces f_L and f_R respectively. We also need the oriented angle θ_L going from the line perpendicular to the basis and passing by X_L to the face f_L and likewise, the angle θ_R going from the line perpendicular to the basis and passing by X_R to the face f_R. We recall that the normal unit vectors for f_L and f_R are known and denoted by $\vec{n}_{f_L}$ and $\vec{n}_{f_R}$ respectively, and the tangential ones by $\vec{t}_{f_L}$ and $\vec{t}_{f_R}$ respectively. At last, we point out that the c-line that we seek, for $c_{lo} \le c \le c_{up}$, is parallel to and lies between the c_{lo}- and c_{up}-lines so that it also forms a similar trapezoid.

An illustration of this configuration is provided in Fig. 2 in which we also show the decomposition of the trapezoid that we will use to obtain the analytical formula. This decomposition is given by the three following elements:

- a **rectangle** formed by the trapezoid basis of length $B = \sqrt{(x_R - x_L)^2 + (y_R - y_L)^2}$ and the unknown height h,
- a **right triangle** on the left defined by the unknown height h and the algebraic length $L_L = h \tan\theta_L$ with angle θ_L being oriented,
- a **right triangle** on the right defined by the unknown height h and the algebraic length $L_R = -h \tan\theta_R$ with angle θ_R being oriented,

for which the areas are respectively denoted by A_B, A_L and A_R and the volumes by V_B, V_L and V_R. This distinction is used in the axisymmetric case since areas and volumes are obviously equal in the planar case.

This decomposition allows to write the trapezoidal volume as the sum of the positive rectangle volume V_B and of the two signed triangles volumes V_L and V_R and still holds for the minimal and maximal values of c_k even though the trapezoid are degenerated into triangles (since $B = 0$ and so $A_B = V_B = 0$). Let us note that as shown in Fig. 3, only four configurations are possible according to the signs of θ_L and θ_R.

Let us point out that the normal vectors to the four sides of the trapezoid are known: the interface normal vector $\vec{n}$ for the parallel basis and the normal to faces f_L and f_R for the two other sides. Therefore the four corner angles of the trapezoid can be computed. In our case, we will only need the tangents of θ_L and θ_R given by

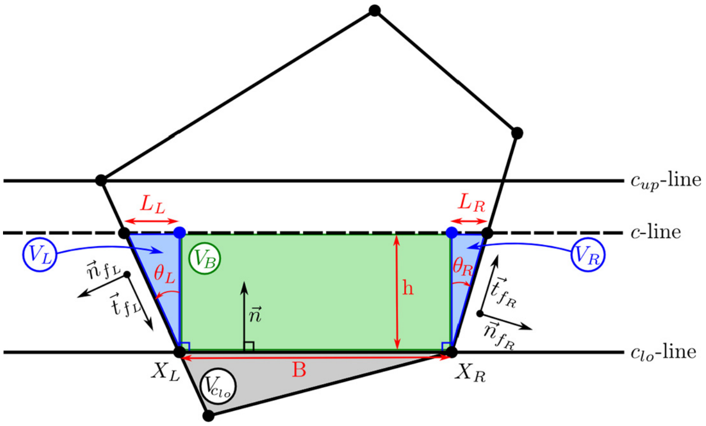

Fig. 2. An example of trapezoidal configuration generated by a given c-line (bounded by c_{lo} and c_{up}) inside a generic convex polygon. The volume of the trapezoid is obtained as the sum of the positive rectangle volume V_B and the two signed triangles volumes V_L, V_R for which the lengths L_L, L_R are algebraic and angles θ_L, θ_R are oriented.

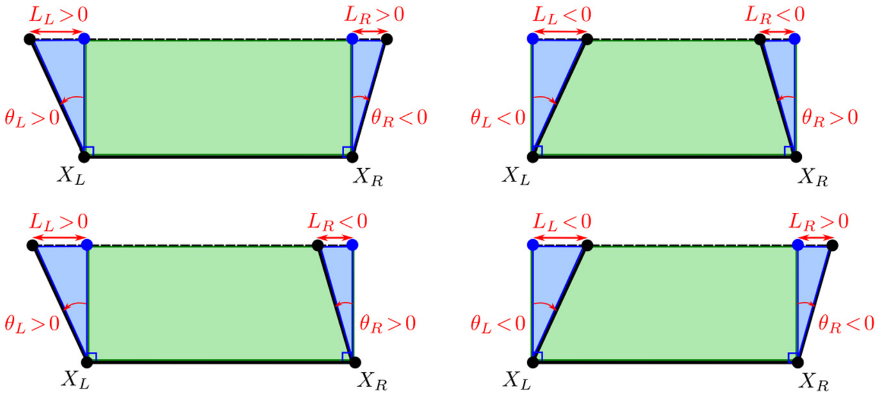

Fig. 3. The four possible trapezoidal configurations depending on the signs of algebraic lengths L_L, L_R and oriented angles θ_L, θ_R. Note that X_L and X_R may be the same point.

$$\tan\theta_L = \frac{\vec{n}\cdot\vec{n}_{f_L}}{\vec{n}\cdot\vec{t}_{f_L}} \quad \text{and} \quad \tan\theta_R = \frac{\vec{n}\cdot\vec{n}_{f_R}}{\vec{n}\cdot\vec{t}_{f_R}}. \tag{1}$$

We now derive the analytical formula for the volume in function of $h = (c - c_{lo})$ that is used during the cell decomposition step to compute the volume when $c = c_k$ and that is *inverted* in order to compute the line constant c of the reconstructed interface. By *inverted*, we mean that we have to compute the value h such that $V(h) = V_h$ that is equivalent to finding the root of polynomials $V(h) - V_h$ of degree two and three for the planar and axisymmetric cases, respectively.

