Title: NUMERICAL RECIPES FOR MOLDING FILLING SIMULATION

Author(s): Doug Kothe
          Damir Juric
          Kin Lam
          Bryan Lally

Submitted to: The 8th International Conference on the Modeling of Casting, Welding and Advance Solidification Processes (MCWASP VIII).
              San Diego, CA, July 1998

Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by the University of California for the U.S. Department of Energy under contract W-7405-ENG-36. By acceptance of this article, the publisher recognizes that the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. The Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.
DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.
DISCLAIMER

Portions of this document may be illegible electronic image products. Images are produced from the best available original document.
NUMERICAL RECIPES FOR MOLD FILLING SIMULATION

Doug Kothe\textsuperscript{1}, Damir Juric\textsuperscript{2}, Kin Lam\textsuperscript{3}, and Bryan Lally\textsuperscript{4}

Los Alamos National Laboratory
Los Alamos, NM, 87545, USA

\textsuperscript{1}Materials Science and Technology Division
Structure/Property Relations Group MST-8, MS-G755
dbk@lanl.gov

\textsuperscript{2}Theoretical Division
Fluid Dynamics Group T-3, MS-B216
djuric@lanl.gov

\textsuperscript{3}Engineering Sciences and Applications Division
Engineering Analysis Group ESA-EA, MS-P946
klam@lanl.gov

\textsuperscript{4}Materials Science and Technology Division
Structure/Property Relations Group MST-8, MS-G755
lally@lanl.gov

Abstract

Has our ability to simulate the filling of a mold progressed to a point where an appropriate \textit{numerical recipe} achieves the desired results? If “results” are defined to be topological robustness, computational efficiency, quantitative accuracy, and predictability, all within a computational domain that faithfully represents complex three-dimensional foundry molds, then the answer unfortunately remains \textit{no}. Significant interfacial flow algorithm developments have occurred over the last decade, however, that could bring this answer closer to “maybe”. These developments have been both evolutionary \textit{and} revolutionary, will continue to transpire for the near future. Might they become useful numerical recipes for mold filling simulations? Quite possibly. Recent progress in algorithms for interface kinematics and dynamics, linear solution methods, computer science issues such as parallelization and object-oriented programming, high resolution Navier-Stokes (NS) solution methods, and unstructured mesh techniques, must all be pursued as possible paths toward higher fidelity mold filling simulations. A detailed exposition of these algorithmic developments is beyond the scope of this paper, hence we choose to focus here exclusively on algorithms for interface kinematics. These \textit{interface tracking} algorithms are designed to model the movement of interfaces relative to a reference frame such as a fixed mesh. Current interface tracking algorithm choices are numerous, so is any one best suited for mold filling simulation? Although a clear winner is not (yet) apparent, pros and cons are given in the following brief, critical review. Highlighted are those outstanding interface tracking algorithm issues we feel can hamper the reliable modeling of today’s foundry mold filling processes.
Introduction And Perspective

Interfacial flows possessing multiple distinct, immiscible fluids bounded by topologically complex interfaces are ubiquitous in natural and industrial processes. The active or passive filling of a mold (e.g., by pressure injection or gravity pour) in a casting process is an excellent example of a complicated industrial interfacial flow. In a typical mold filling scenario, at least four "fluids", each bounded and separated from the other by a discernible interface, are present: the mold material, interpreted as a dense, stationary fluid; the mold cavity material, which is usually vacuum, air, or an inert gas; the molten liquid filling the mold cavity; and the solidified liquid, formed subsequent to mold fill provided adequate cooling has taken place.

Simulation of an interfacial flow process such as mold filling via discrete numerical solution of appropriate partial differential equations (PDEs) is arguably the principal path towards fundamental understanding of these flows. Motivating numerical simulation of interfacial flows is a common "Achilles heel", or analytical intractability: the presence of interfaces that both move with and act back on the local flow field. Interface position is known only at initial time; thereafter it must be determined as part of the overall flow solution. It is not surprising, then, that the quest for accurate numerical simulation of interfacial flows has attracted countless talented researchers over the past half century.

