INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2007; **55**:693–712
Published online 5 April 2007 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1490

Adaptive VOF with curvature-based refinement

Mayank Malik[‡], Eric Sheung-Chi Fan and Markus Bussmann[*, †]

Department of Mechanical and Industrial Engineering, University of Toronto, 5 King's College Road, Toronto, Ont., Canada M5W 3G8

SUMMARY

Adaptive refinement is implemented in the context of the volume-of-fluid (VOF) methodology in order to study the efficacy of resolving interfaces adaptively based on the local value of curvature. The usual uniform mesh VOF implementation is modified slightly to ensure accurate advection of fluxes between cells at different resolutions. Normals and curvatures are calculated accurately *via* height functions. Results of a series of tests indicate that in most instances the use of adaptive refinement (when compared to uniform refinement with a similar number of cells) leads to more accurate VOF advection. The results also clearly show that curvature-based adaptive refinement leads to a distribution of errors along an interface that is nearly independent of curvature. Copyright © 2007 John Wiley & Sons, Ltd.

Received 9 September 2006; Revised 9 February 2007; Accepted 17 February 2007

KEY WORDS: VOF; volume tracking; adaptive; curvature; height function

1. INTRODUCTION

Adaptive refinement is commonly applied to the simulation of one-fluid flows, but the use of adaptivity in interfacial flow simulation is less common. And in many instances, when applied to interfacial flow simulation on an Eulerian mesh (regardless of how the interfaces are represented: level sets, markers, volume fractions, etc.), the refinement criterion is often nothing more than the presence of the interface, which leads to interfaces represented at a uniform level of refinement, and the remainder of the domain typically refined less (e.g. [1–12]). For a comprehensive overview of adaptivity applied to Eulerian methodologies (primarily level sets, but others as well) for interfacial flow simulation, see the recent review by Losasso *et al.* [13].

*Correspondence to: Markus Bussmann, Department of Mechanical and Industrial Engineering, University of Toronto, 5 King's College Road, Toronto, Ont., Canada M5W 3G8.
†E-mail: bussmann@mie.utoronto.ca
‡Present address: The University of Texas at Austin, Petroleum and Geosystems Engineering, 1 University Station C0300, Austin, TX 78712-0228, U.S.A.

Such an approach is a significant step forward from uniform mesh approaches, especially when the interfacial physics are key to the phenomenon under study. Such is often the case, because surface tension acts at the interface, and because interfaces between fluids are often associated with large jumps in fluid properties like density and viscosity.

But why simply use the presence of the interface as the criterion for adaptivity? In many circumstances, interfaces between fluids will demonstrate large variations in what we might term 'complexity'. Figure 1 illustrates two examples of what we mean: the first is a sketch of an interface along which the curvature varies dramatically; and the second illustrates a thin necking filament of fluid. In both cases, some sections of the interfaces are more 'complex' than others, and ought to be better resolved particularly if fluid phenomena such as rupturing and merging are of interest.

In more concrete terms, consider the splash of a liquid droplet onto a solid surface, an instance of which is illustrated in Figure 2 [14]. There are various length scales present in such a phenomenon,

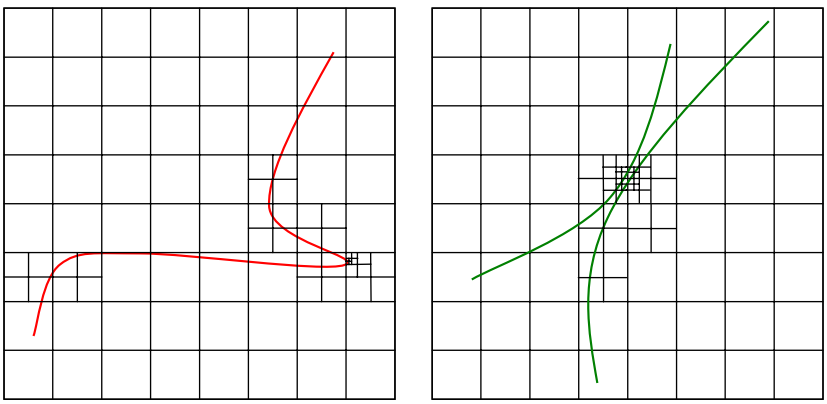


Figure 1. Two interfaces that demonstrate spatially varying 'complexity': on the left, the curvature varies dramatically; on the right, a thin necking filament of fluid.

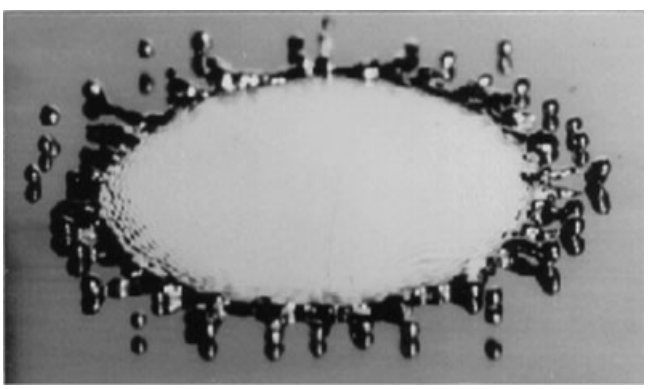


Figure 2. The splash of a droplet of molten tin onto a heated solid substrate [14].

DOI: 10.1002/fld

from the diameter of the impacting drop (not shown) to the diameter of a satellite. Uniform mesh simulations of this and similar phenomena have been performed (e.g. [14–16]), but given the range of length scales and the distribution of complex interfacial features, adaptive refinement of interfaces would seem to be of value in better utilizing a fixed computing resource.

Curvature-based refinement is common in Lagrangian methods, where the mesh is aligned with an interface and is advected with it; the moving-mesh method of Dai and Schmidt [17] is but one example. However, we are aware of only a couple of examples of adaptive interface refinement applied to fixed mesh simulations, and in no instance are details provided of the efficacy of the methodology. Sussman *et al.* [18] implemented curvature-based refinement in the context of an adaptive level set method, and presented results in which interface curvature was utilized as a refinement criterion. And in a very recent paper on an unstructured volume-of-fluid (VOF) flow solver, Löhner *et al.* [19] mention curvature-based refinement, but it is unclear whether this technique was applied to generate any of the results in the paper.

Here, we examine the use of curvature-based adaptivity applied to the VOF (or volume tracking) methodology, although the ideas presented could equally well be applied to other interface representations. The objective is to study the efficacy of adaptive refinement of interfaces, as measured by both global and local measures of error obtained from various tests of the algorithm, by comparison with corresponding results obtained when the interface is uniformly resolved.

The remainder of this paper is organized as follows: in Section 2, we present the methodology: a brief overview of volume tracking on a uniform mesh, the modifications required to volume track on an adaptive mesh, the approach to mesh refinement and coarsening, details of the calculation of interface normals and curvatures on an adaptive mesh, and finally a flowchart of one complete timestep. In Section 3, results and a discussion of various tests of curvature-based adaptivity are presented. The paper is summarized in Section 4.

2. METHODOLOGY

2.1. Uniform mesh volume tracking

Volume tracking is a so-called interface capturing method: rather than track an interface, fluid volume fractions are tracked and advected with the flow. In this work, a modified Youngs' [20] piecewise linear interface calculation method is implemented; the following provides a brief overview.

Volume fractions are discrete values of a continuous indicator function f, equal to one within fluid no. 1 and zero elsewhere, as illustrated in Figure 3(a). Referring to Figures 3(b) and (c), a volume fraction of one denotes a cell that contains only fluid no. 1; a volume fraction of zero denotes a cell that contains only fluid no. 2; and a volume fraction between zero and one implies the existence of an interface. At each timestep, the interface is reconstructed with a linear segment (in 2D) that is oriented according to a calculated normal, and positioned so that it partitions the cell according to the known volume fraction. The linear segment is defined as

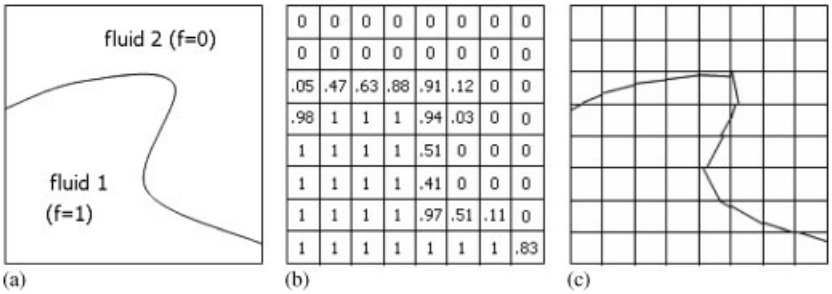
$$n_x \cdot x + n_y \cdot y + \rho = 0 \tag{1}$$

where n_x and n_y are the x and y components of the normal and ρ is the line constant determined from conservation of volume.

