Proceedings of the
3rd ASME/JSME Joint Fluids Engineering Conference
July 18-22, 1999, San Francisco, California, USA

FEDSM99-7109

A SECOND-ORDER ACCURATE, LINEARITY-PRESERVING VOLUME TRACKING ALGORITHM FOR FREE SURFACE FLOWS ON 3-D UNSTRUCTURED MESHES

D. B. Kothe [*], **M. W. Williams, K. L. Lam,**
D. R. Korzekwa, and P. K. Tubesing

Los Alamos National Laboratory
Los Alamos, New Mexico 87545

E. G. Puckett

Department of Applied Mathematics
University of California, Davis
Davis, California 95616
egpuckett@ucdavis.edu

ABSTRACT

The design of an optimal casting process begins with a fundamental understanding of the free surface flow dynamics brought about by the discharge of molten metal into a mold cavity. For the foundries at the Los Alamos National Laboratory (LANL), this discharge is initiated by a gravity pour process which results in a turbulent and topologically complex 3-D flow as the mold cavity is filled. To better understand the mold-filling portion of LANL casting processes, a new casting simulation tool[1] has been recently developed. A key method embodied in this tool is a novel volume tracking algorithm capable of faithfully replicating the kinematics of free surfaces moving within 3-D domains partitioned by generalized unstructured meshes. The method exhibits several desirable properties: robustness, second-order temporal and spatial accuracy, and a suitability for *any* unstructured mesh comprised of elements bounded by ruled surface faces. Details necessary for algorithm implementation, e.g., exact volume truncation expressions, are given along with several illustrative numerical simulations.

INTRODUCTION

Computational simulations of the mold filling portion of a casting process demands an accurate and robust algorithm for tracking the molten metal free surface. Many candidate algorithms are available today, and many more are likely to be devised; see (Kothe et al., 1998; Kothe, 1999) for reviews. The requirements list we impose upon an interface tracking algorithm optimal for mold filling simulations is long and formidable. We seek an algorithm that:

- is globally *and* locally mass conservative;
- maintains at least second order accuracy in time and space;
- maintains compact interface discontinuity width;
- is topologically robust;
- is amenable to three-dimensions;
- is amenable to general unstructured meshes;
- can accomodate additional interfacial physics models;
- can track interfaces bounding more than two materials;
- is computationally efficient;
- can be implemented by novices; and
- can be readily maintained, improved, and extended.

We have expended considerable time and effort in quantitatively comparing most popular candidate algorithms in 2-D (Rider and Kothe, 1995). While equivalent 3-D comparison studies have yet been undertaken, we currently conclude that a clear interface tracking algorithm "winner" is not apparent at this time. Each algorithm has readily identifiable strengths and weaknesses, which, if understood and quantified, could result in a hybrid, unified algorithm possessing the strengths of many different algorithms.

[*]Address all correspondence to this author at *dbk@lanl.gov*.
[1]See *http://public.lanl.gov/mww/HomePage.html* for details.

WHY VOLUME TRACKING?

To date we have embraced volume tracking algorithms, where we have found them useful on 2-D structured (Rider and Kothe, 1998) and unstructured (Mosso et al., 1997) meshes as well as 3-D structured meshes (Kothe et al., 1996). We have devised and implemented volume tracking algorithms which reconstruct piecewise linear (planar) fluid interfaces from discrete fluid volume data. If the piecewise linear interface reconstruction geometry is linearity-preserving, i.e., reconstructs planar interfaces exactly, then we declare the algorithm to be spatially second order. By detailing the algorithm's extension to 3-D unstructured meshes in the following, the first seven requirements previously listed have at least been addressed (although not adequately); work remains before the remaining four requirements can be addressed, and focused analysis is needed on all requirements before victory can be declared. Volume tracking has been our choice to date because other algorithms have fallen short in satisfying some of the more important requirements such as topological robustness and the maintainence of local conservation and compact interface width.