In the numerical simulation of a mold filling process, the mold can be assumed to be relatively stationary, with movement brought about only by residual stress or distortion. Its bounding interface, although often topologically complex, can then coincide with appropriate segments of the (typically unstructured) mesh surfaces partitioning the computational domain. The interface delineating the cavity gas and injected molten liquid, however, is not as easily modeled. During a typical mold filling process, this interface will possess topologies that are not only irregular but also dynamic, undergoing gross changes such as merging, tearing, and filamenting as a result of the flow and interface physics such as surface tension and phase change. The interface topology requirements facing an algorithm required to track the cavity gas/molten liquid interface are therefore formidable. The cavity gas/molten liquid interface generally cannot coincide with the mesh unless the mesh is allowed to move and change topologies arbitrarily. Such Lagrangian mesh capabilities, while attractive, are not typically feasible for practical use foundry process simulation.

From our perspective, schemes that statically partition the computational domain (with a stationary grid) are the basis for a robust method designed to simulate flows possessing interfaces that are topologically complex. When Lagrangian methods are tasked to model such flows, solution accuracy and robustness deteriorates in a manner proportional to the irregularity of the interface topology. Eulerian methods must therefore be embraced. In the recent past, however, it was not clear that Eulerian methods could adequately meet the interfacial flow challenge. The last decade, fortunately, has witnessed remarkable and exciting progress: many talented researchers have been attracted to this problem, and as a result Eulerian interfacial flow methodology has advanced to a point where quantitatively accurate flow solutions can often be realized. The wide applicability of interfacial flow simulations has finally enticed enough people and resources to pay noticeable dividends.

In the next section, we motivate the need for specialized algorithms designed to model cavity gas/molten liquid interface kinematics during mold filling. These algorithms are specifically developed and tuned to track or capture discontinuities. In the following section we take a brief "tour" through most of the current interface tracking algorithms. The review is not specific to a choice of finite difference, finite volume, and finite element methods of
Why Is Mold Filling Simulation So Difficult?

While the answer to this question is probably obvious, consider first the physical phenomena likely to be present in a typical mold filling process:

- unsteady, incompressible (or slightly compressible) flow of multiple, immiscible fluids;
- interface kinematics;
- interface dynamics (surface tension, wetting effects);
- convective, diffusive, and radiative heat transfer;
- solidification of multi-component alloy systems having arbitrary phase diagrams;
- microstructural physics (nucleation, growth, kinetics); and
- material response effects (residual stress, distortion, shrinkage, plastic flow).

While developing models to simulate each individual phenomena above presents formidable challenges, it is the simultaneous occurrence of these phenomena while a mold is being filled that presents the modeling challenge. During the filling of a mold, for example, one region of the casting might contain molten liquid free surface flow, another might be experiencing mushy zone porous flow, while still another contains solidified material that is generating internal stress as it cools. Each phenomena takes place on disparate length scales. Much research remains before a fundamental understanding of these inherently non-linear phenomena and their interplay with one another matures to a point where universal, first-principles models are realized.

Currently, then, mold filling simulations cannot be carried out without a requisite set of modeling assumptions. Standard practice has been to choose some subset of physics perceived relevant to the problem at hand, model the unresolvable (microstructural) length scales, and assume certain physical phenomena take place sequentially rather than simultaneously. For example, one might neglect material strength effects (item one), model alloy species microsegregation with an averaged model (item two), and simulate the mold filling and cavity solidification separately with two different computational tools (item three). Comprehensive computational tools based upon less restrictive modeling assumptions are needed in the future. This is particularly true with regard to which physics can and cannot be "allowed" to take place simultaneously. Any present and future mold filling simulation tool must be able to adapt quickly and easily to changes in physical models and numerical algorithms.

Now, back to the issue of modeling interface kinematics. Relative to the host of physical phenomena present during a mold fill process, can the motion and topology of the cavity gas/molten liquid interface really play a large role in determining the final characteristics of the cast part? In short, yes. This is especially true, however, if mold cavity geometries are complex and heat transfer rates are such that appreciable cooling (and/or solidification) of the molten liquid during fill can take place. If the cavity gas/molten liquid interface
Kinematics are not modeled correctly during mold fill, then the resulting erroneous interface topology and history could lead to flawed heat transfer predictions along the mold cavity boundary. Furthermore, since the distribution of (any) porosity and molten liquid momentum, energy, alloying species begins with the mold fill process, these could also be modeled incorrectly if the interface tracking algorithm is not a faithful representation. An accurate and reliable mold filling simulation therefore starts with a high fidelity interface tracking algorithm.

