Proceedings of ALGORITMY 2000
Conference on Scientific Computing, pp. 11–19

THE SEGMENT PROJECTION METHOD APPLIED TO MULTIPHASE FLOW SIMULATIONS

A.-K. TORNBERG AND B. ENGQUIST

Abstract. Propagating interfaces are present in many different processes, and include wave fronts and different phase and material boundaries. The numerical tracking of interfaces is therefore an important part of the simulation of many physical phenomena.

The segment projection method is a new method for interface tracking. Each curve is represented as a union of overlapping curve segments that are discretized on one-dimensional Eulerian grids. Hence, the curve is explicitly discretized with points on the curve, but the discretization is Eulerian, in difference to commonly used Lagrangian techniques.

The method has been applied to simulations of immiscible incompressible multiphase flow. In this application, the dynamics is strongly influenced by the location of the interfaces separating the different fluid phases. The interfaces indicate discontinuity in density and viscosity and determine the surface tension forces.

1. Introduction. Interfaces or internal boundaries are present in many different applications, such as high frequency wave propagation, solidification and multiphase flows. Interface tracking methods are developed to numerically describe and track such interfaces when they move and deform as determined by the underlying physics. The rules of interface motion and interaction depend on the application. If the interfaces represent wave fronts, one interface might pass over another without interaction. In in the case of merging bubbles however, two interacting interfaces will reconnect and create one single interface.

In interface tracking methods, the interfaces are represented by continuously updated discretizations. These discretizations can be Lagrangian or Eulerian in nature, depending on the interface tracking technique that is used.

In the Lagrangian approach, marker particles are used to define the interfaces. This approach was used by Peskin [4], and more recent immersed boundary methods as well as front-tracking methods are also based on Lagrangian markers, [1, 5, 9]. Each interface has a separate representation and interface interactions that do not include merging are natural. If merging of two interfaces is wanted, the two separate discretizations must be merged into one, which requires appropriate modifications. This is rather complicated, since there is no restriction in the positions of the discrete points of the two merging interfaces.

A different idea is employed in the level-set method, which was introduced by Osher and Sethian [3]. In this method, the interfaces are implicitly defined as the zero level set of a continuous function, and this function is updated in order to capture the motion of the interfaces. The level-set function is discretized on a Eulerian grid that is defined on the computational domain. To update this function, a vector field for advection is needed throughout the domain. If the velocities are prescribed only at interfaces, extension velocities can be computed everywhere in the domain, see e.g. [6]. The level-set method has been applied to many different problems, see [2, 6, 7]. In the level-set method, topological merging will always occur when interfaces defined by the same level-set function get close relative to the resolution of the grid. Merging is therefore naturally incorporated in the level-set method. Instead, explicit action must be taken to prevent merging if it is not wanted. This is in general not a trivial task.

In this paper, we introduce a new method which can be viewed as a compromise between the front-tracking and level-set methods. The new segment projection method relies on the partitioning of an interface into several parts. In the case of two dimensional calculations, the interfaces are curves in the plane. Each curve is described by a union of overlapping curve segments, where the segments are chosen such that they can be given as functions of one spatial variable. The segment functions are discretized using one-dimensional Eulerian grids in the x and y directions. Hence, the curve is explicitly discretized with points on the curve as in the front-tracking method, thereby keeping the lower dimensionality, but the discretization is Eulerian as in the level-set method.

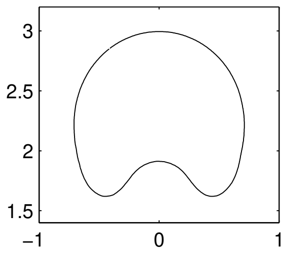
As in the front-tracking method, each interface has a separate representation, which is most natural for non-merging interfaces. It is however easier to define a merging algorithm in the segment projection method, due to the Eulerian discretization of the segment functions.

2. The Segment Projection Method. Introduce the interface Γ bounding a region Ω_A that may be multiply connected. The region Ω_A is contained in the computational domain $\Omega \subset \mathbb{R}^2$. In general, Γ will consist of several separate interfaces, i.e.

