V European Conference on Computational Fluid Dynamics
ECCOMAS CFD 2010
J. C. F. Pereira and A. Sequeira (Eds)
Lisbon, Portugal,14-17 June 2010

A PLIC-VOF IMPLEMENTATION ON PARALLEL 3D UNSTRUCTURED MESHES

Lluís Jofre*, Oriol Lehmkuhl† *, Jesús Castro* and Assensi Oliva*

*Centre Tecnològic de Transferència de Calor, Universitat Politècnica de Catalunya,
ETSEIAT, Colom 11, 08222 Terrassa (Barcelona), Spain
e-mail: cttc@cttc.upc.edu
†Termo Fluids, S.L.
Magí Colet, 8, 08204 Sabadell (Barcelona), Spain
e-mail: termofluids@yahoo.es

Key words: Free Surfaces, Volume Tracking, Volume of Fluid method, Unstructured Mesh, Interface Reconstruction, Advection

Abstract. *The numerical simulation of interfacial and free surface flows is a vast and interesting topic in the areas of engineering and fundamental physics, such as the study of liquid-gas interfaces, formation of droplets, bubbles and sprays, combustion problems with liquid and gas reagents, study of wave motion and others.*

One of the most powerful and robust methods for interface simulation in fixed grids is the Volume-of-Fluid (VOF). In this method, the fluids are represented by a scalar field C_k, known as volume fraction, that represents the portion of volume filled with fluid k. Given a velocity field, interfaces are then tracked by evolving fluid volumes in time. At any time in the solution, an exact interface location is not known. Interface geometry is instead inferred (based on assumptions of the particular algorithm) and its location is reconstructed from local volume fraction data (Interface Reconstruction). The reconstructed interface is then used to compute the volume fluxes necessary for the convective terms in the volume evolution equation (Advection).

The objective of this work is to implement a fast, accurate and parallelizable VOF method well suited to 3D unstructured meshes. The selected interface reconstruction algorithms will be the Youngs' (first order) and the LVIRA (second order). In the other hand, the advection step will be computed by the means of an unsplit-advection volume tracking algorithm.

In the paper, the VOF method will be tested for different test problems. First, a study of reconstruction accuracy, it is most easily assessed by analyzing the reconstruction of known geometries, such as a hollowed sphere. Second, a rotation test, where a velocity field is imposed and the advection algorithm is tested.

1 INTRODUCTION

The contact of immiscible fluids or phases in motion, produces a thin region that separates them called interface. This kind of flows are called free surface or interfacial flows and are found in fields as varied as engineering, fundamental physics, geophysics and others. Typical examples of this phenomena are bubbles, drops, sprays, jets, waves, clouds, etc.

In the last decades, due to continuous improvement of computational power, numerical techniques are becoming more important to study interface phenomena and understand the physics of such flows[1]. Computational fluid dynamics makes feasible to study bubble formation, droplet collision, interfacial instabilities and heat transfer, wave motion, etc.

1.1 Resolution of interfacial flow methods

There are different methods for interface tracking, developed over the decades for specific problems, but they are classified in three main classes:

- fixed-grid: interface cuts across the fixed grid. Continuum Advection[2]; Level Set[3]; Volume Tracking (Volume-of-Fluid) methods[4,5,6,7].

- moving-grid: interface is a boundary between two subdomains of the moving grid. Lagrangian[8]; Free Lagrangian[9].

- grid-free: no grid is needed. Particles[10]; Marker methods[11,12,13].

Among the different methods, the Volume-of-Fluid (VOF) is one of the most widely used and successful in computational fluid dynamics simulation of interfacial and free surface flows. The VOF method preserves mass in a natural way, presents no problem for reconnection or breakup of the interface and it works well in parallel computers[1].

1.2 Survey of documented volume tracking methods

The first Volume-of-Fluid methods were introduced in the 1970s. Three of the most important implementations were DeBar[4], Noh and Woodward[14] and Hirt and Nichols[5]. Each of these methods chose a different reconstructed interface geometry: the DeBar algorithm used a piecewise linear approximation, Noh method used a piecewise constant approximation and the Hirt algorithm used a piecewise constant/stair-stepped approximation. They all used an operator split advection for time integration.

The following implementations were based on the DeBar's piecewise linear approximation and introduced several improvements in interface reconstruction and in time integration. The Youngs'[15] method was first-order accurate for interface reconstruction and Rider and Kothe[16,17] introduced a formulation for general 2D meshes and unsplit advection for time integration. Improvements were made to accomplish second-order accurate approximation to the interface by Puckett[18] and Pilliod[19].