Following reconstruction, volume fractions are advected by the velocity field. Given a flow field $\mathbf{u}$, the advection of f is governed by the equation:

$$\partial f/\partial t + \nabla \cdot (\mathbf{u} f) = 0 \tag{2}$$

DOI: 10.1002/fld



0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
.05	.47	.63	.88	.91	.12	0	0
.98	1	1	1	.94	.03	0	0
1	1	1	1	.51	0	0	0
1	1	1	1	.41	0	0	0
1	1	1	1	.97	.51	.11	0
1	1	1	1	1	1	1	.83

Figure 3. (a) An interface between two fluids; (b) the corresponding volume fractions; and (c) the piecewise linear reconstruction of the interface.

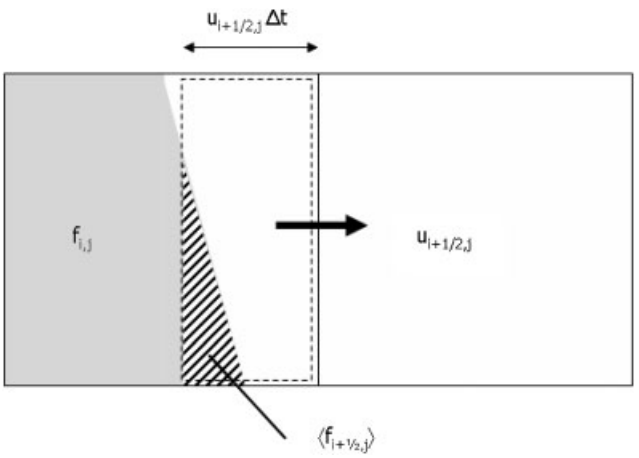


Figure 4. Geometric advection of fluid volume $\langle f_{i+1/2,j} \rangle$ from one cell to another.

When applied to discrete volume fractions, Youngs [20] discretized the equation in an operator-split manner that has been shown to be first-order accurate [21]. Given a known field $f_{i,j}^n$ (the discrete volume fractions at time step n), Equation (2) is discretized as

$$f_{i,j}^* = \frac{f_{i,j}^n V_{i,j} - (u_{i+1/2,j}\langle f \rangle_{i+1/2,j} - u_{i-1/2,j}\langle f \rangle_{i-1/2,j})\Delta y_j \Delta t}{V_{i,j}^*} \tag{3a}$$

$$f_{i,j}^{n+1} = \frac{f_{i,j}^* V_{i,j}^* - (v_{i,j+1/2}\langle f \rangle_{i,j+1/2} - v_{i,j-1/2}\langle f \rangle_{i,j-1/2})\Delta x_i \Delta t}{V_{i,j}} \tag{3b}$$

$f_{i,j}^*$ is the interim field that results from a sweep in one direction (in this case x), $V_{i,j}$ is the cell volume, u and v are face velocities evaluated as the exact average volume flux across a face (so that

DOI: 10.1002/fld

a given divergence-free velocity field is also discretely so), $V^*_{i,j} = V_{i,j} - (u_{i+1/2,j} - u_{i-1/2,j})\Delta y_j \Delta t$ is the interim cell volume, $\Delta t = t^{n+1} - t^n$, and $\langle f \rangle$ refers to the volume fraction (the hatched area illustrated in Figure 4) of the flux volume (the outlined area) that is calculated geometrically from the reconstructed interface. Advection in the other direction (in the case of Equation (3b), the y direction) completes one full time step. The order of advection is alternated from one timestep to the next to minimize directional bias.

2.2. *Adaptive mesh volume tracking*

Unlike a uniform mesh, an adaptive mesh consists of cells of different sizes. In this work, the adaptivity is implemented *via* a quadtree data structure as described by Samet [22], and as previously implemented, for example, by Greaves [3]. The domain begins as a unit square of one cell; smaller cells are created by recursive subdivision of a 'parent cell' into four 'child cells' until a desired

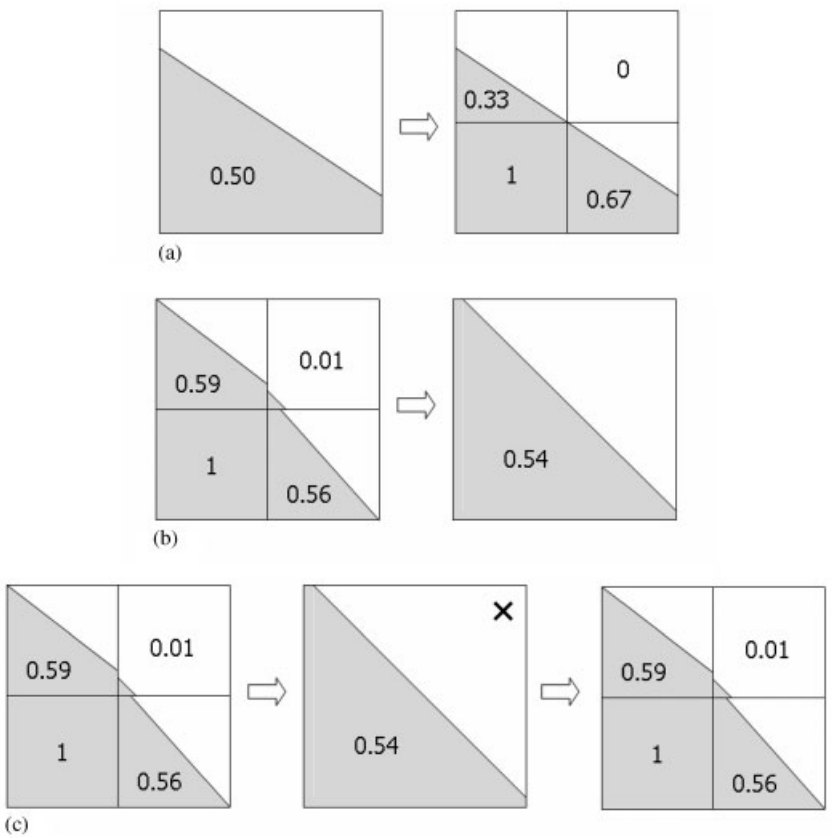


Figure 5. (a) Refinement: fine cell volume fractions are calculated from the reconstructed interface in the parent cell; (b) coarsening: the coarse cell volume fraction is calculated as the average of the four child cells; and (c) a special refinement procedure whereby fine cell volume fractions are returned to their pre-coarsened values.

DOI: 10.1002/fld

level of refinement is achieved. The size of a cell depends on its refinement level: if we designate the unit square to be level 0, then the size of any cell is $\ell = (1/2)^{\text{level}}$. In the results to be presented, the refinement level of an interface cell is determined by a criterion, bounded between prescribed minimum and maximum levels.

Before discussing volume tracking on an adaptive mesh, we present the approach to calculating volume fractions upon refinement/coarsening. Refinement involves dividing a cell into four child cells and determining the volume fraction for each child. If the parent cell is completely full ($f=1$) or empty ($f=0$), all child cells will have volume fractions of one or zero, respectively. If the parent cell contains an interface, the volume fraction of each child cell is determined geometrically as illustrated in Figure 5(a) from the reconstruction of the interface in the parent cell. Coarsening is simpler: the volume fraction of a parent cell is the average of the volume fractions of the four child cells, as shown in Figure 5(b). Finally, once in a while four cells are coarsened only to have the refinement criterion immediately indicate that the now coarse cell ought to be refined; to prevent loss of accuracy, the child cells are returned to their pre-coarsened state, as illustrated in Figure 5(c).

