

Feature identification algorithms for sharp-interface flows

By V. Le Chenadec[†], S. Mirjalili, M. Mortazavi AND A. Mani

The implicit representation of interfaces, e.g., by means of a volume fraction field or a Level Set function, renders the identification of non-local topological features such as droplets, ligaments, or sheets, tedious. This report introduces simple concepts and algorithms in the context of the Volume-of-Fluid method that address the underlying challenges in a systematic way. The algorithms and their computational costs are discussed, and applied to the computation of two-point interface statistics in the large-scale computation of a turbulent hydraulic jump.

1. Background

Two-phase flows of immiscible fluids occur in a wide range of phenomena encountered in environmental sciences and engineering; the interest in their prediction is therefore significant. The continuous development of high-performance computing technologies, and the level of prediction reached by the numerical solvers, enable increasingly complex configurations to be tackled. This has consequently spurred the development of numerical algorithms able to carry first-principle simulations of complex interfacial flows. Recent developments have highlighted the benefits of first-principle discretizations in providing accurate solutions of the underlying stiff partial differential equations. In particular, the challenge posed by the presence of an interface has been shown to be effectively addressed by a new generation of geometric Volume-of-Fluid methods, with enhanced flexibility over more traditional one-dimensional schemes (Le Chenadec & Pitsch 2013). The objective of this work is to enhance these capabilities to analyze complex interface topologies, and ultimately to couple first-principle solvers with subgrid-scale models.

First-principle simulations have progressively enabled the scientific discovery of complex two-phase flows. The analysis of the large datasets that are produced, however, remains, for the most part, limited to the application of single-phase flow concepts, and key two-phase specific statistics are currently not available. It is challenging, for example, to infer break-up or coalescence rates from first-principle atomization computations. These quantities, however, are essential to develop and validate lower-fidelity approaches such as ELSA-type approaches (Blokkeel *et al.* 2004). Similarly, droplet impact statistics, although critical to the prediction of surface generation in environmental flows, are simply not available. These deficiencies can largely be attributed to the challenge inherent to their dynamic detection, and the proposed work represents a first step in this direction.

This report focuses on the static analysis of interface topologies in the context of Volume-of-Fluid methods, that is how, given a set of volume fraction values specified over a computational mesh, topological features (droplets, ligaments, ...) may be efficiently identified, and two-point interfacial statistics may be computed.

[†] Department of Aerospace Engineering, University of Illinois at Urbana-Champaign

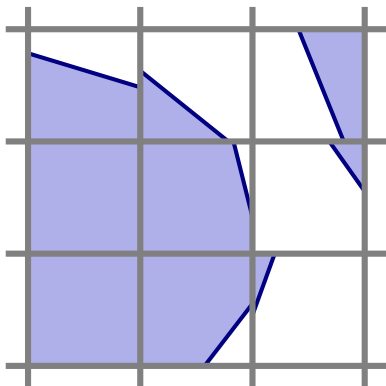


FIGURE 1. Disjoint subsets (dark lines) representing the interface in PLIC-VoF.

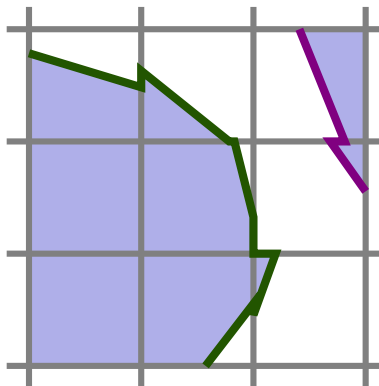


FIGURE 2. Proposed water-tight shell connecting the disjoint PLIC subsets.

2. Motivation and outline

Implicit (or front-capturing) methods identify the interface as an iso-level of a continuous (Level Set) or discontinuous (Volume-of-Fluid) function. For complex turbulent flows, they are advantageous over front-tracking methods which mesh the interface and track it in a Lagrangian fashion, and where topological changes are handled manually as opposed to implicitly. As a result, the notion of distinct contiguous structures (bubbles, droplets, ligaments, . . .), although inherent to the front-tracking technique, does not exist in implicit methods. This notion, therefore, needs to be defined by specifying a set of tagging rules. Applying these rules effectively consists of solving a problem of elliptic nature: the information (e.g. a unique integer identifier per structure) is propagated across all contiguous cells in a sequential manner. As a consequence, for large structures, a very large number of cells will be browsed. For parallel applications, this will result in a significant number of communications (on the order of the number of processes, potentially), which would induce a significant overhead.