In recent years, improvements on second-order accuracy interface reconstruction, unsplit time integration and computational costs have been accomplished for 3D interfacial flows. Liovic et al.[20] and Hernández et al.[21] proposed two interesting unsplit 3D advection algorithms and Miller and Colella[22] introduced a second-order efficient 3D interface reconstruction method on structured meshes.

2 VOLUME OF FLUID METHOD

The colour function $C_k(\vec{x})$ of fluid k is defined as an identity function

$$C_k(\vec{x}) = \begin{cases} 1 & \text{if there is fluid k} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Then, the discrete volume fraction C_k associated to the characteristic function of the kth fluid for a general volume is calculated as

$$C_k = \frac{\int C_k(\vec{x}) dV}{\int dV} \tag{2}$$

where C_k ranges from 0 to 1 and its sum over all k is unity. The volume of the kth fluid V_k and the total volume V are given by

$$V_k = C_k V; \qquad V = \sum_k V_k \tag{3}$$

The Volume-of-Fluid mathematical formulation starts from the mass conservation equation

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{u}) = 0 \tag{4}$$

where ρ is the fluid density and $\vec{u}$ is the fluid velocity. Introducing the definition $\rho = \sum_k C_k \rho_k$, assuming that $\vec{u}_k = \vec{u}$, and substituting these definitions into Eq. (4), an evolution equation is obtained for each volume fraction C_k:

$$\frac{\partial C_k}{\partial t} + \nabla \cdot (C_k \vec{u}) = 0 \tag{5}$$

Integrating Eq. (5) over cell volume, V_c, and using a first-order time discretization gives

$$V_k^{n+1} - V_k^n + \delta t \int_{V_c} \nabla \cdot (C_k \vec{u}) dV = 0 \tag{6}$$

where superscript n refers to discrete time level and δt is the time step. Invoking Gauss's theorem and assuming incompressible fluid, $\nabla \cdot \vec{u} = 0$, Eq. (6) can be approximated discretely as

$$V_k^{n+1} - V_k^n + \sum_f \delta V_{k,f}^n = 0 \tag{7}$$

where subscript f refers to a face cell, the area integral has been converted to a discrete sum over control volume faces and $\delta V_{k,f}$ is the kth volume flux across face f. The total volume flux across face f is given by $\delta V_f = \vec{u}_f \cdot \vec{n}_f A_f \delta t = \sum_k \delta V_{k,f}$; where $\vec{u}_f$ is the velocity at face f, $\vec{n}_f$ is the face unit-outward-facing normal vector and A_f is the face area. Dividing Eq. (7) by the cell volume, V_c, gives the volume fraction evolution equation:

$$C_k^{n+1} - C_k^n + \frac{1}{V_c} \sum_f \delta V_{k,f}^n = 0 \tag{8}$$

Volume-of-Fluid methods essentially solve Eq. (8) for each fluid k. The most important aspect is the geometrically-based estimation of the kth fluid volume fluxes, $\delta V_{k,f}$, calculated by truncating the total volume fluxes, δV_f, by an interface reconstruction of the kth fluid.

Volume tracking algorithms consist of two main steps: Interface Reconstruction and Interface Advection. The details of each of these steps are presented in Sec. (3) and (4).

3 INTERFACE RECONSTRUCTION

In piece-wise linear interface calculation (PLIC) methods, the interface between two fluids is represented, for each cell, by a plane

$$\vec{n} \cdot r - d = 0 \tag{9}$$

where r is a point in the plane, $\vec{n}$ is the unit normal to the plane, and d is the signed distance from the origin to the plane.

The principal reconstruction constraint is local volume conservation, i.e. the reconstructed interface must truncate the cell verifying

$$C_k = \frac{V_k}{V_c} \tag{10}$$

where C_k and V_k are the volume fraction and volume of fluid k of the cell c, and V_c is the volume of the entire cell.

Since a unique interface configuration does not exist, the interface geometry must be inferred based on local data and the assumptions of the particular algorithm. PLIC methods differ in how the normal $\vec{n}$ is computed. For a given normal, d is uniquely defined from Eq. (10).

3.1 Youngs' method

In the Youngs interface reconstruction method[15], the interface normal, $\vec{n}$, is computed by approximating the gradient of the volume fraction function C_k as

$$\vec{n} = -\nabla \cdot C_k \tag{11}$$

In the case of a 3D unstructured mesh consisting of generalized polyhedra, it is convenient to use a least squares gradient procedure[23].

To complete the interface reconstruction, one needs to find the constant d in Eq. (9) such that intersection of the corresponding half-space and cell c satisfies Eq. (10). This is accomplished using the Brent's method[24].

