ESAIM: PROCEEDINGS AND SURVEYS, Vol. ?, 2019, 1-10
Editors: Will be set by the publisher

CURVED INTERFACE RECONSTRUCTION FOR 2D COMPRESSIBLE MULTI-MATERIAL FLOWS

Igor Chollet[1], Giulia Lissoni[2], Théo Corot, Philippe Hoch, Thomas Leroy, [3] and Laurent Dumas[4]

Abstract. A curved interface reconstruction procedure is presented here in the case of a 2D compressible flow made of two or more materials. Built with a dynamic programming procedure already introduced in [1], the curve interface is continuous and volume preserving in each cell. It is applied here to general test cases with non cartesian grids as well as triple point configurations.

Résumé. Une procédure de reconstruction d'interface courbe est présentée ici dans le cas d'un écoulements 2D compressible constitué de 2 matériaux ou plus. Construite avec un algorithme de programmation dynamique introduit dans [1], l'interface obtenue est continue et préserve les volumes dans chaque cellule. La méthode est appliquée ici sur des cas tests généraux pour des maillages non structurés et des configurations de point triple.

1. Introduction

Interface reconstruction (IR) methods are encountered in numerical simulation of multi-material or multi-fluid flows. In a Volume of Fluid (VOF) approach in a case of two materials, denote C the volume fraction of material 1 encountered in volume V:

$$C = \frac{1}{V}\int_V \chi(x,y,z)dxdydz \tag{1}$$

where χ denotes the indicator function of material 1:

$$\chi(x,y,z) = \begin{cases} 1 & \text{if } (x,y,z) \in \text{ material } 1 \\ 0 & \text{if } (x,y,z) \in \text{ material } 2 \end{cases}$$

The objective of IR methods is to define a geometric interface separating material 1 and material 2 with the following properties:

- P1: volume fractions conservation,
- P2: continuity of the interface,
- P3: robustness,
- P4: low or moderate computational cost.

[1] Sorbonne Université , institut des sciences du calcul et des donnes, ISCD, F-75005 Paris, France

[2] Université Côte d'Azur, Inria, CNRS, LJAD, Parc Valrose, 06108 Nice, France

[3] CEA DAM DIF, BP 12, 91297 Arpajon Cedex, France

[4] Laboratoire de Mathématiques de Versailles, UVSQ, CNRS, Université Paris-Saclay, 78035 Versailles, France.

There exists various IR methods based on a VOF approach. The first one that has been introduced in 1982 is due to D.L. Youngs [3]. Even though this method is still largely used up to now because of its simplicity and robustness, it suffers from the non continuity of the interface.

Another family of IR methods doesn't rely on the VOF scheme. Introduced by S. Osher [2], the level-set method consists in defining the interface between the two materials as an isoline contour of a function $\Phi : \mathbb{R}^3 \to \mathbb{R}$ advected by the flow field:

$$\Gamma = \{(x, y, z), \Phi(x, y, z) = 0\},$$

where

$$\frac{\partial \Phi}{\partial t} + v \cdot \nabla \Phi = 0$$

with $\Phi < 0$ (respectively > 0) in the case of $(x, y, z) \in$ material 1 (respectively 2).

In particular, the normal to the interface is given by:

$$n = \frac{\nabla \Phi}{|\nabla \Phi|},$$

and its curvature by:

$$\kappa = \nabla \cdot \frac{\nabla \Phi}{|\nabla \Phi|}.$$

Compared to a VOF scheme, Φ is a solution of a similar equation to the one satisfied by C but, contrarily to the VOF approach, the isoline $\Phi = 0$ exactly defines the interface. This method, very popular in free surface simulations, has here the major drawback of not fulfilling the mandatory conservation property P1. Some variants of this method have been proposed that remedy to this drawback. One of the idea is to define a new control point inside each mixed cell but it is done at the expense of an increased cost and a reduced robustness.

Recently, in [1], a new reconstruction method which ensures continuity of the interface and preserves volume fractions have been introduced. This new interface reconstruction method, called DPIR (*Dynamic Programming Interface Reconstruction*) is introduced in the next section and will be used as a starting point for the presented work. It consists of two main steps. First, minimize a suitable energy functional gives a continuous linear interface. Secondly, add a control point in each cell to find the correct volume fractions. This last step in usually made by searching the point in the normal direction of the interface, in the line passing through the center of this one.