In the sharp interface limit, the definition of a liquid structure is unambiguous: it simply consists in a contiguous region of the physical space that is occupied by the liquid phase. At the discrete level, in the context of implicit methods, the raw information available is local: cell-wise in Volume-of-Fluid methods (volume fraction), or point-wise in Level Set methods (signed distance to the interface) for example. The non-local notion of contiguity therefore needs to be restored: an extension in the context of PLIC-VoF is proposed in PLICshell. This definition is then used to perform structure identification (identification), as well as ray tracing to compute two-point interfacial statistics in the simulation of a hydraulic jump (application).

3. Water-tight interface definition

Both tagging and ray tracing algorithms require a mesh of the interface to be built. Depending on the technique used to represent the interface, there may exist challenges in defining a mesh that does not contain any holes. We discuss these issues in the context of PLIC-VoF.

In these methods, the interface is represented as a set of disjoint lower-dimensional

subsets (segments in 2D, polygons in 3D). A variety of techniques have been devised over the years to yield accurate subsets, such as Parker and Youngs' method (Parker & Youngs 1992), the LVIRA method (Pilliod & Puckett 2004), the Coupled Level Set and Volume-of-Fluid method (Sussman 2003), and the Moment-of-Fluid method (Dyadechko & Shashkov 2008) to name a few. The reconstruction algorithm itself is not relevant to the following discussion, and this aspect will therefore not be further discussed.

One of the limitations of PLIC-VoF is the absence of connectivity between the PLIC subsets (Figure 1). Large holes or gaps between the PLIC subsets introduce numerical artifacts such as Flotsam/Jetsam during the transport of an interface, which lead to spurious subgrid-size features that deteriorate the quality of the solution. In the tagging and ray tracing algorithms presented in this work, either these topological singularities should be eliminated, or a work-around needs to be devised to define closed subspaces.

3.1. Construction of a distance function

A first work-around consists in assigning the signed distance from the PLIC subsets to a set of vertices, for example the mesh nodes, and to identify the interface as the zero iso-level of what is effectively a Level Set function. The signed distance to the PLIC subsets is indeed a perfectly well-defined quantity.

There are a few drawbacks to this approach. First, the signed distance from the PLIC subsets, although continuous, is not differentiable and hence not smooth. It also presents inconsistent features when the magnitude of the holes between the PLIC subsets increases. Furthermore, the introduction of a Level Set function does not fundamentally solve the occurrence of non-closed interface mesh, but only shifts the problem to an auxiliary representation. Although one could resort to a stair-step definition following the lattice defined by the vertices, this would not be an accurate representation of the interface. In most circumstances, a higher-order interface representation is suitable, a tessellation of which is not readily available. One remedy is to resort to the Marching Cubes algorithm (Lorenson & Cline 1987), which presents ambiguities (Dürst 1988) that, if not accounted for, also yield holes in the interface mesh.

3.2. Water-tight PLIC shell

Since the construction of a distance function does not circumvent the limitations of the PLIC-VoF, we propose the following definition, which adheres more closely to the PLIC reconstruction. As noted previously, the set consisting of the union of all PLIC subsets is not sufficient to define a closed interface. There exists, however, a simple way to expand this set adequately: it suffices to add the fraction of the mesh lattice (the union of cell faces) which are wet on one side and dry on the other. The resulting set is illustrated in Figure 2, where the latter contribution corresponds to the horizontal and vertical segments.

Despite its simplicity, we found this definition to be the most effective way to form a closed interface around separated structures.

4. Efficient structure identification

In this section, we delineate a systematic way of defining coherent topological structures (droplets, bubbles, sheets, ...). Particular attention is paid to the operation count and scalability in order to yield satisfactory performances.

4.1. Selection rules

In this section we present an algorithm to agglomerate cells according to a set of rules in domain decomposition methods. These rules, defined by the user, appear in the template algorithms 1 and 2 as two callback functions: (i) `tag`; this function, which returns a boolean value, determines whether a cell i is to be agglomerated. For example, if cells containing liquid are to be agglomerated into droplets, `tag` will return true for cells containing some liquid, and false otherwise. (ii) `merge`; determine whether two cells i and j belong to the same cluster. In the case of droplet identification, for example, if i and j share a face, and the intersection of the wetted areas on both sides of the common face is non-empty, `merge` would return a true value. These rules can of course be extended to common edges (in 3D only) and vertices (2D and 3D) by the user.