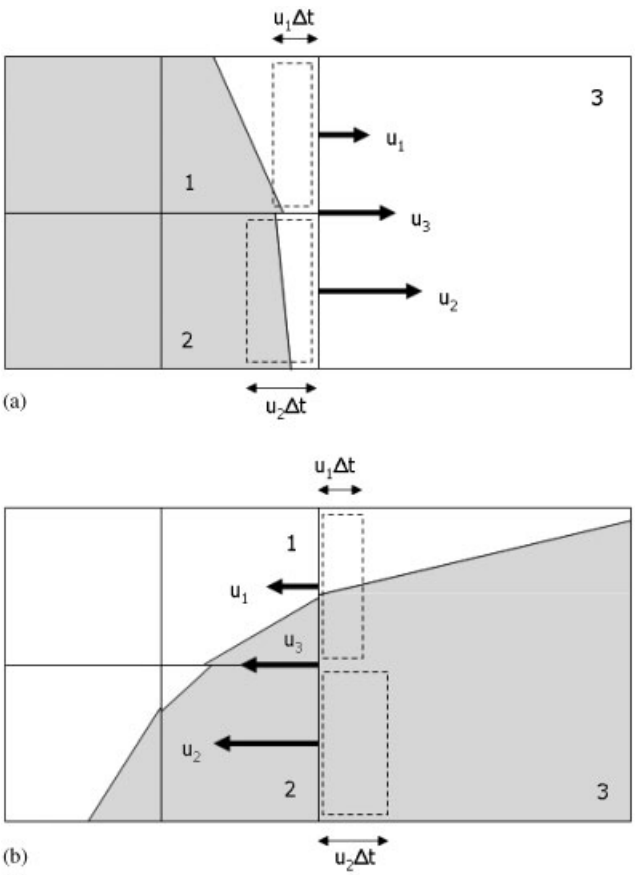


Figure 6. Volume advection: (a) from fine cells to a coarse cell and (b) from a coarse cell to fine cells.

DOI: 10.1002/fld

Moving on to a description of volume tracking, interface reconstruction is exactly the same on an adaptive mesh as on a uniform one: a linear segment, defined by a normal and a line constant, is used to represent the interface. Advection, on the other hand, is different when it occurs between cells of different size, as will happen when an interface is adaptively refined. Referring to Figure 6, whether advection occurs from the two fine cells into a coarse one (Figure 6(a)) or from a coarse into two fine cells (Figure 6(b)), flux volumes are determined at the scale of the fine cells, which in the case of Figure 6(b) implies calculating fluxed volume fractions in a way not required on a uniform mesh.

2.3. Normal and curvature calculation

The accurate calculation of normals $\mathbf{N}=\nabla f$ and curvatures $\kappa=-\nabla\cdot(\nabla f/|\nabla f|)$ from VOF data has long been an issue; recent work [23] makes clear that simple finite difference discretizations of ∇f yield normals that do not converge with mesh refinement, and values of κ that deteriorate with refinement. During implementation of the adaptive methodology presented here, we saw clear evidence of such behaviour that made it impossible to use curvature as a refinement criterion. As a result, we then implemented an entirely different approach, the so-called height function methodology [24]; this is the reason we refer to this implementation as a modified Youngs' [20] method. This methodology yields normals and curvatures that converge with mesh refinement (these are first- and second-order accurate, respectively [23]), at a fraction of the computing cost associated with other recent methods (e.g. [21, 25]) that utilize iterative approaches to determining these quantities.

To summarize, consider Figure 7(b) that illustrates a 3×7 stencil about an interface cell (i, j). The stencil is oriented in the direction more normal to the interface; fluid 'heights' are then

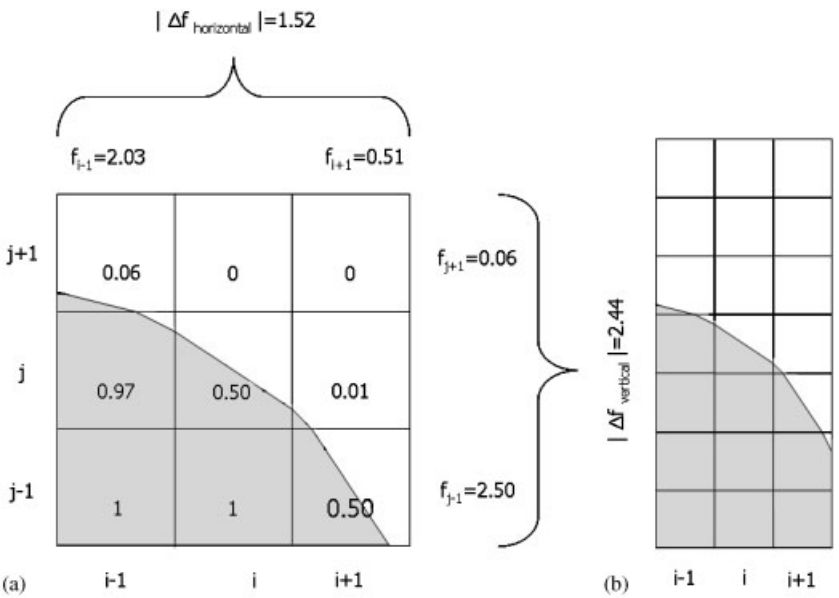


Figure 7. (a) The 3×3 stencil used to determine the orientation of (b) the resulting 3×7 stencil.

DOI: 10.1002/fld

calculated as

$$h_i = \sum_{k=j-3}^{k=j+3} f_{i,k} \Delta y_k \tag{4}$$

The interface normal and curvature are functions of the heights:

$$N_x = h_{i+1} - h_{i-1} \quad \text{and} \quad N_y = -2\Delta x_i \tag{5}$$

$$\kappa = h_{xx}/(1 + h_x^2)^{3/2} \tag{6}$$

where

$$\begin{aligned} h_{xx} &= (h_{i+1} - 2h_i + h_{i-1})/\Delta x_i^2 \\ h_x &= (h_{i+1} - h_{i-1})/2\Delta x_i \end{aligned} \tag{7}$$

where the sign of $\mathbf{N}$ is chosen to point towards fluid no. 1. If the orientation of the interface is more vertical than horizontal, the 3×7 stencil becomes a 7×3.

To choose the stencil orientation, we consider the 3×3 stencil around the cell of interest (i, j) and calculate the sums of volume fractions in the rows and columns about the perimeter of the stencil, as illustrated in Figure 7(a). If the absolute difference between the sum of volume fractions in rows $j + 1$ and $j - 1$ is greater than the difference between the sums in columns $i + 1$ and $i - 1$, the stencil is extended vertically; otherwise, the stencil is extended horizontally.

Height functions are constructed at the level of refinement of each interface cell. If neighbouring cells are coarser, they are temporarily refined; if neighbouring cells are finer, they are temporarily coarsened. To refine a coarser neighbouring cell, the normal and line constant associated with Equation (1) must already be available. For this reason, normals are calculated level by level, from coarse to fine. Once height functions have been calculated, neighbouring cells are returned to their original level of refinement.

Finally, it has been shown recently [23] that height functions only yield accurate normals and curvatures when interfaces are well resolved within the 3×7 stencil. Well-resolvedness is of course the topic of this paper: as the curvature of an interface increases, so should the level of resolution. Nevertheless, in a few instances we encountered fluid configurations within 3×7 stencils that led to poor estimates of normals and curvatures. In particular, a poor stencil is characterized by non-monotonic changes in volume fraction along the heights; an example is illustrated in Figure 8(a). If the slope and curvature of the interface passing through cell (i, j) is of interest, the stencil should ideally include no other interfaces. Yet the cell $(i + 1, j - 3)$ contains just such an interface, evidenced by a volume fraction $(f_{i+1,j-3} = 0.96) < (f_{i+1,j-2} = 0.97)$. In such cases, for the purposes of calculating fluid heights, we set values such as $f_{i+1,j-3}$ to one. In all instances that we examined, this small fix yielded improved values of normals and curvatures.

