Master Thesis

Evaluation of Interface Tracking Schemes with Finite Element Discretizations

By

Jing Xu
Matr.-Nr. 112025

Submitted to the Faculty of Electrical Engineering and Information Technology in partial fulfillment of the requirements for the degree of Master of Science in Automation and Robotics

Supervised by

Prof. Dr. Stefan Turek
Institute for Applied Mathematics and Numerics (LSIII)
Department of Mathematics
University of Dortmund

MSc. Shu-Ren Hysing
Institute for Applied Mathematics and Numerics (LSIII)
University of Dortmund

14, November 2006
Abstract

This thesis is concerned with the evaluation of performance of different numerical methods to track moving and deforming interfaces with finite element discretizations. We compare and contrast the behavior of the following interface tracking methods: the Volume of Fluid (VOF) method, the level set (LS) method, and the hybrid particle level set (PLS) method. We demonstrate the capabilities of these interface tracking methods to model topology changes by evaluating and comparing the performance of the three different methods applied to four different test cases: simple translation, solid body rotation, a single vortex, and a complex deformation field.

Our results indicate that the level set method is accurate, robust and good at smoothly capturing interfaces, but suffer an excessive amount of mass loss in under-resolved regions of the flow. The VOF method turned out to be stable but is in general less accurate than the level set method. The particle level set method is easily the most accurate of those tested. It employs massless marker particles along interface to provide characteristic information. The method maintains the nice geometric properties of the level set method along with the robustness and the ease-of-implementation. Although the particle level set method failed to work in the practical application of the rising bubble problem, some suggestions were given how to construct a velocity field for which particles do not stray significantly from the interface.

**Keywords:** Volume of Fluid (VOF) method, level set (LS) method, hybrid particle level set (PLS) method, Finite Element Method.
Acknowledgments

My sincere gratitude and deep appreciation goes to Prof. Dr. Stefan Turek, known for his intellectual exuberance. His eternal, contagious zest and vigor for new challenges impressed me deeply. I sincerely thank him for his entrustment of me with the project from which I have gained tremendous amount of knowledge and experience. And I would also like to thank him for his kindness to let me work in such an inspiring and pleasant atmosphere. I consider it as my good fortune that I have had the opportunity to work under his supervision and with such a talented and friendly team.

A special thanks goes to my minor supervisor, Shuren Hysing, for his continuous support in my thesis work. He helped me debug and correct the mistakes in my programming and was always there to meet and give advice, to proof-read and mark up my chapters. He showed me different ways to approach a research problem and the need to be persistent to accomplish any goal. Without his expertise, kindness, his patience in solving problems and his constant guidance, I could not have accomplished this thesis.

Last, but not least, I wish to express my gratitude to my husband, for sharing his experience of the thesis writing endeavor with me, for listening to my complaints and frustrations, and for believing in me. And thanks my parents and every other member of my family, whose love, belief and encouragement made me able to achieve my goal.
# Contents

1 Introduction .............................................. 1
   1.1 Problem Description ............................... 1
   1.2 Interface Tracking Techniques .................. 2
   1.3 Application of Interface Tracking Techniques .... 3
   1.4 Main Objective ...................................... 5
   1.5 Organization of the Thesis ......................... 5

2 Standard Interface Tracking Methods ................. 7
   2.1 Volume of Fluid Method ........................... 7
   2.2 Level Set Method ................................... 8
   2.3 TP2D Solver .......................................... 10
   2.4 Test Cases and Results ............................. 12
       2.4.1 Test Cases .................................. 12
       2.4.2 Results for the Test Cases ................ 17

3 Particle Level Set Method ............................... 29
   3.1 Numerical Implementation .......................... 29
       3.1.1 Particle Initialization ....................... 29
       3.1.2 Time Integration ............................... 30
       3.1.3 Error Correction ............................... 31
       3.1.4 Re-initialization ............................... 32
       3.1.5 Particle Reseeding ............................ 32
   3.2 Results for the Test Cases ........................ 33
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 Practical Application of the Particle Level Set Method</td>
<td>49</td>
</tr>
<tr>
<td>4.1 Rising Bubble Problem</td>
<td>49</td>
</tr>
<tr>
<td>4.2 Particle Level Set Solver</td>
<td>49</td>
</tr>
<tr>
<td>4.3 Analysis of Result</td>
<td>52</td>
</tr>
<tr>
<td>5 Conclusions</td>
<td>57</td>
</tr>
<tr>
<td>Bibliography</td>
<td>60</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Problem Description

Evolving boundaries or interfaces are part of many problems in science and engineering. As introduced in the book [1] by Sethian, consider a boundary, either a curve in two dimensions or a surface in three dimensions, separating one region from another. We are given a normal velocity field $F$, which in general depends on space, time, properties of the boundary (such as the normal direction and the curvature), as well as an indirect dependence from physical simulations using the boundary shape. This curve/surface, which we mean by interface in the following, moves in a direction normal to itself (where the normal direction is oriented with respect to an inside and outside) with the speed function $F$, see Figure 1.1. The goal is to accurately model the evolution of the boundary under the velocity $F$. In many cases we are concerned only with the motion of the interface in its normal direction.