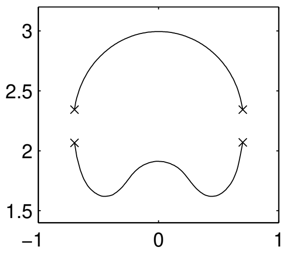
$$\Gamma = \bigcup_j \gamma_j. \tag{2.1}$$

Each of the interfaces γ_j encloses a region of Ω_A, possibly together with parts of $\partial\Omega$, the boundary of Ω. In the example of two-phase flow, the region Ω_A bounded by Γ will be the region occupied by one of the two fluids.

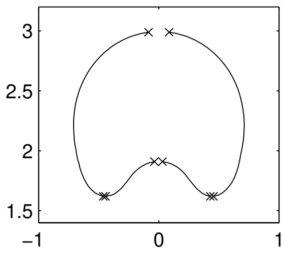
In the segment projection method, each interface γ_j is represented by a union of overlapping curve segments. The segments are chosen such that they can be given as functions of one coordinate variable, i.e. the segments are represented by functions $f_i(x)$ and $g_j(y)$. One example of a curve and its segment representation is given in Figure 2.1.



(a) The interface.



(b) The x-segments.



(c) The y-segments.

FIG. 2.1. *Example of a curve and its segment representation. The x-segments are functions $f_i(x)$, $i = 1, 2$. The y-segments are functions $g_j(y)$, $j = 1, \ldots, 4$.*

We define a uniform x-discretization from the minimum x-value of the computational domain (x_{min}), to the maximum x-value of the domain (x_{max}). The number of intervals is N_x, and the distance between two discrete points is Δx. The same is done in the y-direction, with N_y intervals, and a grid size of Δy. For the x-discretization, we have

$$x_k = x_{min} + k\,\Delta x, \quad k = 0, \dots, N_x, \tag{2.2}$$

and for the y-discretization

$$y_k = y_{min} + k\,\Delta y, \quad k = 0, \dots, N_y. \tag{2.3}$$

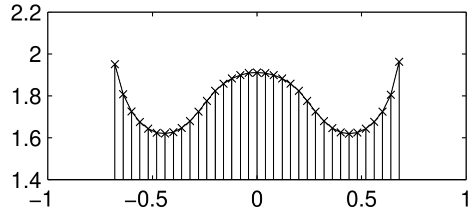
For each segment, the domain of the independent variable must be defined. A segment described as a function of x, an x-segment, is defined from $\tilde{x}_0$ to $\tilde{x}_1$, and a segment described as a function of y, a y-segment, is defined from $\tilde{y}_0$ to $\tilde{y}_1$. In the discrete case, corresponding integer variables k_0 and k_1 are defined so that $x_{k_0-1} < \tilde{x}_0 \le x_{k_0}$ and $x_{k_1} \le \tilde{x}_1 < x_{k_1+1}$. The discretized x-segment is defined from x_{k_0} to x_{k_1}. Similarly, we define k_0 and k_1 also for each y-segment. For each x-segment, the points on the curve are then given by

$$(x_k, f_i(x_k)) = (x_k, f_k), \quad k = k_0, \dots, k_1, \tag{2.4}$$

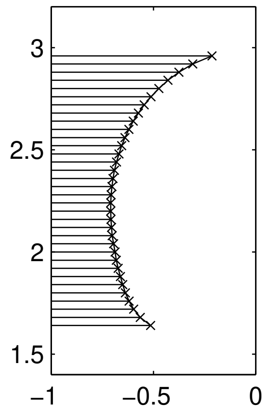
and for each y-segment

$$(g_j(y_k), y_k) = (g_k, y_k), \quad k = k_0, \dots, k_1. \tag{2.5}$$

The discrete points are in general different for an x-segment and a y-segment describing the same part of the curve. The variables k_0 and k_1 are in general different for each segment. The uniform discretizations of two of the segments in Figure 2.1 are indicated in Figure 2.2.