2.4. Refinement criterion

The primary refinement criterion here is the product of the curvature and the cell dimension, which is required to be less than a specified constant: $\kappa \cdot \Delta x < C$, up to a maximum level of refinement. There are, however, two other conditions for refinement in our implementation: when the difference in refinement level between two neighbouring cells is greater than one, the coarser cell is refined; and when advection occurs from an interface cell to a coarser non-interface cell, the coarser cell is refined as well, to prevent loss of accuracy. In what follows, we refer to these last two criteria as 'quadtree neighbour dependency' and 'quadtree advection dependency,' respectively.

DOI: 10.1002/fld

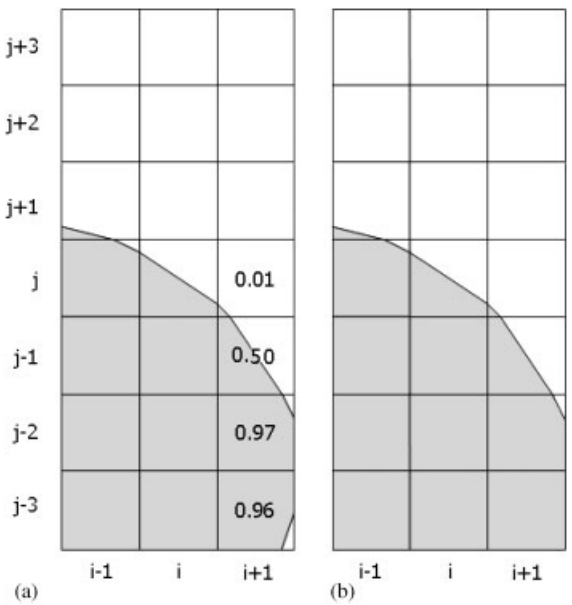


Figure 8. (a) A 3×7 stencil that includes a second interface and (b) a modified stencil.

2.5. Timestep flowchart

To summarize, the following is a flowchart of the algorithm for one timestep.

At the beginning of a simulation, all interface cells are refined to the maximum level in order to calculate an accurate volume fraction distribution; normals and curvatures are then calculated, and cells are coarsened as appropriate. Then the timestepping cycle begins; during each cycle:

1. cells are checked for 'quadtree advection dependency,' and refined if necessary;
2. normals are calculated in a level-by-level approach from coarse to fine, and volume fractions are advected in one direction;
3. normals are re-evaluated, and volume fractions are advected in the other direction;
4. curvatures are calculated, and cells are coarsened according to the criterion;
5. curvatures are calculated again, and cells are refined if $\kappa \cdot \Delta x > C$; then cells are checked again, this time for 'quadtree neighbour dependency,' and refined if necessary;
6. the timestep is complete; return to step 1, and switch the order of advection.

3. RESULTS

We now present the results of four test cases to illustrate the efficacy and error characteristics of adaptive VOF with curvature-based mesh refinement. The error we calculated for each case is defined as

$$E = \sum_{\text{all}} (|f_{i,j}^{\text{T}} - f_{i,j}^{\text{O}}|) \cdot V_{i,j} \tag{8}$$

DOI: 10.1002/fld

Table I. Summary of simulation results.

Test	Level	Refinement criterion C ($\kappa \times \Delta x < C$)	Δt ($\times 10^{-3}$)	No. of Δt	Average no. of interface cells*	Error E ($\times 10^{-3}$)	% E decrease from uniform
Circle stretch	5		6.2500	640	72	2.800	
	4–6	0.18	6.2500	640	71	1.607	43
	6		3.1250	1280	143	0.787	
	5–7	0.075	3.1250	1280	141	0.378	52
	7		1.5625	2560	285	0.182	
	6–8	0.0375	1.5625	2560	282	0.086	53
Ellipse stretch	5		6.2500	250	51	1.126	
	4–6	0.25	6.2500	250	50	0.712	37
	6		3.1250	500	102	0.128	
	5–7	0.11	3.1250	500	102	0.074	42
	7		1.5625	1000	203	0.019	
	6–8	0.055	1.5625	1000	201	0.016	14
Ellipse rotation	5		6.2500	640	62	7.025	
	4–6	0.12	6.2500	640	66	8.749	−25
	6		3.1250	1280	123	2.591	
	5–7	0.06	3.1250	1280	120	2.472	5
	7		1.5625	2560	246	0.820	
	6–8	0.03	1.5625	2560	248	0.970	−18
Slotted disk rotation	6		3.1250	1280	200	3.863	
	5–7	0.1	3.1250	1280	199	3.394	12
	7		1.5625	2560	399	1.434	
	6–8	0.045	1.5625	2560	404	1.008	30

*The number of interface cells is the average over the entire simulation.

DOI: 10.1002/fld

The summation is over all cells, the superscripts O and T refer to initial and final values, and $V_{i,j}$ refers to the area of cell (i, j). In each test, an initial interface returns to its original position at the end of a simulation; E then represents the error between the initial and final values of volume fractions summed over the entire domain. A summary of all results is presented in Table I.

In all cases, the tests were run on adaptive meshes, first with the constraint that all interface cells be refined to the same level α, and then on an adaptive mesh with interface cells refined based on curvature, limited between levels $\alpha-1$ and $\alpha+1$. For conciseness, we refer to the first as a 'uniform' calculation and the second as a 'curvature-based' one. For each curvature-based calculation, the constant C in the refinement criterion $\kappa \cdot \Delta x < C$ was chosen so that the number of interface cells (averaged over time) was approximately the same for the uniform and curvature-based calculations, to enable a comparison. Timestep size was the same for an adaptive and corresponding uniform mesh result; timesteps were chosen to limit the Courant number to 0.4 for all simulations.

The first case, illustrated in Figures 9 and 10, begins as a circle of radius 0.25 centred at the origin; the following velocity field:

$$u = -\frac{\pi}{4}(x^2 - y^2), \quad v = \frac{\pi}{2}xy \tag{9}$$

stretches the circle into a triangular shape in 640 timesteps; the velocity field is then reversed and the interface advected for another 640 timesteps so that the interface returns to its initial configuration. Figure 9 shows the simulation results on a level 6 uniform mesh; Figure 10 shows the corresponding curvature-based results, for $\kappa\Delta x < 0.075$. It is important to note that we present these two figures, and subsequent ones, not to compare the interface reconstructions, but to contrast the adaptive and uniform meshes. In fact, we superimposed images such as 9(a), 9(e), and 10(e) to compare the interface reconstructions, but the superimposed images were difficult to distinguish.

Comparing Figures 9(c) and 10(c), the efficacy of the curvature-based approach is most evident. Initially the circle is resolved equally well: the curvature-based refinement calculates curvatures that lead to refinement of the circle at the same level as the uniform results. But as the circle is stretched, the curvature increases at the corners of the triangle, which then are resolved more accurately, at the expense of the sides of the triangle that require less resolution. Overall, the curvature-based approach yields an error 52% less than the uniform one. Comparisons of coarser (level 5 *versus* levels 4–6) and finer (level 7 *versus* levels 6–8) results demonstrate similar decreases (Table I).

Curvature-based results of the second case are illustrated in Figure 11. An initial ellipse, defined by $x^2/0.12 + y^2/0.01 = 1$, is advected by the following velocity field:

$$u = -\frac{8}{5}x, \quad v = \frac{8}{5}y \tag{10}$$

The ellipse is compressed horizontally and stretched vertically; the field is then reversed and the interface returns to its initial configuration. The results are similar to those of the first test: the use of adaptivity in representing the interface leads to a significant decrease in the final error E.