The laws governing motion of interfaces may be directly connected to the shape of the interface, this for example occurs in motion by mean curvature in geometrical applications. An interface may also be passively tracked where it is moved according to a given vector field. Very often the shape of the interface affects the vector field of advection, as it is determined according to some physical law. Interface tracking techniques are developed to describe and track moving and deforming interfaces or curves that represent various kinds of internal boundaries, interfaces, and fronts.
1.2 Interface Tracking Techniques

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

Eulerian methods generally refer to an implicit description of an interface. That is, the interface may arbitrarily cut and move through the computational cells, and thus usually one employs fixed grids here. Lagrangian methods on the other hand treat interfaces explicitly where the cell edges are aligned with and follow the interfaces. Thus in these approaches the grids are required to move with the interfaces which usually complicated the algorithms significantly (especially when breaking up and merging of interfaces should be taken into account).

The first algorithm of discontinuous interface-capturing type was called the Volume of Fluid (VOF) method. The VOF method consists of three ingredients: a scheme to locate the interface, an algorithm to track the interface sharply when moving through a computational grid, and a means of applying boundary conditions at the interface. In paper [4], the author gives an overview of VOF. In VOF, the interface is defined by calculating the fractional volume of each material occupied in each computational cell. These numbers range from zero (no material) to one (completely filled with material). The interfaces exist in the cells with fractional volumes. The volume fractions are updated during the calculation according to appropriate advection equations. At each time step, the interface
position is reconstructed cell wise using the fractional volume of a cell and its nearest neighbors. The curvature of the interface can be estimated using finite-difference approximates based on the neighboring fractional volumes. Once constructed, the interface can be advanced using a fractional step method. That is, the interface is reconstructed and evolved in each spatial dimension separately.

In contrast to the representation of the interface as a discontinuity line within a discontinuous interface-capturing framework, a continuous approach can be defined where the interface is a zero level set of some continuous function. This gives rise to the notion of a “level-set approach”. The level set method embeds the propagating interface $\Gamma$ as a zero level set of a higher dimensional function $\phi$ defined by its distance $d$ from some position $x$ as $\phi(x, t = 0) = \pm d$ which can be positive or negative depending on which side of $\Gamma$ one is interested in. This level set function is discretized on a Eulerian grid that is defined on the computational domain and updated in order to follow the position of the interfaces. The level set function is a smoothly varying distance function which can produce very accurate numerical solutions for the interface location.

1.3 Application of Interface Tracking Techniques

The main application of the described techniques to fluid mechanics is to track an evolving interface. The level set method has had major successes in this area. It can be used to treat problems in compressible flow, incompressible flow, flows having singular vorticity, Stefan problems, and so on [5]. This can be done by viewing the boundary as an interface, and then using the level set method to keep track of the boundary. The advantage of this approach is two-fold. First, it is easy to evaluate quantities such as the local curvature which determine the interface tension of the interface separating different fluids. Second, the approach is unchanged in three dimensions and higher.

Another major application of level set methods is in the area of computing problems in geometric curve and interface evolution. The reason is that the embedding inherent in this initial value partial differential equation approach means that one can rely on the
smoothness of the level set function to deal with singularities in the evolving interface in question. Thus, problems in corner, cusp, singularity development, and topological change can easily be handled. A variety of geometrical problems have been studied using these techniques, as well as theoretical results about curve and interface evolution. These include minimal interfaces, construction of geodesic shortest paths, flow under the second derivative of curvature and interface diffusion, and the construction of self-similar interfaces.

VOF method is another choice to solve interface problems. The VOF-type algorithms typically employ a segregated treatment for the system “flow variables - interface” and finite difference or finite volume approximations on fixed grids [6]. Because of the ability of the method to allow topological changes in the evolving front, differing regions can merge and split apart.