3.2 Least squares volume-of-fluid interface reconstruction algorithm

In the LVIRA interface reconstruction method[19], the interface normal, $\vec{n}$, is computed by minimizing the error functional

$$E(\vec{n}) = \left(\sum_{nb} (C_{k,nb}{}^{ref} - C_{k,nb}(\vec{n}))^2 \right)^{1/2} \tag{12}$$

where subscript nb refers to the neighbour cells around cell c, $C^{ref}_{k,nb}$ is the reference volume fraction of neighbour nb and $C_{k,nb}(\vec{n})$ is the actual (reconstructed) volume fraction of neighbour nb taken by extending the interface of central cell c, under the constraint that the corresponding plane exactly reproduces the volume fraction in the cell under consideration, see Fig. (1).

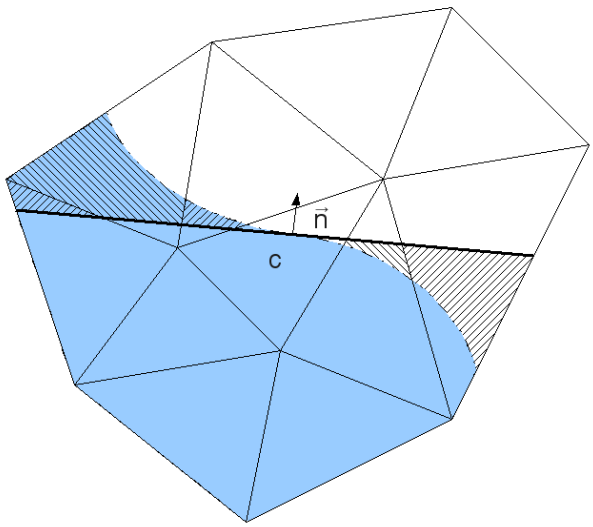

Figure 1: LVIRA error representation on a 2D unstructured mesh. Given a central cell c, its plane interface reconstruction (boldface line) is extended to the neighbour cells and for each one $C_{k,nb}{}^{ref} - C_{k,nb}(\vec{n})$ is calculated (hatched areas) and added to LVIRA error $E(\vec{n})$.

The normal $\vec{n}$ can be described by polar coordinates, therefore, LVIRA implementation requires an algorithm for the minimization of a non-linear function of two variables, e.g. the Powell's method[24].

LVIRA is much more computationally expensive than Youngs, since for every cell c an error has to be minimized, but it is second-order accurate (reconstructs planar interfaces exactly) while Youngs is just first-order.

4 INTERFACE ADVECTION

After reconstructing the interface, Sec. (3), volume fractions for the kth fluid are advected forward in time as described in Eq. (8). There are two main procedures for interface advection in multiple spatial dimensions: direction-split and unsplit.

Direction-split advection consists on a separated time integration for each dimension $C^n \rightarrow_x C^* \rightarrow_y C^{**} \rightarrow_z C^{n+1}$. For each step, one-dimensional volume fluxes are generated by considering individual velocity vector components, then, volume fractions are updated and interface is reconstructed.

Unsplit advection realizes the time integration in one step $C^n \rightarrow_{x,y,z} C^{n+1}$, taking into account velocity vector components acting in all directions and updating volume fractions and reconstructing interfaces just once.

Direction-split volume tracking is easier to implement for 3D flows than unsplit advection, but generates a direction splitting error[17] and requires at least three interface reconstruction sweeps to complete a volume tracking update. Therefore, unsplit advection is more accurate and faster than direction-split but its implementation is more difficult.

4.1 A new 3D unsplit advection algorithm

The method proposed starts from the volume fraction evolution equation, Eq. (8), uses an unsplit advection time integration scheme, and assumes that the interface has been reconstructed, Sec. (3). The algorithm will be presented step-by-step and is applied to each cell:

1. Calculate the volume flux through every face of the cell

$$\delta V_f = \vec{u}_f \cdot \ \vec{n}_f A_f \delta t \tag{13}$$

   where $\vec{u}_f$ is the velocity at the face, $\vec{n}_f$ is the unit normal vector to the face pointing out of the cell, A_f is the face area and δt is the time step.

2. Construct a flux polyhedron of volume δV_f at each face of the cell.

3. Truncate the faces' volume fluxes, δV_f, by the reconstructed interfaces to get the kth fluid volume fluxes across the face, $\delta V_{k,f}$.

4. Calculate the new kth fluid volume fraction, C_k^{n+1}, by evaluating Eq. (8).

Steps 2 and 3 are the basic and most important steps of the algorithm, they are described in detail in Sec. (4.1.1) and (4.1.2), respectively.