**Interface Tracking Algorithms: A Tour**

What specific capabilities might one design into an interface tracking algorithm targeted for mold simulation? The list is long, but no longer than what other interfacial flow simulations might require. Our design requirements are not excessive, yet they are formidable. To date no one algorithm discussed in this paper (or any to our knowledge) adequately meets each and every design requirement itemized randomly below. For high fidelity mold filling simulations, we desire an interface tracking algorithm that:

- **R1.** is globally and locally mass conservative;
- **R2.** maintains (at a minimum) second order temporal and spatial accuracy;
- **R3.** maintains compact interface discontinuity width;
- **R4.** is topologically robust;
- **R5.** is amenable to three-dimensions;
- **R6.** is amenable to meshes of arbitrary element type and connectivity;
- **R7.** can accommodate additional interfacial physics models;
- **R8.** can track interfaces bounding more than two materials;
- **R9.** is computationally efficient;
- **R10.** can be implemented by novices (given ample documentation); and
- **R11.** can be readily maintained, improved, and extended.

We could add the requirement that the algorithm be *implicit* in time, but this is unnecessary in most mold filling situations because resolution of the dynamics is desired. Implicit tracking algorithms are also difficult to implement efficiently.

Interface tracking algorithms generally fall into one of two methods categories, with each category containing several schemes (discussed in the next section):

- *Tracking* methods such as moving-mesh, front tracking, vortex dynamics, and particle schemes;
- *Capturing* methods such continuum advection, volume tracking, level set, and phase field schemes.

A tracking method is Lagrangian in nature, whereby the position history of discrete points $x_i$ lying on the interface are tracked for all time by integrating the evolution equation

$$\frac{dx_i}{dt} = V_i,$$  \hspace{1cm} (1)
where $V_i$ is the velocity with which interface point $x_i$ moves. For moving-mesh and front tracking schemes, the points $i$ correspond to the discrete points on the grid line representing the interface. For particle-based schemes, the points $i$ correspond to individual particles (with known identity) lying along the interface. In capturing methods, the interface is not explicitly tracked, but rather “captured” using a characteristic (color) function $C$ that is the discontinuous Heaviside function in the limit of zero mesh spacing. For example, in the two-fluid case, $C$ will take on constant values ($C_1$ or $C_2$) away from the interface:

$$C = \begin{cases} 
C_1 & \text{in fluid 1;} \\
C_2 & \text{in fluid 2;} \\
> C_1, < C_2 & \text{at the interface;}
\end{cases}$$

where we assume $C_2 > C_1$. For finite mesh spacing, $C$ is not perfectly discontinuous, hence the region with $C_1 < C < C_2$ has finite width, on the order of the mesh spacing. Since a point on the interface must remain there, the evolution equation for $C$ is simply a statement of Lagrangian invariance:

$$\frac{DC}{Dt} = \frac{\partial C}{\partial t} + (V \cdot \nabla) C = 0,$$

where $V$ is the velocity at the interface. Exact Lagrangian information is not retained in capturing methods (discarded instead for $C$), hence the interface location is not known exactly. Its position is defined as the transition region where $C_1 < C < C_2$, so interface location is known only to within the transition width. The color function $C$ is considered to be the volume fraction (usually labeled $f$) in volume tracking methods, the level set ($\phi$) in level set methods, and the phase field ($\phi$) in phase field methods. In capturing methods, knowledge $C$ allows one set of model equations to apply everywhere in the domain. Away from the interface, the equations reduce to the correct pure fluid equations, and within the interface, they contain appropriate discrete delta functions (formed from $C$) for interfacial terms.}

Although differences between each scheme can be inferred from the discussions in the next section, some categorical tracking/capturing differences pervade. In capturing methods, the grid is usually fixed, hence topological robustness (requirement R4) is inherent. Requirement R3 (compact interface width), on the other hand, is difficult to meet because of the potential for numerical diffusion in discretization of the $(V \cdot \nabla) C$ term in equation (3) in regions where $C$ abruptly varies. In tracking methods, the interface is maintained as a discontinuity by design (R3 is easily met), yet topological robustness (R4) and three-dimensionality (R5) can be elusive since discrete points remain on the interface. Currently no interface tracking algorithm is without some weakness, so analysts using these algorithms must be aware and stay inside of the parameter space bounding the algorithm's applicability and reliability.