A WALK THROUGH THE ALGORITHM

Our volume tracking algorithm seeks discrete numerical solutions to

$$\frac{\partial f_k}{\partial t} + \mathbf{u}\cdot\nabla f_k = 0\,, \tag{1}$$

where $\mathbf{u}$ is the flow velocity and f_k is the volume fraction of material k. Here we invoke a one-field approximation, as derived in (Kothe, 1999). Since f_k delineates the presence (or absence) of each fluid, f_k serves as a Heaviside function H for each material k. Equation (1) is therefore an evolution equation for the location of each fluid, with the volume fractions f_k discretely approximating H. The volume fractions f_k are bounded by $0 \le f_k \le 1$, where

$$f_k = \begin{cases} 1, & \text{inside fluid } k\,; \\ > 0, < 1, & \text{at the fluid } k \text{ interface}\,; \\ 0, & \text{outside fluid } k\,. \end{cases} \tag{2}$$

Since fluid volumes are volume-filling, volume fractions must sum to unity, $\sum_k f_k = 1$, throughout the domain. In seeking solutions to Eq. (1), fluid *volumes* are marched forward in time as solutions to the volume integral of Eq. (1) (Rider and Kothe, 1998). Key differentiating aspects of a given volume tracking algorithm include the temporal integration scheme and the accuracy with which fluid truncation volumes at control volume faces are estimated. Truncation volumes follow from a required reconstructed interface geometry assumption. For this work, we fit the fluid volume data to a reconstructed interface whose geometry is piecwise linear (planar), given by the equation

$$\hat{\mathbf{n}}\cdot\mathbf{x} - \rho = 0\,, \tag{3}$$

where $\mathbf{x}$ is a point on the plane and ρ is the plane constant. This approximation is a good one if the radius of curvature of the interface is at least two to three times the characteristic mesh spacing.

We now summarize our volume tracking algorithm template:

1. Estimate the interface topology from discrete f_k data. For piecewise linear schemes, this requires an estimation for the interface normal $\hat{\mathbf{n}}$.
2. Reconstruct the interface in each cell by locating the interface surface within the cell in a volume conservative manner. For piecewise linear schemes, this requires finding the plane constant ρ in Eq. (3).
3. Define the flux volume boundaries at each control volume face.
4. Compute the fluid volume truncated by the interface surface within each flux volume ($V_{\rm tr}$).

Various methods for accurate estimation of the interface normal $\hat{\mathbf{n}}$ can be found in (Kothe et al., 1996; Williams et al., 1999a); in the following we detail how $V_{\rm tr}$ can be computed exactly within volumes bounded by logical hexahedra typical of most unstructured meshes.

Reduction to a Surface Problem

Let the truncation volume $V_{\rm tr}$ be the volume of the portion of the interior of the hexahedron behind the interface plane p. The problems to be solved are

- the *direct* problem: given the cell, ρ, and $\hat{\mathbf{n}}$, find V; and
- the *inverse* problem: given the cell, ρ, $\hat{\mathbf{n}}$, and V, find ρ.

The truncation volume is given by

$$\begin{aligned} V_{\rm tr} &= \int_{\rm v} 1\, d\tau = \frac{1}{3}\int_{\rm v} \nabla\cdot(\mathbf{x}-\hat{\mathbf{n}}\rho)\, d\tau(\mathbf{x}) \\ &= \frac{1}{3}\Big[\sum_{f=1}^{6}\int_{\rm tr}(\mathbf{x}-\hat{\mathbf{n}}\rho)\cdot d\mathbf{S}_f(\mathbf{x}) + \int_{\rm tr}(\mathbf{x}-\hat{\mathbf{n}}\rho)\cdot d\mathbf{S}_p(\mathbf{x})\Big]\,, \end{aligned} \tag{4}$$

where $d\tau$ is an element of volume, $d\mathbf{S}_f$ is a vector element of surface on cell face f, and $d\mathbf{S}_p$ is a vector element of surface on the truncating plane. The surface integrations above are confined to the portions of the surfaces behind the plane (tr), and the $d\mathbf{S}_f$ elements point along the outward normals on each S_f. In Eq. (4), since $d\mathbf{S}_p = \hat{\mathbf{n}}|\mathbf{S}_p|$, then $(\mathbf{x}-\hat{\mathbf{n}}\rho)\cdot\hat{\mathbf{n}} = 0$, hence

$$V_{\rm tr} = \sum_{f=1}^{6} V_f\,, \quad \text{where} \quad 3V_f = \int_{\rm tr}(\mathbf{x}-\hat{\mathbf{n}}\rho)\cdot d\mathbf{S}_f(\mathbf{x})\,. \tag{5}$$

Hence, to compute $V_{\rm tr}$, we consider each truncated face f separately.

