Journal of Computational Physics **183,** 83–116 (2002)
doi:10.1006/jcph.2002.7166

A Hybrid Particle Level Set Method for Improved Interface Capturing

Douglas Enright,*,[1,2] Ronald Fedkiw,†,[1,2] Joel Ferziger,‡,[2] and Ian Mitchell*,[3]

*Scientific Computing—Computational Mathematics Program, †Computer Science Department, and ‡Mechanical Engineering Department, Stanford University, Stanford, California 94305

Received August 9, 2001; revised July 9, 2002

In this paper, we propose a new numerical method for improving the mass conservation properties of the level set method when the interface is passively advected in a flow field. Our method uses Lagrangian marker particles to rebuild the level set in regions which are underresolved. This is often the case for flows undergoing stretching and tearing. The overall method maintains a smooth geometrical description of the interface and the implementation simplicity characteristic of the level set method. Our method compares favorably with volume of fluid methods in the conservation of mass and purely Lagrangian schemes for interface resolution. The method is presented in three spatial dimensions. © 2002 Elsevier Science (USA)

1. INTRODUCTION

Level set methods have become widely used for capturing interface evolution especially when the interface undergoes extreme topological changes, e.g., merging or pinching off. The application of level set methods to a wide variety of problems including fluid mechanics, combustion, computer vision, and materials science is discussed in the recent review articles by Osher and Fedkiw [20] and Sethian [34] as well as the recent books of Osher and Fedkiw [21] and Sethian [33]. The great success of level set methods (and other Eulerian methods) can be attributed to the role of curvature in numerical regularization such that the proper vanishing viscosity solution is obtained. This connection between curvature and the notion of entropy conditions and shocks for hyperbolic conservation laws was explored by Sethian [29, 30]. Any Lagrangian scheme used to solve the same problem, including the

[1] Research supported in part by an ONR YIP and PECASE Award N00014-01-1-0620 and NSF DMS-0106694.

[2] Research supported in part by DOE under the ASCI Academic Strategic Alliances Program (LLNL Contract B341491).

[3] Research supported in part by DARPA under the Software Enabled Control Program (AFRL Contract F33615-99-C-3014).

0021-9991/02 $35.00

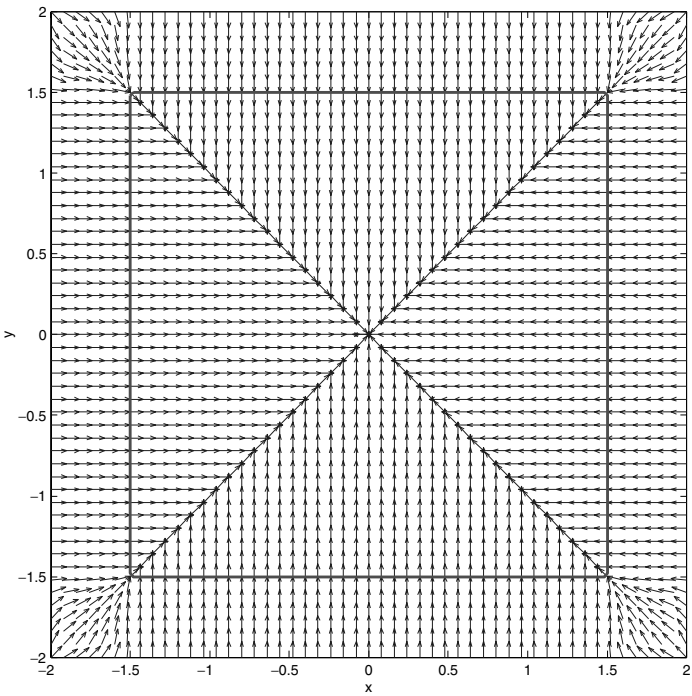


FIG. 1. Shrinking square: Initial interface location and velocity field.

discretization of the interface by marker particles, cannot readily achieve a similar result since there is no a priori way to build regularization into the method. Lagrangian particles faithfully follow the characteristics of the flow but must be deleted (usually "by hand") if the characteristics in a given region merge.

The ability to identify and delete merging characteristics is clearly seen in a purely geometrically driven flow where a curve is advected normal to itself at constant speed. Figure 1 shows an initial square interface described by the $\phi = 0$ isocontour of a level set function along with the associated velocity field defined by $\nabla\phi/|\nabla\phi|$. Note that the flow field has merging characteristics on the diagonals of the square. Figure 2 shows an adequate numerical solution computed using the level set method to correctly shrink the box as time increases. On the other hand, a Lagrangian front-tracking model of the interface will not calculate the correct motion. We demonstrate this by seeding passively advected particles interior to the zero isocontour of the level set function as shown in Fig. 3. As the particle positions evolve in time, they follow the characteristic velocities of the flow field as shown in Fig. 4. Note how the particles incorrectly form long, slender filaments on the diagonals of the square where characteristics merge and should be deleted (as is correctly done by the level set method). Helmsen [13] and others have used "de-looping" procedures in an attempt to remove these incorrect particle tails. While these procedures are sometimes manageable in two spatial dimensions, they become increasingly intractable in three spatial dimensions.

Almost from their inception, level set methods have been used to model multiphase immiscible incompressible flows; see [41]. These methods are attractive because they admit a convenient description of topologically complex interfaces and are quite simple to implement. Lagrangian style front-tracking schemes, e.g., Tryggvason *et al.* [43] and Unverdi and Tryggvason [44] and marker particles [6, 7, 12, 25, 26, 42], have also been

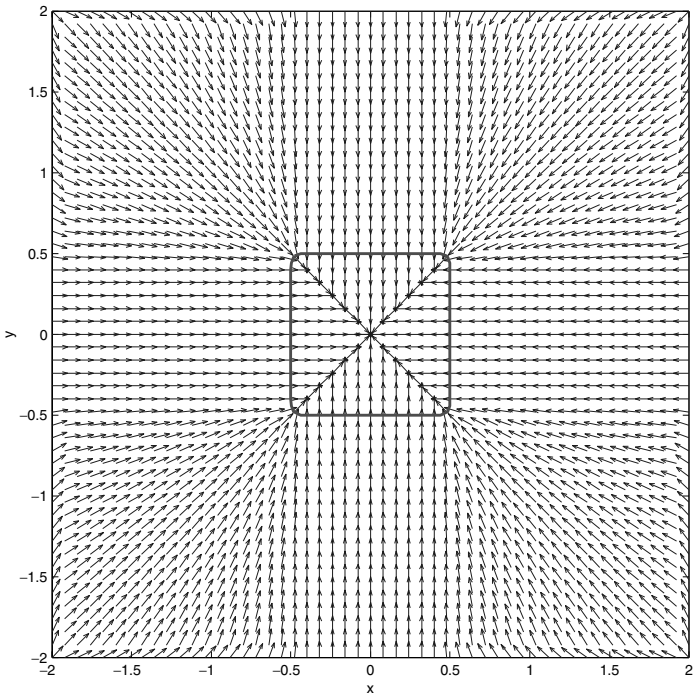


FIG. 2. Shrinking square: Final interface location of the (correct) level set solution.

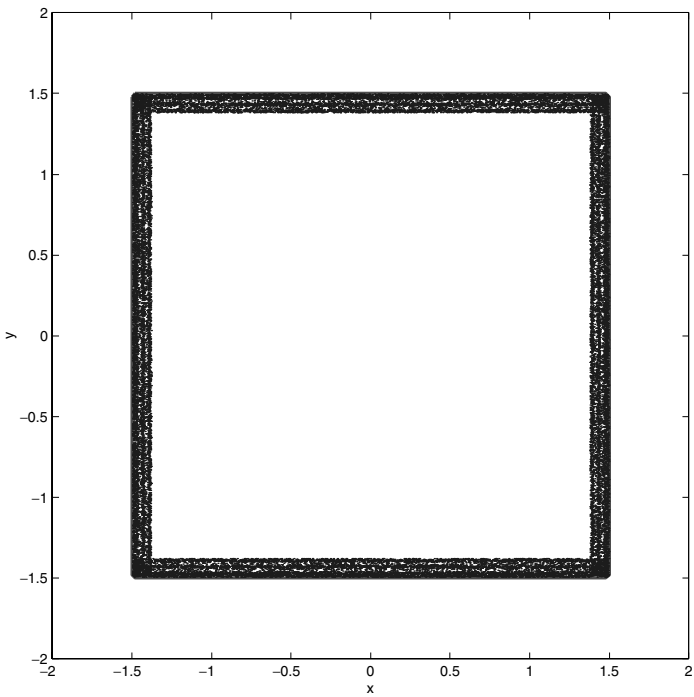


FIG. 3. Shrinking square: Passively advected particles are initially seeded inside the interface.

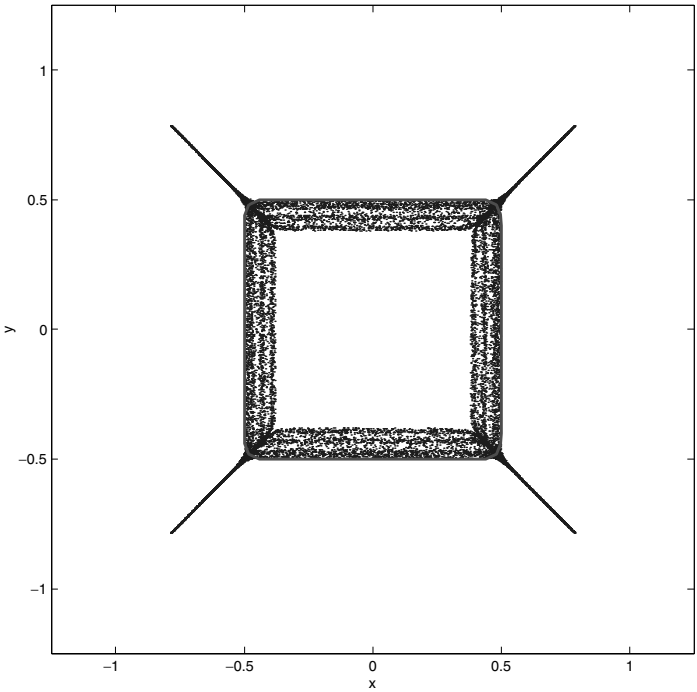


FIG. 4. Shrinking square: Final (incorrect) location of the passively advected marker particles.

used to model this class of problems. In addition, a popular alternative Eulerian front capturing scheme, the volume of fluid (VOF) method [14, 24], is widely used. All of these methods have had varying degrees of success in correctly modeling flows with large vortical components. To compare the fidelity of these schemes, Rider and Kothe proposed a set of test problems [27, 28] which approximate flows with large vortical components. In a comparison of various Lagrangian and Eulerian methods for these flows, Rider and Kothe found that Lagrangian tracking schemes maintain filamentary interface structures better than their Eulerian counterparts. In the same study, it was noted that when fluid filaments become too thin to be adequately resolved on the grid, level set methods lose (or gain) mass while VOF methods form "blobby" filaments to locally enforce mass conservation. Both of these errors decrease the accuracy of the predicted interface location. Attempts to improve mass conservation in level set methods have led to a variety of reinitialization methods, from the original redistancing algorithm of Sussman *et al.* [41] to a more recent method by Sussman and Fatemi [39] and Sussman *et al.* [40], which constrains the reinitialization scheme to approximately conserve area (volume). In addition, the use of higher order ENO/WENO [15] approximations for the spatial derivatives during the convection and reinitialization steps has been proposed.

Despite a lack of explicit enforcement of volume conservation, Lagrangian schemes are quite successful in conserving mass since they preserve material characteristics for all time as opposed to regularizing them out of existence as may happen with Eulerian front-capturing methods. For underresolved flows, Eulerian capturing methods cannot accurately show if characteristics merge, separate, or are parallel. This indeterminacy can cause level set methods to calculate the weak solution to the problem and delete characteristics when they appear to be merging. Osher and Sethian [22] constructed level set methods to deal with

the case in which characteristics *do* merge as seen in Fig. 2, i.e., to recognize the presence of shocks and delete merging characteristic information. Now we are faced with a difficult question concerning the appropriateness of using level set schemes designed to merge characteristics automatically when we lack knowledge about the characteristic structure in underresolved regions of the flow. Moreover, in the case of fluid flows, we know a priori that there are no shocks present in the fluid velocity field, and thus characteristic information in that field should never be incorrectly deleted. This is the case for both incompressible and compressible flows. For compressible flows, only acoustic characteristic fields form shocks, while the linearly degenerate fluid velocity does not.