(a) The lower x-segment in Figure 2.1b), segment function $f_1(x)$.



(b) The left most y-segment in Figure 2.1c), segment function $g_1(y)$.

FIG. 2.2. *The segments are discretized using one-dimensional uniform grids.*

The number of segments needed to describe a curve depends on the shape of the curve. An extremum of a function $f_i(x)$ defines a break point that separates the y-segments. Similarly, an extremum of a function $g_j(y)$ defines a break point in the x-segments. To complete the description, information about the connectivity of segments is needed.

Given the definition of an x-segment, create an ordered set containing the start and end points of the segment, together with the extremum points of the segment. With the number of extremum points equal to M, denote the positions of these points by x_0^e to x_{M+1}^e ($x_0^e = x_{k_0}$, $x_{M+1}^e = x_{k_1}$). Then, for each interval

$$(x_m^e, x_{m+1}^e), \quad m = 0, \dots, M, \tag{2.6}$$

it is necessary to keep track of which y-segment that this part corresponds to. For the y-segments, in the same manner we define points (y_m^e, y_{m+1}^e), and corresponding x-segments for each part.

Given a velocity field $\mathbf{u} = (u, v)$ by which the curve should move, the segments $y = f(x,t)$ and $x = g(y,t)$ are updated according to the partial differential equations

$$\frac{\partial f}{\partial t} + u\,\frac{\partial f}{\partial x} = v, \tag{2.7}$$

$$\frac{\partial g}{\partial t} + v\,\frac{\partial g}{\partial y} = u. \tag{2.8}$$

Note that there is only one spatial variable present in both these equations. If the curve is open, boundary conditions must be defined for the segments defining the start and the end of the curve. However, if it is closed, there is an overlap of segments defined in x and y, and we can update the boundary values for one segment from the segment in the opposite direction. These one-dimensional partial differential equations can be discretized using some standard finite difference technique, such as a Lax-Wendroff scheme.

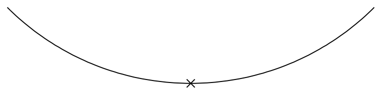
If new extrema have appeared or disappeared after an advection step, the structure of the segments defined as functions of the other spatial variable needs to be modified. The four generic cases are indicated in Figure 2.3. If two new extrema appear around existing extremum in an x-segment (illustrated by $a) \to b)$ in Figure 2.3), two new y-segments must be added in between the extrema. In the reversed situation ($b) \to a)$ in the same figure), the two y-segments in the region between the three old extrema need to be removed.

If two new extrema appear away from existing extrema in an x-segment ($c) \to d)$ in Figure 2.3), the original y-segment must be split in two parts, and a new y-segment must be added in between the two extrema. In the reversed situation, the y-segment between the two old extrema, needs to be removed. The two y-segments defined on each side of the segment to be removed, must be merged into one segment. The same actions must be taken for corresponding x-segments if extrema appear or disappear in a y-segment.

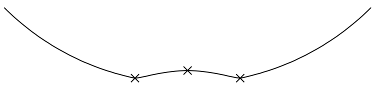
In practice, no short segments are however added or removed in this process. Instead of keeping very short segments, an empty segment is referenced instead. An empty segment is a segment that can be referenced as a corresponding segment, but that contains no information.

Therefore, we need to monitor the use of empty segments. Using the notation introduced around equation (2.6), for each interval

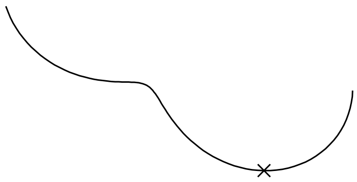
$$\left(x_m^e, x_{m+1}^e\right), \quad m = 1, \dots, M-1, \tag{2.9}$$



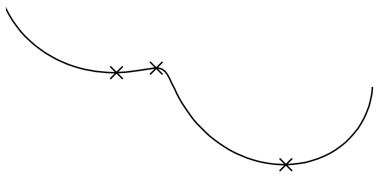
(a) Segment with one extremum.