Which schemes have been used for mold filling simulations? Several have been reported in the literature, yet the benchmark mold fill problem presented at the previous international conference on the Modeling of Casting, Welding, and Advance Solidification Processes (MCWASP VII) [1] indicates that most of the nine groups bold enough to tackle this difficult problem employed variations of volume tracking method to track the cavity gas/molten liquid interface. Other methods are just (if not more) viable, and it is our guess that if this benchmark were to be repeated in the future, more entries using a wider variety of interface tracking algorithms would be submitted.

In the following short review, we comment on the ability of each algorithm to meet our requirement list, which we use as a barometer for determining relative strengths and weak-
nesses... We do attempt to keep the discussion objective, but our comments are undoubtedly biased by our own research experiences and tastes. Definitive, objective comments and conclusions should only be guided by detailed verification and validation simulations, which space does not permit. For additional reviews written by researchers with in general different tastes and research experiences, the reader is encouraged to consult references [2–10].

Moving-Mesh Methods

Moving-mesh methods encompass all techniques that move the physical position of discrete grid points in the computational domain by integrating equation (1) forward in time. A moving-mesh method is Lagrangian if every point is moved, and mixed (Lagrangian-Eulerian) if grid points in a subset of the domain are moved. Mold filling simulations provide an excellent use for mixed methods [11], where the mold/cavity computational domain remain stationary (Eulerian) and the molten liquid is followed with a Lagrangian mesh [12,13]. Moving-mesh methods have been popular in both the finite volume and finite element communities for decades, and are arguably among the oldest of numerical methods. With the mesh boundary-fitted to the physical domain, the system of equations may be transformed into logical space [10,14,15] or expressed as integrals over discrete control volumes [16]. Useful overviews can be found in classical [17] and more recent textbooks [8].

If the interfacial flow problem to be modeled is one-dimensional (usually not interesting) or multi-dimensional with regular interface topologies, than a Lagrangian mesh method will almost always yield the most accurate solutions, hence is the recommended method of choice. “Regular” topologies here refer to single-valued interface topologies that do not undergo tearing, stretching, or merging. Unfortunately regular topologies are not characteristic of those encountered in mold filling simulations.

Discrete control volumes (elements) in the computational domain encompass the same parcel of fluid in Lagrangian mesh methods. If the velocity field has appreciable spatial variation (shear, vorticity), then elements must distort and freely deform with the flow, yet they cannot because of their geometric limitations (such as element nodes connected with logically straight edges). Herein lies the principal drawback of these methods: unavoidable element distortion leads to deterioration of solution accuracy and eventual termination of the simulation if element connectivity rules are violated. A solution to this problem is to remesh the domain, either by keeping the original mesh connectivity constant or allowing it to change. Lagrangian mesh methods that allow connectivity changes (including removal or creation of elements) are known as free lagrange methods [18]. Both conventional Lagrangian mesh methods [19] and free Lagrange methods [20,21] have been shown to be useful for incompressible interfacial flows. The principal problem with remeshing operations is the potential for global numerical diffusion; in fact, when connectivity changes are allowed, many have argued that the method may not even converge to a solution. These issues must be quantified and overcome before Lagrangian mesh methods can be competitive and viable for mold filling simulations.

In summary, we believe that Lagrangian mesh methods meet or exceed requirements R1, R3, R6, R7, and R8; fall short of requirements R4, R5, and R10; and could go either way on requirements R2, R5, R9, and R11.

Explicit Tracking Methods

Explicit front tracking has its roots in the original MAC method [22] and its extensions by Daly [23]. The interface is represented discretely by Lagrangian markers connected to form a front which lies within and moves through the stationary Eulerian mesh. This is
contrary to moving-mesh methods, where every grid point, whether it be along or away the interface, moves as a Lagrangian marker. As the front moves and deforms, interface points are added, deleted and reconnected as necessary. The interface can in principle exhibit arbitrarily complex interface deformations and topology changes, provided the algorithm is capable of making logical decisions about topology change. In the front tracking method, topological changes do not occur naturally, i.e., they do not result from a set of localized interface physics models. They result instead from algorithm intervention, via decisions such as whether or not to “cut” a front into pieces (by creating a new linked marker particle list) or to join two fronts into one. The algorithm is also not explicitly mass conservative as designed, although conservation tends to be reasonably adhered to for high Lagrangian marker densities (number of markers per interface length) and topologically regular interfaces. Further detail on the front tracking method can be found in [10,24–26].