The final two tests are fundamentally different than the first two, in that the velocity field is a solid body rotation, so that the shape of the interface should not change. Figure 12 illustrates the results of an ellipse (equivalent to that of the second test) that is rotated about its centre. Figure 12(a) shows the initial uniform level 7 ellipse; Figure 12(b) shows the final uniform level 7 result; and Figure 12(c) shows the final curvature-based level 6–8 adaptive result. Unlike the previous cases, the benefits of adaptivity are less obvious: the high-curvature ends of the ellipse are

DOI: 10.1002/fld

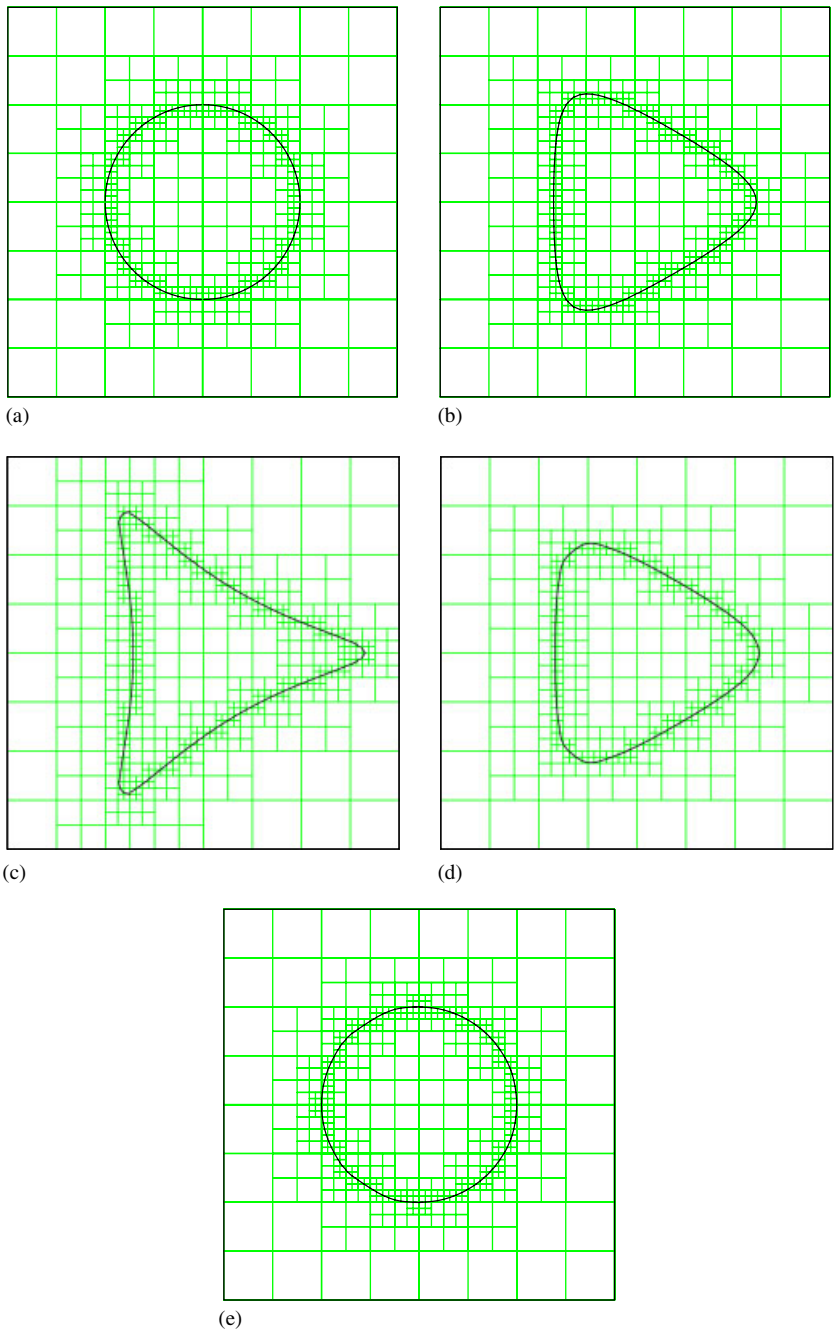


Figure 9. Circle stretch on a level 6 uniform mesh at timesteps: (a) 0; (b) 320; (c) 640; (d) 960; and (e) 1280.

DOI: 10.1002/fld

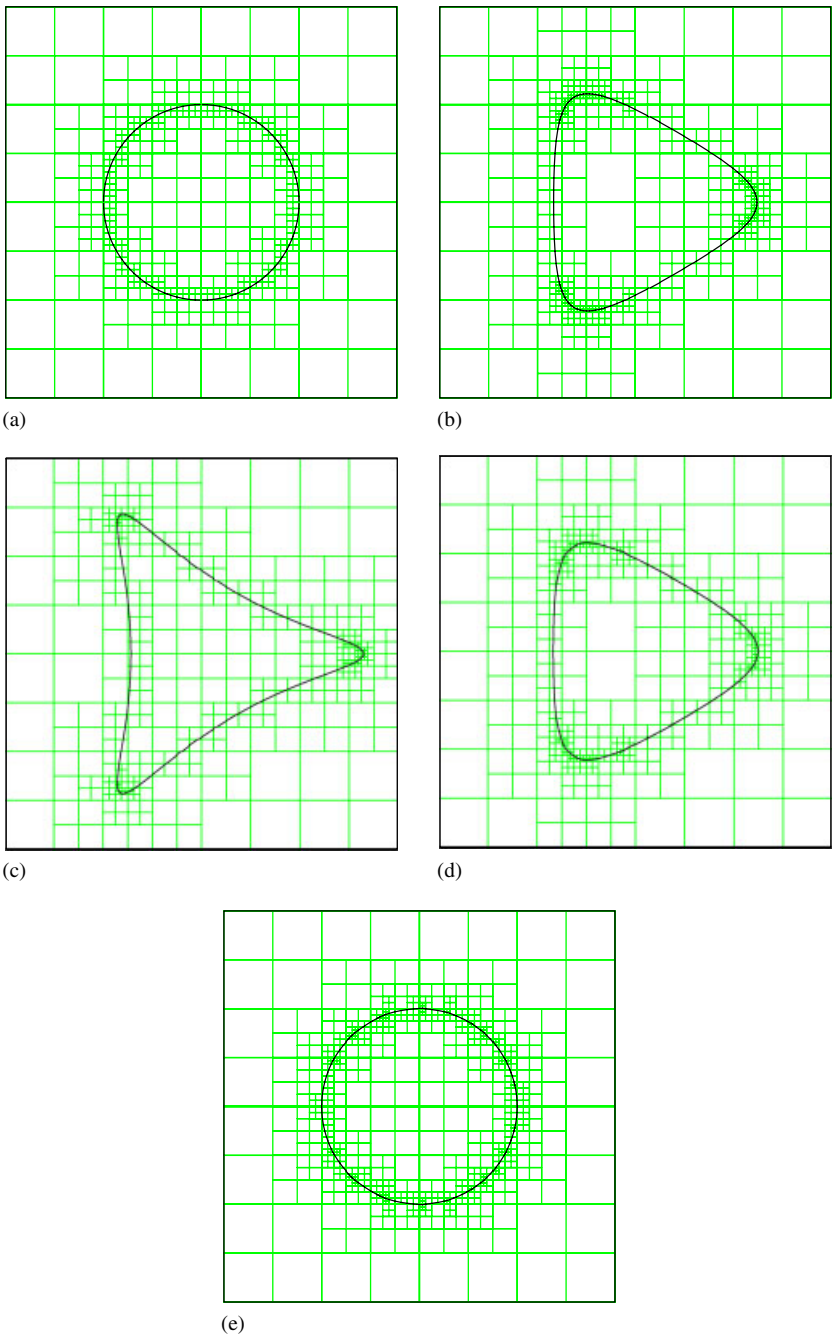


Figure 10. Circle stretch on a level 5–7 curvature-based adaptive mesh at timesteps: (a) 0; (b) 320; (c) 640; (d) 960; and (e) 1280.