Keeping this in mind, we briefly turn our attention away from level set methods. Eulerian VOF schemes have the accuracy problems noted above and, in the attempt to maintain local mass conservation, "blobby" flotsam and jetsam can spuriously appear [17]. The reconstructed interface is not smooth or even continuous, lowering the accuracy of the geometrical information (normals and curvature) at the interface compromising the entire solution. Several researchers have worked to improve the accuracy of the VOF geometrical information using convolution; see, e.g., [45, 46]. Sussman and Puckett [37] combined the VOF and level set methods in order to alleviate some of the geometrical problems of the VOF method. The resulting scheme is completely Eulerian and does not incorporate any of the front-tracked characteristic information needed in underresolved regions. Instead, the VOF local mass constraint is still blindly applied. On the other hand, front-tracking schemes pose many difficulties in the reconstruction of the interface especially in three spatial dimensions for pinching and merging droplets [43]. The use of marker particles is an attractive idea since the lack of connectivity makes implementation much easier than front-tracking. Unfortunately, one is left with a less than satisfactory description of the interface geometry making accurate calculations with surface tension difficult. In fact, even front-tracking schemes need smoothing near the interface in order to obtain smooth geometrical quantities [48]. Level sets have a simplicity not matched by many of the above methods while still maintaining nice geometric properties. However, they suffer from excessive mass loss especially in underresolved regions.

In this paper, we propose a new method which combines the best properties of an Eulerian level set method and a marker particle Lagrangian scheme. Our method randomly places a set of marker particles near the interface (defined by the zero level set) and allows them to passively advect with the flow. In fluid flows, particles do not cross the interface except when the interface-capturing scheme fails to accurately identify the interface location. If the marker particles initially seeded on one side of the interface are detected on the opposite side, this indicates an error in the level set representation of the interface. We fix these errors by locally rebuilding the level set function using the characteristic information present in these escaped marker particles. This allows the level set method to obtain a subgrid scale accuracy near the interface and works to counteract the detrimental mass loss of the level set method in underresolved regions. The particles play no other role in the calculation and the smooth geometry of the interface is determined by the level set function alone. Also, since the marker particles are disconnected, the ease and simplicity of coding level set methods are maintained by this "particle level set" method. Numerical results based upon a series of two- and three-dimensional interface stretch tests proposed and inspired by Rider and Kothe [27, 28] are described to demonstrate that the new particle level set technique compares favorably with VOF methods with regard to mass conservation and with purely Lagrangian schemes with regard to interface resolution.

Before proceeding, we remark that this new method was designed to track material interfaces for both incompressible and compressible flows where characteristics are not created or destroyed. In these instances, we can use marker particles to accurately track characteristic information without considering shocks and rarefactions where particles need to be created and destroyed in a consistent fashion. The particle level set method has not yet been extended to treat more complex flows such as those involving geometry, e.g., motion normal to the interface or motion by mean curvature. However, extending the particle level set method to treat the reinitialization equation is straightforward since the exact solution dictates that both the interface (zero level set) and the marker particles should be still.

2. NUMERICAL METHOD

2.1. Level Set Method

The underlying idea behind level set methods is to embed an interface Γ in R^3, which bounds an open region $\Omega \subset R^3$ as the zero level set of a higher dimensional function $\phi(\vec{x}, t)$. The level set function has the following properties,

$$\phi(\vec{x}, t) > 0 \quad \text{for } \vec{x} \in \Omega$$
$$\phi(\vec{x}, t) \leq 0 \quad \text{for } \vec{x} \notin \Omega,$$

where we include $\phi = 0$ with the negative ϕ values so that it is not a special case. The interface lies between $\phi > 0$ and $\phi = 0$ but can of course be identified as $\phi = 0$. Note that ϕ is a scalar function in R^3, which greatly reduces the complexity of describing the interface, especially when undergoing topological changes such as pinching and merging.

The motion of the interface is determined by a velocity field, $\vec{u}$, which can depend on a variety of things including position, time, and geometry of the interface, or be given externally for instance as the material velocity in a fluid flow simulation. In most of the examples below, the velocity field is externally given, and the evolution equation for the level set function is given by

$$\phi_t + \vec{u} \cdot \nabla\phi = 0. \tag{1}$$

This equation only needs to be solved locally near the interface; e.g., see [1, 23].

It is convenient to make ϕ equal to the signed distance to the interface so that $|\nabla\phi| = 1$. This ensures that the level set is a smoothly varying function well suited for high-order accurate numerical methods. Unfortunately, as noted in [41], the level set function can quickly cease to be a signed distance function especially for flows undergoing extreme topological changes. Reinitialization algorithms maintain the signed distance property by solving to steady state (as fictitious time $\tau \to \infty$) the equation

$$\phi_\tau + \text{sgn}(\phi_0)(|\nabla\phi| - 1) = 0, \tag{2}$$

where $\text{sgn}(\phi_0)$ is a one-dimensional smeared out signum function approximated numerically in [41] as

$$\text{sgn}(\phi_0) = \frac{\phi_0}{\sqrt{\phi_0^2 + (\Delta x)^2}}.$$

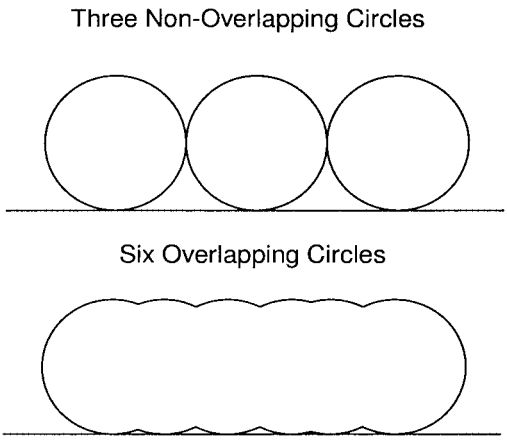


FIG. 5. The top picture shows how three equally sized nonoverlapping circles leave large spaces when we attempt to represent a straight line interface, while the bottom picture shows how six equally sized overlapping circles more readily resolve the line.

Efficient ways to solve Eq. (2) to steady state via fast marching methods are discussed in [31, 32]. Again, Eq. (2) only needs to be solved locally near the interface. We use a fifth-order accurate Hamilton–Jacobi WENO scheme [15] to calculate the spatial derivatives in both Eqs. (1) and (2).

Geometrical quantities can be calculated from the level set function, including the unit normal,

$$\vec{N} = \frac{\nabla \phi}{|\nabla \phi|}, \tag{3}$$

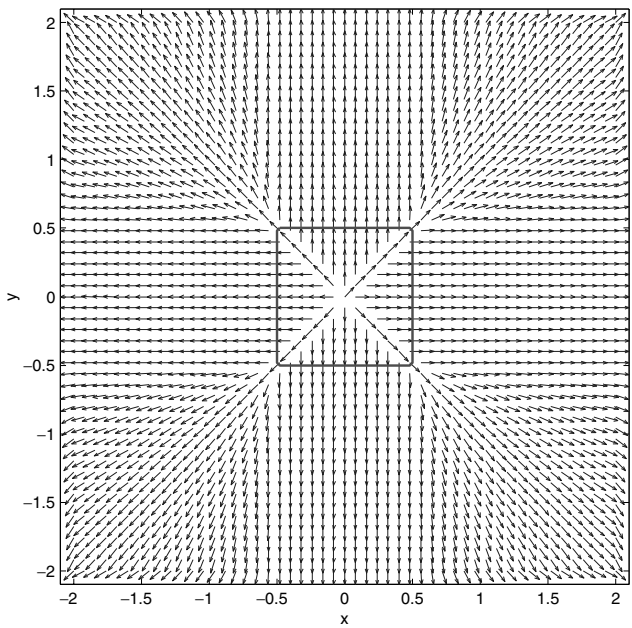


FIG. 6. Expanding square: Initial interface location and velocity field.

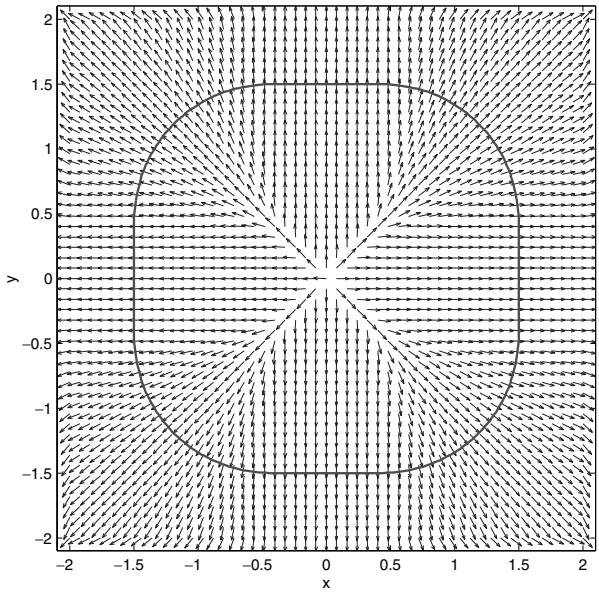


FIG. 7. Expanding square: Final interface location of the level set solution.

and the curvature,

$$\kappa = \nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right). \tag{4}$$

The spatial derivatives in Eqs. (3) and (4) can be calculated using standard second-order accurate central differencing when the denominators are nonzero. Otherwise, one-sided differencing is used.

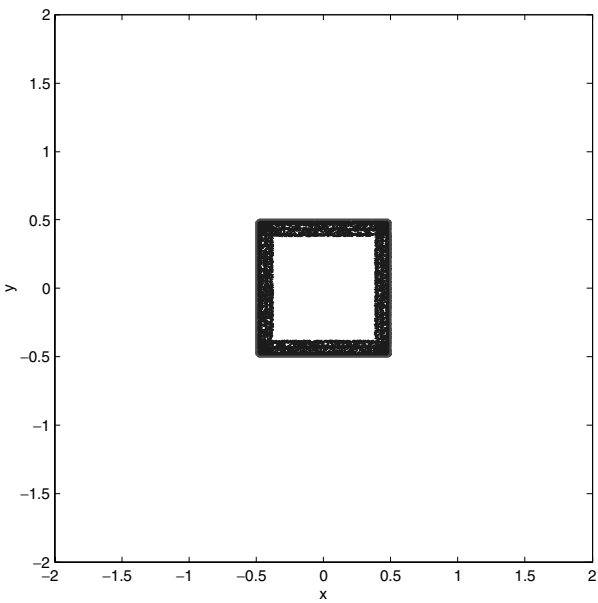


FIG. 8. Expanding square: Passively advected particles are initially seeded inside the interface.

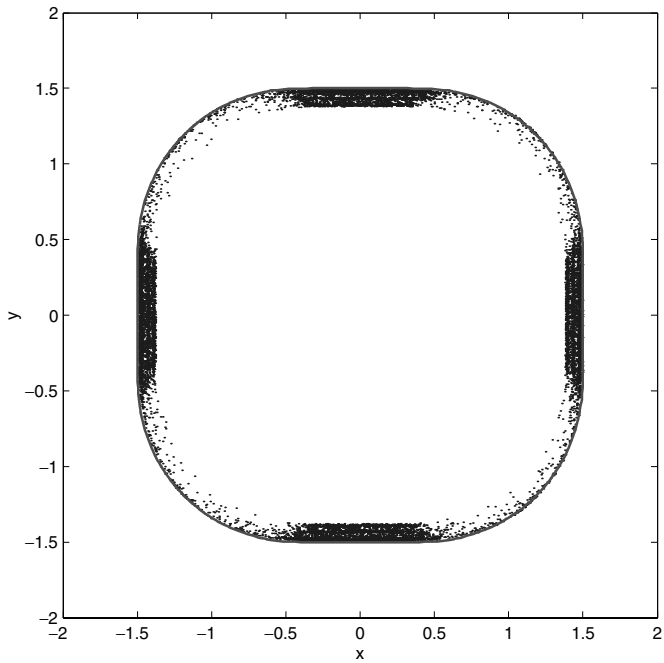


FIG. 9. Expanding square: Final location the passively advected marker particles.

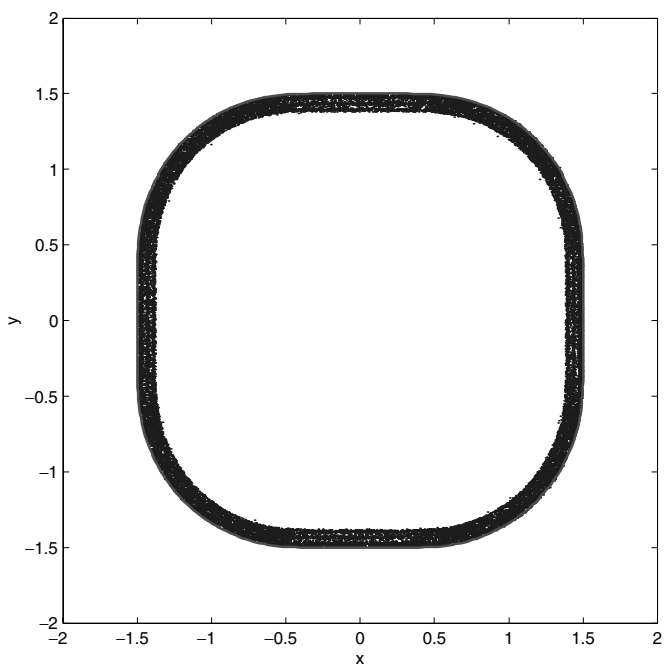


FIG. 10. Expanding square: Location of interior particles after one application of the reseeding algorithm.