The application of interface tracking methods to materials science problems is also a natural fit. Typical examples include solid-liquid boundaries, boundaries in metal layers in etching and deposition, photolithography development simulations, and solid-gas boundaries. Interface tracking methods have provided computational models to track the evolution of interface profiles during various stages of the manufacturing process.

The use of interface tracking methods in image analysis and computer vision has exploded in recent years. This application stems from the need to extract useful anatomical features from medical images such as MRI or CAT scans. Interface tracking techniques also have many other applications, such as used in a variety of aspects of computer graphics, solving problems in path planning and robotic navigation, computer-aided-design and so on.

There is a vast amount of literature devoted to the application of interface tracking methods. [5] is a review article of the level set method in which recent variants and extensions are discussed and various application of the method has been introduced. [7] presents solution algorithms for tracking interfaces with piecewise linear Volume of Fluid (VOF) methods on Eulerian grids in two and three dimensions. [8] and [9] are the examples of applying level set methods to crystal growth. [10] and [11] discuss the application of
interface techniques to problems in etching and deposition in semiconductor manufacturing. For more applications of the interface tracking methods the reader is referred to [12], [13], [14], [15].

1.4 Main Objective

The main objective of this work is two-fold. One is to compare two standard interface tracking methods – the VOF and level set methods. In order to do so, four test cases are presented and served as benchmark problems to test the performance of these two methods. The other is code implementation of the particle level set method through modifying the existing TP2D code. The ability of the new method to track the interface is also tested with the same test cases and compared with the standard methods. Finally we test the practical application of the particle level set method to a bubble simulation.

1.5 Organization of the Thesis

The first chapter begins with a brief description of interface tracking problems, then an introduction to standard interface tracking methods and their applications in various engineering disciplines is given. The next section briefly outlines the objectives of the work.

Chapter 2 begins with an overview of standard methods for tracking interfaces, including the Volume of Fluid (VOF) and level set methods. Next, four different test cases which can be used to evaluate the performance of the interface tracking techniques are introduced, then the numerical simulation results using the VOF and level set methods are presented and the comparison of these two methods are made based on the computed errors and the interface resolution.

A new interface tracking method – the hybrid particle level set method – is described in Chapter 3. The numerical implementation of the particle level set method is then briefly described. The same four test cases are utilized to test the method’s capability of track-
ing an interface undergoing deformation, after which the simulation results are compared with those calculated by the standard level set method.

Chapter 4 focuses on the application of the particle level set method to a rising bubble problem. The particle level set method is modified in order to yield a smooth interface solution. Results solved by the level set method and the modified particle level set method are presented and compared to a reference solution. The reason of the failure of the application of the particle level set method to the rising bubble problem is analyzed and a possible solution is also presented.

Chapter 5 summarizes the findings and overviews the results from the numerical simulations.
Chapter 2

Standard Interface Tracking Methods

Interface tracking techniques are necessary to efficiently compute accurate numerical approximations to partial differential equations with moving interfaces in the solution. The accurate modeling of these types of flows requires high fidelity algorithms for the kinematics and dynamics of interfaces. The numerical techniques chosen to model the interfaces are especially important for Eulerian methods designed to simulate flows with interfaces of arbitrarily complex topology. In these schemes the computational grid remains fixed, so an interface tracking algorithm must minimize diffusion by maintaining a compact interface thickness without sacrificing the robustness necessary to meet the topology demands. The algorithm must also be possible to extend to three dimensions, and incorporation of additional interface physics should be straightforward. Two commonly used interface tracking techniques are the Volume of Fluid (VOF) and the level set (LS) methods.

2.1 Volume of Fluid Method

The Volume of Fluid (VOF) method was first reported in [16], and more completely in [17]. In the VOF method, the initial (known) fluid interface geometry is used to compute fluid volume fractions for each computational cell. This task amounts to computing the volume enclosed by the fluid interface in each cell containing an interface segment. Exact interface information is then discarded in favor of the discrete volume fraction data. Interfaces are subsequently “tracked” by evolving fluid volumes in time with the solution of a standard convection equation.
\[ \frac{\partial f}{\partial t} + \vec{u} \cdot \nabla f = 0 \]  

(2.1)

Volume fractions result from normalization of fluid volumes relative to the cell control volume. At any time in the solution, an exact interface location is not known, i.e., a given distribution of volume fraction data does not guarantee a unique interface topology. The interface geometry is instead inferred (based on assumptions of the particular algorithm) and its location is “reconstructed” from local volume fraction data. Interface locations are then used to compute the volume fluxes necessary for the convective term in the volume evolution equation. Typical implementations of these algorithms are one-dimensional, with multi-dimensionality obtained through operator splitting. In short, the assumed interface geometry, interface reconstruction and volume flux calculation typically comprise the unique features of a given VOF method.