Interface information (position, forces) is passed between the moving Lagrangian interface and the stationary Eulerian mesh using ideas borrowed from the immersed interface method [27,28]. With this technique, the sharp interface is approximated by a smooth distribution function that is used to distribute the sources at the interface over mesh points nearest the interface. In this way, the front is given a finite thickness on the order of the mesh size to provide stability and smoothness. Numerical diffusion is absent since this thickness remains constant for all time. The front tracking distribution function is our previously-mentioned color function $C$, hence this portion of the algorithm is akin to capturing methods. Mass conservation is violated in the front tracking method because no single contour of the function $C$ will in general overlay on the mesh with interface formed by connecting the Lagrangian markers.

Front tracking methods have been successfully applied to a variety of interfacial flow problems: gas dynamics [24,25], incompressible flow with [29,30] and without [31,32] phase change, and microstructure evolution in alloy solidification [33,34], to name a few. Only a few years ago it was not clear that front tracking methods could adequately meet the topological challenges posed by 3-D simulations, but Tryyvason and coworkers [31] have to a large extent shown this to be possible. In summary, we currently believe the front tracking method meets or exceeds requirements R2, R3, R7, and R8; falls short of requirements R1, R10, and R11; and could go either way on requirements R4, R5, R6, and R9. Key outstanding issues are whether the conservation properties and topological robustness are adequate for 3-D mold filling simulations. Implementation complexities associated with a 3-D front tracking method could continue to preclude its widespread acceptance and use. It is, however, a method must be pursued for high fidelity simulations of interfacial flows in materials processing.

Particle-Based Methods

In modeling fluid flow, particle-based, or particle hydrodynamic methods [35], are characterized by the use finite-sized “particles” to represent macroscopic fluid parcels. The Lagrangian form of the NS equations are integrated on numerous globs of fluid (the “particles”) having properties such as mass, momentum, and energy. Particle-based methods have been surprisingly successful in modeling highly distorted multi-dimensional interfacial flows, exhibiting robustness that has been exploited in a wide range of physics.

The first particle-based method was the ingenious particle-in-cell (PIC) method originally developed by Harlow and co-workers at the Los Alamos National Laboratory in the 1950’s [36–42]. The classical PIC method targets these weaknesses by modeling the advection terms in the conservation equations with movement of the fluid particles relative
to a fixed Eulerian mesh. The PIC prescription eliminates grid distortion problems via use of a fixed Eulerian mesh, and difficult nonlinear advection is simply modeled as relative particle-grid motion. Further, if each particle's identity and physical position are known, material interfaces are automatically tracked. In Harlow's classical PIC method [41], particle properties are first assigned to the grid, typically using a nearest grid point (NGP) interpolation, so that particles in a cell are therefore the only contributors to the cell's properties. The Lagrangian equations of motion are then solved on an Eulerian grid. Cell- and grid-based quantities are updated with these tentative solutions. The particles are next moved with a velocity that is linearly-interpolated, known as area weighting (in two dimensions), from the tentative velocities of neighboring cells. A particle, upon changing cells, carries temporary mass-proportioned energy and momentum from the cell it exits into the cell it enters. Finally, grid quantities are reassigned by again summing over all particles in a given cell (NGP interpolation). The advantages of the classical PIC [43]. The second approach, known as full PIC, borrows from plasma PIC simulation methodology [44,45] by allowing each particle to carry all the necessary fluid information, which results in increased computational time and memory storage. The full PIC formulation has been adopted by McCrory, et al. [46], Leboeuf, et al. [47], Marder [48], and FLIP in its original form by Brackbill and Ruppel [49]. Allowing the particles to carry momentum and energy virtually eliminates the diffusion problem encountered in classical PIC (item (3) above), but

Other interesting particle-based approaches include modern incarnations of the “marker particle” method. The aspect of this method that makes it similar to the earlier MAC method is that the particles do not carry either volume or mass, but rather identity. The entire flow domain is filled with particles each carrying an identity. The particles act as markers for the identity of fluid at a given position in time and space. These identities are mapped to the underlying Eulerian grid via interpolation. The marker particle method is a natural successor to the PIC and MAC methods developed at Los Alamos over the past forty years [37,50,51]. Unlike the PIC methods, this marker particle method will not conserve mass in the usual fashion.