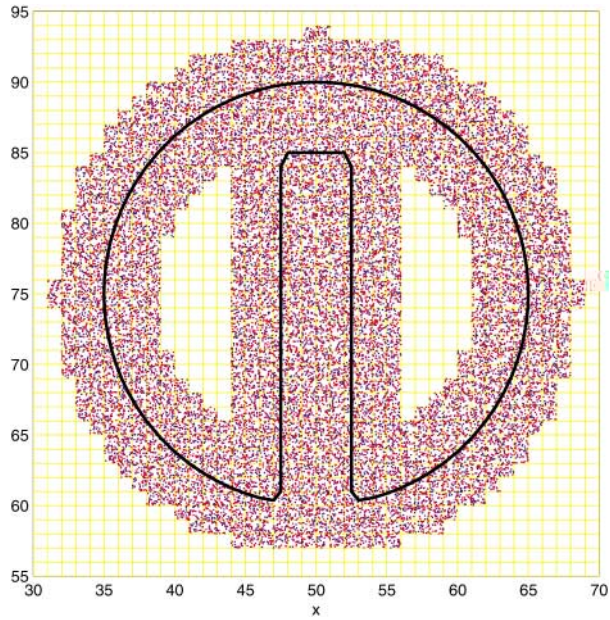


FIG. 11. Initial placement of particles on both sides of the interface.

2.2. Massless Marker Particles

Two sets of massless marker particles are placed near the interface with one set, the *positive* particles, in the $\phi > 0$ region and the other set, the *negative* particles, in the $\phi \leq 0$ region. It is unnecessary to place particles far from the interface since the sign of the level set function easily identifies these regions. This greatly reduces the number of particles needed

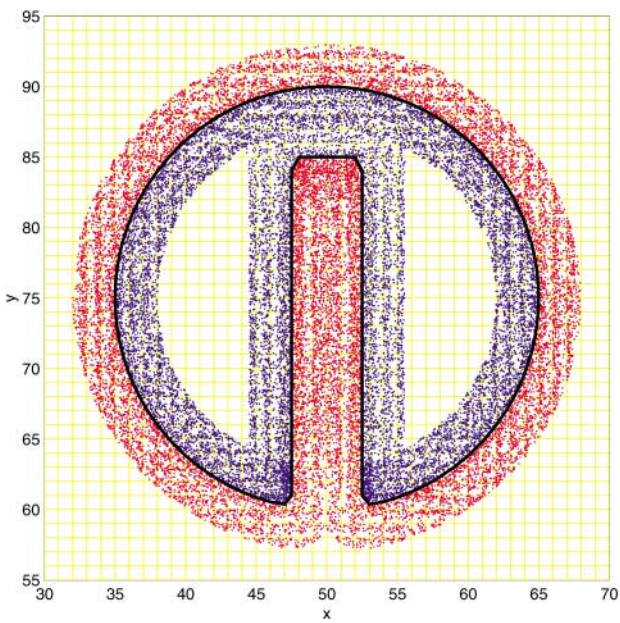


FIG. 12. Particle positions after the initial attraction step.

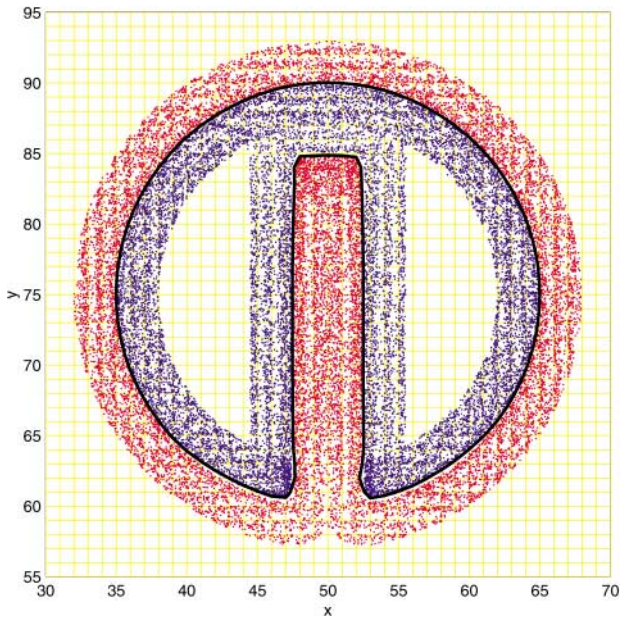


FIG. 13. Particle level set solution after one revolution.

in a given simulation. Traditional marker particle schemes [12, 26] place particles throughout the domain, although Amsden [2] proposed a method that only requires particles near the free surface. More recently, Chen *et al.* [7] introduced the surface marker method which also uses only surface particles.

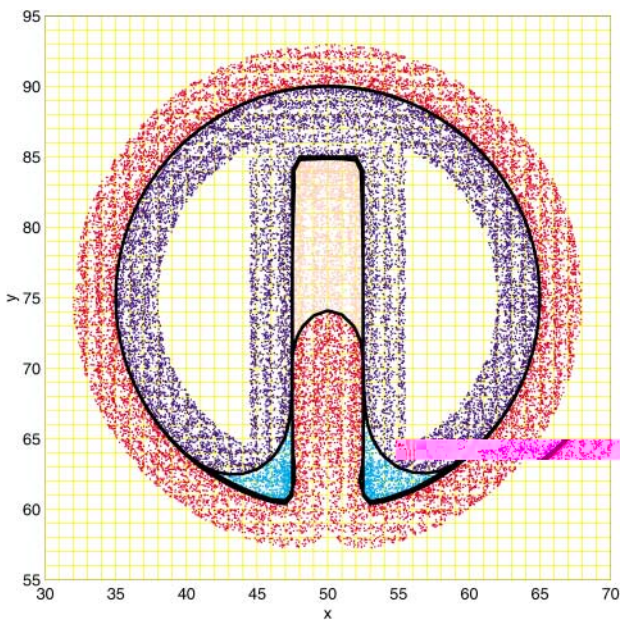


FIG. 14. Comparison of the level set solution, particle level set solution and theory after one revolution. The light red particles and light blue particles have escaped from the level set solution.

The particles are advected with the evolution equation

$$\frac{d\vec{x}_p}{dt} = \vec{u}(\vec{x}_p), \tag{5}$$

where $\vec{x}_p$ is the position of the particle and $\vec{u}(\vec{x}_p)$ is its velocity. The particle velocities are trilinearly interpolated from the velocities on the underlying grid. This trilinear interpolation limits the particle evolution to second-order accuracy. While it is not difficult to implement higher order accurate interpolation schemes (with the appropriate limiters), we have found trilinear interpolation to be sufficient and prefer it for efficiency considerations (i.e., it is fast). We use the grid velocity at time n, although the velocity at time $n + 1$ could be used as well. A third-order accurate TVD Runge–Kutta method—see [16, 35]—is used to evolve the particle positions forward in time.

The particles are used to both track characteristic information and reconstruct the interface in regions where the level set method has failed to accurately preserve mass. For the purpose of interface reconstruction, we allow our particles to overlap as illustrated of the bottom of Fig. 5. This allows us to reconstruct the interface exactly in the limit as the number of particles approaches infinity. Nonoverlapping spheres will not reconstruct the interface as the number of particles approaches infinity. For example, the top of Fig. 5 shows how three equally sized nonoverlapping circles leave large spaces when used to represent a straight line interface, while the bottom of the figure shows that six equally sized overlapping circles more readily resolve the line. Since the particles do not represent finite amounts of mass, allowing them to overlap does not create any inconsistencies in the method. The particles are used to track characteristic information, and allowing overlap merely means that some of the characteristic information is represented in duplicate by more than one particle. Since the particles are allowed to overlap at the end of the time step as well, carrying around duplicate characteristic information does not hinder our scheme in any way.

For the purpose of interface reconstruction, a sphere of radius r_p is centered at each particle location, $\vec{x}_p$. The radius of each particle is bounded by minimum and maximum values based upon the grid spacing. Maximum and minimum radii which appear to work well are

$$r_{min} = 0.1 \min(\Delta x, \Delta y, \Delta z) \tag{6}$$

$$r_{max} = 0.5 \min(\Delta x, \Delta y, \Delta z). \tag{7}$$

This allows multiscale particle resolution of the interface. This particular choice of bounds on the particle radii (from 10 to 50% of a grid cell) was the first we tried, and further experimentation might give improved results. However, as shown in the examples section, these bounds already give surprisingly good results.

We use an array-based implementation to store the particle information, since connectivity information is not required. This simplifies the integration of the particle evolution equation (5). Depending on the frequency of particle insertions and deletions, certain performance optimizations of the particle array storage can be made. For example, one can use an oversized particle array which supports additional insertions without resizing as well as a list of the indices of valid particles so that constant repacking of the array can be avoided.

2.3. Particle Level Set Method

2.3.1. *Initialization of Particles*

Initially particles of both signs are placed in cells that have at least one corner within $3\max(\Delta x, \Delta y, \Delta z)$ of the interface; i.e., for a given cell, we check whether $|\phi| < 3\max(\Delta x, \Delta y, \Delta z)$ at any of the eight corners. The number of particles of each type (positive or negative) per cell is set to a default of 64 particles (16 in 2D or 4 in 1D, i.e., four particles per spatial dimension), although this number is user definable. While there are a number of particle placement strategies that could be used to accurately sample the cell—e.g., see [10] for a discussion of "jitter"—we simply randomly position each particle in the cell. This basic technique worked surprisingly well as can be seen in the examples section. However, more sophisticated sampling techniques might improve the numerical results. An example of this initial particle seeding is shown in Fig. 11.

After the initial seeding, the particles are attracted to the correct side of the interface (i.e., positive particles to the $\phi > 0$ side and negative particles to the $\phi \leq 0$ side) into a band between a distance of $b_{min} = r_{min}$ (the minimum particle radius) and $b_{max} = 3\max(\Delta x, \Delta y, \Delta z)$ of the interface. The original uniform seeding of the particles generates a random distribution of particles in the direction tangent to the interface. To obtain a random distribution of the particles in the direction normal to the interface, we chose an isocontour $\phi_{goal} \in (\pm b_{min}, \pm b_{max})$ using a uniform random distribution. Then we carried out an attraction step with the aim of placing the particle on the $\phi = \phi_{goal}$ level set contour.

The particles are attracted to the appropriate isocontours taking advantage of geometrical information contained within the level set function ϕ. Near the interface, the normal vectors give the direction to the nearest point on the interface. To attract a particle at $\vec{x}_p$ with a current interpolated level set value of $\phi(\vec{x}_p)$ to a different $\phi = \phi_{goal}$ level set contour along the shortest possible path, one calculates

$$\vec{x}_{new} = \vec{x}_p + \lambda(\phi_{goal} - \phi(\vec{x}_p))\vec{N}(\vec{x}_p), \tag{8}$$

with $\lambda = 1$. For underresolved regions or regions where the quality of the geometric information contained within the level set function has been degraded, Eq. (8) may not put the particle on the desired contour or even in the appropriate band. To overcome these difficulties, several iterations of the above scheme may be needed. If Eq. (8) places a particle outside the computational domain, λ is successively halved until the particle stays within the domain. Then, if Eq. (8) (with this new λ) puts the particle in the appropriate band, i.e., $(\pm b_{min}, \pm b_{max})$, we accept the new particle position even though it may not be on the $\phi = \phi_{goal}$ contour. Otherwise, we halve λ once more, determine the new particle position, and repeat the process with λ again initially set to 1 and $\vec{x}_p$ set to this newly calculated position. Using $\lambda/2$ should move the particle closer to the interface where the geometric information should be more accurate. If, after a set maximum number of iterations (e.g., we use 15), the particle is still not within the desired band, it is deleted. Figure 12 shows Zalesak's disk after particle attraction step has been completed.

Finally, each particle radius is set according to

$$r_p = \begin{cases} r_{max} & \text{if } s_p\phi(\vec{x}_p) > r_{max} \\ s_p\phi(\vec{x}_p) & \text{if } r_{min} \leq s_p\phi(\vec{x}_p) \leq r_{max} \\ r_{min} & \text{if } s_p\phi(\vec{x}_p) < r_{min}, \end{cases} \tag{9}$$

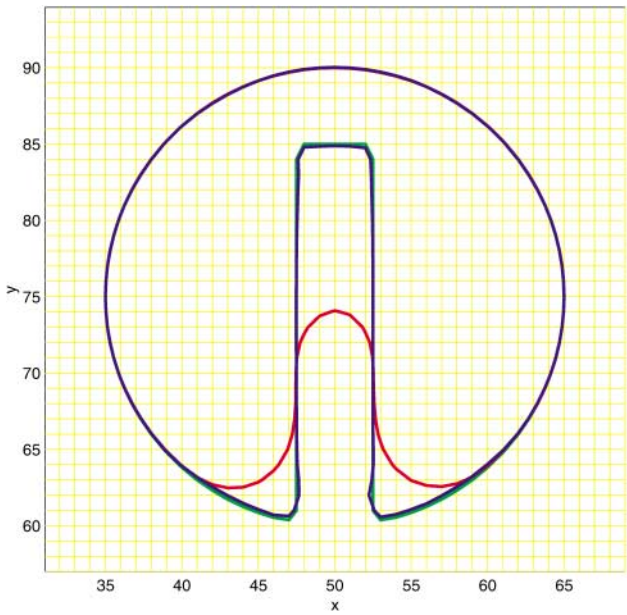


FIG. 15. Comparison of the level set solution (red), particle level set solution (blue), and theory (green) after one revolution.

where s_p is the sign of the particle ($+1$ for positive particles and -1 for negative particles). This equation adjusts the particle size such that the boundary of the particle is tangent to the interface whenever possible, i.e., while adhering to the possibly stricter restriction that the always positive particle radius is bounded by r_{min} and r_{max}.