4.2. Identification algorithm

The identification process is implemented in two stages. First, a serial step is performed over each subdomain independently (see Algorithm 1 in Appendix A). Here, a unique identifier is determined for each subcluster. Subclusters are constructed by performing the `merge` operations over the cells owned by the local process p only (set denoted by Ω^p). No information from the cells adjacent to the subdomain Ω^p (for example, from cells contained in the ghost layer Ω_{ghost}^p) is used at this point. When the number of processes P is larger than unity, multiple instances of an identifier may exist on different processes. To resolve potential conflicts with the second stage described in the next paragraph, the range of identifiers $[[T_{\min}^p, T_{\max}^p]]$ is made unique by introducing a process-dependent shift. This involves a single `mpi_allgather` communication, and provides access to the ranges

$$T_{\min}^1 \leq \dots \leq T_{\max}^p + 1 = T_{\min}^{p+1} \leq \dots \leq T_{\max}^P + 1, \quad (4.1)$$

to all processes.

Second, a parallel step is performed, where a unique identifier is determined for each cluster (Algorithm 2 in Appendix A). This step consists in merging the previously determined subclusters across subdomain boundaries according to the merge rule. For all local subclusters sitting on the subdomain inner boundary Ω_{bound}^p , the list of non-local subclusters connected through the ghost layer Ω_{ghost}^p is created. A new identifier is then introduced for each local subcluster and initialized to the subcluster identifier. For each local subcluster, this identifier is then updated to the minimum of the values associated with each connected subclusters, including the local subcluster itself. This operation is iterated until a stationary value is reached. This, of course, requires pair-wise communications of the updated values (line 14 in Algorithm 2): these communications, however, are symmetric provided the merge rule is also symmetric (that is, $(\text{merge}(i, j) = \text{merge}(j, i))$ which in practice is always the case. Finally, the stopping criterion signaling when unique cluster identifiers have been found also involves communications (`mpi_allreduce`).

We also note that the cell and process neighborhoods (line 11 in Algorithm 1, as well as lines 4 and 14 in Algorithm 2), as a result of the definition of the ghost layer Ω_{ghost}^p and the pair-wise communication patterns (line 14 in Algorithm 2), depend on the merge rule, in particular, whether cells are allowed to connect across vertices (and edges in 3D) in addition to faces.

4.3. Efficiency and flexibility

We conclude this section by highlighting a few important aspects of the proposed procedure. Regarding the output of this algorithm, by construction, the set of cluster identifiers

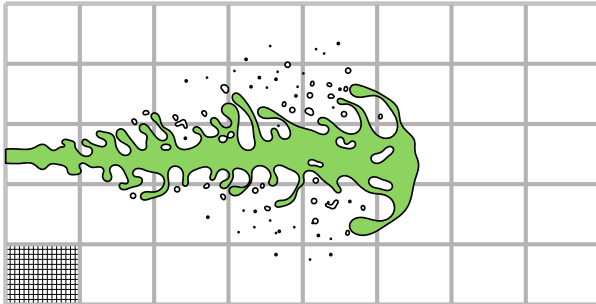


FIGURE 3. Identification of a liquid core in a domain decomposed structured mesh.

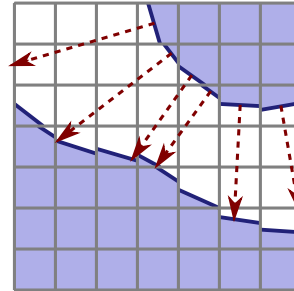


FIGURE 4. Rays traced outwards of the top-left cluster.

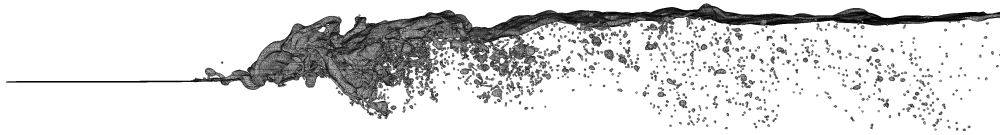


FIGURE 5. Instantaneous PLIC subsets characterizing the interface location.

is a subset of the subcluster identifiers; given the knowledge of the ranges of subcluster identifiers owned by each process (Eq. (4.1)), a simple binary search, of $\log(P)$ cost, can be performed to find a cluster's owner. The parallelism of tasks such as monitoring and outputting cluster statistics (position, momentum, geometric moments, ...) is therefore inherent to the proposed data structures (except for the replicated storage of the P integers defining the ranges). As a result, the gathering of cluster statistics is easily set up with a single `mpi_alltoallv` call, and performed with `mpi_alltoallv` calls.