DOI: 10.1002/fld

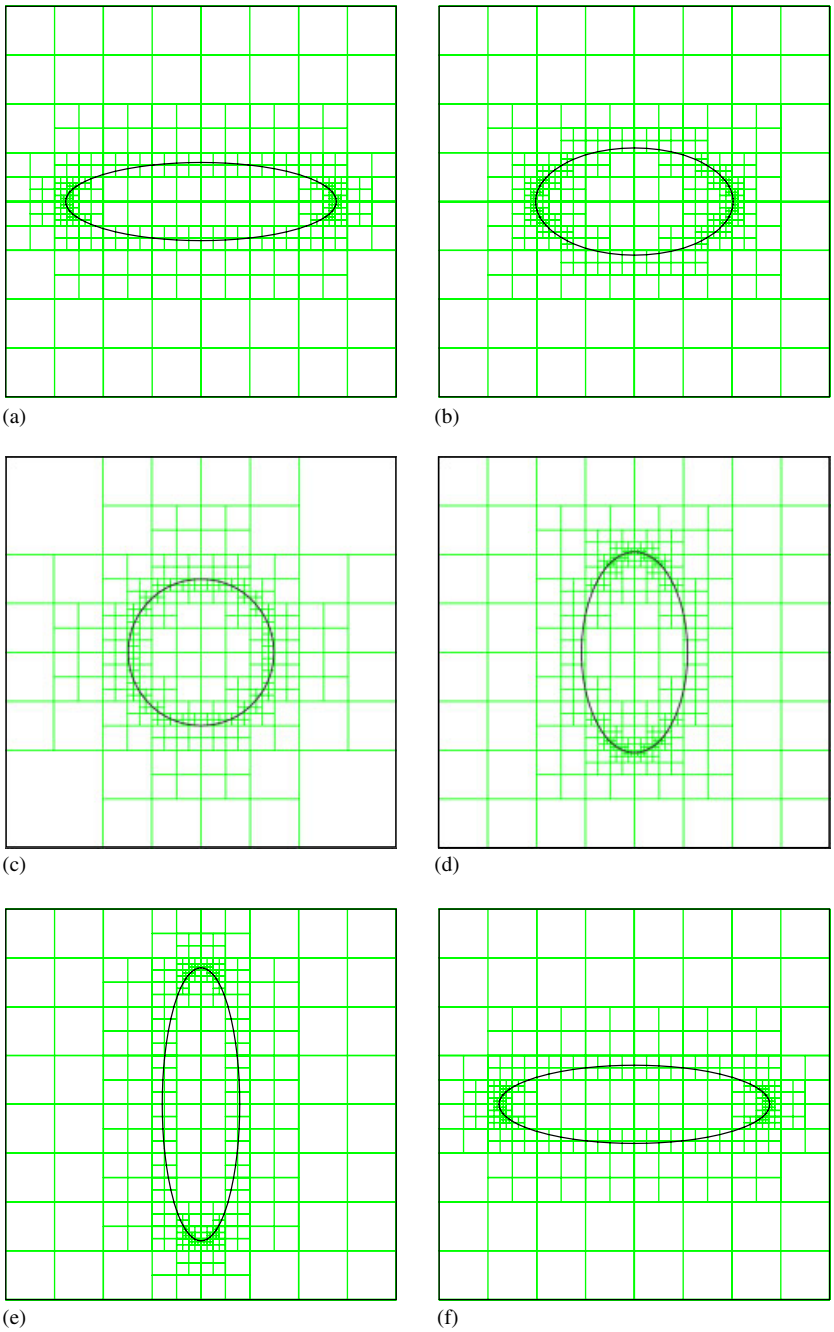


Figure 11. Ellipse stretch on a level 5–7 curvature-based adaptive mesh at timesteps: (a) 0; (b) 63; (c) 125; (d) 188; (e) 250; and (f) 500.

DOI: 10.1002/fld

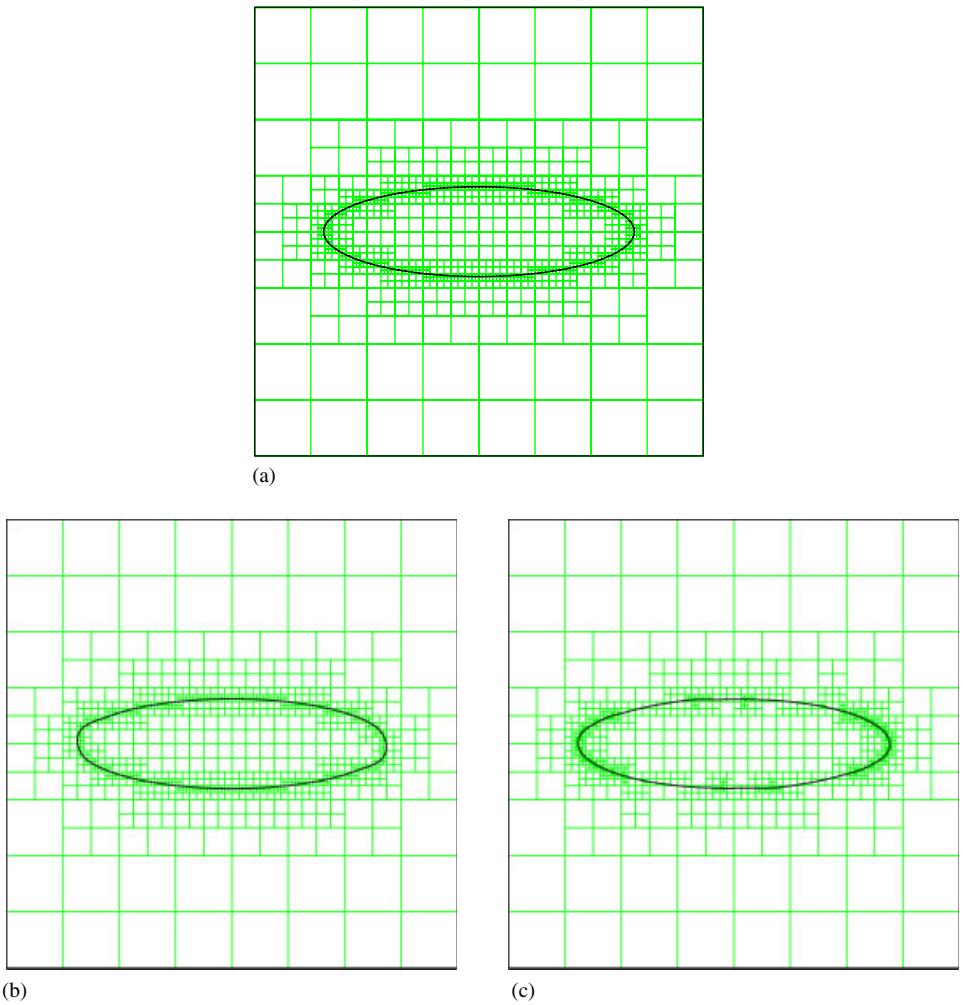


Figure 12. Ellipse rotation: (a) the initial level 7 ellipse; and final ellipses after one complete rotation; (b) a level 7 uniform calculation; and (c) a level 6–8 curvature-based adaptive calculation.

Table II. Level 6–8 rotation results for ellipses of different aspect ratio a/b.

$(a/b)^2$	Refinement criterion C	Average no. of interface cells	Error E ($\times 10^{-3}$)	% E decrease from uniform
6	0.036	263	0.765	−64
12	0.030	248	0.970	−18
24	0.026	239	0.932	19
36	0.024	232	1.065	15
48	0.023	231	1.105	15

Results correspond to 2560 timesteps with $\Delta t = 1.5625 \times 10^{-3}$.