The Volume of Fluid method has become the preferred choice when dealing with severe topological changes of the interface, since it requires no a priori assumptions on the nature of the fluid interface and as such no special techniques to handle changes in the topology of the interface. In [18], the author summarizes the advantages and disadvantages of this method. The VOF method is robust, preserve mass inherently, topology changes are implicit to the algorithm and the scheme is local using only those \( f \) values in neighboring cells thus allowing for parallelization. Its disadvantages include an inability to resolve details of the interface smaller than the mesh size, additionally, the accuracy of viscous stress and surface tension is reduced when the curvature and orientation are not determined accurately. The implementation of these techniques for three dimensional problems, in particular when unstructured grids are employed, is not always straightforward and the computational cost for large three-dimensional computations can become excessively high. Another aspect that can be critical in VOF methods is the evaluation of the interface curvature, essential in applications where surface tension effects are relevant.

### 2.2 Level Set Method

The level set method was introduced in [19] for the numerical solution of front propagating problems with curvature-dependent motion and then extended to a variety of physical
applications. The underlying idea behind the level set method is to embed an interface $\Gamma$ in $\mathbb{R}^m$ which bounds an open region $\Omega \subset \mathbb{R}^m$ as the zero level set of a higher dimensional function $\phi(\vec{x}, t)$. The level set function has the following properties,

\begin{align}
\phi(\vec{x}, t) > 0 & \quad \text{for} \quad \vec{x} \in \Omega \quad (2.2) \\
\phi(\vec{x}, t) \leq 0 & \quad \text{for} \quad \vec{x} \not\in \Omega \quad (2.3)
\end{align}

where we include $\phi = 0$ with the negative $\phi$ 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 $\phi$ is a scalar function in $\mathbb{R}^m$ 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, geometry of the interface, or be given externally for instance as the material velocity in a fluid flow simulation. In all of the following examples, the velocity field is externally given. The evolution equation for the level set function is given by

$$\phi_t + \vec{u} \cdot \nabla \phi = 0 \quad (2.4)$$

This equation essentially only needs to be solved locally near the interface. A so called narrow band approach but in the following it is solved globally for the sake of convenience.

It is also convenient to make $\phi$ 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, the level set function can quickly cease to be a signed distance function especially for flows undergoing extreme topological changes. Re-initialization algorithms maintain the signed distance property by solving to steady state (as fictitious time $t \to \infty$) the equation

$$\phi_t + sgn(\phi_0)(|\nabla \phi| - 1) = 0 \quad (2.5)$$
where \( \text{sgn}(\phi_0) \) is a one-dimensional smeared out signum function approximated numerically in [14] as

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

(2.6)

Efficient ways to solve equation 2.5 to steady state via fast marching methods are discussed in [20]. Again, equation 2.5 only needs to be solved locally near the interface.

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

\[
\vec{N} = \frac{\nabla \phi}{|\nabla \phi|}
\]

(2.7)

and the curvature,

\[
\kappa = \nabla \cdot \left( \frac{\nabla \phi}{|\nabla \phi|} \right)
\]

(2.8)

The main advantage of the level set method, when compared with the VOF method, is that the advection equation 2.4 is solved for a continuous function, rather than a discontinuous one. Moreover, the evaluation of geometrical quantities, such as interface normals and curvature, is much easier and potentially more accurate. On the other hand, the VOF method guarantees better mass conservation properties. Indeed, in the level set method, the mass conservation properties strongly depend on the numerical schemes adopted for the solution of equation 2.4 and on the re-initialization procedure.

\section{2.3 TP2D Solver}

TP2D is short for Transport Phenomena in two dimensions. The TP2D code is an extension of the FeatFlow incompressible flow solver to treat immiscible fluids [21]. TP2D is based on a finite element discretization in space with non-conforming \( \tilde{Q}_1 Q_0 \) basis functions for the flow variables and a conforming \( Q_1 \) bilinear approximation for the level set
function. The computational grid used, which provided a basis for the discretization in space, was simply a tensor product grid of quadrilaterals. The level set/VOF (Volume of Fluid) solver in the TP2D code of our work is essentially solving the following partial differential equation

$$\frac{\partial \phi}{\partial t} + (\mathbf{w} \cdot \nabla) \phi = 0$$

where the scalar \( \phi \) is initialized at \( t = 0 \) either as a signed distance function for level set calculations or a Heaviside function for VOF simulations. The velocity vector field \( \mathbf{w} \) is thus transporting the scalar \( \phi \) in time, and thus also implicitly our interface (represented by the contour line \( \phi = 0 \) for level set calculations and \( \phi = 0.5 \) for VOF code). This equation is discretized in time with a standard \( \theta \)-method which yields the following equation to solve:

Given \( \phi^n \) and the time step \( \Delta t = t_{n+1} - t_n \), then solve for \( \phi = \phi^{n+1} \)

$$\frac{\Phi - \phi^n}{\Delta t} + \theta[(\mathbf{w} \cdot \nabla) \phi] = (\theta - 1)(\mathbf{w} \cdot \nabla)\phi^n \quad (2.9)$$

where \( \theta = 1 \) will be the simple first-order Backward Euler-scheme (BE), and \( \theta = 0.5 \) the second-order Crank-Nicolson-scheme (CN). We here used \( \theta = 0.5 \) in our simulation throughout this study.

After time discretization we discretize in space with the finite element method. We choose bilinear quadrilaterals (\( Q_1 \)) as the trial space to search for a solution in. After space discretization we are left with a matrix system to solve in each time step of the type

$$(\mathbb{M}_L + \Delta t \theta \mathbb{K}) \mathbf{u}^{n+1}_h = \mathbf{g}$$

where \( \mathbf{u}^{n+1}_h \) is the sought unknown solution vector at time level \( n + 1 \), \( \mathbb{M}_L \) is the lumped mass matrix and \( \mathbb{K} \) is the iteration matrix. They are defined by

$$\mathbb{M} = \int_{\Omega} \psi_i \psi_j \, d\Omega$$

$$\mathbb{K} = \int_{\Omega} \frac{\partial \psi_i}{\partial x_k} \psi_j \, d\Omega$$
where $M_L$ is constructed by the usual row-sum lumping of $M$. Finally $g$ is the right hand side vector represented by

$$g = [M_L + \Delta t(\theta - 1)K]u^l_h.$$ 

Artificial stabilization is added in the form of FEM-TVD with a MC-limiter [22].

### 2.4 Test Cases and Results

#### 2.4.1 Test Cases

In order to judge how good the above techniques really are in the tracking of moving interfaces, certain test problems have been devised [23]. Rigorous test problems which exist for some methods, such as continuum advection methods, are lacking in the case of discontinuity tracking [7]. An acceptable tracking method must be able to translate and rotate bodies without significant distortion or degradation of fluid interfaces, thus we use the following two test cases of simple translation and solid body rotation problems. Mass should be conserved rigorously. Neither of these two tests should pose a serious problem for any well-designed interface tracking scheme.

Flows near interfaces often possess a strong vortical component due to sharp gradients in fluid properties such as density and viscosity. Thus tests which impose strong vorticity and which stretch and tear the interface are required. A second set of tests were designed to examine exactly such vortical flows. The first of these tests uses a single vortex which spins fluid elements into a filament that spirals toward the vortex center. The second one imposes a flow field which forces the flow to undergo large topological changes. In the converged limit, elements will not tear, but form thin filaments. The flow field for both examples is solenoidal. In addition, a mechanism is provided for time-reversing the flow, returning it to the initial condition. By forcing the flow to return to its initial state, quantitative comparison and evaluation (e.g. error norms, convergence tests) can be made.

Each of the four test problems have identical initial conditions defined by Rider & Kothe [23], that is: a circle of radius 0.15 is centered at (0.50, 0.75) within a $1 \times 1$ computa-
tional domain, as shown in Figure 2.1.

![Initial condition for all test cases.](image)

**Figure 2.1:** Initial condition for all test cases.

All boundaries are periodic. A scalar field is initialized in the domain equal to unity and zero inside and outside the circle respectively, for the VOF method. This field can be thought of as a fluid volume fraction for a circular body of fluid, i.e, the volume fraction is 100% inside the circle and 0% outside. For the level set method, a distance function is defined as

$$\phi = \sqrt{(x - 0.5)^2 + (y - 0.75)^2} - 0.15$$

The first two problems, translation and solid body rotation, are standard in the literature. The last two, a single vortex and deformation field, are much more challenging because the flow field deforms (stretches and tears) fluid elements.

**Case 1: simple translation**

A uniform and constant velocity field is imposed such that the circular body translates diagonally across the mesh (45° angle). The velocity field is
Figure 2.2 shows the velocity field. After 1 time unit, the body returns to its initial position from where error measurements can be made. The body should not change shape as a result of this movement.