Although presented in the context of droplet identification, callback functions (tag, merge), supplemented by ray tracing capabilities for example, may also be designed to identify ligaments Herrmann (2005) or thin sheets, a potential application being the coupling to subgrid-scale models Kim & Moin (2011).

An important aspect is, of course, the underlying computational cost which, in distributed applications, is dominated by the communications. In the case of the structured mesh represented in Figure 3 where the computational domain is decomposed into 8×5 blocks, the largest cluster (liquid core) extends horizontally over 6 blocks, and Algorithm 2 is therefore expected to require 6 iterations. This simple example suggests the maximum number of steps to scale like $\mathcal{O}(P^{1/D})$ (D being the physical dimension). Communications are therefore expected to be the bottleneck in large-scale applications. The data structures adopted here have been designed to accommodate recursive coarsening of the domain decomposition in order to achieve logarithmic complexity.

5. Application to a hydraulic jump computation

This section focuses on the application of the proposed developments to the geometric analysis of a hydraulic jump computation (Mortazavi *et al.* 2012). This analysis is motivated by small-scale phenomena, such as the formation of micrometer size bubbles (Mirjalili & Mani 2013), that occur over a range of scales out of reach of current as

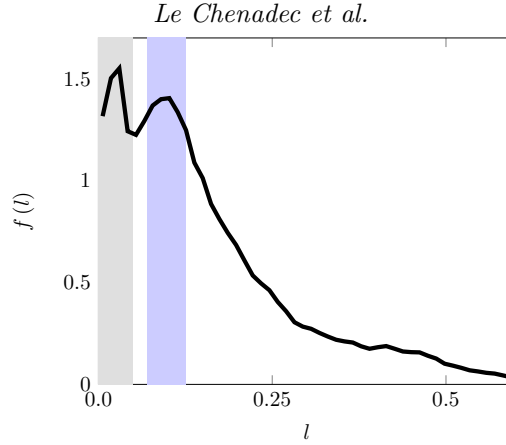


FIGURE 6. Probability density function $f(l)$ in non-dimensional form.

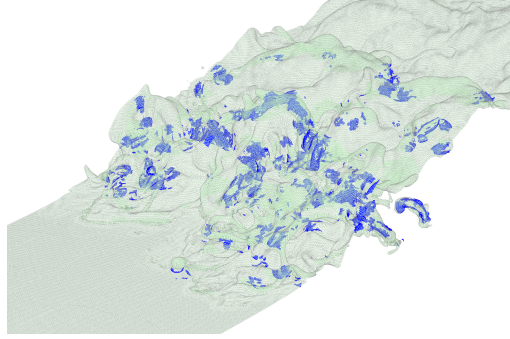


FIGURE 7. Liquid sheets of thickness comprised between $4\Delta x$ and $6\Delta x$.

well as future large-scale computations (Figure 5). The modeling of these phenomena is therefore of significant interest.

5.1. Ray tracing and two-point statistics

This section describes how ray tracing capabilities are leveraged to compute two-point interface statistics, such as the probability $f(l) dl$ for a ray departing from the interface propagating normal to it to cross the interface within a distance range $[l - dl/2, l + dl/2]$. In order to estimate this statistic in the most relevant part of the flow (shown in Figure 7), we first proceed by excluded all interface subsets (liquid and gas) that are not adjacent to the ambient gas and ambient liquid clusters simultaneously. This is done by invoking the identification procedure previously described, where the largest gas and liquid clusters are identified as the ambient clusters.