(b) Segment with three extrema.



(c) Segment with one extremum.



(d) Segment with three extrema.

FIG. 2.3. *The four different cases: Development from a) to b) or from b) to a). Development from c) to d) or from d) to c).*

we need to check if there should be a segment discretized in the y-direction, or if there should only be a reference to an empty segment. Denote the value at x_m^e by f_m^e. If $|f_m^e - f_{m+1}^e| < 2\Delta y$, there should not be an opposite segment (i.e. y-segment) in this interval. If such a segment exists, it should be removed, and an empty segment should be referenced instead. On the other hand, if $|f_m^e - f_{m+1}^e| > 2\Delta y$, there should be an opposite segment corresponding to this interval. If there is no such segment, then it should at this point be created by interpolation from this segment, and added to the structure. For each interval (x_m^e, x_{m+1}^e), we make a reference to the corresponding y-segment, or to an empty segment. The same procedure is made for each y-segment, where $|g_m^e - g_{m+1}^e|$ is compared to $2\Delta x$.

The domain of definition of a segment might need to be extended or reduced after the advection. The extension of this domain is defined by the position of extrema in segments in the opposite direction. If it is extended, so that new values of the discretization need to be defined, these values are interpolated linearly from the corresponding segment discretized in the opposite direction. In this process, we update k_0 and k_1 for all segments.

In overlapping regions, two segments define the same part of the curve, and we can not allow discrepancies to develop in the representation. For both segments, new values are computed as weighted averages of the values of the segment function and the values interpolated from the segment defined in the opposite direction. The weighting function is a function of the slopes of the segments, since the most accurate values should be defined for the segment function with the smallest slope. We use cubic interpolation.

3. The Multiphase Flow Problem. We study two-dimensional incompressible flows including immiscible fluids. In this presentation, we assume that we have two different fluids, fluid A and fluid B. The density and viscosity at a fixed time are given by

$$(\rho(\mathbf{x}), \mu(\mathbf{x})) = \begin{cases} (\rho_A, \mu_A) & \text{for } \mathbf{x} \text{ in fluid A,} \\ (\rho_B, \mu_B) & \text{for } \mathbf{x} \text{ in fluid B.} \end{cases} \tag{3.1}$$

In general $\rho_A \neq \rho_B$ and $\mu_A \neq \mu_B$, so that $\rho(\mathbf{x})$ and $\mu(\mathbf{x})$ are discontinuous across each interface separating fluid A and B. Refer to Figure 3.1 for an example of a configuration of the two fluids A and B.

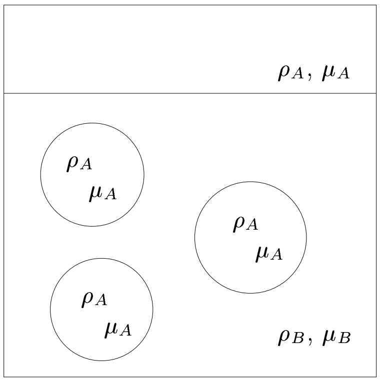


FIG. 3.1. *Example of a configuration involving two fluids A and B, with different density and viscosity (ρ_A, μ_A) and (ρ_B, μ_B).*

The equations describing this immiscible multiphase flow are the Navier-Stokes equations for incompressible flow. The contribution of the surface tension forces and the gravity forces are added as source terms. The equations can be written

$$\rho \left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} \right) = -\nabla p + \nabla \cdot \left(\mu \left(\nabla \mathbf{u} + \nabla \mathbf{u}^T\right)\right) + \mathbf{f} + \rho \mathbf{g}, \tag{3.2}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{3.3}$$

in $\Omega \subset \mathbb{R}^2$, together with appropriate boundary conditions and initial conditions.