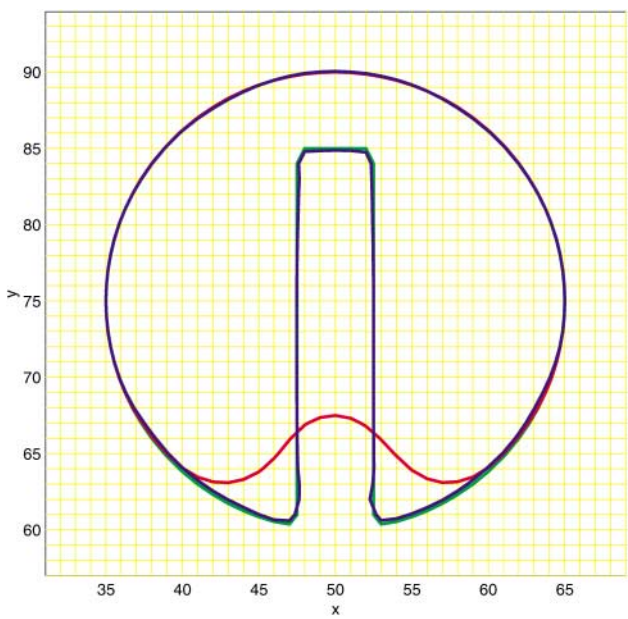


FIG. 16. Comparison of the level set solution (red), particle level set solution (blue), and theory (green) after two revolutions.

2.3.2. *Time Integration*

The marker particles and the level set function are separately integrated forward in time with a third-order accurate TVD Runge–Kutta method; see [16, 35]. Separate temporal integration allows for the possibility of using a different ordinary differential equation solver for the particle evolution.

2.3.3. *Error Correction of the Level Set Function*

Identification of error. After each complete Runge–Kutta cycle, the particles are used to locate possible errors in the level set function due to the nonphysical deletion of (incorrectly perceived) merging characteristics. Particles that are on the wrong side of the interface by more than their radius, as determined by the locally interpolated $\phi(\vec{x}_p)$, are considered to have escaped. These escaped particles indicate that characteristics have probably been incorrectly merged through regularization; i.e., the level set method has computed a weak solution. Since these weak solutions are only first-order accurate, one can typically use a relatively low-order accurate method for both the particle evolution and the particle correction algorithms, i.e., relatively lower order accurate as compared to the fifth-order accurate WENO method used for the evolution of the level set function. Using any information from the particles greatly improves the quality of the computed results, since the particles provide us with characteristic information that was discarded entirely by the level set function. Moreover, lower order accurate methods are relatively more efficient in decreasing the computational overhead involved in augmenting the level set calculations with the proposed particle methods.

In smooth well-resolved regions of the flow where the level set method is highly accurate, the particles do not drift a nontrivial distance across the interface allowing us to maintain a high-order accurate level set solution. That is, a particle is not defined as escaped when some portion of its sphere crosses the interface. Otherwise, escaped particles would appear all the time due to various types of numerical errors (even roundoff error). If we allow infinitesimal errors to force particle repairing of the level set function, we would need high-order accuracy for the particle evolution and correction methods. Instead, we define a particle as escaped only when it crosses the interface by more than its radius. Since the particle radius is $O(\triangle x)$, error identification only occurs when the particle solution and the level set solution differ to first-order accuracy. With at least second-order accurate evolution methods for the particles and the level set function, one would not expect to see any error identification in well-resolved regions of the flow. However, in underresolved regions of the flow, where the level set method generates a first-order accurate weak solution, the second-order accurate particle evolution will identify errors in the level set representation of the interface that need to be repaired. While one can change the escape condition for the particles to be any (nonzero) multiple of the particle's radius, we have found that the current choice produces good numerical results. So far, we have not experimented with other choices, and conceivably they might lead to even better numerical results.

Quantification of error. The spheres associated with each particle can be thought of as locally defined level set functions. We represent the sphere centered at each particle using a level set function

$$\phi_p(\vec{x}) = s_p(r_p - |\vec{x} - \vec{x}_p|), \tag{10}$$

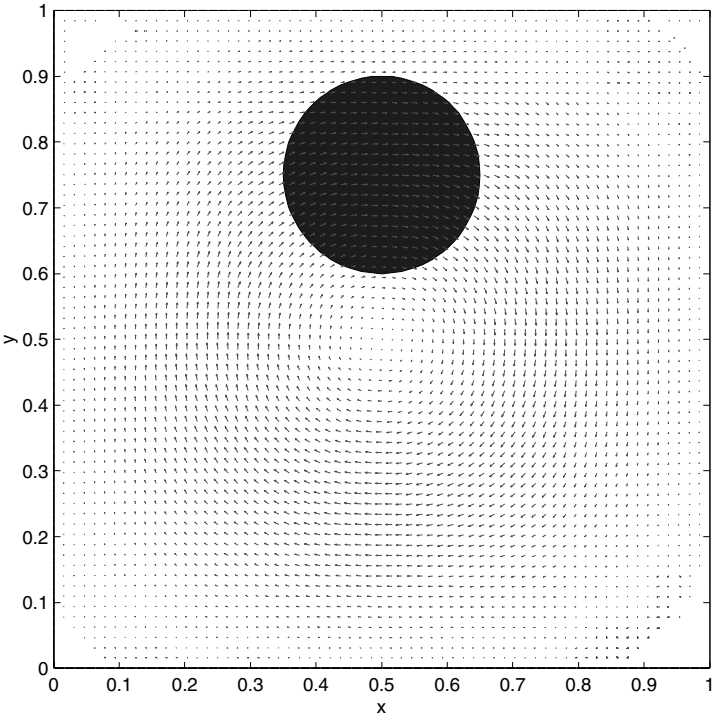


FIG. 17. Initial data and velocity field for the vortex flow.

where s_p is the sign of the particle, i.e., ± 1. The zero level set of ϕ_p corresponds to the boundary of the particle sphere. These level sets are only defined locally on the eight corners of the cell containing the particle. The local values of ϕ_p are the particle predictions of the values of the level set function on the corners of the cell. Any variation of ϕ from ϕ_p indicates possible errors in the level set solution.

Reduction of error. We use the escaped positive particles to rebuild the $\phi > 0$ region and the escaped negative particles to rebuild the $\phi \leq 0$ region. For example, take the $\phi > 0$ region and an escaped positive particle. Using Eq. (10), the ϕ_p values of the eight grid points on the boundary of the cell containing the particle are calculated. Each ϕ_p is compared to the local value of ϕ and the maximum of these two values is taken as ϕ^+. This is done for all escaped positive particles creating a reduced error representation of the $\phi > 0$ region. That is, given a level set ϕ and a set of escaped positive particles E^+, we initialize ϕ^+ with ϕ and then calculate

$$\phi^+ = \max_{\forall p \in E^+} (\phi_p, \phi^+). \tag{11}$$

Similarly, to calculate a reduced error representation of the $\phi \leq 0$ region, we initialize ϕ^- with ϕ and then calculate

$$\phi^- = \min_{\forall p \in E^-} (\phi_p, \phi^-). \tag{12}$$

ϕ^+ and ϕ^- will not agree due to the errors in both the particle and level set methods as well as interpolation errors, etc. We merge ϕ^+ and ϕ^- back into a single level set by setting ϕ

equal to the value of ϕ^+ or ϕ^-, which is least in magnitude at each grid point,

$$\phi = \begin{cases} \phi^+ & \text{if } |\phi^+| \le |\phi^-| \\ \phi^- & \text{if } |\phi^+| > |\phi^-|. \end{cases} \tag{13}$$

The minimum magnitude is used to reconstruct the interface (instead of, for example, taking an average), since it gives priority to values that are closer to the interface.

Note that if the particle locations (from the particle evolution equation) are calculated to second-order accuracy and the $O(\triangle x)$ escape condition is used, this error reduction step only occurs in regions of the flow where the level set method has computed a weak solution and deleted characteristics entirely. Therefore, the accuracy requirements of the error reduction step can be rather low while still producing markedly improved numerical results as shown in the examples section. While higher order accurate error reduction methods might produce better results, there is a trade-off in loss of efficiency especially when dealing with a large number of particles.

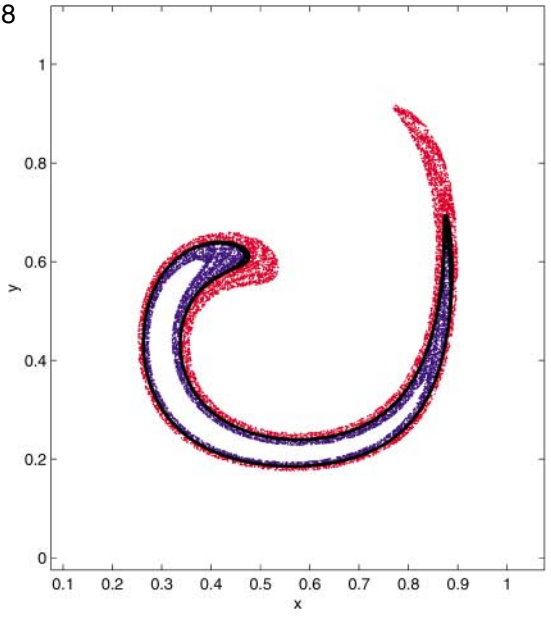
We also note that there are a number of standard techniques that can be applied to further increase efficiency, but most of these are merely implementation details that cloud the purpose of this paper. As an example, one might worry that every particle needs to have a costly square root evaluated at each of the eight corners of the cell containing it. However, we only need to do this for escaped particles in underresolved regions of the flow greatly reducing the number of particles that need to be considered. Furthermore, each grid node is only affected by the closest positive particle and the closest negative particle. Thus, the particles can be sorted based on the square of their distance, and after the closest particles are identified, only two square roots per grid node are needed. That is, the number of square roots needed is equal to two times the number of grid points in underresolved regions of the flow and is independent of the number of particles or escaped particles used in the calculation.

2.3.4. *Reinitialization*

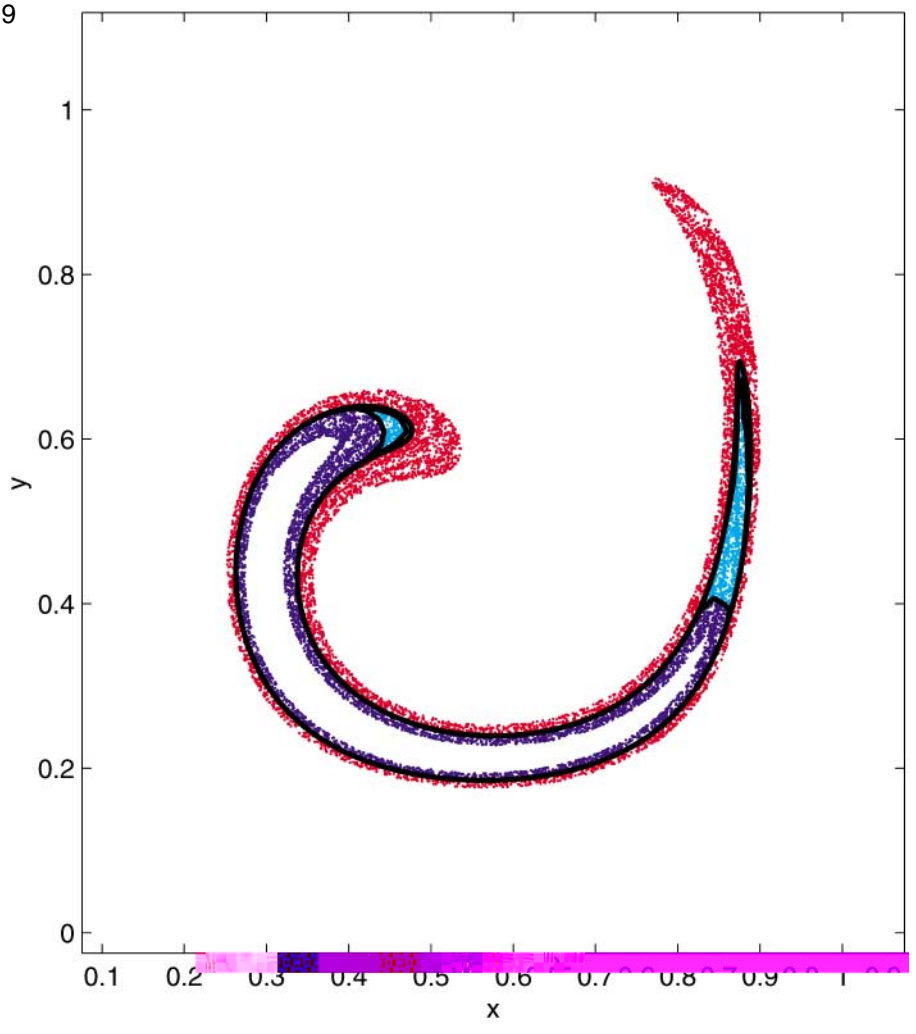
Since the particle level set method relies on ϕ being an approximate signed distance function, we reinitialize the level set function using Eq. (2) after each combination of a Runge–Kutta cycle and an error correction step. Unfortunately, reinitialization may cause the zero level set to move, which is not desirable, so we use the particle level set method to correct these errors as well. During the reinitialization step, we *do not* want the particles to follow the characteristics present in Eq. (2) where particles flow away from the interface. Instead, we keep the particles stationary corresponding to the desired velocity of the zero level set. We then use the particles to identify and correct any errors produced by the reinitialization scheme.