2.1. Formula for the planar geometry

We start with the planar geometry. This is the simplest situation since volumes are not position-dependent. It is thus easy to compute the volumes (*i.e.* areas) of the three elements using basic formulae and obtain the trapezoid volume V as:

$$V(h) = A_B + A_L + A_R = hB + \frac{hL_L}{2} + \frac{hL_R}{2} = hB + h^2\left(\frac{\tan\theta_L - \tan\theta_R}{2}\right). \tag{2}$$

Note that the formula only uses the length of the trapezoid basis B and the tangents of the two oriented angles θ_L and θ_R that we have previously defined in Eq. (1).

2.2. Formula for the axisymmetric geometry

We now consider the axisymmetric geometry. For the sake of simplicity and to avoid confusion in notation, we still use the coordinates (x, y) instead of using the classical (r, z); x is thus the radial coordinate and y the axial one. In this case, volumes are given as the product of the elements area with the x-coordinate of the elements centroid, therefore it is position-dependent and obtaining the formulae is slightly more complex. In order to simplify this step, we consider for each element a change of reference frame. More specifically, we combine a translation and a rotation so that the computation of the centroids position is trivial and then bring back their coordinates in the original frame. The translation is used to make one of the element node be the new frame origin as detailed below for each element while the rotation is such that the interface is horizontal in the new frame. The rotation angle is thus given by $\tilde{\theta} = \pi/2 - \bar{\theta}$ where $\bar{\theta}$ is defined by the line normal vector $\vec{n} = (n_x, n_y)$ such that $\cos(\bar{\theta}) = n_x$ and $\sin(\bar{\theta}) = n_y$. Therefore coming back to the original frame makes use of the rotation of angle $-\tilde{\theta}$ corresponding to the matrix $R_{-\tilde{\theta}} = \begin{pmatrix} n_y & n_x \\ -n_x & n_y \end{pmatrix}$.

In the following, we separate the computation of the three elements volumes and provide details on the change of reference frame separately. Finally we gather the three formulae to obtain the trapezoid volume as a function of h.

Volume of the rectangle

The change of reference frame is obtained by the translation of vector $(-x_L, -y_L)$ to bring the point X_L to the origin followed by the rotation of angle $\tilde{\theta}$ so that the rectangle basis is aligned with the x-axis. In this frame of reference, the centroid is simply given by $(B/2, h/2)$. It is then straightforward to obtain the x-coordinate of the centroid in the original reference frame as

$$\bar{x}_B(h) = x_L + \frac{B}{2}n_y + \frac{h}{2}n_x,$$

and hence the volume of the rectangle as

$$\frac{V_B(h)}{2\pi} = A_B(h)\bar{x}_B(h) = h\left(B\left(x_L + \frac{B}{2}n_y\right)\right) + h^2\left(\frac{B}{2}n_x\right).$$

Volume of the left triangle

We here apply the same change of reference frame as in the rectangle case. Thus in the new reference frame, the centroid of the triangle is given by $(-L_L/3, 2h/3)$ and in the original reference frame the x-coordinate of the centroid is

$$\bar{x}_L(h) = x_L - \frac{L_L}{3}n_y + \frac{2h}{3}n_x,$$

and so the signed volume of the left triangle is

$$\frac{V_L(h)}{2\pi} = A_L(h)\bar{x}_L(h) = \frac{h^2}{2}(\tan\theta_L x_L) - \frac{h^3}{6}\big(\tan\theta_L(\tan\theta_L n_y - 2n_x)\big).$$

Volume of the right triangle

In this case, the translation of vector $(-x_R, -y_R)$ brings the point X_R to the new origin while we apply the rotation of angle $\tilde{\theta}$. The centroid of the triangle is this reference frame is $(L_R/3, 2h/3)$ which yields to the x-coordinate of the centroid in the original reference frame

$$\bar{x}_R(h) = x_R + \frac{L_R}{3}n_y + \frac{2h}{3}n_x = x_L + \left(\frac{L_R}{3} + B\right)n_y + \frac{2h}{3}n_x,$$

and to the following signed volume for the right triangle

$$\frac{V_R(h)}{2\pi} = A_R(h)\bar{x}_R(h) = \frac{h^2}{2}(-\tan\theta_R x_L - \tan\theta_R B n_y) + \frac{h^3}{6}\big(\tan\theta_R(\tan\theta_R n_y - 2n_x)\big).$$

Volume of the trapezoid

Finally after algebraic manipulations, we obtain the volume of the trapezoid as

$$\frac{V(h)}{2\pi} = \frac{V_B(h) + V_L(h) + V_R(h)}{2\pi} = h\left(B\left(x_L + \frac{B}{2}n_y\right)\right) + \frac{h^2}{2}\big(B(n_x - \tan\theta_R n_y) - (\tan\theta_R - \tan\theta_L)x_L\big) + \frac{h^3}{6}\big((\tan\theta_R - \tan\theta_L)\big((\tan\theta_L + \tan\theta_R)n_y - 2n_x\big)\big). \quad (3)$$

Note that in this case, the formula is moreover depending on the position of the left node of the trapezoid basis x_L. This must be interpreted as the contribution to the volume formula of the position of the trapezoid.

Before ending this section by detailing the global algorithm of our technique and in particular how formulae of Eqs. (2) and (3) are used, let us draw some remarks about how our approach compares to the non-iterative approach of [1]. In order to get the cubic polynomial that we just derived, authors of [1] make use of a cubic interpolation between four volumes. More specifically, once the solution is bounded, the volumes associated to $h = 0$ and $h = c_{up} - c_{lo}$ are known, but two other volumes are still needed for the interpolation. Therefore they choose to compute the volumes associated to $h = (c_{up} - c_{lo})/3$

and $h = 2(c_{up} - c_{lo})/3$ (it could be any values strictly in $[0, c_{up} - c_{lo}]$) for which they need to compute the two cross sections lengths that themselves require the computation of two intersection points for each cross section. In consequence, the interpolation step of their approach is costly compared to our geometric approach as confirmed by the computational cost comparisons provided in the numerical results section.