4.1.1 Construction of faces' volume fluxes

As described in Sec. (4.1), it is necessary to construct a volume flux polyhedron for each face. The flux polyhedron proposed in this work satisfies:

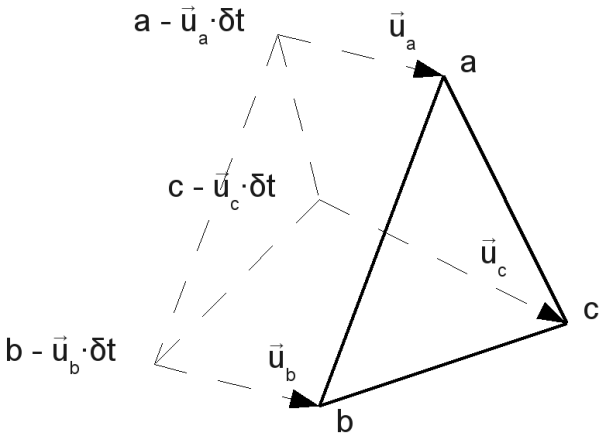

Figure 2: Face's volume flux polyhedron. The polyhedron is constructed by using the vertexes of the face (a,b,c) and by tracing back the Lagrangian trajectories for time δt of the face's vertexes ($a - \vec{u}_a \delta t$, $b - \vec{u}_b \delta t$, $c - \vec{u}_c \delta t$).

- Non-overlapping: to avoid overlapping, as explained by Rider[16], the polyhedron is constructed from the velocities at the vertexes of the face. The vertexes of the polyhedron, see Fig. (2), consist of: the vertexes of the face (a,b,c) and the traced back Lagrangian trajectories for time δt of the face's vertexes ($a - \vec{u}_a \delta t$, $b - \vec{u}_b \delta t$, $c - \vec{u}_c \delta t$).

- Volume adjustment: in order to verify Eq. (13), the polyhedron's volume is adjusted. The back face, created by $a - \vec{u}_a \delta t$, $b - \vec{u}_b \delta t$ and $c - \vec{u}_c \delta t$, is a non-planar surface, then, when approximating the surface by a set of planar surfaces the centroid of this back face can be readjusted to create a flux polyhedron with a volume equal to δV_f.

The constructed flux polyhedron, in the case of 3D flows, is a set of non-planar surfaces that create a non-convex polyhedron. The non-planar surfaces are approximated by a set of planes and the resulting non-convex polyhedron is split up in several convex figures.

4.1.2 Truncation of faces' volume fluxes

Once the face volume fluxes are constructed, they have to be truncated by the interface reconstruction to get the kth fluid volume fluxes across the faces, $\delta V_{k,f}$. Given a face, see Fig. (3), its flux polyhedron is constructed (boldface lines), then, for each surrounding

cell (a, b, c, d, e, f, g, h and i) the polyhedron is truncated by the cell faces (continuous lines) and the reconstructed interface (dashed lines), finally, the addition of the individual volumes (hatched areas) corresponds to $\delta V_{k,f}$. When the kth fluid volume fluxes across all the faces have been calculated, Eq. (8) can be evaluated for each cell taking in account that if the fluid enters, the $\delta V_{k,f}$ is positive and negative if it leaves the cell.

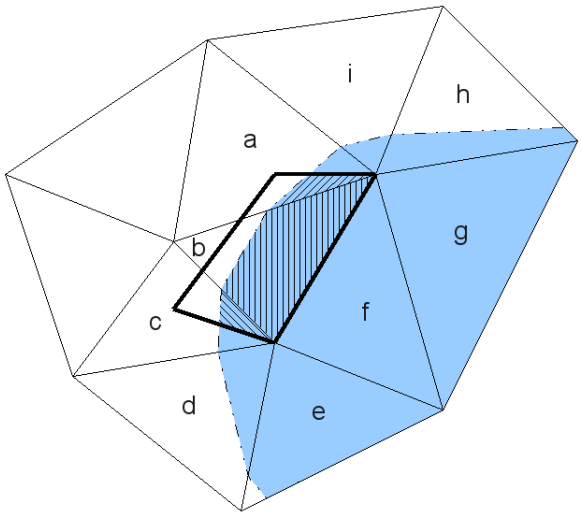

Figure 3: Truncation of the face's volume flux polyhedron to calculate the $\delta V_{k,f}$, represented on a 2D unstructured mesh. Given a face, its volume flux polyhedron is constructed (boldface lines), then, for each surrounding cell (a, b, c, d, e, f, g, h and i) the polyhedron is truncated by the cell faces (continuous lines) and reconstructed interface (dashed lines) and the addition of the individual volumes (hatched areas) corresponds to $\delta V_{k,f}$.

4.2 Algorithm details

The standard time step[17] for structured meshes is defined from the CFL[25] constraint as

$$\delta t < \frac{\text{CFL } h}{u} \tag{14}$$

where the CFL value ranges from 0 to 1, h refers to mesh size and u is a characteristic velocity.