DOI: 10.1002/fld

certainly better represented by the curvature-based adaptive mesh, but apparently at the expense of a significantly poorer representation elsewhere: the error of the curvature-based adaptive result is 18% greater than that of the uniform result. This case was run at two coarser resolutions as well, as summarized in Table I; the results were similarly unimpressive. To better understand this, we ran level 6–8 simulations of the rotation of ellipses of other aspect ratios (by varying the size of the minor axis), and compared errors to corresponding uniform ones. The results are presented

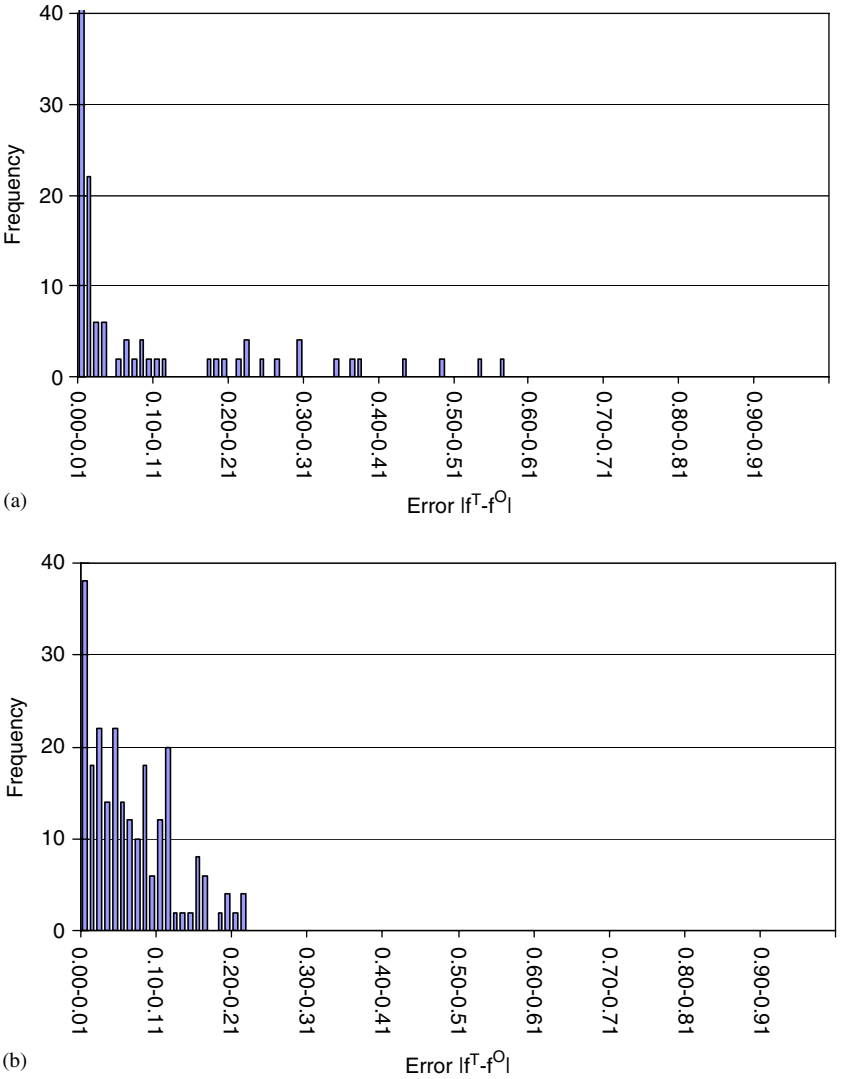


Figure 13. Histograms of error distribution at the end of an ellipse rotation (Figure 12): (a) a level 7 uniform calculation (standard deviation of 0.119) and (b) a level 6–8 curvature-based calculation (standard deviation of 0.055). Note that in (a), the number of cells with an error between 0.00 and 0.01 is actually 154.

DOI: 10.1002/fld

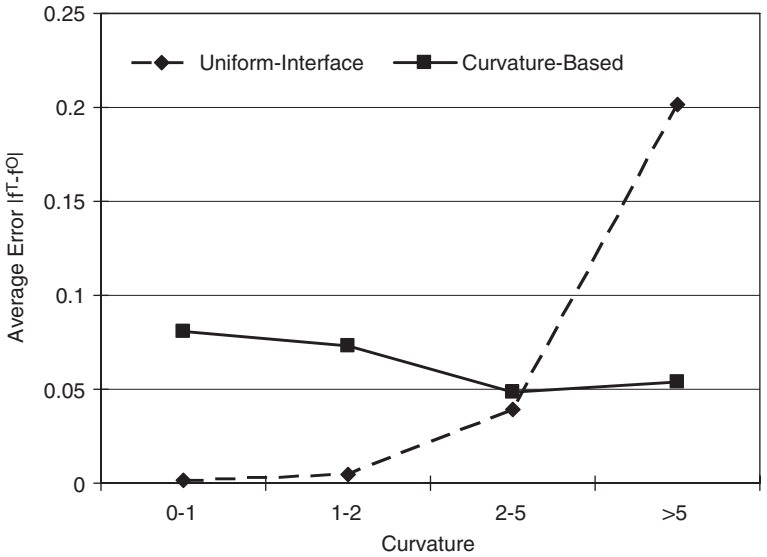


Figure 14. Average error $|f^{\mathrm{T}} - f^{\mathrm{O}}|$ *versus* curvature for the uniform and adaptive results of the rotating ellipse (Figure 12).

in Table II, and show that the benefits of adaptivity increase with the complexity of the interface (defined in this case as the aspect ratio), and that adaptivity as evaluated here (comparing a uniform result to a three level result) can actually lead to larger errors on problems that do not include a significant variation of curvature.

This third test nevertheless provides an opportunity to examine the distribution of error along an interface. Figure 13 presents histograms of error $|f^{\mathrm{T}}_{i,j} - f^{\mathrm{O}}_{i,j}|$ for both the uniform and curvature-based adaptive results (to facilitate a comparison, the final results of both cases were converted to a uniform level 7 mesh before calculating the errors). As expected, the histogram for the uniform case indicates a large number of cells with very small errors (note that the first column in Figure 13(a) actually extends to 154) and a few cells with large errors; these are cells near the high curvature ends of the ellipse. The adaptive results, on the other hand, are characterized by a more uniform yet narrower distribution of error: fewer cells with very small errors, but a maximum error that is little more than a third of the maximum error on the uniform ellipse. In statistical terms, the few large errors of the uniform case lead to a standard deviation of the error of 0.119, more than twice the standard deviation of the curvature-based results (0.055). Finally, Figure 14 reinforces the same message: on a uniform mesh, the average error in a cell increases dramatically with curvature; on the curvature-based adaptive mesh, the errors remain nearly constant independent of curvature.

Finally, Figure 15 illustrates the results of the fourth test, the solid-body rotation of a slotted disk (the disk radius is 0.3, the slot is 0.125 wide, and 0.3 high). Figure 15(a) shows a uniform level 7 disk; Figure 15(b) shows the final slotted disk from a level 7 uniform mesh simulation; and Figure 15(c) shows the final slotted disk of a corresponding curvature-based adaptive simulation. In this case, the high curvature is entirely at the corners of the slot; the perimeter of the disk is less curved; and the straight sides of the slot obviously have zero curvature. Close examination of the corners demonstrates clearly that the curvature-based adaptive results have resolved these more