2.3. The complete volume-matching algorithm

We give hereafter the general algorithm of our volume-matching technique and detail the use we make of formulae of Eqs. (2) and (3). Assuming that the unit normal vector $\vec{n}$ to the interface is known, the proposed technique to match a volume $\bar{V}$ consists of

Step 1 Compute the line constants of the lines parallel to the interface and passing through all cell nodes $X_1, \ldots, X_N$ and sort them in increasing order such that $c_1 < \ldots < c_N$.

Step 2 Find the bounding line constants c_{lo} and c_{up} such that $V_{c_{lo}} < \bar{V} < V_{c_{up}}$ by computing the increasing c_k-volumes ($V_{c_1} < \ldots < V_{c_N}$). To this end, use $h = (c_{k+1} - c_k)$ in Eqs. (2) and (3) to compute the volume of each additional trapezoid in the planar and axisymmetric geometries, respectively.

Step 3 In the trapezoid defined by the c_{lo}- and c_{up}-lines, compute the interface line constant $c = c_{lo} + h$ where h is the only root in $[0, c_{up} - c_{lo}]$ of the polynomial $V(h) - V_h$ with $V_h = (\bar{V} - V_{c_{lo}})$ and $V(h)$ defined by Eqs. (2) and (3) in the planar and axisymmetric geometries, respectively.

Consequently formulae of Eqs. (2) and (3) are used in two different ways when bracketing and finding the interface line constant. More specifically, during the bracketing step (Step 2) these formulae are simply evaluated with $h = (c_{k+1} - c_k)$ to get the volume of the current trapezoid T_k, while during the volume-matching step (Step 3) they are used in a root finding process to find the only value of h that corresponds to the given volume $\bar{V}$. Let us now draw some remarks about these two different uses.

When computing the volume of the trapezoid T_k during the bracketing step, our approach does not need the cell cross section with the c_{k+1}-line contrary to [4]. This avoids the computation of one intersection node and one square root to compute the length of the cross section and so, tends to reduce the computational cost of our method. Nevertheless, it is clear that if c is not bounded by c_k and c_{k+1}, we need to compute the intersection point X_K^{k+1} between the c_{k+1}-line and the cell to define the next trapezoid basis. But within our configuration it is straightforwardly given by

$$X_K^{k+1} = X_K^k + (c_{k+1} - c_k)\big(\tan\theta_K^k \vec{t} + \vec{n}\big),$$

where K stand for L or R depending on the situation, X_K^k is the current trapezoid basis node and the interface tangential vector $\vec{t}$ is defined as $(-n_y, n_x)$.

It is also important to emphasize again that our approach avoids the costly computation of the complete polygonal (and its volume) resulting from the intersection of the lines with the cell as done in classical methods or in the method of [9].

Finally the volume-matching step is reduced to finding the root in $[0, c_{up} - c_{lo}]$ of the polynomial $V(h) - V_h$ that is of degree two and three in the planar and axisymmetric geometries, respectively. This problem is simplified by the fact that only one root exists in this interval and that regardless of the geometry, we have $V(0) - V_h \le 0$ and $V(c_{up} - c_{lo}) - V_h \ge 0$.

However at the numerical level, we separate the planar and axisymmetric cases since the computation of the root from the exact formula for a polynomial of degree three is generally less efficient than using an iterative method when an accurate interval for the root is known. Therefore we always use a Newton's method to find the root when dealing with axisymmetric geometry, but use the exact formula in the planar case for which it is straightforward to remark that the root always has the form $(-B + \sqrt{\Delta})/(2A)$ when writing the polynomial $V(h) - V_h$ as $Ah^2 + Bh + C$.

3. Numerical tests

In this section, we prove over several numerical tests the increased efficiency of our technique over the Brent's method and the non-iterative approaches of [4,1] and [9]. We detail in Section 3.1 the conditions of the numerical tests that we use to assess the accuracy and computational cost of our volume-matching approach alone and give the results for the planar and axisymmetric geometries in Sections 3.2 and 3.3, respectively. Then in the last subsection we apply our method to an example within a complete interface reconstruction algorithm.

3.1. Testing conditions

During the course of a classical simulation of multi-material flow, the material volume fractions fluctuate between 0 and 1 and may take all values in between, while the vector normal to the interface in each cell may cover all 2D directions. In addition the shape of cells may be different from one cell to another for unstructured meshes. Therefore we will consider a large number of configurations to render these realistic conditions and will define them in a deterministic manner for the sake of reproducibility. Moreover to asses the efficiency of our method, we will use as a reference the Brent's-like iterative