Case 2: solid body rotation

A constant-vorticity velocity field is imposed at the center of the domain, e.g.:

\[
    u = 2\pi(0.5 - y), \quad v = 2\pi(x - 0.5).
\]

The velocity field is shown in the following Figure 2.3. This field will cause all fluid elements to rotate around the center. The body will undergo one rotation in 1 time unit, and should ideally not change shape as a result of this rotation.

Case 3: single vortex

A single vortex is imposed with a velocity field defined by

\[
    u = -2\sin^2(\pi x)\sin(\pi y)\cos(\pi y), \quad v = 2\sin(\pi x)\sin^2(\pi y)\cos(\pi y).
\]
Figure 2.3: A constant-vorticity velocity field for case 2, solid body rotation.

Figure 2.4(a) shows the non-constant vorticity velocity field centered in the box. This velocity field stretches out the circle into a very long, thin fluid element which progressively wraps itself towards the center of the box. Figure 2.4(b) shows the solution for the single vortex problem at $t = 1$ when $t_{\text{end}} = 2$.

Case 4: deformation field

A far more radical deformation of the interface will occur with the complex velocity field given by:

$$u = \sin(4\pi(x + \frac{1}{2})) \sin(4\pi(y + \frac{1}{2})), \quad v = \cos(4\pi(x + \frac{1}{2})) \cos(4\pi(y + \frac{1}{2})).$$

The velocity field and the solution for the deformation field at $t = 1$ are shown in Figures 2.5(a) and 2.5(b) respectively. This gives a stringent test for interface tracking methods. The tests above are time-reversed in the sense of advancing the flow forwards for a time and then reversing the velocity field in order to return the fluid object to its initial position by multiplied by $\cos(\pi t / T)$. Errors can then be measured by comparison to the initial state.
Figure 2.4: Single vortex flow and its solution at $t = 1$.

Figure 2.5: Deformation velocity field and its solution at $t = 1$. 
2.4.2 Results for the Test Cases

Visual inspection of the results is one obvious way to compare simulations. But in order to determine how accurate our simulations are, we introduce the following quantities which are useful to describe the temporal evolution of the interfaces quantitatively.

**Density error.** The density L1 error norm is defined as:

\[ ||u_{err}||_1 = \frac{1}{NVT} \sum_{i=1}^{NVT} [\max(0, -\text{sign}(u_i(t = 0))) - \max(0, -\text{sign}(u_i(t = t_{end})))] \]

where \( NVT \) is the number of nodes, \( u \) is the solution vector, and \( \text{sign} \) is the signum function defined by

\[
\text{sign}(a) = \begin{cases} 
-1, & a < 0 \\
0, & a = 0 \\
1, & a > 0.
\end{cases}
\]

This error measure calculates a density function equal to unity inside the interface and zero outside. This density function is used to calculate the difference between the initial and final states, these should ideally coincide if we have a perfect interface tracking algorithm.

**Circularity.** The “degree of circularity”, introduced by Wadell [24], can in two dimensions be defined as

\[ C = \frac{P_a}{P_b} = \frac{\text{perimeter of area-equivalent circle}}{\text{perimeter of interface}} = \frac{\pi d_a}{P_b} \]

Here \( P_a \) denotes perimeter or circumference of a circle with diameter \( d_a \) and area equal to that of the interface under consideration with perimeter \( P_b \). Obviously for a perfect circular interface the circularity will be equal to unity and decrease as the interface is deformed.

**Mass/Area.** An obvious quantity to measure is mass/area conservation which ideally should be preserved perfectly. This is one of the most common measures that is used in validation of interfacial flow algorithms. Although it does give information regarding the potential accuracy of the simulation, it is not sufficient to determine whether the result
is converged or even physically correct. In the following we will be concerned with the relative area change, that is

\[ \Delta A_{rel} = \frac{\left| \text{Area}(t = t_{end}) - \text{Area}(t = 0) \right|}{\left| \text{Area}(t = 0) \right|}. \]

In addition to this experimental convergence rates for the quantities were computed as

\[ EOC = \log_{10}\left( \frac{||\epsilon^{l-1}||}{||\epsilon^l||} \right) / \log_{10}\left( \frac{h^{l-1}}{h^l} \right) \]

where \( l \) is the level of refinement and \( h \) is the mean cell edge length.

The results for the test cases computed with the TP2D code are presented here. All computations were performed on rectangular tensor product grids with cell sizes \( h = 1/[40, 80, 160, 320] \). The time step was fixed to \( \Delta t = h/16 \).