In this paper there is three main goals. First, the DPIR method is extended for curved interfaces (section 3). It is of interest in particular in the case of curved meshes, where an exact reconstruction of the interface is expected. In order to be a real candidate for being used in multi-material hydrodynamic simulation using ALE remap methods, the DPIR method must be able to deal with distored meshes. Although the principle of the method remains unchanged (one minimization step, one correction step), several improvment are proposed (section 4), in particular to deal with strongly distorded cells and small volume fraction issues. Finally, this work ends with a generalization of the method for three materials (section 5). Interface reconstruction for multi-material simulations is a complicated issue, and a comparison of several existing methods can be found in [?]. The proposed method applies the DPIR method for all the materials *without choosing any material ordering* and a suitable average is applied to obtain the final interfaces. The method is tested on two test cases with triple point configuration on cartesian meshes, giving encouraging results for futur unstructured meshes cases.

2. Interface reconstruction with DPIR

The interface reconstruction problem can be seen as a minimization problem of the total sum of all volume fractions errors. In the particular case of a 2D cartesian grid, it writes as:

$$J(y) = \sum_{(i,j)\in\{1,\dots,N_x\}\times\{1,\dots,N_y\}} |vol_{y,i,j} - vol_{i,j}|^2 \tag{2}$$

where $t \mapsto y(t)$ is the associated interface curve and $vol_{i,j}$ denotes the targeted volume fraction in cell (i, j).

Because of its partially separated structure, dynamic programming can be used to minimize the cost function J. A local correction step is then added to make decrease the cost function to an exact zero value. The detailed DPIR algorithm is decribed below.

2.1. First step: minimization of J with dynamic programming

In this first step, the interface curve $t \mapsto y(t)$ is supposed to be piecewise linear on each cell (see figure 1).

The minimization problem will thus consists in finding a finite number of points $(M_i)_{0\le i\le N}$ such that $M_0 = M_N$, located on the edges of mixed cells. More precisely, the possible locations of points M_i are obtained after finding the so-called *internal* and *external curves* that will bound the interface curve (see figure 1). The internal and external curves are also polygonal curves with nodes located at the mesh nodes and are obtained by a simple search algorithm among mixed cells (corresponding to cells such that $0 < vol_{i,j} < 1$). A dynamic programming procedure is applied here to find the piecewise closed linear curve that will minimize the cost function J at a cost $\mathcal{O}(NL^2)$ where L is the discretization number of each cell edges.

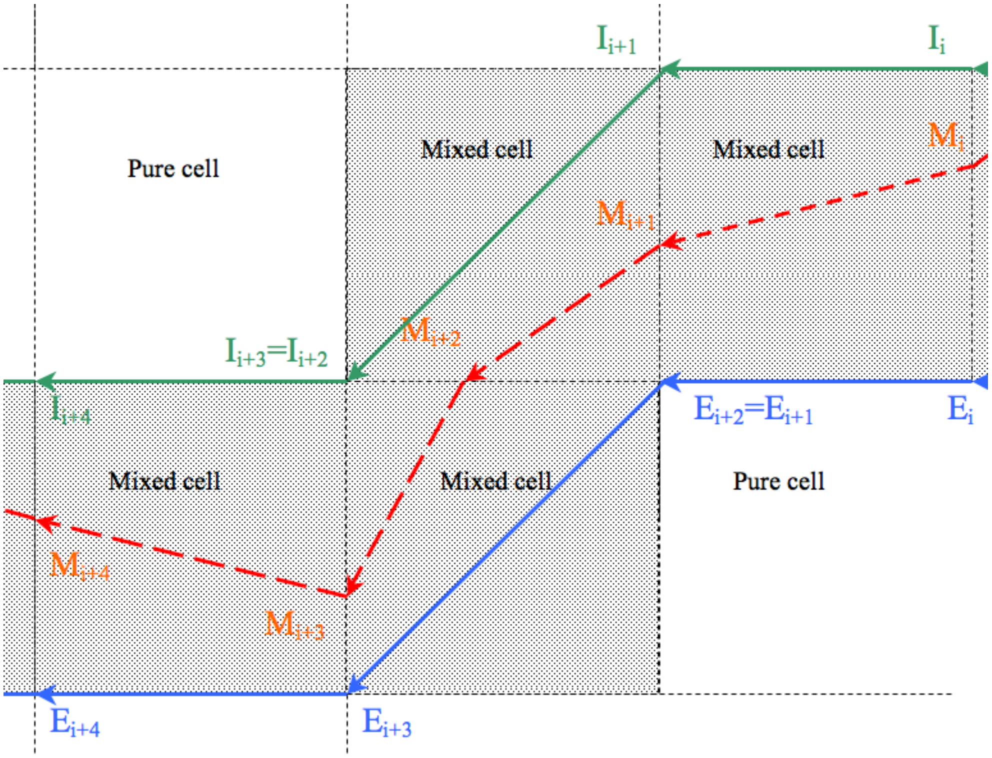


Figure 1. The interface curve (dotted line) with the internal (points I_i) and external (points E_i) curves.