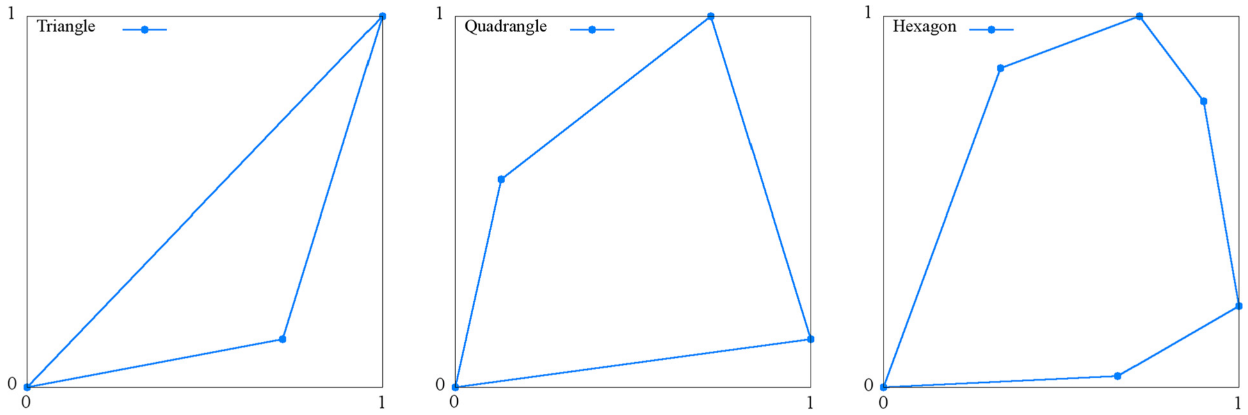

Fig. 4. Illustration of the three non-regular cells used in the numerical tests.

method of Rider and Kothe [13] but will also consider the non-iterative method of [4,1] to show the superiority of our approach. Both errors in volume-matching and computational times will be compared.

Details on the test case configurations are given in the first part of this subsection while the second one is dedicated to comparison measures. In the last part, we give details about the parameters used in the three methods.

Test cases: Cell shapes, volume fractions and normal vectors

The numerical tests we propose are performed in four different cell shapes; a regular unit square for the sake of comparison with [9] for instance, and three non-regular ones that mimic the types of cells that may appear with unstructured meshes: a triangle, a quadrangle and a hexagon. The nodes coordinates of these non-regular cells, represented in Fig. 4, are given by

$$\begin{aligned} &\textit{Triangle}: && (0,0),\ (0.72,0.13),\ (1,1),\\ &\textit{Quadrangle}: && (0,0),\ (1,0.13),\ (.72,1),\ (0.13,0.56),\\ &\textit{Hexagon}: && (0,0),\ (0.66,0.03),\ (1,0.22),\ (0.9,0.77),\ (0.72,1),\ (0.33,0.86). \end{aligned}$$

Note that these cells are contained in a square of unit area to prevent round-off errors and that it does not induce any loss of generality since a simple scaling as the one described in Appendix A can be used to recover an analogous situation.

We now define the two sets S_1 and S_2 of configurations (*i.e.* couples formed by a normal vector $\vec{n}$ and a volume fraction α) that will be used as test cases. To define these two sets, we simply gather the combinations obtained by considering a set $S_{\vec{n}}$ of normal vectors and two sets of volume fractions S_α^1 and S_α^2 respectively, that is

$$S_1 = \{(\vec{n},\alpha) \mid \vec{n} \in S_{\vec{n}}, \alpha \in S_\alpha^1\} \quad \text{and} \quad S_2 = \{(\vec{n},\alpha) \mid \vec{n} \in S_{\vec{n}}, \alpha \in S_\alpha^2\}.$$

The normal vectors of the set $S_{\vec{n}}$ are defined by the sines and cosines of 360 angles starting from zero and increasing by one degree, that is

$$S_{\vec{n}} = \{\vec{n} = (\cos\theta, \sin\theta) \mid \theta = 0^\circ, 1^\circ, 2^\circ, \ldots, 359^\circ\}.$$

We chose to consider two different sets of volume fractions S_α^1 and S_α^2 to emphasize the fact that our method is almost not affected by the values that are used, contrary to the Brent's-based method for which the number of iterations increases when volume fractions are close to zero or one as described in [13]. Both sets contain 10,000 values so that the total number of combination in S_1 and S_2, denoted N_{comb}, is 3.6 millions. Such a big N_{comb} is necessary to have large enough computational times to ensure fair comparison between methods.

The set S_α^1 uses a constant increment between consecutive volume fractions ranging from 10^{-4} to 1, that is

$$S_\alpha^1 = \{\alpha = m.10^{-4} \mid m = 1, 2, 3, \ldots, 10^4\},$$

while the set S_α^2 focuses more on small volume fractions by considering 1000 uniformly distributed volume fractions in each of the 10 intervals $[10^{-p}, 10^{-p+1}]$ with $p = 1, \ldots, 10$, that is

$$S_\alpha^2 = \left\{\alpha = 10^{-p} + \frac{m}{10^3}\left(10^{-p+1} - 10^{-p}\right) \,\middle|\, m = 1, 2, \ldots, 10^3 \text{ and } p = 1, 2, \ldots, 10\right\}.$$

The first set is certainly closer to the reality of physics computation, but at the same time the set S_α^2 is relevant in that on the one hand volume fractions smaller than 10^{-4} often appear during simulation and on the other hand it is important to assess the insensitivity of our method to volume fractions close to 0 or 1 that badly affect the Brent's method cost.

It is worth noticing that ideally the computational time needed by any method to solve the volume-matching problem for the two sets S_1 and S_2 would only be dependent on the number of combinations and not on the different values of the volume fractions.

Comparison measures

To compare the proposed method to the two existing ones, we consider several measures both in terms of accuracy and computational cost. First, we consider two error norms that measure the global accuracy of the volume matching, a L^∞ norm

$$\max_{1\le i\le N_{comb}} \left|V_c^i - \bar{V}^i\right|,$$

and a L^1 norm

$$\frac{1}{N_{comb}} \sum_{i=1}^{N_{comb}} |V_c^i - \bar{V}^i|,$$

where the superscript i refers to a particular configuration $(\vec{n}, \alpha)$ in S_1 or S_2 so that the volume to match is $\bar{V}^i = \alpha V_{cell}$ while V_c^i is the c-volume corresponding to the interface line constant c computed by the method under consideration.

Finally we compare the computational times between the two methods using Relative Time Units (R.T.U.). More specifically, for each geometry we set that one R.T.U. is the computational time taken by our method to complete the set S_1 for the triangular cell. This way, the comparisons between methods and also combinations sets are made easier.