In performing the surface integrals above, we must first define the properties of each control volume, which is a computational cell characterized as a *logical cube* having eight vertices, twelve edges, and six faces; vertex positions arbitrary; edges that are straight lines; and faces that are ruled surfaces. Note that this

definition easily allows the cell *in physical space* to be a tetrahedron, prism, pyramid, or hexahedron, since any of the eight vertex physical coordinates can coincide.

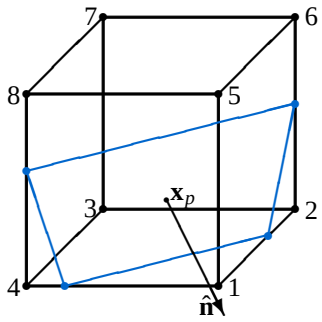

Figure 1. A LOGICAL CUBE TRUNCATED BY A PLANE WHOSE CONSTANT ρ IN EQ. (3) IS $\hat{\mathbf{n}} \cdot \mathbf{x}_p$, WHERE $\mathbf{x}_p$ IS A POINT ON THE PLANE.

Ruled Surfaces

Before defining a ruled surface, consider the following definitions for the four-vertex cell face shown in Fig. 2. Let the four vertices of a face be labeled $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{x}_3$, and $\mathbf{x}_4$, ordered counterclockwise, with the outside of the face above the plane of the paper, as shown in Fig. 2. The points $(\mathbf{x}_1, \mathbf{x}_3)$ and $(\mathbf{x}_2, \mathbf{x}_4)$ are

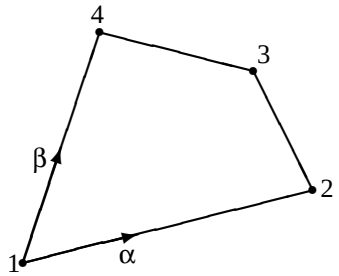

Figure 2. A RULED SURFACE IS DEFINED BY FOUR (IN GENERAL NONPLANAR) POINTS CONNECTED BY STRAIGHT LINES. THE SURFACE IS PARAMETERIZED BY α AND β.

therefore diagonal pairs. The successor vertex, $\mathbf{x}_i'$, is defined as the vertex next to and ahead of $\mathbf{x}_i'$ in the counterclockwise rotation, i.e., $\mathbf{x}_1' = \mathbf{x}_2$, $\mathbf{x}_2' = \mathbf{x}_3$, $\mathbf{x}_3' = \mathbf{x}_4$, and $\mathbf{x}_4' = \mathbf{x}_1$. Similarly, the double-successor vertex, $\mathbf{x}_i''$, is the diagonal partner vertex, and the triple-successor vertex, $\mathbf{x}_i'''$, is the predecessor vertex, i.e., $\mathbf{x}_1''' = \mathbf{x}_4$.

Several vectors and scalars associated with a ruled surface can be defined. First, a deviation vector $\mathbf{B}$ is given by

$$\mathbf{B} = \mathbf{x}_1 - \mathbf{x}_2 + \mathbf{x}_3 - \mathbf{x}_4 \,, \tag{6}$$

hence $\mathbf{B} = 0$ only if the face is a parallelogram. When $|\mathbf{B}| \neq 0$, then $|\mathbf{B}|$ measures the deviation of the ruled surface from the parallelogram configuration. Next, the vector $\mathbf{k}$, given by

$$\mathbf{k} = (\mathbf{x}_3 - \mathbf{x}_1) \times (\mathbf{x}_4 - \mathbf{x}_2) \,, \tag{7}$$

possesses a magnitude which is twice the face area if the face $\mathbf{x}_i$ lie in a plane. The area of the ruled surface in Fig. 2 is also $|\mathbf{k}|/2$, *independent* of whether the four $\mathbf{x}_i$ lie in a plane. A volume v_{tet}, given by

$$v_{\text{tet}} = (\mathbf{x}_1 - \mathbf{x}_2) \times (\mathbf{x}_2 - \mathbf{x}_3) \cdot (\mathbf{x}_3 - \mathbf{x}_4) \,, \tag{8}$$