The relative error norms and convergence rates of test cases 1 and 2 using the VOF and level set methods at the final time \( t_{end} = 1 \) are shown in Tables 2.1, 2.2, 2.3 and 2.4 respectively. The second column of the tables shows the density L1 error norm. For the simple translation problem, the results indicate that the level set method gives first order convergence for the density error norm with \( h = 1/80 \) and second order convergence on finer grid sizes, while VOF only gives first order convergence rate for the density error norm. Figure 2.6 are graphical outputs of the simple translation problem solved by both VOF and level set methods at the final time \( t_{end} = 1 \). As can be seen from the errors in Table 2.1 and 2.2 as well as Figure 2.6, the level set method preserves the circle much better compared to VOF at the final time. For the solid body rotation problem, the level set method also shows much smaller density error norm compared to the VOF method. It is obvious that results solved with the level set method have much smaller relative area change than those solved with VOF for both test cases. Figure 2.7 are graphical outputs of the solid body rotation problem solved by the VOF and level set methods at time \( t = 0.5 \).

Both the VOF and level set methods perform well in both the translation and solid body rotation test cases. The results exhibit good mass conservation for both methods because of the lack of topology change. The solution quality increases predictably as the grid is refined. Judging from the density error and relative area change of various grid re-
finement levels, the level set method perform better compared to VOF and has a higher convergence rate for the quantities than VOF (for example, level set method converges with almost fourth-order while VOF only has first-order convergence for the solid body rotation problem when $h = 1/160$).

Results for the single vortex problem solved with VOF and level set methods at the final time $t_{end} = 2$ are shown in Tables 2.5 and 2.6 respectively. The level set method begins to lose its distinct advantage over VOF method which is due to numerical diffusion in the distance function solution and the loss of mass. However, it still converges at second-order for the density error norm and relative area change, which is better than the convergence rate for VOF, refer to Figures 2.8 and 2.9.

Tables 2.7 and 2.8 show the results of deformation field problem using both VOF and the level set methods at the final time $t_{end} = 2$. In this test case, the level set solution has lost much of its integrity and the mass loss is clearly evident (see Fig. 2.10 and Fig. 2.11). As can be seen from the Figures 2.11(a), 2.11(c) and 2.11(e), the return to initial state at $t_{end} = 2$ solved by the level set method is very deformed and cannot preserve the shape of the circle. The mass loss is reduced with refined grids, but remains large on coarser meshes. The VOF solution preserves the large scale features of the flow. The final state at $t_{end} = 2$ is not of high quality, but the general shape of the circular body is still recognizable on the finest grid. The severe mass loss shows itself as the flattening of the circle top and bottom in Figure 2.11(f)

As can be seen from the comparison and analysis above, the level set method can provide good results when the flow is simple. When the body being tracked deforms the level set method experiences severe mass loss and then loses its accuracy. The VOF method turned out to be stable but is in general less accurate than the level set method although for the deformation field test case VOF was clearly more accurate.
<table>
<thead>
<tr>
<th>1/h</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.01309</td>
<td>N/A</td>
<td>0.19453</td>
<td>N/A</td>
<td>0.98716</td>
</tr>
<tr>
<td>80</td>
<td>0.00793</td>
<td>0.7236</td>
<td>0.07483</td>
<td>1.3786</td>
<td>0.99695</td>
</tr>
<tr>
<td>160</td>
<td>0.00494</td>
<td>0.6826</td>
<td>0.02827</td>
<td>1.4043</td>
<td>0.99892</td>
</tr>
<tr>
<td>320</td>
<td>0.00293</td>
<td>0.7526</td>
<td>0.01088</td>
<td>1.3781</td>
<td>0.99936</td>
</tr>
</tbody>
</table>

**Table 2.1**: Relative error norms and convergence rates for simple translation problem with VOF.

<table>
<thead>
<tr>
<th>1/h</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.00476</td>
<td>N/A</td>
<td>0.03422</td>
<td>N/A</td>
<td>0.99112</td>
</tr>
<tr>
<td>80</td>
<td>0.00213</td>
<td>1.1572</td>
<td>0.00071</td>
<td>5.5909</td>
<td>0.99862</td>
</tr>
<tr>
<td>160</td>
<td>0.00031</td>
<td>2.7895</td>
<td>0.00434</td>
<td>-2.6118</td>
<td>0.99957</td>
</tr>
<tr>
<td>320</td>
<td>5.8E-05</td>
<td>2.4061</td>
<td>0.00080</td>
<td>2.4396</td>
<td>0.99990</td>
</tr>
</tbody>
</table>