Methods parameters

The three methods have been implemented in the same code such that the computational costs comparison is as fair as possible. It has been written in Fortran and compiled by Intel ifort with the -O2 and -ipo optimization options. We hereafter give the implementation details that needs to be highlighted for each of the three methods.

For the technique we propose, the interface line constant is found using the exact formula for the root in the planar geometry and a Newton's method in the axisymmetric one. In the latter case, we always use $h = (c_{up} - c_{lo})/2$ as initial guess and $|V(h) - V_h| < \epsilon$, with ϵ set to 10^{-15}, as stopping criterion for all computations.

For the other non-iterative method, we use the exact formula given in [4] for the planar case. For the axisymmetric case, since few details are given in [1], we choose to perform the cubic interpolation relative to the h variable as in our method. It is simplified by the fact that the trapezoidal volume is zero for $h = 0$. In this case, we also separate the centroid computation for the first and last degenerated trapezoids (that are triangles) to speed up the computation. At last, the root of the cubic polynomial is found by the same exact Newton's method that we use for our method.

Finally we implemented the Brent's iterative method as described in [13] with a slight modification of the line constant bracketing. Indeed, as proposed in [9], we first compute and sort the values of c_k for each node and start with computing the c_k-volumes corresponding to the *central* index $\mathtt{INT}[(N+1)/2]$, thus limiting the number of c_k-volumes computations needed to bracket the solution. It decreased the cost of the Brent's method in all the test cases we have carried out by about 10%. During the line constant computation, the Brent's algorithm is stopped when the difference between two consecutive candidate roots is smaller than 10^{-14} or when $|V(h) - V_h|$ is smaller than machine tolerance. These values have been chosen such that the errors in volume-matching are of the same magnitude than the ones with our method. We point out that increasing these values does not dramatically affect the computational times but clearly increases the errors and refer the reader to [9] for a study with different tolerance values.

3.2. Results for the planar geometry

We consider in this section the planar geometry case and gather in Table 1 the R.T.U., L^∞ and L^1 errors obtained for the sets S_1 and S_2 and for each of the four cell shapes using both the Proposed, the non-iterative method of [4] (referred to as 'MOF-like') and the Brent's methods. We also provide the average number of iterations (Avg. It. Nb.) of the Brent's algorithm in the last column.

As expected the errors are exact to machine precision for all methods since the tolerance in the Brent's method has been chosen to ensure this property. Nevertheless it is worth noticing that the maximal errors (L^∞) over the sets are always more accurate by about one order of magnitude for the proposed method compared to the two other methods (MOF-like and Brent's).

Let us first consider the set S_1, the proposed method is 5 times faster than the classical method for the square cell and between 6 and 8 times faster for the non-regular shapes. The results on the square is explained by a smaller number of iterations (analogous to the ones in Tables 1 and 2 of [9]) for the Brent's method. Moreover, as expected the comparisons with the non-iterative method of [4] reveals closer computational times but our approach is always at least 1.6 times faster with a better accuracy. It is worth remarking that indirect comparisons to the other non-iterative method proposed in [9] shows that our method is about 3.5 times faster. Indeed for the square cell their method is about 1.4 times faster than the Brent's method coupled with the efficient bracketing procedure while ours is 5 times faster.

Table 1
Results in planar geometry. R.T.U. and volume-matching errors for the different cell shapes obtained by the Proposed method, the MOF-like method of [4] and the Brent's method. One R.T.U. corresponds to the computational time needed by the Proposed method to compute the results of the Set 1 on the triangular cell. When iterative method is used, the average number of iterations over the whole set of combinations is also specified.

Set	Method	Cell Shape	R.T.U.	L^∞ Error	L^1 Error	Avg. It. Nb.
Set 1	Proposed	Triangle	1.00	1.94e-16	2.41e-17	N/A
		Square	1.06	2.78e-16	2.48e-17	N/A
		Quadrangle	1.16	3.89e-16	4.85e-17	N/A
		Hexagon	1.55	6.94e-16	7.04e-17	N/A
	MOF-like	Triangle	1.70	3.30e-15	4.23e-17	N/A
		Square	1.95	1.44e-15	5.55e-17	N/A
		Quadrangle	2.14	4.02e-15	6.47e-17	N/A
		Hexagon	2.83	3.00e-15	8.63e-17	N/A
	Brent's	Triangle	8.40	3.18e-15	1.22e-16	7.168
		Square	5.30	3.72e-15	6.72e-17	3.340
		Quadrangle	8.43	3.83e-15	2.15e-16	6.445
		Hexagon	9.39	5.22e-15	2.23e-16	6.046
Set 2	Proposed	Triangle	0.94	2.78e-16	1.05e-17	N/A
		Square	1.03	3.33e-16	6.82e-18	N/A
		Quadrangle	1.04	3.89e-16	1.41e-17	N/A
		Hexagon	1.26	6.38e-16	1.69e-17	N/A
	MOF-like	Triangle	1.68	5.88e-15	1.39e-17	N/A
		Square	1.74	1.78e-15	1.12e-17	N/A
		Quadrangle	1.80	4.02e-15	1.78e-17	N/A
		Hexagon	2.10	3.61e-15	1.97e-17	N/A
	Brent's	Triangle	18.34	3.53e-15	4.23e-17	18.498
		Square	17.64	3.50e-15	3.99e-17	17.066
		Quadrangle	18.58	3.61e-15	5.48e-17	17.510
		Hexagon	19.50	4.22e-15	5.77e-17	16.188

Furthermore the proposed method is essentially insensitive to the values of volume fractions used in the tests as shown when running the combinations of set S_2. While the Brent's method is almost two times slower (with 2.5 more iterations) for the same number of combinations due to small volume fractions, the proposed method is even slightly faster than for the set S_1 because of its incremental nature. The conclusions of comparisons between our approach and the one of [4] still hold for the set S_2.