Unstructured meshes require a different approach to calculate the correct time step, δt. Imposing that the volume of the flux polyhedron, Eq. (13), has to be equal to the volume of the cell, V_c, gives

$$V_c = \vec{u}_f \cdot \vec{n}_f A_f \delta t \tag{15}$$

using constant σ, the minimum time step is calculated as

$$\delta t < min \left\{ \frac{\sigma V_c}{\vec{u}_f \cdot \vec{n}_f A_f} \right\} \tag{16}$$

where the σ value ranges from 0 to 1, V_c is the volume of the cell, and $\vec{u}_f$, $\vec{n}_f$ and A_f are the velocity, the outward-unit normal and the area for each face of the cell, respectively. When applying this condition to structured meshes, the equation is equivalent to Eq. (14).

Some errors are introduced to the solution when advecting volumes in time by resolving Eq. (8), generating undershoots ($C_k < 0$), overshoots ($C_k > 1$) and wisps (fluid in void regions or vice versa). These errors are caused basically by discretization errors and velocity fields with non-zero divergence. When any of the previous errors occur is useful to use a local redistribution algorithm like the one proposed by Harvie and Fletcher[26].

5 TESTS

5.1 Reconstruction tests

The hollowed sphere test is used to examine the accuracy of the reconstruction methods presented in Sec. (3). This test problem is stationary; no advection is performed and hence there is no error due to discretization in time. In the test, a sphere of radius 0.4 (convex surface) is initialized in a unitary domain, and a spherical core of radius 0.2 (concave surface) is then hollowed out of it.

The interface reconstruction error is measured as the difference between the exact interface and the reconstructed one. An L_1 error norm is used, which in the continuous limit is the integral[27]

$$E_{L_1} = \int |\chi(\vec{x}) - \tilde{\chi}(\vec{x})| dV \tag{17}$$

where $\chi(\vec{x})$ is the exact interface topology and $\tilde{\chi}(\vec{x})$ is its approximation obtained using an interface reconstruction method.

cartesian	Youngs	LVIRA	unstructured	Youngs	LVIRA
10^3	4.80×10^{-3}	6.76×10^{-3}	10^3	2.62×10^{-2}	3.05×10^{-2}
20^3	1.40×10^{-3}	1.41×10^{-3}	20^3	2.81×10^{-3}	3.30×10^{-3}
40^3	6.36×10^{-4}	4.44×10^{-4}	40^3	9.92×10^{-4}	9.04×10^{-4}
80^3	3.15×10^{-4}	1.73×10^{-4}	80^3	4.74×10^{-4}	2.97×10^{-4}

Table 1: E_{L_1} in interface reconstruction in the hollowed sphere test, using the Youngs and LVIRA schemes on cartesian and unstructured meshes.

The error norm L_1 results for the hollowed sphere reconstructions tests are shown on Tab. (1). The Youngs algorithm exhibits better results for coarse grids (10^3 cells) but LVIRA performs better when the grid is refined (40^3 cells). In the other hand, the LVIRA algorithm is much more time consuming since it needs a 2D minimization. The results for the cartesian meshes are of same magnitude as the ones of the equivalent unstructured meshes, except for the very coarse ones where the cartesian mesh presents better results. The calculation of the interface reconstruction is faster on unstructured meshes than on cartesian ones due to the less number of faces per cell. In order to understand the magnitude of the error, the profiles of the interface reconstruction, cutted by a plane to view the internal part, for the different tests are shown from Fig. (4) to Fig. (7).

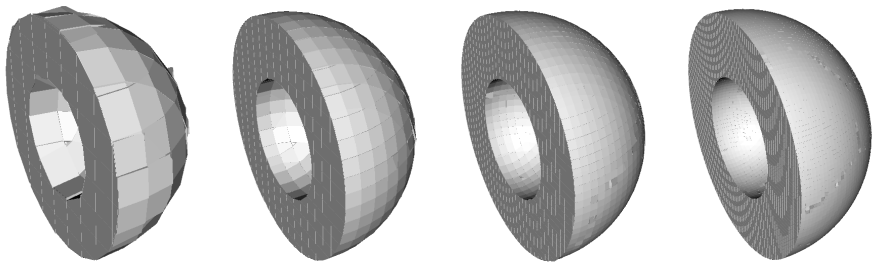

Figure 4: Hollowed sphere reconstruction using Youngs algorithm on a cartesian mesh.