is six times the volume of the tetrahedron formed by the four $\mathbf{x}_i$ in Fig. 2. This volume is zero only if the four $\mathbf{x}_i$ lie in a plane. The cross-vectors $\mathbf{X}_i$ are defined by

$$\mathbf{X}_i = (\mathbf{x}_i''' - \mathbf{x}_i) \times (\mathbf{x}_i - \mathbf{x}_i') \,, \tag{9}$$

e.g., $\mathbf{X}_1 = (\mathbf{x}_4 - \mathbf{x}_1) \times (\mathbf{x}_1 - \mathbf{x}_2)$. The cross-vector magnitude is twice the area of the surface bounded the three vertices in its definition. We also define the signs ε_i, $1 \le i \le 4$ as $\varepsilon_1 = \varepsilon_3 = +1$ and $\varepsilon_2 = \varepsilon_4 = -1$. Given the definitions above, we see that $2v_{\text{tet}} = \mathbf{B} \cdot \mathbf{k}$, and, if $|\mathbf{B}| = 0$, then $\mathbf{X}_1 = \mathbf{X}_2 = \mathbf{X}_3 = \mathbf{X}_4$.

Let α and β in Fig. 2 be parametric variables with ranges $0 \le \alpha \le 1$, $0 \le \beta \le 1$. Then $(1-\alpha)\mathbf{x}_1 - \alpha\mathbf{x}_2$ and $(1-\alpha)\mathbf{x}_4 - \alpha\mathbf{x}_3$ are points on the lines $(\mathbf{x}_1, \mathbf{x}_2)$ and $(\mathbf{x}_4, \mathbf{x}_3)$, respectively. Given this relationship, one can write an expression for any point $\mathbf{x}$ on a ruled surface as

$$\mathbf{x} = \mathbf{x}_1 + \alpha(\mathbf{x}_2 - \mathbf{x}_1) + \beta(\mathbf{x}_4 - \mathbf{x}_1) + \alpha\beta\mathbf{B} \,. \tag{10}$$

If the face is planar, then $\mathbf{x}$ is a point inside the quadrilateral, but, more generally, $\mathbf{x}(\alpha, \beta)$ given by Eq. (10) is a 2-D surface segment of 3-D space whose borders are the straight lines $(\mathbf{x}_1, \mathbf{x}_2)$, $(\mathbf{x}_2, \mathbf{x}_3)$, $(\mathbf{x}_3, \mathbf{x}_4)$, and $(\mathbf{x}_4, \mathbf{x}_1)$. Through any point (α_0, β_0) on this surface, there are two straight lines, namely $\mathbf{x}(\alpha_0, \beta)$, $0 \le \beta \le 1$, and $\mathbf{x}(\alpha, \beta_0)$, $0 \le \alpha \le 1$, through it which lie entirely on the surface. Thus it is called a *ruled* surface. The ruled surface can, under suitable translation and rotation of coordinates, also be considered a parabolic hyperboloid whose surface area is the minimum area that can be passed through its four straight lines.

The ruled surface element **dS** is given by

$$\mathbf{dS} = \left[\mathbf{X}_1 + \alpha(\mathbf{X}_3 - \mathbf{X}_4) + \beta(\mathbf{X}_3 - \mathbf{X}_2)\right] d\alpha d\beta \,. \tag{11}$$

Integrating **dS** over the ruled surface, on obtains:

$$\int_0^1 d\alpha \int_0^1 d\beta \left[\mathbf{X}_1 + \alpha(\mathbf{X}_3 - \mathbf{X}_4) + \beta(\mathbf{X}_3 - \mathbf{X}_2)\right] = \frac{1}{2}\mathbf{k} \tag{12}$$

The trace of the ruled surface on the truncating plane given by Eq. (3) is a hyperbola on that plane, and the trace of the truncating plane on the ruled surface is a hyperbola on the (α, β) surface.