3.3. Results for the axisymmetric geometry

We now turn to the axisymmetric geometry case and give in Table 2 the R.T.U., L^∞ and L^1 errors obtained for the sets S_1 and S_2 for the four cell shapes with the three methods. However the average number of iterations refers to the Newton's method for our method and the one of [1] while it still refers to the Brent's algorithm for the reference method.

For this geometry, all methods use an iterative process but the Newton's method always needs less iterations than the Brent's one and most importantly a Newton's iteration is much less expensive than a Brent's one as it only consists of one evaluation of a polynomial of degree three and one evaluation of its derivative.

Let us first point out that the cost of the proposed technique in the axisymmetric case increases compared to the planar situation with a factor of about 1.6 for the set S_1 and 2.4 for the set S_2. This is what we expected because of the more complex nature of the formula for the axisymmetric geometry requiring the use of an iterative Newton's method.

Regardless of this cost increase, the proposed method is still more efficient than the Brent's method, at least 4.8 times faster for the set S_1 and at least 7.7 times for the set S_2 while the errors are at machine precision as wanted. Furthermore, the reduction in computational cost with our method compared to the MOF-like one is more important in this geometry than in the planar one as expected (see the remark at the end of Section 2) since it is at least 2 times faster on S_1 and 1.8 on S_2 with always a better accuracy. At last, the relative insensitivity of our method to the values used in the volume-matching problem is confirmed in the axisymmetric geometry as shown in the results with the set S_2.

Finally our approach brings an important reduction in computational cost compared to the classical iterative method and a slight one compared to the non-iterative MOF-like method in addition to proposing a clearer geometric derivation of the exact formula.

3.4. Example within a complete interface reconstruction algorithm

We now want to assess the gain brought by our approach in a complete interface reconstruction problem. To this end, we consider a biohazard-like symbol (depicted in the right panel of Fig. 5) on an irregular Voronoi mesh computed from 6348 randomly distributed generators (except on the domain boundaries). It leads to 6348 cells in total and 704 multi-material

Table 2
Results in axisymmetric geometry. R.T.U. and volume-matching errors for the different cell shapes obtained by the Proposed method, the MOF-like method of [4] and the Brent's method. One R.T.U. corresponds to the computational time needed by the Proposed method to compute the results of the Set 1 on the triangular cell. When iterative method is used, the average number of iterations over the whole set of combinations is also specified.

Set	Method	Cell Shape	R.T.U.	L^∞ Error	L^1 Error	Avg. It. Nb.
Set 1	Proposed	Triangle	1.00	1.55e-15	1.33e-16	4.892
		Square	0.96	2.00e-15	1.80e-16	3.173
		Quadrangle	1.06	1.78e-15	1.51e-16	4.402
		Hexagon	1.26	1.78e-15	1.52e-16	3.881
	MOF-like	Triangle	2.08	2.39e-14	3.70e-16	4.892
		Square	1.94	1.29e-14	3.72e-16	3.173
		Quadrangle	2.31	2.03e-14	4.02e-16	4.402
		Hexagon	2.76	1.55e-14	3.96e-16	3.881
	Brent's	Triangle	5.68	1.30e-14	6.12e-16	7.457
		Square	4.79	1.52e-14	7.34e-16	5.770
		Quadrangle	5.78	1.33e-14	7.18e-16	7.195
		Hexagon	6.18	1.20e-14	7.92e-16	6.648
Set 2	Proposed	Triangle	1.41	1.89e-15	9.63e-17	10.721
		Square	1.41	2.22e-15	8.72e-17	9.967
		Quadrangle	1.41	1.78e-15	9.43e-17	10.045
		Hexagon	1.49	1.60e-15	9.56e-17	9.353
	MOF-like	Triangle	2.50	3.09e-14	1.38e-16	10.721
		Square	2.44	1.24e-14	1.22e-16	9.969
		Quadrangle	2.53	1.42e-14	1.32e-16	10.045
		Hexagon	2.64	1.38e-14	1.26e-16	9.353
	Brent's	Triangle	11.30	1.75e-14	1.41e-16	17.788
		Square	10.93	1.71e-14	1.71e-16	16.368
		Quadrangle	11.42	1.38e-14	1.62e-16	16.978
		Hexagon	11.89	1.29e-14	1.78e-16	15.743