Vortex Dynamics Methods

Continuum Advection Schemes

Continuum advection schemes here to refer to traditional methods (originating in the finite difference community) for obtaining discrete numerical solutions to equation (3). This equation is perhaps the simplest form of a hyperbolic equation, hence the countless references and textbooks available on hyperbolic equation solution techniques can be drawn upon. We have found references [53-56] particularly useful in this regard. We classify these schemes as continuum advection schemes since they are usually designed upon the premise that $C$ in the advection equation (3) is smoothly varying (a “continuum”). An obvious (and unacceptable) example of such a scheme is first-order upwind “donor cell” scheme. More realistic examples are the higher-order monotonic van Leer [57], PPM [58], and TVD [59] schemes. Techniques identical or similar to these schemes should be used in mold filling simulations for solutions to (3) if $C$ represents a passive scalar, energy, solute species, momentum, or density away from interfaces.

Continuum advection schemes have been used as interface tracking methods in mold filling simulations [60,61]. Asking such schemes to track interfaces, however, forces them to gen-
erate solutions to (3) when \( C \) is a discontinuous color function. This can be problematic because continuum advection schemes are not generally designed for this purpose. The source of this problem lies in the algebraic treatment of the \((\mathbf{V} \cdot \nabla) C\) term; even with higher-order approximations, unacceptable broadening of the region where \( C \) varies (the interface width) occurs. For example, a compressive, fourth-order PPM method was still found to unacceptably diffuse the interface [62]. Why? Because discontinuities in \( C \) can only remain so after solutions to (3) result from geometric approximations to \((\mathbf{V} \cdot \nabla) C\); continuum advection schemes approximate this term algebraically, i.e., by taking spatial differences in \( C \) directly across the interface. Higher-order approximations to these differences can obviously mitigate this numerical diffusion, yet not nearly well enough. Studies besides our own have found that interface widths can broaden to 4-8 cells even when higher-order methods are used [63]. This is not satisfactory for an interface tracking method tasked to perform 3-D mold filling simulations.

One interesting approach to the interface diffusion problem is to transform the discontinuous \( C \) function into another smooth function \( \tilde{C} \), solve (3) with \( \tilde{C} \), then transform \( \tilde{C} \) back to \( C \). In this way, the continuum advection scheme can generate solutions for which it was designed. This idea, set forth by [] and others, is also the basic premise of the level set method [9]. In summary, continuum advection schemes meet or exceed all of our requirements except for R3, i.e., their inability to maintain a compact interface width. This shortcoming currently renders continuum advection schemes much less attractive than many other interface tracking methods.

**Volume Tracking Methods**

As stated previously, most of the MCWASP VII benchmark mold fill problem submissions employed volume tracking\(^1\) methods. One should not infer from this, however, that this algorithm is the best choice for tracking interfaces in mold filling simulations. It remains the most popular and widely used tracking method for mold filling, but this could be due to reasons other than performance: it is relatively easy to implement (at least in its crude forms), it is an older, well-documented algorithm, it is topologically robust, and its basis in volume fractions lends itself well to incorporation of other physics. Volume tracking methods differ from continuum advection schemes in one principal way: the \((\mathbf{V} \cdot \nabla) C\) in equation (3) is approximated geometrically, based upon knowledge of a (nonunique) “reconstructed” interface position.

Volume tracking methods originated in the early 1970s, when three methods were introduced within a short period of time: DeBar’s method [65], Hirt and Nichols’ VOF (for Volume-Of-Fluid) method [64,66], and Noh and Woodward’s SLIC (for Simple Line Interface Calculation) method [67]. See [68,69] and the references therein for historical perspectives and accounts of the chronological developments that have taken place over the last three decades. In short, substantial evolutionary development and improvement has taken place, rendering most of original methods virtually obsolete. Many of the original methods exhibit first-order convergence in time and space (at best) [70]. These characteristics are just not competitive nor adequate for modern simulations.

Without certain key developments occurring over the last decade, volume tracking methods would not have remained competitive (from an accuracy perspective) with newer, impressive methods such as front tracking and level set techniques. Such developments include spatially second order, linearity-preserving interface geometry reconstructions, multi-

\(^1\)We avoid the term “VOF” for volume tracking methods because of its ambiguity; VOF here refers only to the original Hirt-Nichols [64] algorithm.
dimensional time integration schemes [68,69,71], and extensions to 3-D unstructured (finite element) meshes [72]. Perhaps the most important trend is the movement of volume tracking algorithms away from heuristic “case-by-case” logic toward a mathematically formal algorithm based upon geometric primitives. This trend should facilitate propagation of improvements to older implementations, reduce the likeliness of any two volume tracking algorithms generating vastly different solutions, reduce implementation difficulties (particularly in 3-D), and make it easier for future researchers to find weaknesses and devise improvements.