Let $\mu_i = \hat{\mathbf{n}} \cdot \mathbf{x}_i$, where i denotes any of the four vertices on the ruled surface. Further, define $\rho_i = \min(\rho, \mu_i)$, hence $\rho - \rho_i = 0$ if $\mathbf{x}_i$ is in front of the plane, otherwise $\rho - \rho_i = \mu_i$. Let the $\mathbf{x}_i$ be relabeled $\mathbf{x}_a$, $\mathbf{x}_b$, $\mathbf{x}_c$, and $\mathbf{x}_d$ and set $\mu_a = \hat{\mathbf{n}} \cdot \mathbf{x}_a$, $\mu_b = \hat{\mathbf{n}} \cdot \mathbf{x}_b$, $\mu_c = \hat{\mathbf{n}} \cdot \mathbf{x}_c$, and $\mu_d = \hat{\mathbf{n}} \cdot \mathbf{x}_d$ according to the convention that: $\mu_a \le \mu_b \le \mu_c \le \mu_d$. Given this convention, then, as the plane moves with increasing ρ from $\rho = -\infty$ to $\rho = \infty$, $\mathbf{x}_a$ is the first vertex passed, then $\mathbf{x}_b$, $\mathbf{x}_c$, and $\mathbf{x}_d$ are successively passed. Successor

vertices will also be denoted a', a'', so that $a'' = a+2$. With this notation, all possible plane/ruled surface truncation alteratives divide into six unique cases for V_f:

Case 0: $\rho \leq$ all μ_i, then $V_f = 0$;
Case 1: $\mu_a < \rho \leq \mu_b$, then V_f has only one truncated corner;
Case 2: $\mu_a \leq \mu_b < \rho < \mu_c \leq \mu_d$ and $b \neq a+2$;
Case 3: $\mu_c \leq \rho < \mu_d$, then only one corner has not been truncated;
Case 4: $\mu_d \leq \rho$ ($V_f = V_{f\text{tot}}$);
Case 5: $\mu_b < \rho < \mu_c$ and $b = a+2$, which *can* occur for "bow-tied" surfaces.

Given the integral, K_{nm}, defined as

$$K_{nm} = \int_{\text{tr}} \alpha^n \beta^m \, d\alpha \, d\beta \,, \tag{13}$$

then V_f can be expressed as

$$3V_f = (\mathbf{x}_1 - \hat{\mathbf{n}}\rho) \cdot \big[\mathbf{X}_1 K_{00} + (\mathbf{X}_3 - \mathbf{X}_4) K_{10} + (\mathbf{X}_3 - \mathbf{X}_2) K_{01}\big] - v_{\text{tet}} K_{11} \,. \tag{14}$$

Upon analytically performing the K integrals, a general expression for V_f follows:

$$V_f = \frac{1}{6} \sum_i \varepsilon_i Y_i \frac{(\rho - \rho_i)^2}{\lambda_i} + \frac{v_{\text{tet}}}{2} \sum_i \varepsilon_i \big[J_1(w_i) - 2J_2(w_i) + J_3(w_i)\big] \frac{(\rho - \rho_i)^4}{\lambda_i^2} \,, \tag{15}$$

where

$$\begin{aligned} Y_i &= (\mathbf{x}_i''' - \hat{\mathbf{n}}\rho) \cdot (\mathbf{x}_i - \hat{\mathbf{n}}\rho) \times (\mathbf{x}_i' - \hat{\mathbf{n}}\rho) \,; \\ \lambda_i &= \varepsilon_i (\mu_i - \mu_i')(\mu_i - \mu_i''') \,; \\ w_i &= \frac{\nu}{\lambda_i}(\rho - \rho_i) \,, \quad \text{where} \quad \nu = \hat{\mathbf{n}} \cdot \mathbf{B} \,; \end{aligned} \tag{16}$$

and

$$J_1(w) - 2J_2(w) + J_3(w) = 2 \sum_{n=0}^{\infty} \frac{(-w)^n}{(n+2)(n+3)(n+4)} \,. \tag{17}$$

If w is large, i.e., $w > 10^{-2}$, we compute J_0 and J_1, J_2, J_3 successively by recursion. But if w is small, i.e., $w \leq 10^{-2}$, then compute according to the power series in Eq. (17).

The expression for V_f in Eq. (15) is quite general, being appropriate for cases 1, 3, and 5 mentioned previously. Cases 0 and 4 are trivial, but Eq. (15) may break down for case 2 because one of the λ's in the denominator can vanish. In this case, the terms in Eq. (15) need to be rearranged.