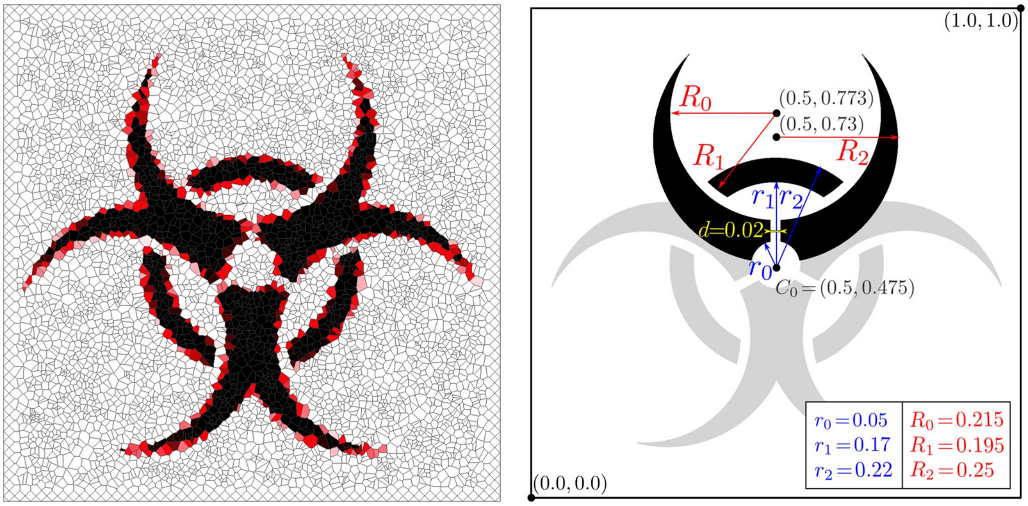

Fig. 5. Interface reconstruction test. Left: Plot of the material volume fractions on the irregular Voronoi mesh: White cells contain only material 2, black cells contain only material 1 while red cells contain both materials. Right: Representation of the biohazard-like symbol with the details to reproduce the fundamental shape. Rotations of center C_0 and angles $2\pi/3$ and $-2\pi/3$ are used to complete the symbol.

ones with the smallest volume fraction being about 2.10^{-9} for both planar and axisymmetric geometries. An illustration of the mesh is given in the left panel of Fig. 5 where the volume fractions of material 1 have also been plotted.

For each cell, the first step of the algorithm used in this test is to detect if the cell is multi-material by looking at the material volume fractions. Then if it is the case, the normal vector to the interface is computed using a weighted least-squares technique [2] and the volume-matching technique described in Section 2.3 is used to find the line constant of the interface. The last step is the storage of the material polygons in memory. Note that in multi-material hydrodynamics simulations with sharp interface treatment, this algorithm is used at each time step to get the interface representation.

We compare in Tables 3 the computational times (in R.T.U.) obtained to complete the interface reconstruction problem described in previous paragraph for all cells both in planar and axisymmetric geometries with the three volume-matching

Table 3
Interface reconstruction test. Total computational times (Total CPU) for the interface reconstruction problem in R.T.U. and percentage corresponding to the volume-matching step (Vol. Match.) with the three volume-matching techniques: the Proposed method, the MOF-like method of [4] and the Brent's method. One R.T.U. corresponds to the computational time needed by the Proposed method in the planar geometry.

Vol. Match. Technique	Planar geometry		Axisymmetric geometry	
	Total CPU (R.T.U.)	Vol. Match. (%)	Total CPU (R.T.U.)	Vol. Match. (%)
Proposed	1.00	52.42	1.17	59.65
MOF-like	1.06	60.12	1.39	70.43
Brent's	3.09	86.34	3.14	86.58

techniques used in previous sections (Proposed, MOF-like and Brent's). We have set one R.T.U. to be the computational time needed for the Proposed technique in the planar geometry when repeating 30,000 times the problem in order to get fair computational times comparisons. Moreover the percentage of the total computational time that corresponds to the volume-matching step alone is also provided.

As expected, the complete interface reconstruction problem is solved in less computational time when using the proposed volume-matching technique due to the gain observed in the previous section when only considering the volume-matching step. In particular, we see that the proposed technique allows to solve the current problem three times faster than when the classical Brent's method is used, and 10–15% faster than when the MOF-like method is used.

Let us finally note that the volume-matching step is reduced to 55–60% of the whole interface reconstruction process with the new method while it is about 85% with the Brent's method, thus explaining the decrease in computational times. Nevertheless this gain will be mitigated when implemented in a complete simulation code since the reconstruction step is usually not be the main part of the algorithm.

4. Summary and conclusion

In this work, we propose a non-iterative interface reconstruction technique for planar and axisymmetric geometries based on analytical formulae that are derived from geometric considerations. Our approach is new and allows a direct use of the analytical formulae to both bracket and find the line constant that is solution of the interface positioning problem, contrary to existing efficient non-iterative methods [4,1] that use classical volume formulae and polynomial interpolation techniques. Our work is only focused on the second step of the interface reconstruction problem, the interface positioning, thus assuming that the normal vector to the interface is already computed by one of the many available techniques ([14] and references therein).

Considering the trapezoidal decomposition of the cell defined by the lines parallel to the interface and passing through the cell nodes, we propose a simple geometric approach to derive analytical formula for the volumes of the particular trapezoids in both planar and axisymmetric 2D geometries. This technique has been tested over four different cell shapes and two different sets of configurations (normal vectors and material volumes) both containing 3.6 millions of combinations. We provided all the details on the conditions so that they are entirely reproducible.

Our method is found 5–19 times faster than the widely used Brent's method, and is moreover mainly insensitive to volume fractions values. When compared to the efficient non-iterative method of [4,1], our approach is slightly faster (1.6–2.0 times) in both the planar and axisymmetric geometries while always being more accurate.

In the future, we plan to apply the same ideas first to develop an exact interface reconstruction method for arbitrary three-dimensional convex cells, and then to extend it to arbitrary non-convex cells. Moreover we plan to test the new volume-matching method in dynamic advection and multi-material flows problems.

Acknowledgements

This work was performed under the auspices of the National Nuclear Security Administration of the US Department of Energy at Los Alamos National Laboratory under Contract No. DE-AC52-06NA25396 and supported by the Advanced Simulation and Computing (ASC) program. Approved for unlimited release under LA-UR-13-28977.

Appendix A. Unit-area scaling valid for planar and axisymmetric geometries

In this appendix, we describe a simple scaling valid for both planar and axisymmetric geometries that may be used to avoid round-off errors during interface reconstruction in very small cells. The main idea is to scale the smallest box containing the cell under consideration to a square cell of unitary area. As such it is valid for any cell shape.

Let us denote by Δx and Δy the width and height of the smallest rectangle containing the cell under consideration. We then define the change of coordinates as

$$\tilde{x} = \frac{x}{\Delta x} \quad \text{and} \quad \tilde{y} = \frac{y}{\Delta y},$$

where $\tilde{x}$ and $\tilde{y}$ are the scaled coordinates.