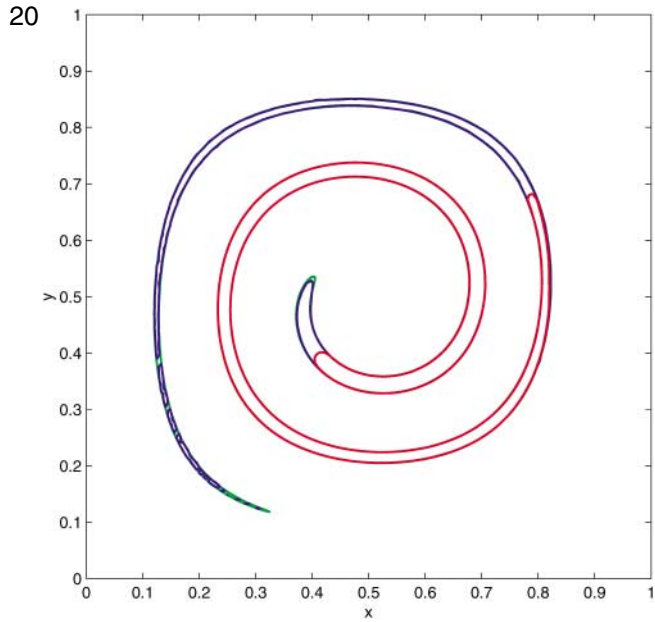
After reinitialization of the level set function, including the identification and reduction of errors using particles, we adjust the radii of the particles according to the current $\phi(\vec{x}_p)$ value as described by Eq. (9). This radius adjustment allows a larger difference between the particle and level set solutions before one assumes that characteristics have been incorrectly deleted. In smooth regions, particles can slowly shrink to the minimum allowable radius before crossing over the interface indicating errors. However, in underresolved regions, particles will jump (possibly relatively far) across the interface in a single time step as the level set method computes a weak solution deleting a large region of characteristic information.

18


1
0.8
0.6
0.4
0.2
0
y
0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
x

19


1
0.8
0.6
0.4
0.2
0
y
0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
x



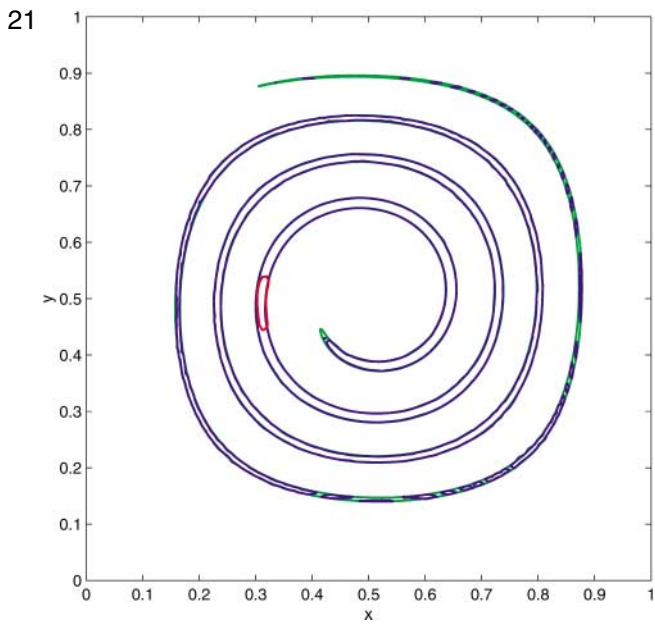


FIG. 18. Comparison of the particle level set solution and a high-resolution front-tracked solution for the vortex flow at $t = 1$.

FIG. 19. Comparison of the level set solution, particle level set solution and a high-resolution front-tracked solution for the vortex flow at $t = 1$. The light blue particles have escaped from the level set solution.

FIG. 20. The level set solution (red), particle level set solution (blue), and a high-resolution front-tracked solution (green) for the vortex flow at $t = 3$.

FIG. 21. The level set solution (red), particle level set solution (blue), and a high-resolution front-tracked solution (green) for the vortex flow at $t = 5$.

In summary, the order of operations is evolve both the particles and the level set function forward in time, correct errors in the level set function using particles, apply reinitialization, again correct errors in the level set function using particles, and finally adjust the particle radii.

2.3.5. *Particle Reseeding*

In flows with interface stretching and tearing, regions which lack a sufficient number of particles will form. This problem has also been observed in particle only methods which seed particles everywhere in the computational domain [11]. To accurately resolve the interface for all time, we will need to periodically readapt the particle distribution to the deformed interface. This idea of adding and deleting particles has been addressed by many authors; see, for example, [18]. We not only add and delete particles in cells near the interface, but also delete particles that have drifted too far from the interface to provide any useful information, e.g., positive particles with $\phi(\vec{x}_p) > b_{max}$ and negative particles with $\phi(\vec{x}_p) < -b_{max}$. The reseeding algorithm should not alter the position of the particles near the interface as they are accurately tracking the evolution of the interface and can provide useful information should they escape in the future. In addition, escaped particles should not be deleted as they indicate that characteristic information too small to be represented with the current grid has been deleted by the level set method. Even if escaped particles are not currently contributing to the level set function because there are not enough of them in a given region, they may agglomerate and contribute in the future. Moreover, recent work [5] has shown that underresolved information can be adapted into a two-phase mixture model until it reaches a critical mass where it can be reabsorbed and properly represented by the interface-tracking scheme.

Reseeding is carried out by first identifying all the nonescaped particles in each cell. Then the local value of the level set function is used to decide if a given cell is near the interface, e.g., within three grid cells. If a cell is not near the interface, all the nonescaped particles are deleted. Otherwise, if a cell near the interface currently has less particles than the previously defined maximum (e.g., 64 in 3D), particles are added to the cell and attracted to the interface.

If there are too many nonescaped particles in a given cell, we create a heap data structure which holds the desired number of particles. Each nonescaped particle in the cell is inserted into the heap based upon the difference between the locally interpolated $\phi(\vec{x}_p)$ value and its radius, i.e., $s_p\phi(\vec{x}_p) - r_p$, since we want to keep the particles that are closest to the interface. The heap is a computationally and memory efficient way to store a priority queue. The particle with the largest $s_p\phi(\vec{x}_p) - r_p$ is placed on top. Once the heap is full and properly sorted, we consider the remaining particles one at a time. For each remaining particle, its $s_p\phi(\vec{x}_p) - r_p$ value is compared with the corresponding value of the particle on top of the heap. If the current particle under consideration has a smaller value than the particle on top of the heap, we delete the particle on top of the heap and replace it with the current particle. A down heap sort is then performed placing the next candidate for removal on top of the heap. On the other hand, if the current particle's $s_p\phi(\vec{x}_p) - r_p$ value is larger than that of the particle on top of the heap, we simply delete it.

The reseeding operation is problem dependent. Viable reseeding strategies include reseeding at fixed time intervals or according to some measure of interface stretching/compression, e.g., arc length in 2D or surface area in 3D. When reseeding based upon a change in surface

area, level set methods allow easy estimation of this quantity. The surface area of the interface is given by

$$\int \delta(\phi)|\nabla\phi|\, d\vec{x},$$

where $\delta(\phi)$ is a numerically smeared out delta function, which to first-order accuracy can be approximated as

$$\delta(\phi) = \begin{cases} 0 & \phi < -\epsilon \\ \dfrac{1}{2\epsilon} + \dfrac{1}{2\epsilon}\cos\left(\dfrac{\pi\phi}{\epsilon}\right) & -\epsilon \le \phi \le \epsilon, \\ 0 & \epsilon < \phi, \end{cases}$$

where $\epsilon = 1.5\Delta x$ is the bandwidth of the numerical smearing.

We note that excessive reseeding to calculate underresolved regions is not recommended since the inserted particles have to be attracted to the interface using the current geometry defined by the level set function. If the interface geometry is poorly resolved, reseeding may not improve the resolution at the interface. In fact, it may be damaged.

To demonstrate the need and feasibility of the reseeding algorithm, we consider the converse to the problem addressed in Figs. 1 and 2. Here we use the velocity field $\vec{u} = -\vec{N}$ as opposed to the $\vec{u} = +\vec{N}$ velocity field used in those figures. This velocity field is shown in Fig. 6 along with the level set initial data. In this example, the level set function experiences a rarefaction at the corners as a single point expands into a quarter circle as shown in Fig. 7. Figure 8 shows an initial seeding of passively advected interior particles while Fig. 9 shows the final location of these particles. Note that they have spread out appreciably in the corners. Figure 10 shows the same result after one application of the reseeding algorithm. Note that the interface is now significantly more accurately resolved by the particles.

3. EXAMPLES

3.1. Rigid Body Rotation of Zalesak's Disk

Consider the rigid body rotation of Zalesak's disk in a constant vorticity velocity field [47]. The initial data are a slotted circle centered at (50, 75) with a radius of 15, a width of 5, and a slot length of 25. The constant vorticity velocity field is given by

$$u = (\pi/314)(50 - y),$$
$$v = (\pi/314)(x - 50),$$

so that the disk completes one revolution every 628 time units.

Figure 11 illustrates the initial seeding of particles on both sides of the interface while Fig. 12 depicts the particle locations after the attraction step has been applied to attract them to the appropriate bands on the correct side of the interface. Blue dots indicate the location of negative particles and red dots indicate the location of positive particles. A 100×100 grid cell computational mesh is shown in the figures illustrating that the slot is only five grid cells across.

Figure 13 illustrates the high quality particle level set solution obtained after one full rotation. Figure 14 illustrates the need for both positive and negative particles. Here, we plot the level set solution and the particle level set solution and theory, along with both

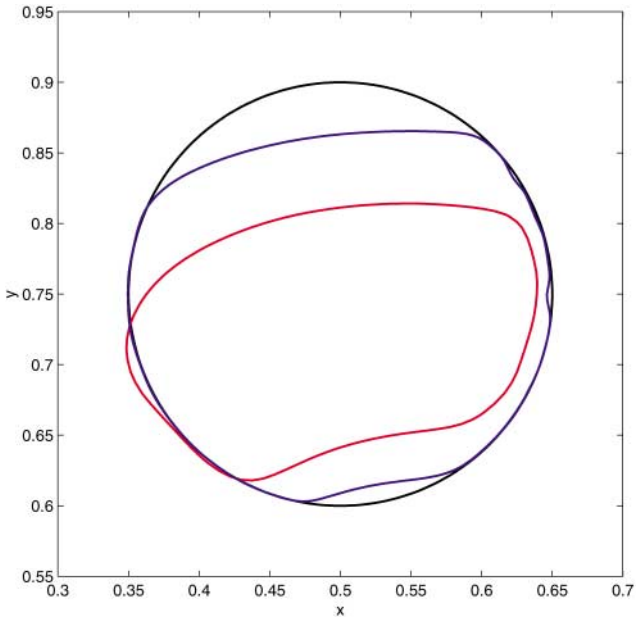


FIG. 22. Level set solution. The exact solution (black) in comparison to grids of size 256 × 256 (blue), 128 × 128 (red), and 64 × 64 (green—disappeared).

the positive and negative particles. The errors in the level set solution are emphasized by plotting escaped positive particles in light red and the escaped negative particles in light blue. This illustrates how the positive particles correct the errors at the two corners at the top of the slot while the negative particles correct the errors at the two corners near the bottom of the slot.