Note that a penalty term of the form $p(y) = \sum_{i=0}^{N-1} \lambda||M_{i+1} - M_i||$ ca be added to the cost function J in this first step in order to reduce the interface length and to avoid wave effects.

2.2. Second step: local correction of volume fractions

Dynamic programming is a very efficient tool to find the minimum of J, and consequently a first approximation of the interface boundary. Note that after this step, the three conditions P2 (continuity), P3 (robustness) and P4 (low cost) on the interface reconstruction are fulfilled. However, a correction step must be added in order to satisfy property P1 (volume fraction conservation in each cell). In the initial DPIR algorithm described in [1], a control point is added in each cell on the perpendicular bisector of each elementary segment to have an exact equality of volume fractions on each cell.

The complete method including the previous two steps, called DPIR (Dynamic Programming Interface Reconstruction) can then be summarized by the following algorithm:

DPIR Algorithm*: for a given distribution of volume fractions on a 2D cartesian grid:*

- **Initialization***: define the internal and external curves that will bound the interface.*
- **Step 1 (global step)***: minimize the cost functional J in (2) with dynamic programming.*
- **Step 2 (local step)***: add a control point in each cell to have an exact equality of volume fractions.*

Some extensions of this initial algorithm are now presented in the next sections.

3. DPIR extension to curved interfaces reconstruction

In order to obtain a curved interface, more suited to particular cases (for a circle reconstruction for instance), rational Bezier curves are introduced in the local correction phase of DPIR.

A second order rational Bezier curve is a parametric curve defined by three control points P_0, P_1 and P_2. In the present case, P_1 will play the role of the control point with an associated weight $\omega \in [0, +\infty]$ (see figure 2).

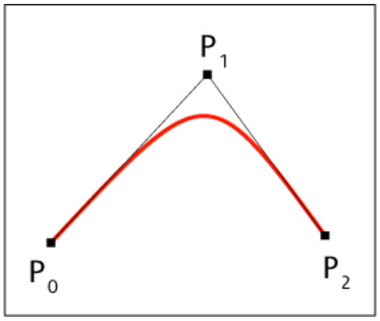


Figure 2. A second order rational Bezier curve

More precisely, we can compute the area of the Bezier curve, A, in the following way:

$$A = f(\omega) \cdot A(P_0, P_1, P_2)$$

where $A(P_0, P_1, P_2)$ is the area of the triangle $\widehat{P_0P_1P_2}$ and

$$f(\omega) = \begin{cases} 0 & \text{if } \omega = 0 \\ \dfrac{2\omega}{1-\omega^2}\left(\dfrac{1}{1-\omega^2}\arctan\left(\sqrt{\dfrac{1-\omega}{1+\omega}}\right) - \dfrac{\omega}{2}\right) & \text{if } \omega \in (0,1) \\ \dfrac{2}{3} & \text{if } \omega = 1 \\ \dfrac{\omega}{\omega^2-1}\left(\omega + \dfrac{1}{\sqrt{1-\omega^2}}\ln\left(\omega - \sqrt{\omega^2-1}\right)\right) & \text{if } \omega > 1 \end{cases}$$

In our algorithm, the value of ω will be fixed. Suppose that M_i, $i = 0, \ldots N$ are the coordinates of the obtained points after the first step of DPIR, we then look for the coordinates P_i of the control points such that the volume fraction defined by the rational Bezier curve associated to (M_i, P_i, M_{i+1}) is equal to the desired one.