Given these recent developments, modern volume tracking methods are competitive with other interface tracking methods [62,73]. They will remain viable for the foreseeable future, but several outstanding issues should be resolved. Some of these include quantification and alleviation of numerical surface tension, improved schemes for time integration and interface normal approximations, and improved efficiency and ease of implementation on arbitrary 3-D meshes. In summary, modern volume tracking schemes currently fall short of requirements R9–R11, and could go either way on requirements R2, R5, and R8. All other requirements are met or exceeded.

Level Set Methods

Since introduction of the level set method by Osher and Sethian [74] only a decade ago, its use has exploded, evidenced by the recent textbook and references therein [9]. To date, the level set method has been used to model interfacial phenomena in the fields of material science, fluid mechanics, image enhancement, computer vision, and grid generation, to name a few. Impressive examples of the ability of the level set method to track interfaces in topologically complex incompressible flows can be found in the recent simulation results of Sussman [75]. The mathematical formalism and rigor grounded in the level set method has helped to attract leading numerical analysts and mathematicians, resulting in its evolution, widespread promotion and use, and increasing range of applicability.

The basic premise of the level set method is to embed the propagating interface (denoted \( \Gamma(t) \)) as the zero level set of a higher dimensional function \( \phi \), which is defined as \( \phi(x,t=0) = \pm d \), where \( d \) is the distance from \( x \) to \( \Gamma(t=0) \), chosen to be positive(negative) if \( x \) is outside(inside) the initial \( \Gamma(t=0) \). If the zero level set coincides with the initial interface, i.e., \( \Gamma(t=0) = \phi(x,t=0) = 0 \), then a dynamical equation for \( \phi(x,t) \) that contains the embedded motion for \( \Gamma(t) \) as the level set \( \phi = 0 \) can be derived [74]:

\[
\frac{\partial \phi}{\partial t} + F|\nabla \phi| = 0,
\]

where \( F \) is the speed of the interface \( \Gamma \) in the outward normal direction. In general, \( F \) is the sum of any applied interface propagation speed, a curvature-dependent speed, and the flow velocity normal to interface, i.e., \( V \cdot \hat{n} \), where \( \hat{n} = \nabla \phi / |\nabla \phi| \) is the unit interface normal. For certains forms of \( F \) this equation takes on a standard Hamilton-Jacobi form, but for the interfacial flows encountered in mold filling, \( F \) is only \( V \cdot \hat{n} \), hence equation (4) equivalent to equation (3).

Level set methods, then, propagate interfaces by integrating the same basic scalar evolution equation as other capturing methods. The difference, however, is that the scalar function \( \phi \) in level set methods is not some discrete representation of a Heaviside \((H)\) function, but rather a smoothly varying distance function. Herein lies a key advantage: highly accurate numerical solutions to equation (3) are possible (e.g., using the continuum advection schemes previously mentioned) since \( \phi \) is smoothly varying. Many such schemes
are readily available and easily implemented [55]. Herein also lies a key disadvantage: since
\( H \) is not directly integrated forward in time, it must be reconstructed each time step from
the distance function \( \phi, \tilde{H}(\phi) \) [75]. This procedure will not be mass conservative [62] since
densities are defined from \( H(\phi) \) rather than an \( H \) that is directly integrated forward in
time according to strict mass conservation (as in volume tracking methods).

Rigorous mass conservation is elusive in the level set method because the distance function
\( \phi \) does not necessarily remain so after solutions to (4) are obtained. This is especially
ture if the interface has undergone large topological changes. So-called “reinitialization”
algorithms have therefore been devised [75, 76] to insure \( \phi \) remains a distance function,
i.e., satisfying \( |\nabla \phi| = 1 \). Reinitialization can (and in general will) move the zero level
set position; this violates mass conservation. More accurate information is needed about
the front (\( \phi = 0 \)) position so that movement of the zero level set does not occur during
reinitialization. The addition of a global mass conservation constraint [75], acting like a
Lagrangian multiplier, have improved the reinitialization conservation properties, but \textit{local}
constraints are still needed. Without local constraints, the zero level set might still move
as much as a cell width during reinitialization, which artifically creates mass locally in
some cells and destroys it in others. This is not acceptable performance for an interface
tracking algorithm used in mold filling simulations. If local, \textit{geometric} information is used
to fix the zero level set position during reinitialization, the algorithm is likely to have many
similarities to volume tracking algorithms [77].