It is useful to definte $V_{f\text{tot}}$, which is our case 4, where all four vertices are behind the interface plane, i.e., all $\mu_i \leq \rho$. Using Eq. (14), and integrating over the entire ruled surface, which yields $K_{00} = 1$, $K_{01} = K_{10} = 1/2$ and $K_{11} = 1/4$, gives:

$$V_{f\text{tot}} = \frac{1}{2}(\mathbf{x}_{\text{cm}} - \hat{\mathbf{n}}\rho) \cdot \mathbf{k} \,, \qquad \mathbf{x}_{\text{cm}} = \frac{\mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3 + \mathbf{x}_4}{4} \,. \tag{18}$$

NUMERICAL EXPERIMENTS

We consider three numerical experiments as a demonstration of the versatility of the volume tracking algorithm. For the first, volume data derived from a known fluid topology is reconstructed as interface planes. This static test assesses the algorithm's ability to replicate complex topologies. For the second, this topology is then moved, hence the time-integration properties of the volume tracking algorithm can be scrutinized. For the third, we task the algorithm to track time-dependent free surfaces in a real flow situation. Here we simulate the side fill of a plexiglass box with water and qualitatively compare the computed free surface topology with the actual topology observed in digital video images of the fill.

The Logo

Consider a 3-D version of the familiar ASME logo. Each letter in the logo is three-dimensionalized by extruding an outline of the letter (an n-sided polygon) in the xy-plane a small distance along the z-axis. After extrusion, each letter fits within a $0.05 \times 0.04 \times 0.01$m box, and the entire logo resides within a $0.17 \times 0.04 \times 0.01$m box.

Static Interface Reconstruction. Figure 3 demonstrates the ability of the volume tracking algorithm to reconstruct the 3-D logo, as seen in the xy-plane. Note that all three meshes are too coarse ($\sim$ 0.30m mesh size) for the volume track-

Figure 3. RECONSTRUCTED INTERFACE PLANES BOUNDING THE ASME LOGO IN THREE DIFFERENT MESHES: AN ORTHOGONAL ($41 \times 26 \times 26$) HEXAHEDRAL MESH (TOP) AND TETRAHEDRAL MESHES CONSISTING OF 63,159 (MIDDLE) AND 123,738 ELEMENTS (BOTTOM).

ing algorithm to reconstruct the hole in the A. Note also that the reconstructed logo has a smoother surface in the hexahedral mesh relative to the tetrahedral meshes. The tetrahedral mesh "fuzziness" exhibited on the logo surface is primarily due to less than second-order accuracy in computing the interface normal $\hat{\mathbf{n}}$. The normal $\hat{\mathbf{n}}$ is estimated with the procedure outlined in (Williams et al., 1999b; Williams et al., 1999a), which is strictly linearity-preserving only for orthogonal hexahedral meshes. As indicated in (Williams et al., 1999a), however, an algorithm for estimating $\hat{\mathbf{n}}$ with convolutions and an appropriate kernel appears promising, able to achieve near linearity-preserving results on complex tetrehedral meshes. This approach is therefore likely to deliver strict linearity-preservation on general unstructured meshes in the near future.

Moving the Interface. Now consider what happens when the logo is allowed to fall into a pool under the force of gravity. The logo is assumed to be an inviscid incompressible fluid ten times more dense than a background fluid (also inviscid and incompressible). The logo initially hovers 0.02m above the pool, which is 0.03m deep. The computational domain, a $0.20 \times 0.12 \times 0.04$m box, is partitioned with $80 \times 48 \times 16$ orthogonal hexahedral cells, offering ample resolution for the volume tracking algorithm to resolve the hole in the letter A. At time zero the logo is allowed to fall, eventually splashing into the pool within 0.20s, as seen in Fig. 4. The splash is not overly energetic because the logo possesses a relatively small free-fall velocity at impact due to its short initial height above the pool. The letters in the logo also experience appreciable drag and distortion imparted to them by the upward-moving background fluid, which has sizeable inertia due to its relatively high density.