In the scaled frame of reference, it is obvious that the smallest box containing the cell has unitary width and height and so unitary area. However the cell itself does not have unitary area in general and the volume of the box is not unitary for the axisymmetric case neither.

We now give the relation between the cell area A in the original frame and $\tilde{A}$ the one in the scaled frame

$$\tilde{A} = \frac{A}{\Delta x \Delta y},$$

which is also the relation between volumes V and $\tilde{V}$ for the planar geometry, while for the axisymmetric case the relation between volumes is given by

$$\tilde{V} = 2\pi \tilde{A}\tilde{\bar{x}} = 2\pi \frac{A}{\Delta x \Delta y}\frac{\bar{x}}{\Delta x} = \frac{V}{(\Delta x)^2 \Delta y},$$

where $\bar{x}$ and $\tilde{\bar{x}}$ are the x-coordinate of the cell centroid in the original and scaled frame of reference respectively.

These relations between volumes in the original and scaled frames of reference ensure the equivalence between matching a prescribed volume fraction α in the scaled frame and matching it in the original one as long as we consider the change of reference frame for the interface equation too. Indeed we have

$$\frac{\tilde{V}_{rec}}{\tilde{V}} = \alpha \quad \overset{planar}{\Longrightarrow} \quad \frac{V_{rec}}{V} = \frac{\tilde{V}_{rec}}{\Delta x \Delta y}\frac{\Delta x \Delta y}{\tilde{V}} = \alpha \quad \overset{axisymmetric}{\Longrightarrow} \quad \frac{V_{rec}}{V} = \frac{\tilde{V}_{rec}}{(\Delta x)^2 \Delta y}\frac{(\Delta x)^2 \Delta y}{\tilde{V}} = \alpha,$$

where V_{rec} and $\tilde{V}_{rec}$ are the volumes defined by the reconstructed interface in the original and scaled reference frames respectively.

Finally, as mentioned the interface equation has to be scaled back to the original frame of reference. Considering the computed interface equation in the scaled reference frame given by $\tilde{n}_x\tilde{x} + \tilde{n}_y\tilde{y} + \tilde{c} = 0$, the equation in the original frame is

$$n_x x + n_y y + c = \left(\frac{\tilde{n}_x}{\Delta x}\right)x + \left(\frac{\tilde{n}_y}{\Delta y}\right)y + \tilde{c} = 0$$

that we can also renormalize in order to have a unit normal vector in the original frame by dividing the normal components and the line constant by $\sqrt{n_y^2 + n_y^2}$.

References

[1] H.R. Anbarlooei, K. Mazaheri, 'Moment of fluid' interface reconstruction method in axisymmetric coordinates, Int. J. Numer. Meth. Biomed. Eng. 27 (2011) 1640–1651.

[2] T.J. Barth, P.O. Fredrickson, Higher-order solution of the Euler equations on unstructured grids using quadratic reconstruction, AIAA Aerosp. Sci. Meeting 28 (1990).

[3] R.P. Brent, Algorithms for Minimization Without Derivatives, Prentice–Hall, Englewood Cliffs, New Jersey, ISBN 0-13-022335-2, 1973, 195 pp.

[4] V. Dyadechko, M. Shashkov, Moment-of-fluid interface reconstruction, LANL Report, LA-UR 07-1537, 2007, pp. 1–57.

[5] V. Dyadechko, M. Shashkov, Reconstruction of multi-material interfaces from moment data, J. Comput. Phys. 227 (2008) 5361–5384.

[6] M.M. Francois, B.K. Swartz, Interface curvature via volume fractions, heights, and mean values on nonuniform rectangular grids, J. Comput. Phys. 229 (2010) 527–540.

[7] D. Gueyffier, J. Li, A. Nadim, R. Scardovelli, S. Zaleski, Volume-of-fluid interface tracking with smoothed surface stress methods for three-dimensional flows, J. Comput. Phys. 152 (1999) 423–456.

[8] R. DeBar, Fundamentals of the KRAKEN Code, LLNL Report, UCIR-760 (1974).

[9] J. López, J. Hernández, Analytical and geometrical tools for 3D volume of fluid methods in general grids, J. Comput. Phys. 227 (2008) 5939–5948.

[10] B.D. Nichols, C.W. Hirt, Methods for Calculating Multi-Dimensional, Transient Free Surface Flows Past Bodies, LANL Report LA-UR-75-1932 (1975).

[11] W.F. Noh, P. Woodward, SLIC (Simple Line Interface Calculation), Lect. Notes Phys., vol. 59, Springer-Verlag, New York, 1976, pp. 330–340.

[12] J.E. Pilliod Jr., E.G. Puckett, Second-order accurate volume-of-fluid algorithms for tracking material interfaces, J. Comput. Phys. 199 (2004) 465–502.

[13] W.J. Rider, D.B. Kothe, Reconstructing volume tracking, J. Comput. Phys. 141 (1998) 112–152.

[14] R. Scardovelli, S. Zaleski, Interface reconstruction with least-square fit and split Eulerian–Lagrangian advection, Int. J. Numer. Methods Fluid 41 (2003) 251–274.

[15] T.G. Vignesh, S. Bakshi, Noniterative interface reconstruction algorithms for volume of fluid method, Int. J. Numer. Methods Fluid 73 (2013) 1–18.

[16] X. Yang, A.J. James, Analytic relations for reconstructing piecewise linear interfaces in triangular and tetrahedral grids, J. Comput. Phys. 214 (2006) 41–54.

[17] D.L. Youngs, Time-dependent multi-material flow with large fluid distortion, in: K.W. Morton, M.J. Baines (Eds.), Numerical Methods for Fluids Dynamics, Academic Press, New York, 1982, pp. 273–285.