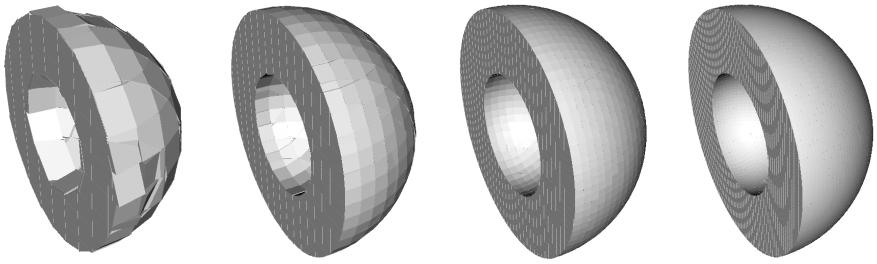

Figure 5: Hollowed sphere reconstruction using LVIRA algorithm on a cartesian mesh.

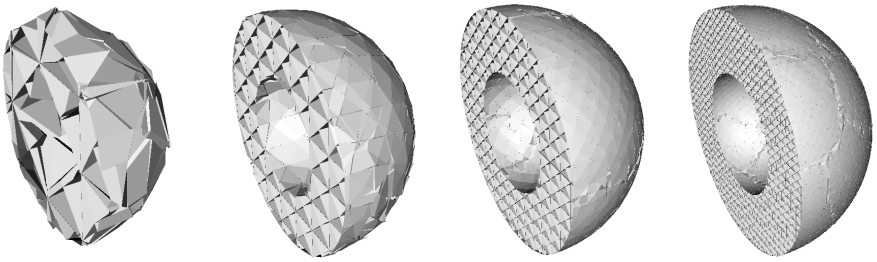

Figure 6: Hollowed sphere reconstruction using Youngs algorithm on an unstructured mesh.

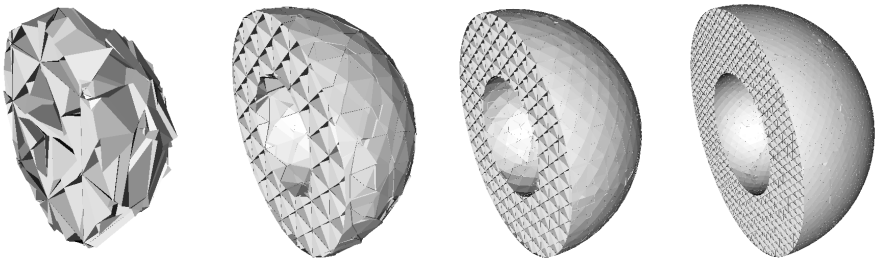

Figure 7: Hollowed sphere reconstruction using LVIRA algorithm on an unstructured mesh.

5.2 Advection tests

The Rotating flow field test is used to verify the implementation of the advection algorithm presented in Sec. (4). The test makes no change in interface topology being ideal to test just the advection algorithm. In the test, a sphere of radius 0.15 and centered at $(0.5, 0.75, 0.5)$ is initialized in a unitary domain, and is advected in a rotational field for the period of time required to return to its initial position. In this paper, a CFL equal to 0.5 will be used.

The advection error is measured as the difference between the initial and after advection colour functions. An L_1 error norm is used, which in the discrete form is the summation[27]

$$E_{L_1} = \sum_c V_c |\tilde{C}_{k,c} - C_{k,c}| \tag{18}$$

where V_c refers to the volume of cell c, and $C_{k,c}$ and $\tilde{C}_{k,c}$ are the colour functions, for fluid k, before and after advection of cell c, respectively.

cartesian	Youngs	unstructured	Youngs
32^3	6.34×10^{-4}	32^3	2.18×10^{-3}
64^3	5.47×10^{-4}	64^3	1.21×10^{-3}
128^3	3.20×10^{-4}	128^3	6.74×10^{-4}

Table 2: E_{L_1} in advection in the rotational flow field test, using the Youngs scheme and a CFL 0.5 on cartesian and unstructured meshes.

The error norm L_1 results for the rotational flow field tests are found in Tab. (2). The Youngs algorithm was the only one used since the LVIRA based on the Powell's method[24] is not efficient enough. The time step was calculated for the different tests from Eq. (16). Cartesian tests were faster due to the less number of neighbours surrounding the faces, as it can be seen in Fig. (3). The reconstructed interfaces for the advection test at times: $T/4$, $2T/4$, $3T/4$ and T, are shown in Fig. (8) and Fig. (9) for cartesian an unstructured meshes, repectively.

The results can be largely improved since some numerical inconsistencies have been found when testing the advection algorithm. There are inaccuracies in some flux polyhedrons when vertex's velocities are tangent to the cell faces. Solving these problems will improve the results and the errors will be reduced as the grid is refined.

When the problems described in the previous paragraph are solved, more advection tests will be performed. The LVIRA algorithm have to be improved, e.g. using the BFGS method[24], to be tested on the advection tests in order to compare the results with the Youngs' ones. Vortical flows, such as the shearing and deformation flow fields[20], will be tested to study the behaviour of the advection algorithm for more demanding situations.