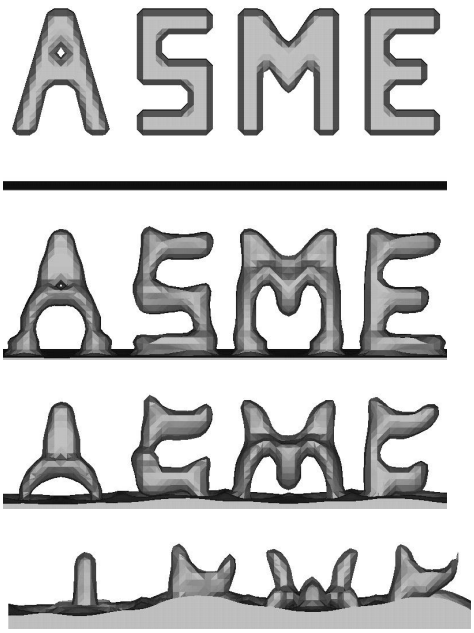

Figure 4. SIMULATION RESULTS FOR THE ASME LOGO FALLING INTO A POOL. SHOWN ARE ISOSURFACES OF THE ONE-HALF VOLUME FRACTION LEVEL AT TIMES OF 0.0s, 0.08s, 0.10s, AND 0.12s, RESPECTIVELY.

Free Surface Flow Validation

We now present very preliminary results for a series of free surface flow validation experiments currently ongoing at LANL. The principal purpose of this series of experiments is the validation of the free surface flow algorithm in a regime relevant to the gravity-pour mold filling portion of LANL casting operations. The experimental plan calls for varying fluid/fluid configurations possessing flow regimes that are either energetic or dominated by surface tension, as measured by the Weber, Bond, and Reynolds numbers. Obvious examples are the classic free surface fluids, water/air (presented here), for a more energetic flow, and gallium/water, for less energetic flow dominated by surface tension. To closely replicate current LANL mold-filling processes, these experimental free surface flows must be truly three-dimensional, initiated by the gravity-fill of a container from one or more apertures along its top or side.

The Gravity-Fill Experiment. The current experimental configuration is shown in Fig. 5. The filling container is a box, whose simple geometry allows the quick and easy generation of computational meshes. The box is constructed from plexiglass so that the filling fluid free surface dynamics can be easily observed with a digital video camera. Square inflow apertures and circular outflow vents on all surfaces of the box are available in various locations. The square inflow aperture holes come in three sizes: 6.35×10^{-03}, 9.525×10^{-03}, or 1.27×10^{-02} m on a side. For the water/air experiment, the box initially contains air at atmospheric pressure, which is vented out of a circular hole along the top surface during the fill. Water supplied from a reservoir ($0.165m$ above the box surface) enters the square inflow aperture under the action of gravity. For these experiments, the inflow velocity, based on observations and pressure-head calculations, is ~ 1.0 m/s, but more precise inflow conditions will be determined in the future. The video camera records thirty digital images each second (see Fig. 6).

Simulation Results. Our first simulation of the side-fill experiment is shown in Fig. 7. The filling box is resolved coarsely with a computational domain partitioned with $10 \times 15 \times 25$ orthogonal hexahedral cells. Our initial results are very promising, with the inflow stream shape (position and inflection) and opposite-side impact time and position captured correctly. The filling dynamics below the inflow stream along the box bottom are not captured as well as the inflow stream, however, because

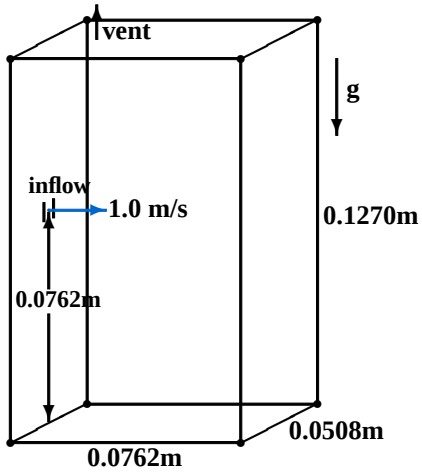

Figure 5. SCHEMATIC OF THE BOX GRAVITY-FILL EXPERIMENT. WATER ENTERS A SQUARE (0.00635m) APERTURE AS SHOWN.