As seen in (2.1), Γ will in general consist of several separate interfaces γ_j, where each interface can either be closed or attached to the boundary of the domain. The domain Ω_A enclosed by Γ, is in this case the regions of the domain occupied by fluid A. The surface tension force is given by

$$\mathbf{f} = \sigma \kappa \hat{\mathbf{n}} \delta_\Gamma, \tag{3.4}$$

where δ_Γ is a measure of Dirac delta function type with support on Γ, i.e the union of all interfaces γ_j. Its action on any smooth test function $\varphi(\mathbf{x})$ is given by

$$\int_\Omega \delta_\Gamma \ \varphi \ d\mathbf{x} = \int_\Gamma \varphi \ d\gamma = \sum_j \int_{\gamma_j} \varphi \ d\gamma, \tag{3.5}$$

where the last quantity denotes the sum of the line integrals of $\varphi(\mathbf{x})$ along the interfaces γ_j. The interfaces γ_j are advected by the flow field $\mathbf{u}(\mathbf{x},t)$, and change with time. The coefficient $\sigma \in \mathbb{R}$ in (3.4) is the surface tension coefficient, $\kappa \in \mathbb{R}$ is the curvature and $\hat{\mathbf{n}} \in \mathbb{R}^2$ the normal vector to Γ. The direction of this force is towards the local center of curvature.

The singular surface tension force $\mathbf{f}$ and the discontinuous coefficients ρ and μ are better represented numerically starting from a weak form of the equations. For this formulation, we introduce the spaces

$$\mathsf{V}_\nu = \{\mathbf{v} \in H^1(\Omega)^2 : \mathbf{v} = \nu \text{ on } \partial\Omega\}, \quad \Pi = \{q \in L^2(\Omega) : \int_\Omega q\, d\mathbf{x} = 0\}.$$

Multiplying equation (3.2) by $\mathbf{v} \in \mathsf{V}_0$, (3.3) by $q \in \Pi$, and integrating over the domain using Green's formula, the following variational formulation of (3.2)-(3.3) is obtained. Find $\mathbf{u}(\mathbf{x},t) \in \mathsf{V}_\nu$ and $p(\mathbf{x},t) \in \Pi$ such that $\forall t \in [0,T]$,

$$m(\rho,\mathbf{u}_t,\mathbf{v}) + \tilde{a}(\mu,\mathbf{u},\mathbf{v}) + b(\mathbf{v},p) + c(\rho,\mathbf{u},\mathbf{u},\mathbf{v}) = f_\gamma(\mathbf{v}) + m(\rho,\mathbf{g},\mathbf{v}), \tag{3.6}$$

$$b(\mathbf{u},q) = 0, \tag{3.7}$$

hold for all $\mathbf{v}$ in V_0 and for all q in Π, respectively. The forms in the weak formulation are given in [8], i.e.

$$\tilde{a}(\mu,\mathbf{u},\mathbf{v}) = \int_\Omega \mu \sum_{i,j=1}^{2} \left(\frac{\partial u_i}{\partial x_j}\frac{\partial v_i}{\partial x_j} + \frac{\partial u_i}{\partial x_j}\frac{\partial v_j}{\partial x_i} \right) d\mathbf{x}, \tag{3.8}$$

$$f_\gamma(\mathbf{v}) = \sigma \int_\gamma \kappa \hat{\mathbf{n}} \cdot \mathbf{v}\, d\gamma. \tag{3.9}$$

The singular terms $\nabla \cdot (\mu\, (\nabla\mathbf{u} + \nabla\mathbf{u}^T))$ and $\mathbf{f} = \sigma\kappa\hat{\mathbf{n}}\delta_\gamma$ in (3.2) are easier to approximate numerically in their weak forms (3.8) and (3.9).

In the multiphase flow problem, interfaces are advected by the flow field. The interfaces are represented using the segment projection method. The advection process was described in the previous section, where the advection equations for the segments were given in (2.7)-(2.8). The velocity field is however affected by the configuration of the volumes of different density and viscosity and the surface tension forces, which are all determined by the positions and shapes of the interfaces. In order to complete the formulation of this problem, we need not only to define how the interface Γ is represented and how the evolution of Γ is determined. In addition, definitions of the density and viscosity fields using the information from the segment projection method are needed, as well as calculation of surface tension forces.