DOI: 10.1002/fld

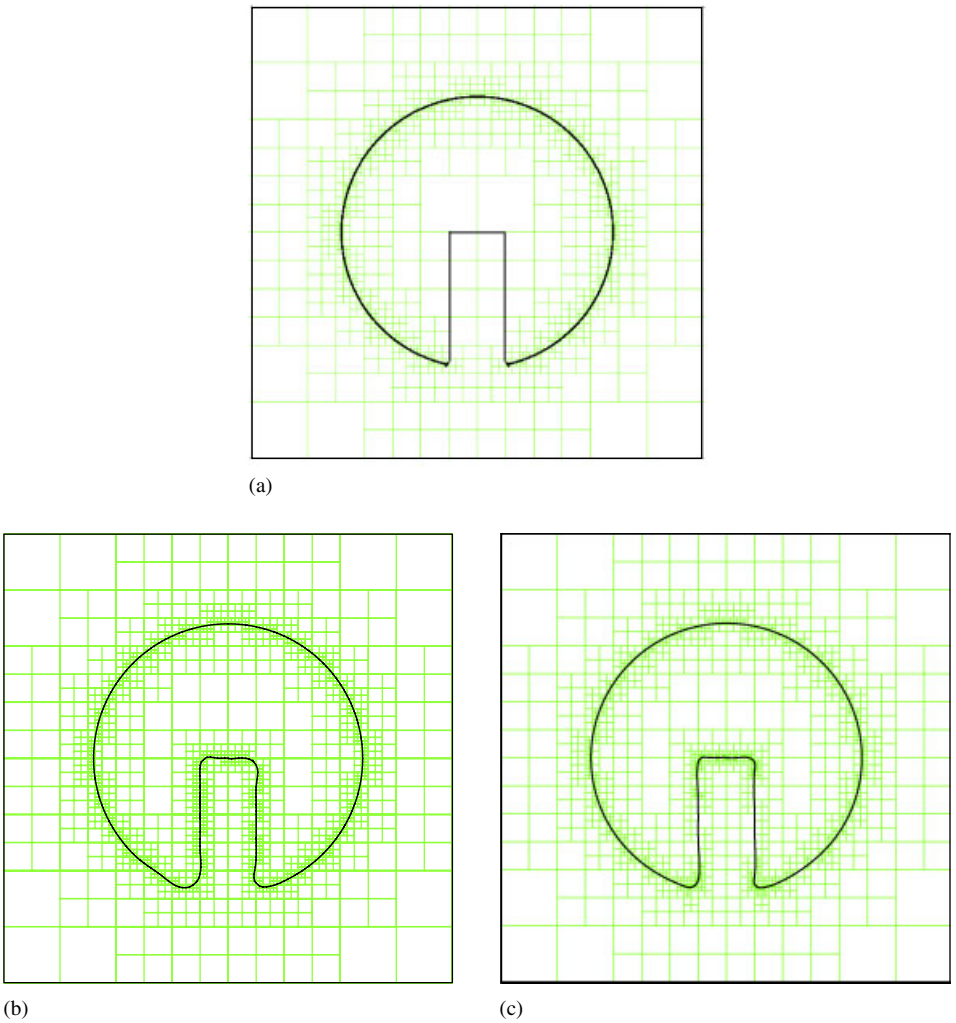


Figure 15. Slotted disk rotation: (a) the initial level 7 slotted disk; and results after one complete rotation; (b) a level 7 uniform calculation; and (c) a level 6–8 curvature-based adaptive calculation.

clearly, while resolving the sides of the slots at a low resolution. Again, this test demonstrates that curvature-based adaptivity leads to an overall reduction in the error.

4. SUMMARY

When adaptive mesh refinement is applied to Eulerian simulations of interfacial flow (using methods like level sets or VOF), in most cases the resolution of interfaces is maintained at a constant level even though curvature can vary dramatically when interfaces are advected by complex flows. This

DOI: 10.1002/fld

led us to study the efficacy of curvature-based refinement of interfaces, implemented here in a VOF method, although the conclusions of this work may well apply to other interface representations as well.

Regarding implementation, the adaptive mesh complicates the VOF advection step when neighbouring cells are at different levels of refinement; and the use of curvature as the refinement criterion led us to implement the height function methodology to calculate normals and curvatures.

From the results of 11 comparisons of adaptive and uniform refinement based on four test cases, the adaptive results were more accurate in nine instances, with errors reduced by up to 50%. In the remaining two instances, the reason for an increase in error is not readily apparent, although obviously the lower errors associated with increased resolution along parts of the interface were more than offset by larger errors elsewhere. Finally, an analysis of the error distribution along an interface clearly illustrates that curvature-based refinement leads to errors that are nearly independent of curvature.

REFERENCES

1. Devals C, Heniche M, Bertrand F, Hayes RE, Tanguy PA. A finite element strategy for the solution of interface tracking problems. *International Journal for Numerical Methods in Fluids* 2005; **49**:1305–1327.
2. Ginzburg I, Wittum G. Two-phase flows on interface refined grids modeled with VOF, staggered finite volumes, and spline interpolants. *Journal of Computational Physics* 2001; **166**:302–335.
3. Greaves D. A quadtree adaptive method for simulating fluid flows with moving interfaces. *Journal of Computational Physics* 2004; **194**:35–56.
4. Greaves D. Simulation of interface and free surface flows in a viscous fluid using adapting quadtree grids. *International Journal for Numerical Methods in Fluids* 2004; **44**:1093–1117. DOI: 10.1002/fld.687
5. Greaves DM. Simulation of viscous water column collapse using adapting hierarchical grids. *International Journal for Numerical Methods in Fluids* 2006; **50**:693–711. DOI: 10.1002/fld.1073
6. Hay A, Visonneau M. Computation of free-surface flows with local mesh adaptation. *International Journal for Numerical Methods in Fluids* 2005; **49**:785–816. DOI: 10.1002/fld.1042
7. Jeong JH, Yang DY. Finite element analysis of transient fluid flow with free surface using VOF (volume-of-fluid) method and adaptive grid. *International Journal for Numerical Methods in Fluids* 1998; **26**:1127–1154.
8. Jeong JH, Yang DY. Three-dimensional finite element analysis of transient fluid flow with free-surface using marker surface method and adaptive grid refinement. *International Journal for Numerical Methods in Fluids* 1999; **29**:657–684.
9. Kohno H, Tanahashi T. Numerical analysis of moving interfaces using a level set method coupled with adaptive mesh refinement. *International Journal for Numerical Methods in Fluids* 2004; **45**:921–944. DOI: 10.1002/fld.715
10. Sussman M. A second order coupled level set and volume-of-fluid method for computing growth and collapse of vapor bubbles. *Journal of Computational Physics* 2003; **187**:110–136.
11. Theodorakakos A, Bergeles G. Simulation of sharp gas–liquid interface using VOF method and adaptive grid local refinement around the interface. *International Journal for Numerical Methods in Fluids* 2004; **45**:421–439. DOI: 10.1002/fld.706
12. Wang JP, Borthwick AGL, Taylor RE. Finite-volume-type VOF method on dynamically adaptive quadtree grids. *International Journal for Numerical Methods in Fluids* 2004; **45**:485–508. DOI: 10.1002/fld.712
13. Losasso F, Fedkiw R, Osher S. Spatially adaptive techniques for level set methods and incompressible flow. *Computers and Fluids* 2006; **35**:995–1010.
14. Bussmann M, Chandra S, Mostaghimi J. Modeling the splash of a droplet impacting a solid surface. *Physics of Fluids* 2000; **12**:3121–3132.
15. Gueyffier D, Zaleski S. Formation de digitations lors de l'impact d'une goutte sur un film liquide. *Comptes Rendus de l'Académie des Sciences—Série IIb*: *Mécanique*, *Physique*, *Chimie*, *Astronomie* 1998; **326**:839–844.
16. Rieber M, Frohn A. A numerical study on the mechanism of splashing. *International Journal of Heat and Fluid Flow* 1999; **20**:455–461.
17. Dai M, Schmidt DP. Adaptive tetrahedral meshing in free-surface flow. *Journal of Computational Physics* 2005; **208**:228–252.

18. Sussman M, Almgren AS, Bell JB, Colella P, Howell LH, Welcome ML. An adaptive level set approach for incompressible two-phase flows. *Journal of Computational Physics* 1999; **148**:81–124.
19. Löhner R, Yang C, Oñate E. Simulation of flows with violent free surface motion and moving objects using unstructured grids. *International Journal for Numerical Methods in Fluids* 2007; **53**:1315–1338. DOI: 10.1002/fld.1244
20. Youngs DL. Time-dependent multi-material flow with large fluid distortion. In *Numerical Methods for Fluid Dynamics*, Morton KW, Baines MJ (eds). Academic Press: New York, 1982; 273–285.
21. Pilliod Jr. JE, Puckett EG. Second-order accurate volume-of-fluid algorithms for tracking material interfaces. *Journal of Computational Physics* 2004; **199**:465–502.
22. Samet H. *The Design and Analysis of Spatial Data Structures*. Addison-Wesley: Reading, MA, 1990.
23. Cummins SJ, Francois MM, Kothe DB. Estimating curvature from volume fractions. *Computers and Structures* 2005; **83**:425–434.
24. Francois MM, Cummins SJ, Dendy ED, Kothe DB, Sicilian JM, Williams MW. A balanced-force algorithm for continuous and sharp interfacial surface tension models within a volume tracking framework. *Journal of Computational Physics* 2006; **213**:141–173.
25. Renardy Y, Renardy M. PROST: a parabolic reconstruction of surface tension for the volume-of-fluid method. *Journal of Computational Physics* 2002; **183**:400–421.