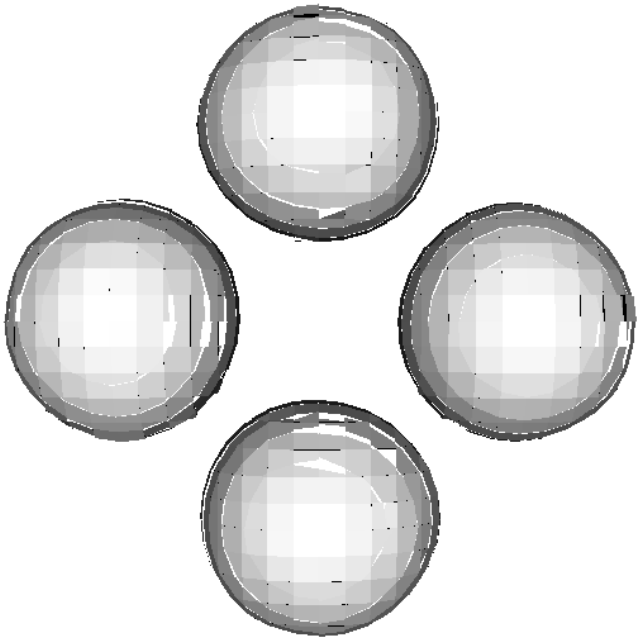

Figure 8: Rotation flow field using Youngs algorithm on a cartesian 32^3 mesh.

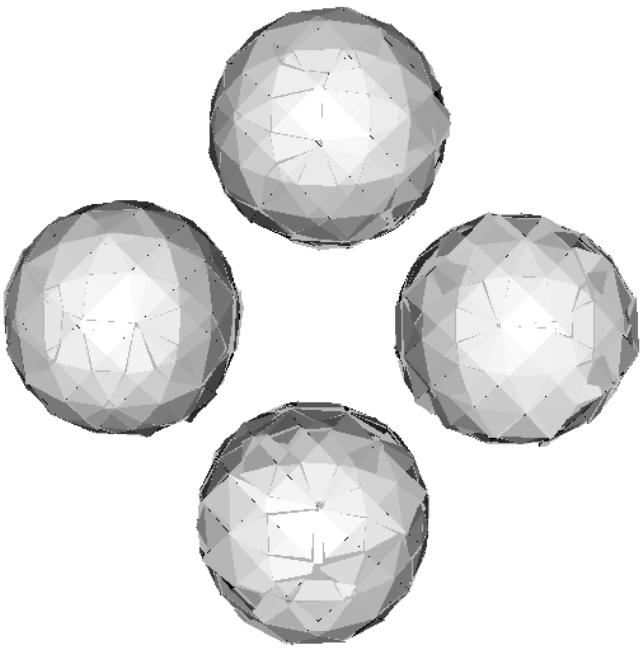

Figure 9: Rotation flow field using Youngs algorithm on an unstructured 32^3 mesh.

6 CONCLUSIONS

A Volume-of-Fluid method based on a new approach for the multidimensional advection has been proposed for tracking interfaces in 3D. For this purpose, a set of simple geometric tasks have been explained in detail. The proposed advection algorithm enforces mass conservation locally by using non-overlapping fluxes and verifying the conservation equation, thus, reducing the formation of undershoots, overshoots and wisps.

The interface reconstruction and advection methods presented have been assessed using different tests. After studying the results some conclusions have been made: the LVIRA error has to be minimized by a more efficient method, e.g. BFGS method[24], and construction inaccuracies have to be solved for some flux polyhedrons when vertex's velocities are tangent to the cells' faces. When these inaccuracies are solved more demanding advection tests will be performed.

ACKNOWLEDGEMENTS

This work has been financially supported by the Ministerio de Educación y Ciencia, Secretaría de Estado de Universidades e Investigación, Spain (ref. ENE2007-67185: Development of high performance parallel codes for the design and optimization of thermal systems and equipments).

References

[1] R. Scardovelli and S. Zaleski. Direct Numerical Simulation of Free-Surface and Interfacial Flow. *Annual Review of Fluid Mechanics*, 31(-):567–603, 1999.

[2] P. Woodward and P. Colella. The Numerical Simulation of Two-Dimensional Fluid Flow with Strong Shocks. *Journal of Computational Physics*, 54(1):115–173, 1984.

[3] M. Sussman, P. Smereka, and S. Osher. A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow. *Journal of Computational Physics*, 114(1):146–159, 1994.

[4] R. DeBar. Fundamentals of the KRAKEN Code. Technical report, Lawrence Livermore National Laboratory, 1974.