The discretization of the Navier-Stokes equations is based on a finite element approximation with piecewise quadratic polynomials in space and a second order BDF difference method in time. To resolve the nonlinearities, we use a direct fixed point iteration. In each iteration, it remains to solve a Stokes problem. The divergence-free constraint is enforced using an iterated penalty method (see [8]). The finite element mesh need not conform to the interfaces. The effects of the interfaces are taken into account through the definition of the density and viscosity fields and the surface tension forces.

The density and viscosity have one set of values in fluid A, (ρ_A, μ_A) and another set of values in fluid B, (ρ_B, μ_B). At a fixed time, given a characteristic function $I(\mathbf{x})$, where

$$I(\mathbf{x}) = \begin{cases} 1 & \text{for } \mathbf{x} \text{ in fluid } A, \\ 0 & \text{for } \mathbf{x} \text{ in fluid } B, \end{cases} \tag{3.10}$$

we can write the density and viscosity as

$$\begin{aligned} \rho(\mathbf{x}) &= \rho_B + (\rho_A - \rho_B)\ \ I(\mathbf{x}), \\ \mu(\mathbf{x}) &= \mu_B + (\mu_A - \mu_B)\ I(\mathbf{x}). \end{aligned} \tag{3.11}$$

For numerical accuracy reasons, we do not want to keep $I(\mathbf{x})$ discontinuous. Let $d(\mathbf{x})$ be the signed distance function defined so that $|d(\mathbf{x})|$ yields the closest distance to any interface, and $d(\mathbf{x}) > 0$ in fluid A, $d(\mathbf{x}) < 0$ in fluid B. Then we have that $I(\mathbf{x}) = H(d(\mathbf{x}))$, where $H(t)$ is the Heaviside function. When $H(t)$ is replaced by an approximation $H_w(t)$, which varies continuously across a transition zone of width $2w$, we naturally obtain a more regular characteristic function $I_w(\mathbf{x})$. The accuracy of the discretization depends on the particular choice of $H_w(t)$, see [8].

The closest distance to the interface can be computed using the segment representations, and the sign of this function can be defined if we also assign two integer indicator variables for each segment.

The definition of the surface tension forces were given in (3.4) as $\mathbf{f} = \sigma\kappa\hat{\mathbf{n}}\delta_\Gamma$, and in their weak form in (3.9) as a sum of line integrals along each interface. To evaluate this interfacial force term, we need to have a representation of each interface, the curvature κ and normal vector $\hat{\mathbf{n}}$ along that interface.

The curvature and normal vectors can be computed from the definition of the segments. The product $\kappa\,\hat{\mathbf{n}}$ that is included in the surface tension forces is defined by

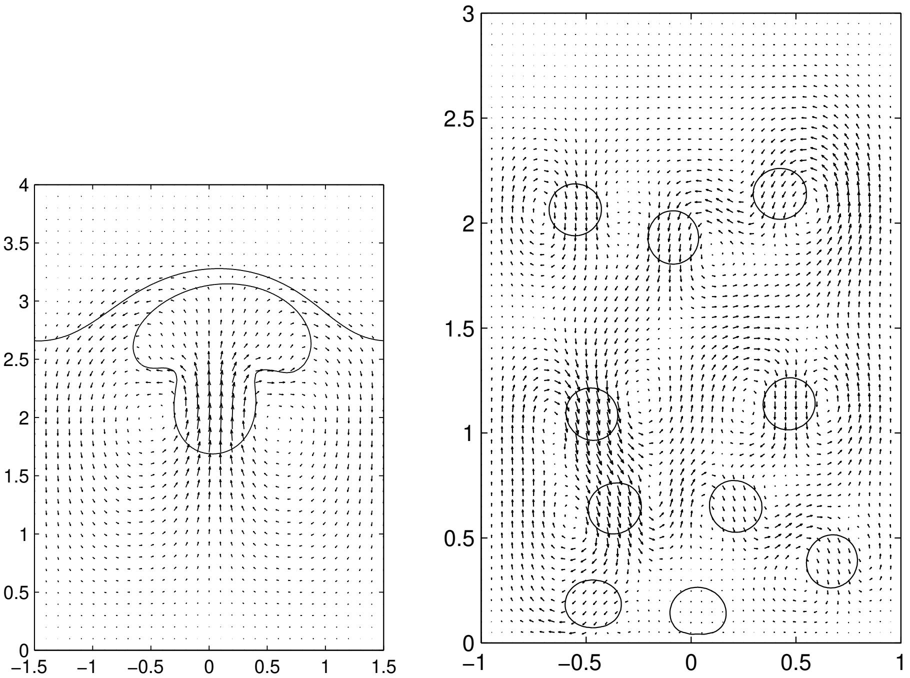
$$\kappa(x)\,\hat{\mathbf{n}}(x) = \frac{f''(x)\,(-f'(x), 1)}{(1 + f'(x)^2)^2}, \tag{3.12}$$