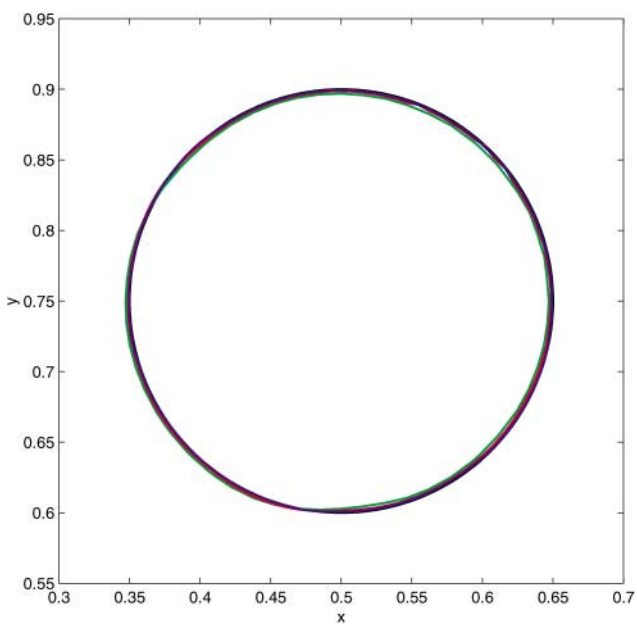


FIG. 23. Particle level set solution. The exact solution (black) in comparison to grids of size 256 × 256 (blue), 128 × 128 (red), and 64 × 64 (green).

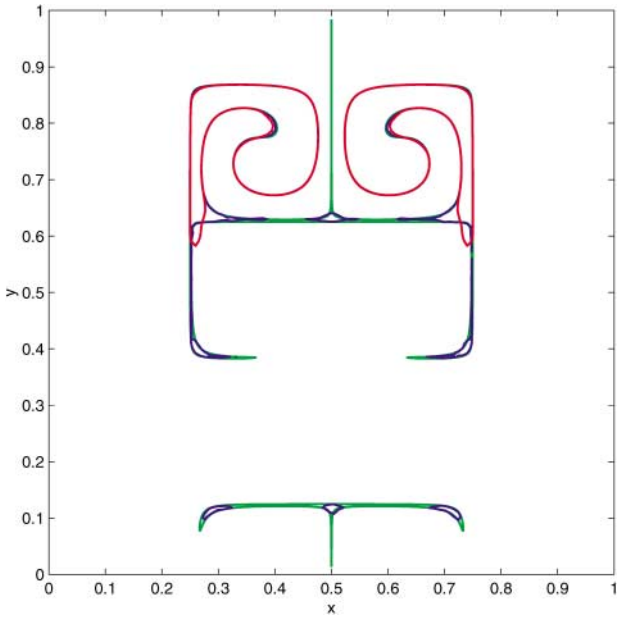


FIG. 24. Comparison of the level set solution (red), particle level set solution (blue), and a high-resolution front-tracked solution (green) for the deformation flow at $t = 1$.

Figures 15 and 16 compare the evolution of a level set only method (red) and our particle level set method (blue) after one and two revolutions, respectively. The exact solution (green) is also plotted for the sake of comparison. As expected, the level set only method applies an excessive amount of regularization in the sharp corners.

Tables I and II compare the area loss (or gain) of the level set method and the particle level set method on three different grids. The area is calculated using a second-order accurate unbiased level set contouring algorithm [4]. In addition, we calculate the accuracy of the interface location using the first-order accurate error measure introduced in [39],

$$\frac{1}{L}\int |H(\phi_{expected}) - H(\phi_{computed})|\, dx\, dy, \tag{14}$$

where L is the length of the expected interface. This integral is numerically calculated as in [39]:

TABLE I
Zalesak's Disk: Level Set Method

	Grid cells	Area	% Area loss	L_1 error	Order
One revolution	exact	582.2	—	—	—
	50	0	100%	4.03	N/A
	100	613.0	−5.3%	0.61	2.7
	200	579.1	0.54%	0.08	2.9
Two revolutions	50	0	100%	4.03	N/A
	100	634.7	−9.0%	0.89	2.2
	200	577.4	0.82%	0.11	3.0

TABLE II
Zalesak's Disk: Particle Level Set Method

	Grid cells	Area	% Area loss	L_1 error	Order
One revolution	exact	582.2	—	—-	—
	50	495.7	14.9%	0.59	N/A
	100	580.4	0.31%	0.07	3.1
	200	581.0	0.20%	0.02	1.5
Two revolutions	50	487.6	16.2%	0.62	N/A
	100	578.0	0.72%	0.09	2.8
	200	580.0	0.38%	0.03	1.4

- partition the domain into many tiny pieces (1000×1000),
- interpolate $\phi_{computed}$ onto the newly partitioned domain and calculate $\phi_{expected}$ for the domain,
- numerically integrate Eq. (14) where $H(\phi)$ is the indicator function for $\phi \leq 0$; i.e., $H(\phi) = 1$ if $\phi \leq 0$ and $H(\phi) = 0$ otherwise.

On the coarsest grid (50×50 grid cells), the level set only solution vanishes before one revolution is completed while the particle level set method still maintains 83.8% of the area even after two rotations.

3.2. Single Vortex

While Zalesak's disk is a good indicator of diffusion errors in an interface-capturing method, it does not test the ability of an Eulerian scheme to accurately resolve thin filaments on the scale of the mesh which can occur in stretching and tearing flows. A flow which exhibits interface stretching is the "vortex-in-a-box" problem introduced by Bell *et al.* [3]. Figure 17 shows the nonconstant vorticity velocity field centered in the box with the largest velocity located half way to the walls of the domain. The velocity field is defined by the stream function

$$\Psi = \frac{1}{\pi} \sin^2(\pi x) \sin^2(\pi y).$$

A unit computational domain is used with a circle of radius 0.15 placed at (0.5, 0.75). The resulting velocity field stretches out the circle into a very long, thin fluid element which progressively wraps itself toward the center of the box.

Figure 18 illustrates the behavior of the particles in the vortex flow field at $t = 1$ using a 64×64 grid cell domain. The positive particles are shown in red while the negative particles are shown in blue. Even at this relatively early time, there was a substantial amount of stretching of the interface, and the particle bands which were initially three grid cells deep were stretched out and compressed. Both the particle level set solution and a high-resolution front-tracked solution are plotted in the figure, although it is difficult to ascertain which is which as they are almost directly on top of each other. The role of the particles in helping to maintain the interface can be seen in Fig. 19, which depicts the level set solution along with light blue negative particles that have escaped from the level set solution. Note that these escaped particles exist at both the head and the tail, where the curvature is large.

TABLE III
One Period of Vortex Flow: Level Set Method

Grid cells	Area	% Area loss	L_1 error	Order
exact	0.0707	—	—	—
64	0	100.0%	0.075	N/A
128	0.0425	39.8%	0.031	1.3
256	0.0634	10.3%	0.008	2.0

The ability of the particle level set method to maintain thin, elongated filaments is shown in both Fig. 20 at $t = 3$ and Fig. 21 at $t = 5$. Both figures were computed using 128×128 grid cells. The interface depicted in Fig. 20 is at the same position as Fig. 13a in [28] and Figs. 4a–4d in [27] for the sake of comparison. Both figures show the level set solution (red), the particle level set solution (blue), and the high-resolution front-tracked solution (green). The particle level set method clearly outperforms the level set method. Only near the head and the tail of the interface where it is about one grid cell wide does the particle level set method fail to compete with the high-resolution front-tracked solution. Also note that while the particle level set method does not exactly conserve area, unlike VOF methods, it exhibits less "blobby" structure than VOF methods; see Fig. 4d in [27]. In underresolved regions, the particles will not be close enough together to accurately represent the interface, and thin filament structures will break apart. However, the particles still track the interface motion with second-order accuracy, and thus the resulting pieces are in accurate locations. In contrast, the interface reconstruction procedure used in VOF methods forces mass in neighboring cells to be artificially attracted to each other; i.e., mass in underresolved regions is inaccurately moved around during the interface reconstruction step resulting in larger blobs with first-order accurate errors in their location.

For the purposes of error analysis, the velocity field is time reversed by multiplying by $\cos(\pi t/T)$, where T is the time at which the flow returns to its initial state; see LeVeque [19]. The reversal period used in the error analysis of the vortex problem is $T = 8$ producing a maximal stretching similar to the interface in Fig. 20. As can be seen from the error Tables III and IV as well as Figs. 22 and 23, the ability of the particle level set method to model interfaces undergoing substantial stretching is quite good. The L_1 errors reported here compare favorably with those reported by Rider and Kothe in [28] using a VOF PLIC method.

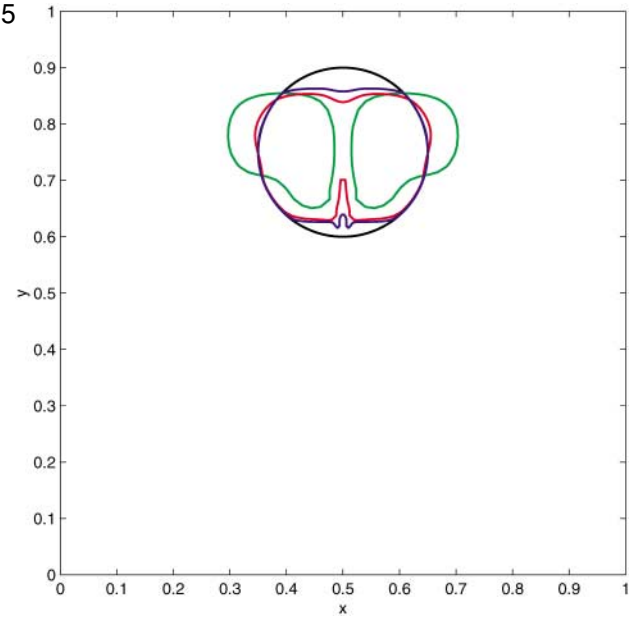
3.3. Deformation Field

An even more difficult test case is the entrainment of a circular body in a deformation field defined by 16 vortices as introduced by Smolarkiewicz [36]. The periodic velocity

TABLE IV
One Period of Vortex Flow: Particle Level Set Method

Grid cells	Area	% Area loss	L_1 error	Order
exact	0.0707	—	—	—
64	0.0694	1.81%	0.003	N/A
128	0.0702	0.71%	0.001	1.1
256	0.0704	0.35%	5.09E-4	1.4

25



26

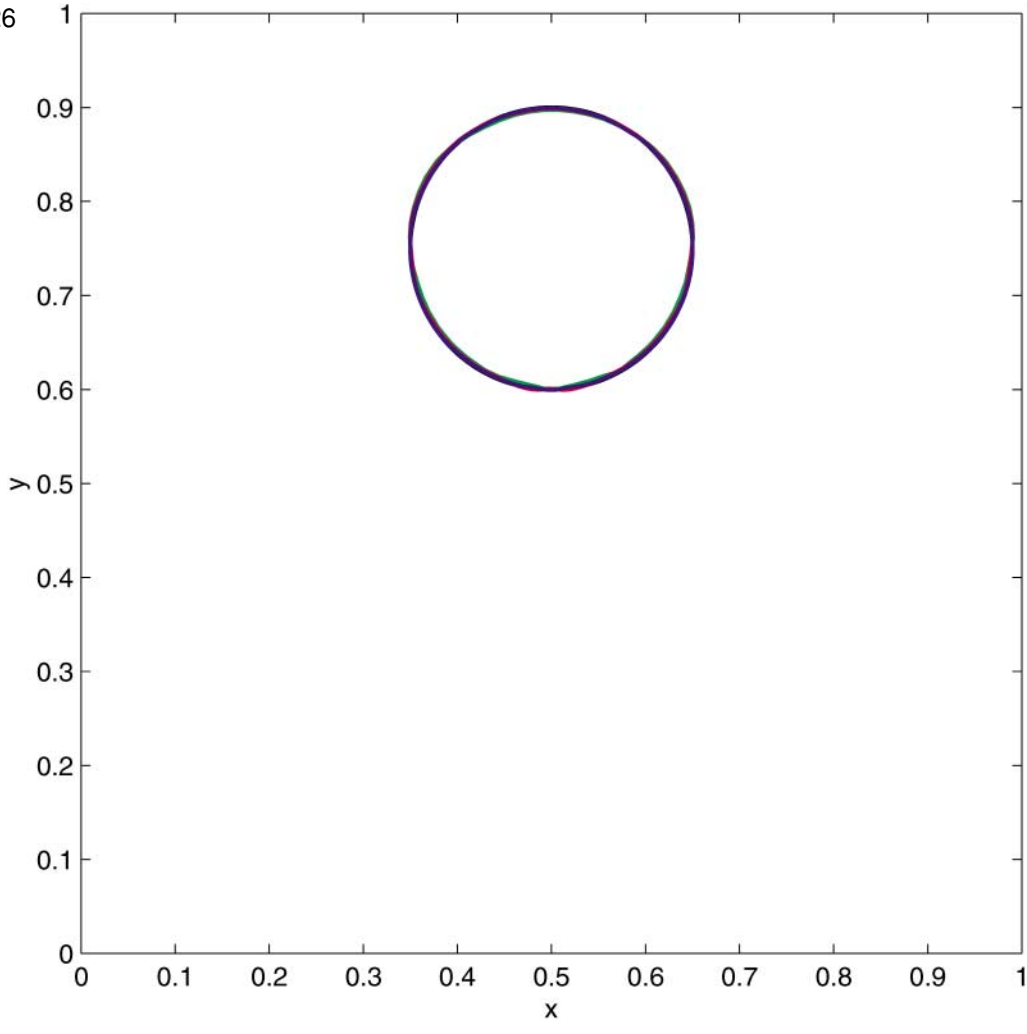


TABLE V
One Period of Deformation Flow: Level Set Method

Grid points	Area	% Area loss	L_1 error	Order
exact	0.0707	—	—	—
64	0.0569	19.5%	0.045	N/A
128	0.0585	17.2%	0.016	1.5
256	0.0622	12.0%	0.009	0.8

field is given by the stream function

$$\Psi = \frac{1}{4\pi}\sin(4\pi(x+0.5))\cos(4\pi(y+0.5)). \tag{15}$$

Periodicity is enforced and the interface crosses the top boundary of the domain to reappear on the bottom as shown in Fig. 24 at $t = 1$ using a time reversed flow field with a period of $T = 2$. The figure shows the level set solution (red), the particle level set solution (blue), and the high-resolution front-tracked solution (green) using a 128×128 grid. Note that the width of some of the filamentary regions are on the order of a grid cell and would be very difficult to resolve without the information provided by the particles. One can increase the particle resolution of the interface using several techniques. More particles can be added at the beginning of the simulation by increasing the particles' initial bandwidth or the number of particles per cell. In addition, one could periodically apply reseeding techniques, although caution should be used when reseeding the interface shown in Fig. 24 as the geometry is poorly defined in the underresolved regions.