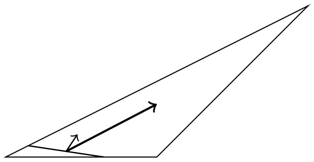
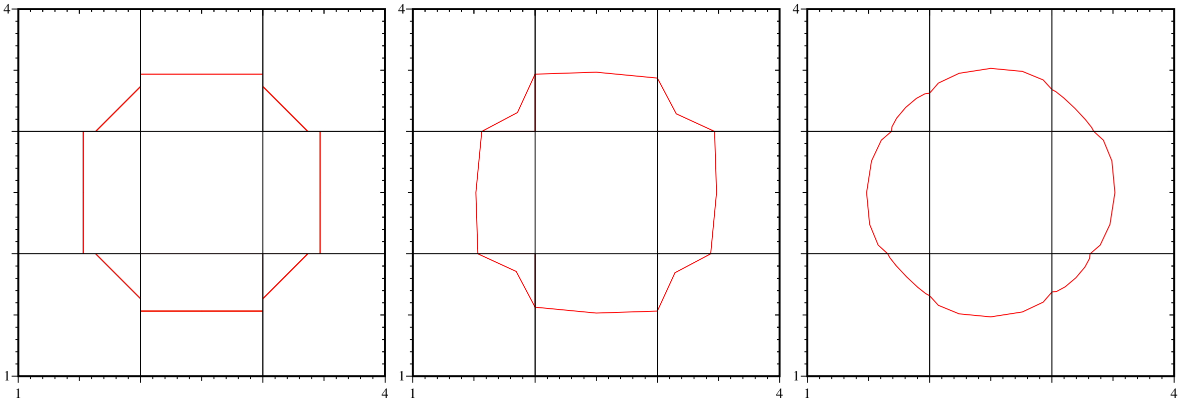


Figure 3. Search direction of the control point: perpendicular bisector vs center of the cell.

In order to do so, a dichotomy procedure is defined. The difference with respect to the initial second step of DPIR lies also in the search direction : instead of adding the control point on the perpendicular bisector of M_i, M_{i+1}, dichotomy is applied towards the center of the cell. This ensures to move into a direction where there is more space to apply the correction inside the cell. This will be crucial in particular in the case of distorted cells (see figure 3).

The corresponding results for a circle reconstruction with this new version of DPIR are presented in figures 34 with a constant value $\omega = \frac{1}{2}$. A comparison with a Youngs reconstruction and a piecewise linear reconstruction with DPIR for a coarse grid is in particular depicted. In this case, the improvement when using rational Bezier curves is clearly visible.



Young's method

Polygonal (DPIR)

Curve (DPIR)

The robustness of DPIR with respect to the mesh quality and to some undesirable effects in its second step (for instance a too large correction that goes outside the cell) is now investigated.

4. Robustness improvement of DPIR

One of the major problem in the initial DPIR algorithm is the one illustrated in figure 5. In some cases the points on the edges M_iM_{i+1} obtained after the first step of DPIR can be to close to the same node leading to a spike when the algorithm tries to balance the volume. It has been observed, on some cases, that the algorithm can't balance the volume since the interface ends up outside the cell.

Three improvements have been made in the DPIR algorithm in order to increase its robustness and are presented below.

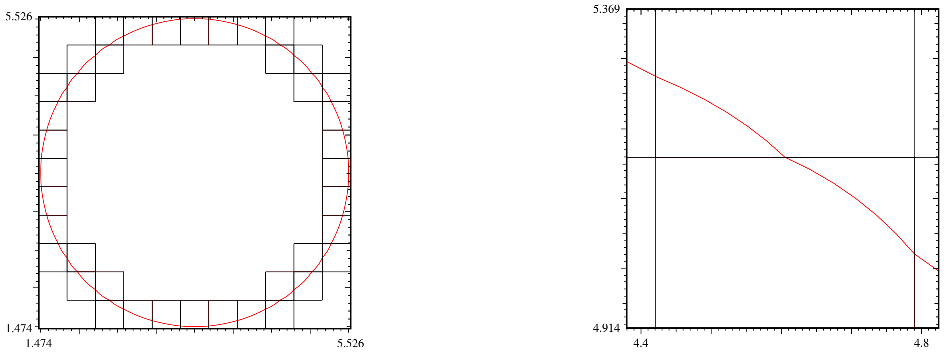


Figure 4. Reconstruction of a circle: zoom around control points in two adjacent mixed cells

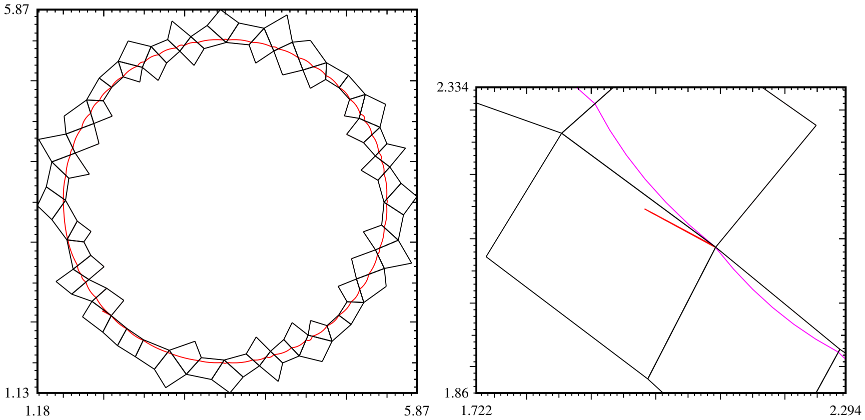


Figure 5. Reconstruction of a circle : some problems with the initial DPIR algorithm.