Phase Field Formulations

Phase field formulations have been applied to crystal growth problems and Hele-Shaw
flows over the past decade [78–81], but only recently have they shown promise for in-
compressible and compressible NS flows [82–84]. Phase field models, like other Eulerian
capturing methods, model interfacial forces as continuum forces by smoothing interface dis-
continuities and forces over thin but numerically resolvable layers. This smoothing allows
conventional numerical approximations of interface kinematics on fixed grids. The phase
field method, applied to the Navier-Stokes equations, also provides a continuum surface
tension model (similar to the CSF model [85]) that is energetically and thermodynamically
consistent [84]. Just recently a phase field model for dendritic solidification in the presence
of melt convection (incompressible NS flow) was developed [86]. Phase field formulations
are indeed showing promise and the ability to provide a powerful vehicle for the direct
numerical simulation of interfacial phenomena.

In the case of free surface flows, the starting point is the van der Waal hypothesis, in
which the interfacial energy density depends upon both \( \phi \) (the phase field) and gradients
in \( \phi \). Cahn and Hilliard [87] extended this hypothesis to dynamical situations by approx-
imating interfacial diffusion fluxes as being proportional to chemical potential gradients.
Jacqmin [84] recently extended the Cahn-Hilliard equation to allow for the presence of flow
(advection). Equation (3) gives Jacqmin’s evolution equation for \( \phi \) is (with \( C \) replaced by
\( \phi \)), except that the RHS side is the Laplacian of the chemical potential rather than zero.
This term is quite interesting; it can be diffusive (positive) or anti-diffusive (negative),
helping to regularize the interface width (not too diffuse or compact). Rather than relying
on special numerical techniques in tracking algorithms to keep interface widths regular,
physical mechanisms in the phase field method do the work. Simple central difference
expressions for \((\nabla \cdot \nabla) C\) were found to be adequate in most cases [84].

The presence of the (anti)diffusive term on the RHS of equation (3) is problematic, however:
at least three cells are needed through the interface so that the Laplacian can be properly
discretized, otherwise the interface will “stick” to the mesh [84]. Current approaches aimed to alleviate this problem are to adaptively refine the interface region; this (or some other solution) will be required for phase field methods to be viable in 3-D. Perhaps the physical principles embodied in these formulations can be combined with the numerical techniques in tracking methods to yield a unified, optimal approach. Phase field formulations for free surface flows are new and exciting, and deserve further attention and exploration.

Parting Thoughts

The wide variety and number of interface tracking algorithms reviewed in this paper are evidence for the many options available for modeling interfacial flows. Keeping in mind that each algorithm has its own unique strengths and weaknesses, which algorithm possesses strengths best suited for mold filling simulation? The best way to answer this question is with objective (and perhaps definitive) scrutiny by executing controlled tests with defined flow fields and known interface topology solutions [62], performing interfacial flow simulations using varied interface tracking algorithms and the same NS flow solver [71,73], and validating the mold filling simulation against experimental data. Not enough of these studies have been performed, hence there is not yet a clear interface tracking algorithm “winner” in our mind. Our own experiences with volume tracking, front tracking, and particle-based methods have led us to believe that these methods have been and will continue to be useful. The more recent level set and phase field methods have also exhibited impressive capabilities. Since one algorithm’s strength is often another’s weakness, a very real (and welcome) possibility is the convergence of many of these methods into one unified method. Not to be overlooked is the possibility that some new, robust and accurate 3-D method is yet to be discovered.

In closing, many modeling issues in addition to interface tracking algorithms have not been covered in this paper, and each issue could ultimately control whether or not a mold filling simulation tool is useful in the foundry. For example, for an incompressible interfacial flow model, a partial list includes surface tension models, cavity gas models [88], NS flow solvers [89], discretization techniques [90], parallelism [91], linear solvers [92], and turbulence models [93].

Acknowledgments

We thank W. Rider and M. Williams for many productive discussions. This work was supported by the United States Department of Energy Accelerated Strategic Computing Initiative Program and the Los Alamos National Laboratory Directed Research and Development Program.

References