[5] C. W. Hirt and B. D. Nichols. Volume of fluid (VOF) Method for the Dynamics of Free Boundaries. *Journal of Computational Physics*, 39(5):201–225, 1981.

[6] L. Leal, J. Castro, P. Pozo, and A. Oliva. Verification and Validation of Volume of Fluid Methods. In *Proceedings of the 3rd International Symposium on Two-Phase Modelling and Experimentation*, pages 1439–1446, 2004.

[7] J. Castro and C. Oliet. Accuracy Evaluation of Volume Tracking Methods for Free Surface Flows. In *Proceedings of the Fifth European Congress on Computational Methods in Applied Sciences and Engineering*, pages 1–2, 2008.

[8] C. W. Hirt, J. L. Cook, and T. D. Butler. A Lagrangian Method for Calculating the Dynamics of an Incompressible Fluid with Free Surface. *Journal of Computational Physics*, 5(1):103–124, 1970.

[9] M. J. Fritts and J. P. Boris. The Lagrangian Solution of Transient Problems in Hydrodynamics using a Triangular Mesh. *Journal of Computational Physics*, 31(2):173–215, 1979.

[10] D. J. Torres and J. U. Brackbill. The Point-Set Method: Front-Tracking without Connectivity. *Journal of Computational Physics*, 165(2):620–644, 2000.

[11] F. H. Harlow and J. E. Welch. Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface. *Physics of Fluids*, 8(12):2182–2189, 1965.

[12] S. O. Unverdi and G. Tryggvason. A Front-Tracking Method for Viscous, Incompressible, Multi-Fluid Flows. *Journal of Computational Physics*, 100(1):25–37, 1992.

[13] J. Glimm, J. W. Grove, X. L. Li, K. M. Shyue, Y. Zeng, and Q. Zhang. 3-Dimensional Front Tracking. *SIAM Journal of Scientific Computing*, 19(3):703–727, 1998.

[14] W. F. Noh and P. R. Woodward. SLIC (Simple Line Interface Calculation). *Lecture Notes in Physics*, 59(-):330–340, 1976.

[15] D. L. Youngs. Time-Dependent Multi-Material Flow With Large Fluid Distortion. In *Numerical Methods for Fluid Dynamics Conference*, pages 273–285, 1982.

[16] W. J. Rider and D. B. Kothe. Reconstructing Volume Tracking. *Journal of Computational Physics*, 141(2):112–152, 1998.

[17] D. B. Kothe, W. J. Rider, S. J. Mosso, J. S. Brock, and J. I. Hochstein. Volume Tracking of Interface Having Surface Tension in Two and Three Dimensions. In *Proceedings of the 34th AIAA Aerospace Sciences Meeting and Exhibit*, pages 1–24, 1996.

[18] E. G. Puckett. A Volume-of-Fluid Method Interface Tracking Algorithm with Applications to Computing Shock Wave Refraction. In *Fourth International Symposium on Computational Fluid Dynamics*, pages 933–938, 1991.

[19] 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, 2004.

[20] P. Liovic, M. Rudman, J. L. Liow, D. Lakehal, and D. Kothe. A 3D Unsplit-Advection Volume Tracking Algorithm with Planarity-Preserving Interface Reconstruction. *Journal of Computational Physics*, 35(10):1011–1032, 2006.

[21] J. Hernandez, J. Lopez, P. Gomez, C. Zanzi, and F. Faura. A New Volume of Fluid Method in Three Dimensions – Part I: Multidimensional Advection Method with Face-Matched Flux Polyhedra. *International Journal for Numerical Methods in Fluids*, 58(8):897–921, 2008.

[22] G. H. Miller and P. Colella. A Conservative Three-Dimensional Eulerian Method for Coupled SolidFluid Shock Capturing. *Journal of Computational Physics*, 183(1):26–82, 2002.

[23] A. Haselbacher and V. Vasilyev. Commutative Discrete Filtering on Unstructured Grids based on Least-Squares Techniques. *Journal of Computational Physics*, 187(1):197–211, 2003.

[24] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C++*. Cambridge University Press, 2002.

[25] R. Courant, K. Friedrichs, and H. Lewy. On the Partial Difference Equations of Mathematical Physics. *IBM Journal of Research and Development*, 11(2):215–234, 1967.

[26] D. J. E. Harvie and D. F. Fletcher. A New Volume of Fluid Advection Algorithm: The Stream Scheme. *Journal of Computational Physics*, 162(1):1–32, 2000.

[27] E. Aulisa, S. Manservisi, R. Scardovelli, and S. Zaleski. Interface Reconstruction with Least-Squares Fit and Split Advection in Three-Dimensional Cartesian Geometry. *Journal of Computational Physics*, 225(2):2301–2319, 2007.