4.1. A new search direction for the control point

A first way to reinforce DPIR robustness has already been introduced in the previous section. It consists in looking for the control point in the direction of the center of the cell, instead of the segment perpendicular bisector. With this change, it is now possible to have a robust reconstruction, even for a distorted cartesian mesh, as shown on figures 6 and 7 for the respective case of a square and a J shape.

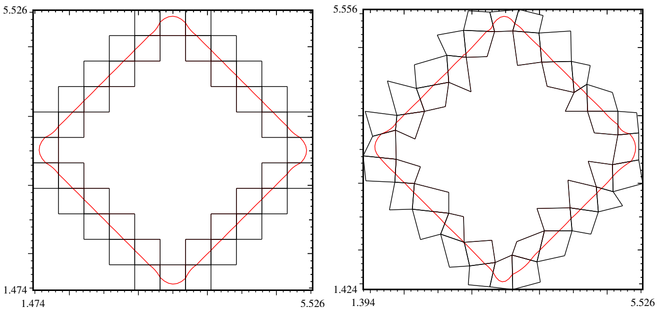


Figure 6. Reconstruction of a square on a distorted cartesian mesh

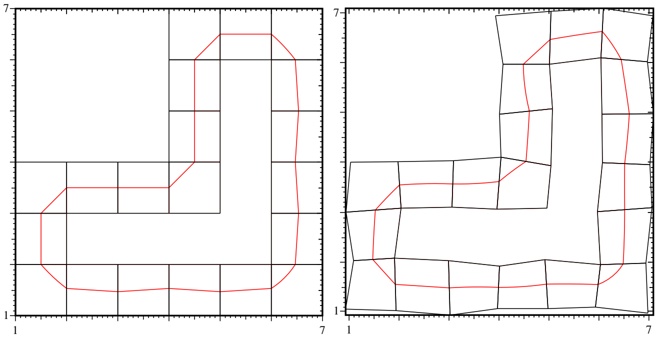


Figure 7. Reconstruction of a J on a distorted cartesian mesh

4.2. A new discretization of the cell edges

A second idea is to change, in the first step of DPIR, the discretization on the segments crossed by the interface: instead of considering uniform discretization, we use Chebyshev points in order to obtain a finer discretization around the corners than in the middle of those segments.

It allows in particular to reduce the number of points on each edge without altering the reconstruction quality.

4.3. A new penalty term

It may happen that for some cells the cost function term $vol(M_i, M_{i+1})$ and the penalty term $\|M_{i+1} - M_i\|$ have very different scale. In order to remedy to this problem, a second penalization term is added to the cost functional in the first step of DPIR. It writes as:

$$\widetilde{p} = \sum_{i=0}^{N-1} \frac{|\mathrm{vol}_{\mathrm{target}} - \mathrm{vol}(M_i, M_{i+1})|}{\mathrm{vol}(M_i, M_{i+1})},$$

that leads to the following minimization problem:

$$\min_{M_0,\dots M_N} \sum_{i=0}^{N-1} |\mathrm{vol}(M_i, M_{i+1}) - \mathrm{vol}_{\mathrm{target}}|^2 + \lambda \|M_{i+1} - M_i\| + \widetilde{p}.$$

In this way, if the volume fraction between M_i and M_{i+1} is too small with respect to the correction, the second penalization $\widetilde{p}$ becomes big and those points are not chosen by the minimization.

The result after the implementation of these three improvements, for the same case of circle reconstruction, is shown in figure 8. It can be seen that all the problems visible on figure 5 have disappeared.

5. DPIR extension to triple point reconstruction

An extension of DPIR for the interface reconstruction of a three (or more) materials case is presented here. Before that, some definitions are recalled.