for the x-segments, and

$$\kappa(y)\,\hat{\mathbf{n}}(y) = \frac{g''(y)\,(1, -g'(y))}{(1 + g'(y)^2)^2}, \tag{3.13}$$

for the y-segments. The derivatives in this formulas can be approximated by divided differences $f'(x_k) \approx D_0 f_k$ and $f''(d_k) \approx D_+ D_- f_k$, and similarly, $g'(y_k) \approx D_0 g_k$ and $g''(y_k) \approx D_+ D_- g_k$.

When two regions of the same fluid merge, the segments must be reconnected to represent the new topology, which requires a specific algorithm. This step in the algorithm gives full control over the merging process and the surface physics may influence if or when merging should take place. The segment projection method has been used for simulations with and without topology changes. In Figure 3.2, results from two different simulations are shown. Comparisons with a level-set method and a front-tracking method have been made, and show good agreement. See [8] for more details.



(a) Bubble dynamics including topology changes.

(b) The sedimentation of viscous drops.

FIG. 3.2. *Results from runs with the segment projection method. The arrows represent the velocity fields and the solid lines the interfaces.*

REFERENCES

[1] J. Glimm, J.W. Grove, X.L. Li, K.-M. Shyue, Y. Zeng, and Q. Zhang. Three-Dimensional Front Tracking . *SIAM Journal on Scientific Computing*, 19:703–727, 1998.

[2] S. Osher and R. Fedkiw. Level Set Methods. To appear in *Journal of Computational Physics.*

[3] S. Osher and J.A. Sethian. Fronts Propagating with Curvature Dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations. *Journal of Computational Physics*, 79:12–49, 1988.

[4] C. S. Peskin. Numerical Analysis of Blood Flow in the Heart. *Journal of Computational Physics*, 25:220–252, 1977.

[5] A.M Roma, C.S. Peskin, and M.J Berger. An Adaptive Version of the Immersed Boundary Method. *Journal of Computational Physics*, 153:509–534, 1999.

[6] J.A. Sethian. *Level Set Methods and Fast Marching Methods: Evolving interfaces in Computational Geometry, Fluid Mechanics, Computer Vision and Materials Science.* Cambridge University Press, Cambridge, 1999.

[7] M. Sussman, P. Smereka, and S. Osher. A Level Set Approach for Computing Solutions to Incompressible Two-Phase Flow. *Journal of Computational Physics*, 114:146–159, 1994.

[8] A.-K. Tornberg. *Interface Tracking Methods with Application to Multiphase Flows.* PhD thesis, Royal Institute of Technology, Department of Numerical Analysis and Computing Science, 2000. ISBN 91-7170-558-9, TRITA-NA 0010.

[9] S.O. Unverdi and G. Tryggvason. A Front-Tracking Method for Viscous, Incompressible, Multifluid Flows. *Journal of Computational Physics*, 100:25–37, 1992.