The probability density function f , shown in Figure 6 in non-dimensional form (the inlet height H of liquid is used for normalization), is computed as follows. For every PLIC subset, a line-of-sight algorithm is used to follow the ray leaving from the PLIC subset center along the interface normal. Once the ray hits the PLIC shell (defined in Section 3), the total length of the segment (shown in red in Figure 4) is computed and the corresponding bin is then incremented by the subset area (the area of the blue subsets in Figure 4). f presents a first peak centered at the mesh spacing. This peak can be attributed to the inaccuracies inherent to the PLIC reconstruction of under-resolved

interfaces. A second peak, highlighted in blue, occurs at $0.1 H$, after which the probability density function decreases monotonically.

5.2. Sheet identification

The identification algorithm may easily be used to agglomerate all neighboring gaseous cells that have been traversed by a ray. It suffices to supply the first callback function tag which assesses whether a cell was traversed by a ray, and merge that connects gas cells according to the PLIC shell definition. The resulting clusters represent gas sheets, which may be used to characterize the dynamics of the film break-up which typically leads to microbubble formation. In Figure 7, the sheets of thickness $5\Delta x \pm \Delta x$ are shown ($\Delta x = H/64$).

A more extensive characterization of the potential impacts requires conditioning on the local curvatures, as well as a mesh of the relative velocity at which they occur. This quantity is actually not straightforward to infer since it is unlikely that the resolution at which the impacts are seen on a snapshot such as the one pictured in Figure 5 is sufficient for the estimate to be accurate (the peak in f occurs ca. $5\Delta x$). Overcoming this hurdle would, however, provide valuable information for phenomenological modeling of the microbubble formation.

6. Conclusion and future work

In this report, we delineated a methodology that was designed to analyze interfacial topologies in complex two-phase flows in a systematic way. A simple yet effective way was proposed of joining the linear segments (planar polygons in 3D) that define the interface topology in PLIC/Volume-of-Fluid methods in a water-tight fashion. This concept was then used to identify coherent interfacial structures such as droplets, bubbles, ligaments or sheets, in a parallel environment.

These capabilities were then used in the context of the large-scale computation of a hydraulic jump. A preliminary analysis of the interface topology in the wave-breaking region was performed by computing low-order, two-point interface statistics. Ongoing work focuses on the extension of this analysis to characterize the dynamics, the objective here being the formulation and coupling of phenomenological models accounting for unresolved subgrid phenomena. The main challenge is that, given the finite precision CFD computations operate at, the detection of break-up and coalescence is defined relative to the mesh spacing. This naturally leads to the question of how to restore relevant subgrid-scale features faithfully in well-resolved simulations.

The tools and analysis performed so far are static, in the sense that the flow variables (include the interface topology) are frozen. Future work will consist in extending these capabilities to on-the-fly processing in order to characterize the dynamics of the fragmentation processes, the surface generation, etc. Based on the coarsening strategy delineated above, the computational cost associated with the communications remains on the order $D^{-1} \log P$, P being the number of processes, D the physical dimension. Additional strategies will also be investigated to reduce it further.

Acknowledgments

The first author would like to acknowledge Profs. Marcus Herrmann and Olivier Desjardins for fruitful discussions regarding the identification of non-local features in interfacial flows. The authors acknowledge use of computational resources from the Certainty cluster awarded by the National Science Foundation to CTR.

Appendix A. Detailed description of the algorithms

Algorithm 1: agglomerate (callback tag, callback merge)

```

1  $N^p \leftarrow 0$ 
2 foreach  $i \in \Omega^p$  do  $T(i) \leftarrow 0$ 
3 foreach  $i \in \Omega^p$  do
4   if !tag( $i$ ) |  $T(i) > 0$  then skip
5    $N^p \leftarrow N^p + 1$  // new non-unique subcluster identifier
6    $T(i) \leftarrow N^p$ 
7   queue  $\leftarrow \emptyset$  // initialize queue
8   queue.push( $i$ )
9   while queue  $\neq \emptyset$  do
10     $j \leftarrow$  queue.pop
11    foreach  $k \in \Omega^p$  neighboring  $j$  do
12      if !tag( $k$ ) |  $T(i) > 0$  | !merge( $j, k$ ) then skip
13       $T(k) \leftarrow N^p$ 
14      queue.push( $k$ )
15    end
16  end
17 end
18 Get  $(N^q)_{q \in \llbracket 1, P \rrbracket}$  from all processes // ALLGATHER communication
19 foreach  $q \in \llbracket 1, P \rrbracket$  do
20    $T_{\min}^q \leftarrow 1 + \sum_{r=1}^{q-1} N^r$ 
21    $T_{\max}^q \leftarrow \sum_{r=1}^q N^r$ 
22 end
23 foreach  $i \in \Omega^p$  do  $T(i) \leftarrow T(i) + T_{\min}^p - 1$  // make subcluster identifier unique