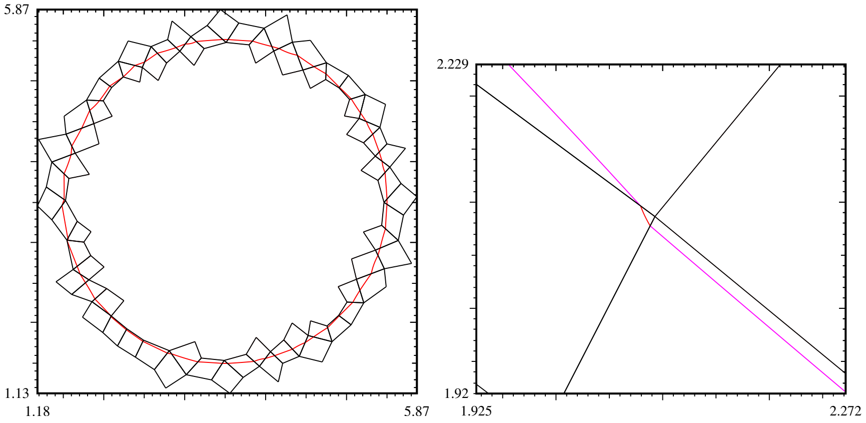


FIGURE 8. Reconstruction of a circle : correction (with $\tilde{p}$).

5.1. Classification of mixed cells and triple point definition

A cell with exactly two strictly positive volume fractions is called a *double cell*. We call *triple cell* a cell of the mesh with three strictly positive volume fractions only. A triple cell with edges crossed not more than once by a material is called a *simple triple cell*. If a single material crosses twice or more a fixed edge of a triple cell, we say that this triple cell contains a *filament*.

In a simple triple cell, we can suppose that it exists a crossing point between all the interfaces between one material and all the others in the cell. We call this crossing point *a triple point*.

5.2. The new algorithm for interface reconstruction

The method we describe below is only suited in the case where the mesh is made of double cells and simple triple cells. It is still divided in two steps : a first step leading to an interface prediction by running three Dynamic Programming algorithms, and a second step for a local correction of all mixed cells.

5.2.1. *First step: one-against-all approach*

A) Consider each material against all the others. Run the dynamic programming step of DPIR on all those materials. The result is shown in Fig. 9.

B) Every double cell is composed of edges with exactly 2 or 0 interface points (from outer and inner materials). Average those two points and fix the new resulting point as the interface final interface point (see Fig. 10).

C) Every simple triple cell has three edges crossed by two interfaces. Glue the interfaces as previously and compute the temporary triple point as the barycenter of the triangle induced by those three new points (see Fig. 10).

5.2.2. *Second step: mixed cells correction*

Each mixed cell is divided in different sub-cells whose number depends on its type (two for double cells and three for triple cells) after this first step. We use a different correction algorithm depending on the type of mixed cell. If the mixed cell is a double cell, we simply use the standard correction step of DPIR in order to correct the partial volumes.

In the case of a simple triple cell, the objective is to obtain an interface regardless of the order of materials in the correction step. We describe here an algorithm that realizes such a correction.

- Compute signs of correction (decide if a material needs to increase or decrease its volume by evaluating the sign of the difference between the current volume in the sub-cell and the targeted volume)

- Correct the material that has a correction sign different of the two others by moving the triple point (considered here as a control point) in the direction given by the leading direction (that is the mean between the two correction directions of sub-cells that have the same correction sign)
- Correct the partial interface between the two others materials using the standard DPIR correction step (adding a control point on the sub-segment). The result of this step is illustrated in Fig. 11.

5.2.3. *Complexity*

The complexity of the algorithm is equal to M times the complexity of DPIR, where M is the number of materials. As all one-against-all DPIR executions are independent, they can be executed in parallel. The correction step only involves very local corrections, that are independent. There are two different options :

- from the edge point of view: average contributions of computed temporary interface on each edge (that can be done in parallel) and correct the volume into each cell (possibly in parallel);
- from the cell point of view: average all interface contributions on each edge of the cell crossed by an interface and correct the volume into each cell (possibly in parallel).

Each local correction involves a (small) dichotomy search optimization that is $\mathcal{O}(log_2(|C|/\epsilon))$, with ϵ the requested precision of this optimization and $|C|$ the diameter of the largest mixed cell of the mesh. If we denote by T the number of mixed cells, the total complexity of the correction step is then $\mathcal{O}(Tlog_2(|C|/\epsilon))$ (triple cells only requires 2 dichotomies).

5.3. **A example of DPIR reconstruction on a triple point configuration**

The new DPIR algorithm is applied to the interface reconstruction of a classical triple point test case where materials 1 and 2 are located inside a half sphere and material 3 outside this sphere. The results are displayed on figures 9, 10 and 11.