As can be seen from the error Tables V and VI as well as Figs. 25 and 26, the ability of the particle level set method to model interfaces undergoing substantial stretching is quite good.

3.4. Rigid Body Rotation of Zalesak's Sphere

Analogous with the two-dimensional Zalesak disk problem, we use a slotted sphere to examine the diffusion properties of the particle level set method in three spatial dimensions. The sphere has radius 15 in a $100 \times 100 \times 100$ domain. The slot is five grid cells wide and 12.5 grid cells deep on a $100 \times 100 \times 100$ grid cell domain. The sphere is initially placed at (50, 75, 50) and undergoes a rigid body rotation in the $z = 50$ plane about the point (50, 50, 50). The constant vorticity velocity field is given by

$$\begin{aligned} u(x, y, z) &= (\pi/314)(50 - y), \\ v(x, y, z) &= (\pi/314)(x - 50), \\ w(x, y, z) &= 0, \end{aligned}$$

so that the sphere completes one revolution every 628 time units.

FIG. 25. Level set solution. The exact solution (black) in comparison to grids of size 256×256 (blue), 128×128 (red), and 64×64 (green).

FIG. 26. Particle level set solution. The exact solution (black) in comparison to grids of size 256×256 (blue), 128×128 (red), and 64×64 (green).

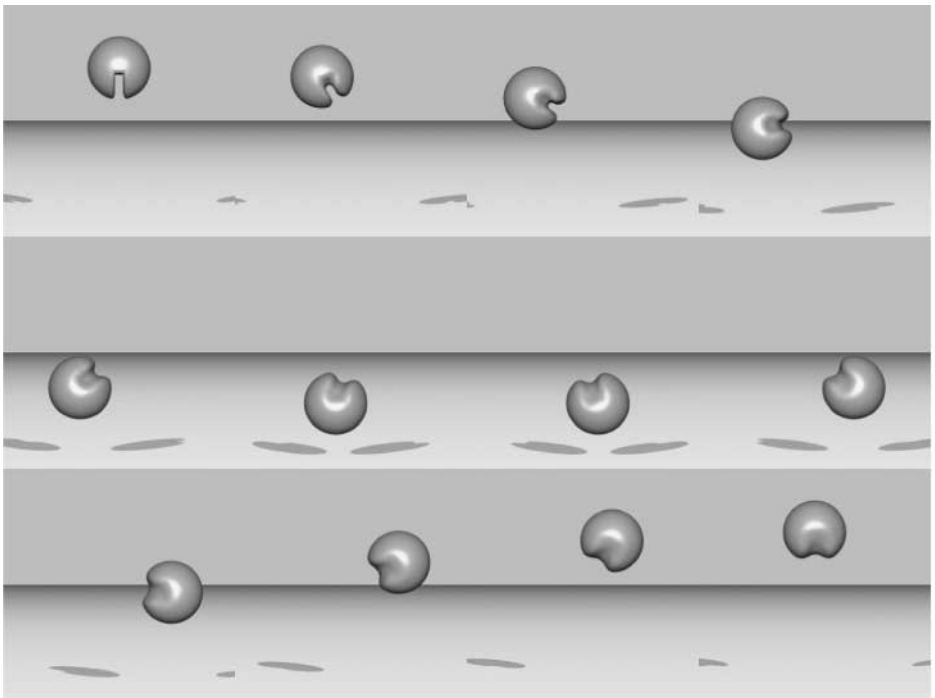


FIG. 27. Zalesak's sphere: Level set solution.

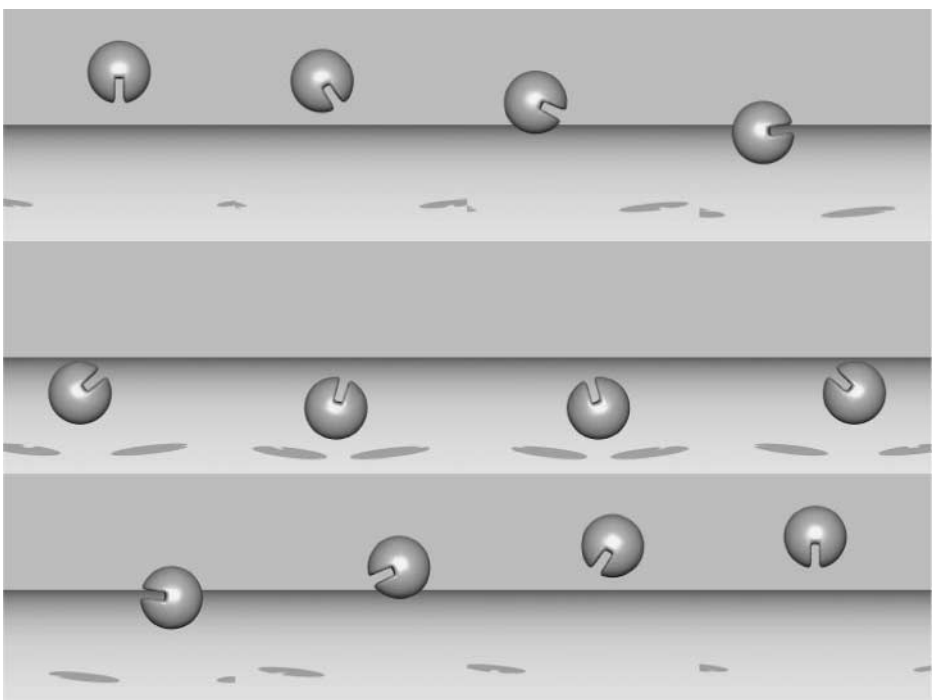


FIG. 28. Zalesak's sphere: Particle level set solution.

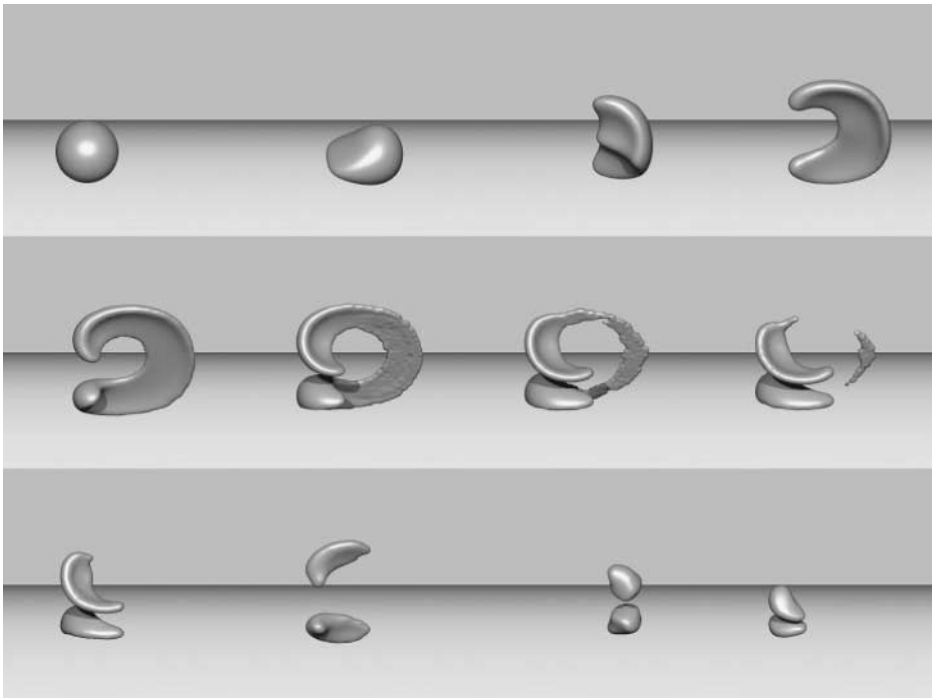


FIG. 29. Deformation test case: Level set solution.

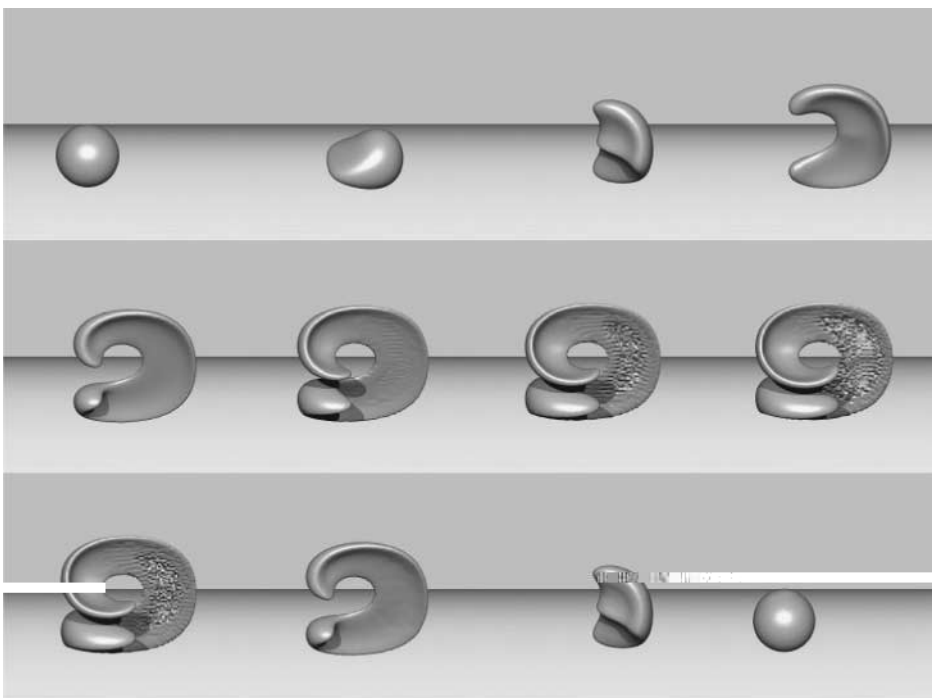


FIG. 30. Deformation test case: Particle level set solution.

TABLE VI
One Period of Deformation Flow: Particle Level Set Method

Grid points	Area	% Area loss	L_1 error	Order
exact	0.0707	—	—	—
64	0.0696	1.59%	0.002	N/A
128	0.0705	0.03%	0.001	1.1
256	0.0705	0.03%	4.4E-4	1.4

The presence of an extra dimension allows for more opportunities to examine an interface-capturing scheme for excessive amounts of regularization. Figures 27 and 28 show the level set solution and the particle level set solution, respectively, at approximately equally spaced time intervals from $t = 0$ to $t = 628$. Note that the final frame at $t = 628$ should be identical to the initial data at $t = 0$. The particle level set method is able to maintain the sharp features of the notch unlike the level set method.

To illustrate the volume preservation properties of our scheme, we estimate the volume of the interior region using a first-order accurate approximation to the integral

$$\int H(\phi)\, d\vec{x}, \tag{16}$$

where H is a numerically smeared out heaviside function given by

$$H(\phi) = \begin{cases} 0 & \phi < -\epsilon \\ \dfrac{1}{2} + \dfrac{\phi}{2\epsilon} + \dfrac{1}{2\pi} \sin\left(\dfrac{\pi\phi}{\epsilon}\right) & -\epsilon \le \phi \le \epsilon \\ 1 & \epsilon < \phi \end{cases}, \tag{17}$$

with $\epsilon = 1.5\triangle x$ the bandwidth of the numerical smearing. It is interesting to note that the level set solution loses only 2.1% of its total volume while the particle level set solution has lost 2.3%. The level set method has both inward and outward dissipation errors in convex and concave regions, respectively, leading to fortuitous cancellation and a misleading appearance of accuracy when volume preservation is considered alone. Similar statements hold for VOF calculations that preserve volume almost exactly, but locate the volume incorrectly, e.g., flotsam and jetsam.