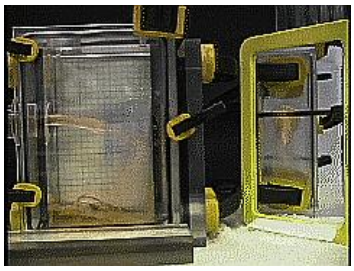

Figure 6. DIGITAL VIDEO IMAGE OF THE WATER/AIR FREE SURFACE 0.60 SECONDS AFTER INFLOW IS INITIATED.

of the coarse simulation resolution and the incorrect 10:1 density ratio. For example, an unrealistically high background fluid density gives the background fluid enough inertia to impart appreciable drag to the filling fluid along the inflow wall at the bottom, causing it to “pile up”. Simulations with the correct density ratio (800:1) are likely to be devoid of this artifact. Currently high density ratios place restrictions on the flow algorithm, which, however, can be alleviated with a nonsolenoidal filtering, as discussed in (Kothe, 1999; Williams et al., 1999a). This work is currently underway, and will be used in subsequent validation simulations.

ACKNOWLEDGMENT

We are indebted to Charles Zemach of Fluid Dynamics Group T-3 at Los Alamos National Laboratory for his analytical volume truncation solutions. This work was supported by the United States Department of Energy Accelerated Strategic Computing Initiative Program.

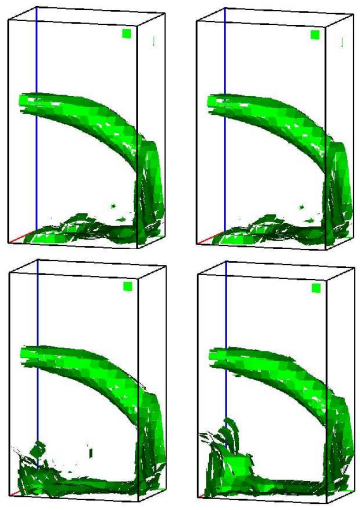

Figure 7. SIDE-FILL SIMULATION RESULTS CORRESPONDING TO FIG. 6, EXCEPT THAT THE WATER/AIR FREE SURFACE IS APPROXIMATED WITH TWO FLUIDS HAVING A 10:1 DENSITY RATIO. INTERFACE PLANES ARE SHOWN AT 0.224s, 0.264s, 0.350s, AND 0.420s.

REFERENCES

D. Kothe, D. Juric, K. Lam, and B. Lally. Numerical recipes for mold filling simulation. In *Proceedings of the Eighth International Conference on Modeling of Casting, Welding, and Advanced Solidification Processes*, 1998.

D. B. Kothe. Perspective on Eulerian finite volume methods for incompressible interfacial flows. In H. Kuhlmann and H. Rath, editors, *Free Surface Flows*, pages 267–331, New York, NY, 1999. Springer-Verlag.

D. B. Kothe, W. J. Rider, S. J. Mosso, J. S. Brock, and J. I. Hochstein. Volume tracking of interfaces having surface tension in two and three dimensions. Technical Report AIAA 96–0859, AIAA, 1996. Presented at the 34rd Aerospace Sciences Meeting and Exhibit.

S. J. Mosso, B. K. Swartz, D. B. Kothe, and R. C. Ferrell. A parallel, volume-tracking algorithm for unstructured meshes. In P. Schiano, A. Ecer, J. Periaux, and N. Satofuka, editors, *Parallel Computational Fluid Dynamics: Algorithms and Results Using Advanced Computers*, pages 368–375, Capri, Italy, 1997. Elsevier Science.

W. J. Rider and D. B. Kothe. Reconstructing volume tracking. *Journal of Computational Physics*, 141:112–152, 1998.

W. J. Rider and D. B. Kothe. Stretching and tearing interface tracking methods. Technical Report AIAA 95–1717, AIAA, 1995. Presented at the 12th AIAA CFD Conference.

M. W. Williams, D. B. Kothe, and E. G. Puckett. Approximating interface topologies with applications to interface tracking algorithms. Technical Report 99–1076, AIAA, 1999. Presented at the 37th Aerospace Sciences Meeting.

M. W. Williams, D. B. Kothe, and E. G. Puckett. Convergence and accuracy of kernel-based continuum surface tension models. In W. Shyy, editor, *Fluid Dynamics at Interfaces*, pages 347–356, Boston, MA, 1999. Cambridge University Press.