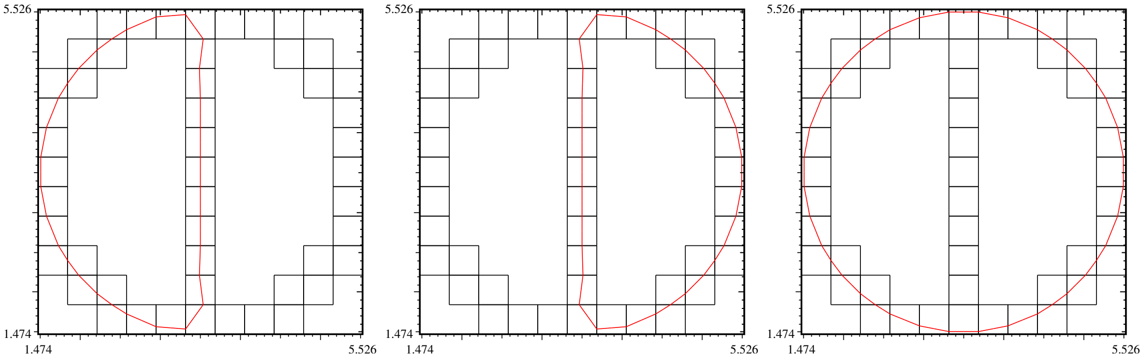


Figure 9. A triple point configuration: three interfaces obtained after the Dynamic programming execution, introduced in section 5.2.1, point A).

With the initial penalization on the interface length, results of dynamic programming on double cells are almost the same before the averaging step. If there is no penalization, oscillations may appear on straight lines. When it happens, the averaging correction compensates those oscillations and allows to recover straight lines.

Due to the individual treatment of materials, the position of interface points on the edges of triple cells is not optimal (even if we can correct partial volumes in it). In particular, a local pushing effect for curved interfaces has been observed, depending on the curvature.

Note that this new algorithm has also been successfully applied for some cases with four materials (see figure 12).

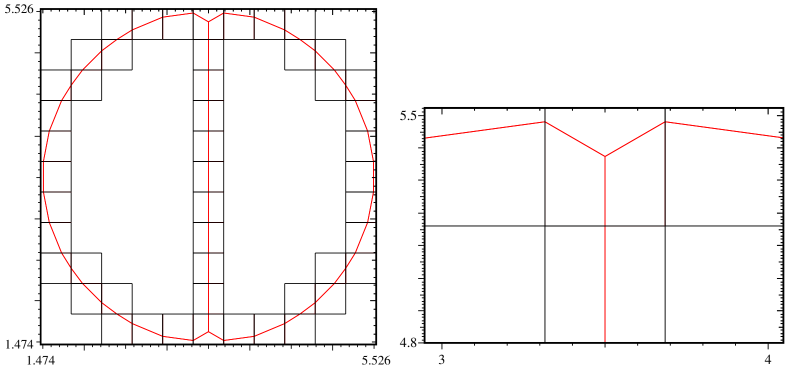


Figure 10. A triple point configuration: result after the first step, see section 5.2.1, points B) and C).

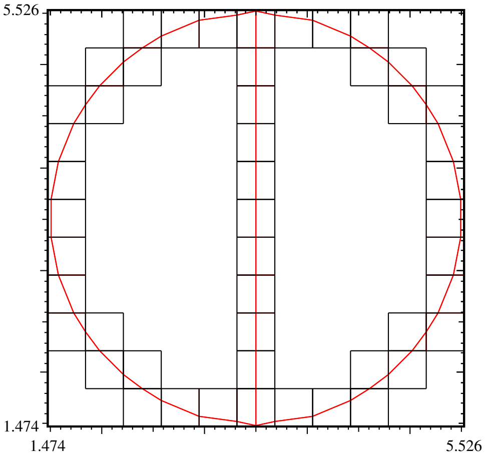


Figure 11. A triple point configuration: final result after the second step, see section 5.2.2.

5.4. **Filament issue**

A filament is a portion of the interface of a given material that crosses twice a same edge. When it appears, one has to detect it before applying the algorithm. An example of a filament issue is presented on Fig. 5.4 : in this case, there exist two adjacent triple cells. The algorithm is not able to detect which cell has the triple point and which one has not. Moreover, the method is not able to reconstruct the interface in the case of an edge crossed twice by the interface itself, i.e. a filament, as the red one on the Fig. 5.4 does. At this point, what happens is that the triple point is assigned randomly to one of those two triple cells and it is not treated correctly. A test case where this situation occured is, for instance, the one illustrated in Fig. 14.

The advantage is that we are able to detect wether there are two adjacent triple cells. Once detected, the idea is to split the edges crossed by the filament and to add a single edge at its extremity. Then the partial volumes are equally distributed on each cell and a single DPIR procedure is applied on each filament (note that the ending point is arbitrarily chosen). Then, the filament contribution is added to the rest of the reconstruction. Up to now, only isolated filaments have been successfully reconstructed using this technique.