```

Algorithm 2: connect (callback merge)

```

// set of ghost clusters connected to local ones
1 foreach  $t \in \llbracket T_{\min}^p, T_{\max}^p \rrbracket$  do  $C_t \leftarrow \emptyset$ 
2 foreach  $i \in \Omega_{\text{bound}}^p$  do
3   if  $t(i) = 0$  then skip
4   foreach  $j \in \Omega_{\text{ghost}}^p$  neighboring  $i$  do
5     if  $T(j) = 0$  | !merge( $i, j$ ) then skip
6      $C_{T(i)} \leftarrow C_{T(i)} \cup \{T(j)\}$ 
7   end
8 end
// compute unique global identifier for each cluster
9 foreach  $t \in \llbracket T_{\min}^p, T_{\max}^p \rrbracket$  do  $F_t \leftarrow t$ 
10  $\Delta \leftarrow 1$ 
11 while  $\Delta \neq 0$  do
12    $\Delta^p \leftarrow 0$ 
13   foreach  $s \in \cup_{t \in \llbracket T_{\min}^p, T_{\max}^p \rrbracket} C_t$  do
14     get  $F_s$  from neighbor process // symmetric pair-wise communication
15   end
16   foreach  $t \in \llbracket T_{\min}^p, T_{\max}^p \rrbracket$  do
17     foreach  $s \in C_t$  do
18       if  $F_s < F_t$  then
19          $F_t \leftarrow F_s$ 
20          $\Delta^p \leftarrow 1$ 
21       end
22     end
23   end
24    $\Delta \leftarrow \max_q \Delta^q$  // global maximum query
25 end
26 foreach  $i \in \Omega^p$  do  $T(i) \leftarrow F_{T(i)}$ 

```

REFERENCES

- BLOKKEEL, G., DEMOULIN, F. X. & BORGHI, R. 2004 Modeling of two-phase flows: An Eulerian model for Diesel injection. In *Proc. THIESEL Conf.* **2**, 87–105.
- DÜRST, M. J. 1988 Additional reference to "Marching Cubes". *ACM SIGGRAPH Computer Graphics* **22**, 243.
- DYADECHKO, V. & SHASHKOV, M. J. 2008 Reconstruction of multi-material interfaces from moment data. *J. Comput. Phys.* **227**, 5361–5384.
- HERRMANN, M. 2005 Refined Level Set Grid method for tracking interfaces. *Annual Research Briefs*, Center for Turbulence Research, Stanford University, pp. 3–18.
- KIM, D. & MOIN, P. 2011 Numerical simulation of the breakup of a round liquid jet by a coaxial flow of gas with a subgrid Lagrangian breakup model. *Annual Research Briefs*, Center for Turbulence Research, Stanford University, pp. 15–30.
- LE CHENADEC, V. & PITSCH, H. 2013 A 3D Unsplit Forward/Backward Volume-of-Fluid Approach and Coupling to the Level Set Method. *J. Comput. Phys.* **233**, 10–33.
- LORENSEN, W. E. & CLINE, H. E. 1987 Marching cubes: A high resolution 3D surface construction algorithm. *ACM SIGGRAPH Computer Graphics* **21**, 163–169.
- MIRJALILI, S. & MANI, A. 2013 Numerical investigation of air film breakup and micro-bubble formation in liquid-liquid impact events. *Bull. Am. Phys. Soc.* **58**.
- MORTAZAVI, M., LE CHENADEC, V., KIM, D. & MANI, A. 2012 On the importance of the Mesler entrainment mechanism in turbulent breaking waves. *Bull. Am. Phys. Soc.* **57**.
- PARKER, B. & YOUNGS, D. 1992 *Two and Three Dimensional Eulerian Simulation of Fluid Flow With Material Interfaces*. Technical Report, UK Atomic Weapons Establishment.
- PILLIOD, J. E. & PUCKETT, E. G. 2004 Second-order accurate volume-of-fluid algorithms for tracking material interfaces. *J. Comput. Phys.* **199**, 465–502.
- SUSSMAN, M. 2003 A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles. *J. Comput. Phys.* **187**, 110–136.