3.5. Three-Dimensional Deformation Field

LeVeque [19] proposed a three-dimensional incompressible flow field, which combines a deformation in the x–y plane with one in the x–z plane. The velocity field is given by

$$\begin{aligned} u(x, y, z) &= 2\sin^2(\pi x)\sin(2\pi y)\sin(2\pi z), \\ v(x, y, z) &= -\sin(2\pi x)\sin^2(\pi y)\sin(2\pi z), \\ w(x, y, z) &= -\sin(2\pi x)\sin(2\pi y)\sin^2(\pi z), \end{aligned}$$

and the flow field is modulated in time with a period of $T = 3$. A sphere of radius 0.15 is placed within a unit computational domain at (0.35, 0.35, 0.35). A $100 \times 100 \times 100$ grid

FIG. 31. Pouring water into a cylindrical glass using the particle level set method.

cell domain was used. This allows the sphere to be entrained by two rotating vortices which initially scoop out opposite sides of the sphere and then compress them causing the sphere to pancake. The top and bottom edges of the pancake are then caught up again in their appropriate vortices causing the surface to become very stretched out. Parts of the interface thin out to about a grid cell and both methods have difficulty resolving this thin interface. Of course, we could use a variety of techniques to increase the initial particle density or to reseed along the way, but the particle level set method recovers rather nicely and loses very little volume even without increasing the particle count. On the other hand, the level set method fails severely for this example.

Figures 29 and 30 show the level set solution and the particle level set solution, respectively, at $t = 0$, $10\triangle s$, $20\triangle s$, $30\triangle s$, $40\triangle s$, $50\triangle s$, $60\triangle s$, $70\triangle s$, $90\triangle s$, $110\triangle s$, $130\triangle s$, and $150\triangle s$, where $\triangle s = 628/150$. Note that the final frame at $t = 150\triangle s$ should be identical to the initial data at $t = 0$. In addition, frame $130\triangle s$ should be identical to frame $20\triangle s$, frame $110\triangle s$ should be identical to frame $40\triangle s$, and frame $90\triangle s$ should be identical to frame $60\triangle s$. The maximum stretching occurs at $t = 75\triangle s$, where the level set solution has lost 49% of the initial volume and the particle level set solution has gained 1.9%. After completing a full cycle, at $t = 150\triangle s$, the level set solution has lost 80% of the initial volume while the particle level set solution has only lost 2.6%.

4. CONCLUSION

We proposed a new numerical method which merges the best aspects of Eulerian front-capturing schemes and Lagrangian front-tracking methods for improved mass conservation in a fluid flow. Level set methods are used to smoothly capture the interface but suffer an excessive amount of mass loss in underresolved regions of the flow. This prevents the resolution of thin interfacial filaments and regions of high curvature. To counteract this problem, characteristic information is provided by massless marker particles. Escaped massless marker particles are used to correct mass loss in the level set function. The method

otherwise maintains the nice geometric properties of level set methods along with the ease and simplicity of implementation. As seen in the examples, the particle level set method compares favorably with volume of fluid methods in the conservation of mass and purely Lagrangian schemes for interface resolution.

While we add particles in a band near the interface, other strategies could be adopted. For example, it is obvious that particles are only needed in regions of high curvature and this optimization would speed up the method significantly. Concerning speed, it is interesting to point out how this method compares to adaptive meshing, for example, as in [38]. A significant advantage of particle methods is that they allow one to avoid the small time step restriction dictated by the small cells produced by adaptively meshing. Also, the data structures are simpler than for cell-based adaptive meshing, and the particle method can be applied more locally than patch-based adaptive meshing.

Recently, in [8], we combined this new particle level set method with an existing fluid code in order to study its behavior in more complicated situations including pinching and merging. An example calculation from [8] showing water being poured into a glass is shown in Fig. 31. This calculation utilized a free-surface Navier–Stokes solver to model the water with the air kept at a constant pressure. A semi-Lagrangian method was used to update the convective terms, while a preconditioned conjugate gradient method was used to solve for the pressure with Dirichlet (fixed) pressure boundary conditions prescribed at the free surface. The glass was modeled as a cylinder of radius 0.25 cut out of a Cartesian computational fluid grid of $55 \times 120 \times 55$ grid cells, with a grid spacing of 0.01 in each direction. Further details concerning the flow solver used and the modeling of objects in the flow field are given in [9]. The ability to maintain the thin sheets of water formed during the topologically complex initial pouring phase as illustrated in Fig. 31 indicates the robustness of the particle level set method as an interface-capturing method for real world fluid flows. This example also illustrates that the particle level set method proposed in this paper can handle complex topological changes in a straightforward fashion as does the level set method.

ACKNOWLEDGMENTS

Ron Fedkiw thanks Nick Foster for stimulating discussions concerning the hybridization of particle and level set methods. This led to the work in [9], where particles were placed on one side of the interface only, and the local interface curvature was used to indicate which particles should be used to rebuild the level set function. While that original technique suffered from a number of technical difficulties, without it, this current work would not have been possible. Doug Enright thanks the Hughes Aircraft and Raytheon Systems companies for previous support as a Howard Hughes doctoral fellow.

REFERENCES

1. D. Adalsteinsson and J. Sethian, A fast level set method for propagating interfaces, *J. Comput. Phys.* **118**, 269 (1995).
2. A. Amsden, *Numerical Calculation of Surface Waves: A Modified ZUNI Code with Surface Particles and Partial Cells*, Technical Report LA-5146 (Los Alamos Scientific Laboratory, Los Alamos, NM, 1973).
3. J. Bell, P. Colella, and H. Glaz, A second-order projection method for the incompressible Navier–Stokes equations, *J. Comput. Phys.* **85**, 257 (1989).
4. R. Caiden, R. Fedkiw, and C. Anderson, A numerical method for two-phase flow consisting of separate compressible and incompressible regions, *J. Comput. Phys.* **166**, 1 (2001).

5. G. Cerne, S. Petelin, and I. Tiselj, Coupling of the interface tracking and the two-fluid models for the simulation of incompressible two-phase flow, *J. Comput. Phys.* **171**, 776 (2001).
6. S. Chen, D. Johnson, and P. Raad, Velocity boundary conditions for the simulation of free surface fluid flow, *J. Comput. Phys.* **116**, 262 (1995).
7. S. Chen, D. Johnson, P. Raad, and D. Fadda, The surface marker and micro cell method, *Int. J. Numer. Meth. Fluids* **25**, 749 (1997).
8. D. Enright, S. Marschner, and R. Fedkiw, Animation and rendering of complex water surfaces, in *SIGGRAPH'02* (in press).
9. N. Foster and R. Fedkiw, Practical animation of liquids, in *SIGGRAPH 01* (2001), p. 15.
10. A. Glassner, *Principles of Digital Image Synthesis* (Morgan Kaufmann San Francisco, CA, 1995).
11. F. Harlow, J. Shannon, and J. Welch, *THE MAC METHOD: A Computing Technique for Solving Viscous, Incompressible, Transient Fluid-Flow Problems Involving Free Surfaces*, Technical Report LA-3425 (Los Alamos Scientific Laboratory, Los Alamos, NM, 1965).
12. F. Harlow and J. Welch, Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface, *Phys. Fluids* **8**, 2182 (1965).
13. J. Helmsen, *A Comparison of Three-Dimensional Photolithography Development Methods*, Ph.D. thesis (U.C. Berkeley, Berkeley, CA, 1994).
14. C. Hirt and B. Nichols, Volume of fluid (VOF) method for the dynamics of free boundaries, *J. Comput. Phys.* **39**, 201 (1981).
15. G.-S. Jiang and D. Peng, Weighted ENO schemes for Hamilton–Jacobi equations, *SIAM J. Sci. Comput.* **21**, 2126 (2000).
16. M. Kang, R. Fedkiw, and X.-D. Liu, A boundary condition capturing method for multiphase incompressible flow, *J. Sci. Comput.* **15**, 323 (2000).
17. B. Lafaurie, C. Nardone, R. Scardovelli, S. Zaleski, and G. Zanetti, Modelling merging and fragmentation in multiphase flows with SURFER, *J. Comput. Phys.* **113**, 134 (1994).
18. G. Lapenta and J. Brackbill, Dynamic and selective control of the number of particles in kinetic plama simulations, *J. Comput. Phys.* **115**, 213 (1994).
19. R. LeVeque, High-resolution conservative algorithms for advection in incompressible flow, *SIAM J. Numer. Anal.* **33**, 627 (1996).
20. S. Osher and R. Fedkiw, Level set methods: An overview and some recent results, *J. Comput. Phys.* **169**, 463 (2001).
21. S. Osher and R. Fedkiw, *The Level Set Method and Dynamic Implicit Surfaces* (Springer-Verlag, New York, 2002).
22. S. Osher and J. Sethian, Fronts propagating with curvature dependent speed: Algorithms based on Hamiliton–Jacobi formulations, *J. Comput. Phys.* **79**, 12 (1988).
23. D. Peng, B. Merriman, S. Osher, H.-K. Zhao, and M. Kang, A PDE-based fast local level set method, *J. Comput. Phys.* **155**, 410 (1999).
24. E. Puckett, A. Almgren, J. Bell, D. Marcus, and W. Rider, A high-order projection method for tracking fluid interfaces in variable density incompressible flows, *J. Comput. Phys.* **130**, 269 (1997).
25. P. Raad, S. Chen, and D. Johnson, The introduction of micro cells to treat pressure in free surface fluid flow problems, *J. Fluids Eng.* **117**, 683 (1995).
26. W. Rider and D. Kothe, A marker particle method for interface tracking, in *Proceedings of the Sixth International Symposium on Computational Fluid Dynamics*, edited by H. Dwyer (1995) p. 976.
27. W. Rider and D. Kothe, Stretching and tearing interface tracking methods, in *12th AIAA CFD Conference*, AIAA 95-1717 (1995).
28. W. Rider and D. Kothe, Reconstructing volume tracking, *J. Comput. Phys.* **141**, 112 (1998).
29. J. Sethian, Curvature and the evolution of fronts, *Comm. Math. Phys.* **101**, 487 (1985).
30. J. Sethian, Numerical methods for propagating fronts, in *Variational Methods for Free Surface Interfaces*, edited by P. Concus and R. Finn (Springer-Verlag, Berlin/New York, 1987).
31. J. Sethian, A fast marching level set method for monotonically advancing fronts, *Proc. Natl. Acad. Sci.* **93**, 1591 (1996).

32. J. Sethian, Fast marching methods, *SIAM Rev.* **41**, 199 (1999).
33. J. Sethian, *Level Set Methods and Fast Marching Methods* (Cambridge Univ. Press, Cambridge, UK, 1999).
34. J. Sethian, Evolution, implementation, and application of level set and fast marching methods for advancing fronts, *J. Comput. Phys.* **169**, 503 (2001).
35. C. Shu and S. Osher, Efficient implementation of essentially nonoscillatory shock capturing schemes, *J. Comput. Phys.* **77**, 439 (1988).
36. P. Smolarkiewicz, The multi-dimensional Crowley advection scheme, *Month. Weather Rev.* **110**, 1968 (1982).
37. M. Sussmam and E. Puckett, A coupled level set and volume-of-fluid method for computing 3D and axisymmetric incompressible two-phase flows, *J. Comput. Phys.* **162**, 301 (2000).
38. M. Sussman, A. Almgren, J. Bell, P. Colella, L. Howell, and M. Welcome, An adaptive level set approach for incompressible two-phase flows, *J. Comput. Phys.* **148**, 81 (1999).
39. M. Sussman and E. Fatemi, An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow, *SIAM J. Sci. Comput.* **20**, 1165 (1999).
40. M. Sussman, E. Fatemi, P. Smereka, and S. Osher, An improved level set method for incompressible two-phase flows, *Comput. Fluids* **27**, 663 (1998).
41. M. Sussman, P. Smereka, and S. Osher, A level set approach for computing solutions to incompressible two-phase flow, *J. Comput. Phys.* **114**, 146 (1994).
42. D. Torres and J. Brackbill, The point-set method: Front tracking without connectivity, *J. Comput. Phys.* **165**, 620 (2000).
43. G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y.-J. Jan, A front-tracking method for the computations of multiphase flow, *J. Comput. Phys.* **169**, 708 (2001).
44. S.-O. Unverdi and G. Tryggvason, A front-tracking method for viscous, incompressible, multi-fluid flows, *J. Comput. Phys.* **100**, 25 (1992).
45. M. Williams, D. Kothe, and E. Puckett, Approximating interfacial topologies with applications for interface tracking algorithms, in *37th AIAA Aerospace Sciences Meeting*, AIAA 99-1076 (1999).
46. M. Williams, D. Kothe, and E. Puckett, Convergence and accuracy of kernal-based continuum surface tension models, in *Fluid Dynamics at Interfaces*, edited by W. Shyy (Cambridge Univ. Press, Cambridge, UK, 1999), p. 347.
47. S. Zalesak, Fully multidimensional flux-corrected transport algorithms for fluids, *J. Comput. Phys.* **31**, 335 (1979).
48. Y. Zhang, K. Yeo, B. Khoo, and C. Wang, 3D jet impact and toroidal bubbles, *J. Comput. Phys.* **166**, 336 (2001).