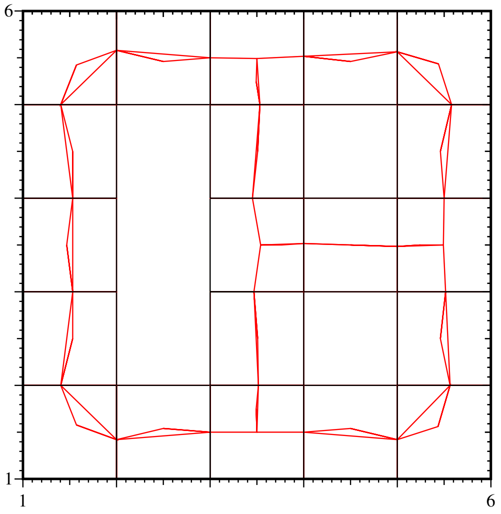


Figure 12. Interface reconstruction in a case with 4 materials.

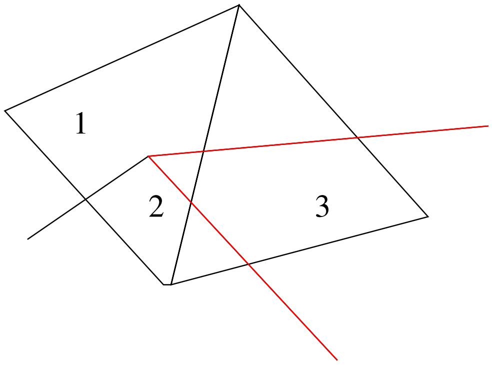


Figure 13. Example of filament. The numbers 1,2,3 indicate the three different materials.

6. Conclusion and perspectives

A new and extended version of DPIR algorithm has been developed that preserves its major advantages, namely the continuity of the interface and the exact conservation of volumes. It can now be applied on more general unstructured grids, and for three (or more) materials cases. In this case, we have focused on the approach Material-DPIR (see [?] for the Material Level-Set Model), but it could be interesting to construct an Interface-DPIR algorithm (as for the Interface Level-Set Model). Moreover, the robustness of DPIR has been enhanced with various techniques, at almost no additional cost. More test cases and a coupling of this new version to an hydrodynamic code with ALE remap need to be tackled in order to fully validate this tool.

7. Acknowledgment

This work has received financial support from the institut des sciences du calcul et des données of Sorbonne Université.

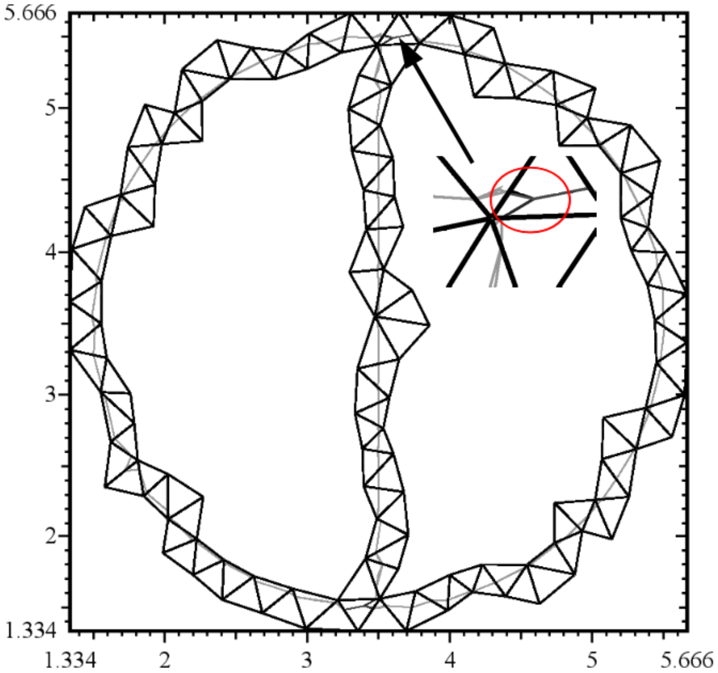


Figure 14. A triple point configuration with a filament issue.

References

[1] L. Dumas. A new volume-preserving and continuous interface reconstruction method for 2d multimaterial flow. *Int. Journ. for Num. Met. in Fluids*, 1(1):1–2, 2017.

[2] S. Osher and R.P. Fedkiw. Level set methods: an overview and some recent results. *Numerical Methods for Fluid dynamics*, 169:463–502, 2001.

[3] D.L. Youngs. Time dependent multi-material flow with large fluid distorsio. *Numerical Methods for Fluid dynamics*, pages 273–285, 1982.