**Table 2.2**: Relative error norms and convergence rates for simple translation problem with the level set method.
\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|c|}
\hline
$1/h$ & Density error & Order & Relative area change & Order & Circularity \\
\hline
40 & 0.01309 & N/A & 0.19459 & N/A & 0.99556 \\
80 & 0.00854 & 0.6167 & 0.08020 & 1.2785 & 0.99895 \\
160 & 0.00486 & 0.8122 & 0.03113 & 1.3650 & 0.99941 \\
320 & 0.00288 & 0.7540 & 0.01223 & 1.3484 & 0.99971 \\
\hline
\end{tabular}
\caption{Relative error norms and convergence rates for solid body rotation problem with VOF.}
\end{table}

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|c|}
\hline
$1/h$ & Density error & Order & Relative area change & Order & Circularity \\
\hline
40 & 0.00535 & N/A & 0.01042 & N/A & 0.99403 \\
80 & 0.00366 & 0.5496 & 0.01714 & -0.7180 & 0.99960 \\
160 & 0.00019 & 4.2452 & 0.00283 & 2.5985 & 0.99993 \\
320 & 8.7E-05 & 1.1430 & 0.000250 & 3.5008 & 0.99999 \\
\hline
\end{tabular}
\caption{Relative error norms and convergence rates for solid body rotation problem with the level set method.}
\end{table}

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|c|}
\hline
$1/h$ & Density error & Order & Relative area change & Order & Circularity \\
\hline
40 & 0.01547 & N/A & 0.18816 & N/A & 0.97736 \\
80 & 0.00793 & 0.9646 & 0.05646 & 1.7357 & 0.99627 \\
160 & 0.00374 & 1.0827 & 0.02161 & 1.3860 & 0.99901 \\
320 & 0.00221 & 0.7580 & 0.00867 & 1.3183 & 0.99968 \\
\hline
\end{tabular}
\caption{Relative error norms and convergence rates for single vortex problem with VOF.}
\end{table}
<table>
<thead>
<tr>
<th>1/h</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.00892</td>
<td>N/A</td>
<td>0.04329</td>
<td>N/A</td>
<td>0.98890</td>
</tr>
<tr>
<td>80</td>
<td>0.00549</td>
<td>0.70156</td>
<td>0.03467</td>
<td>0.3204</td>
<td>0.99759</td>
</tr>
<tr>
<td>160</td>
<td>0.00150</td>
<td>1.8667</td>
<td>0.00854</td>
<td>2.0214</td>
<td>0.99953</td>
</tr>
<tr>
<td>320</td>
<td>0.00034</td>
<td>2.1065</td>
<td>0.00117</td>
<td>2.8677</td>
<td>0.99995</td>
</tr>
</tbody>
</table>

Table 2.6: Relative error norms and convergence rates for single vortex problem with the level set method.

<table>
<thead>
<tr>
<th>1/h</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.04343</td>
<td>N/A</td>
<td>0.57533</td>
<td>N/A</td>
<td>0.68963</td>
</tr>
<tr>
<td>80</td>
<td>0.02850</td>
<td>0.6075</td>
<td>0.35086</td>
<td>0.7116</td>
<td>0.67456</td>
</tr>
<tr>
<td>160</td>
<td>0.01701</td>
<td>0.7444</td>
<td>0.19098</td>
<td>0.8775</td>
<td>0.82982</td>
</tr>
<tr>
<td>320</td>
<td>0.01102</td>
<td>0.6272</td>
<td>0.12938</td>
<td>0.5622</td>
<td>0.94026</td>
</tr>
</tbody>
</table>

Table 2.7: Relative error norms and convergence rates for deformation field problem with VOF.

<table>
<thead>
<tr>
<th>1/h</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.05294</td>
<td>N/A</td>
<td>0.70208</td>
<td>N/A</td>
<td>0.69875</td>
</tr>
<tr>
<td>80</td>
<td>0.03826</td>
<td>0.4688</td>
<td>0.47091</td>
<td>0.5762</td>
<td>0.67805</td>
</tr>
<tr>
<td>160</td>
<td>0.02403</td>
<td>0.6706</td>
<td>0.31090</td>
<td>0.5990</td>
<td>0.67308</td>
</tr>
<tr>
<td>320</td>
<td>0.01506</td>
<td>0.6742</td>
<td>0.20631</td>
<td>0.5916</td>
<td>0.74568</td>
</tr>
</tbody>
</table>

Table 2.8: Relative error norms and convergence rates for deformation field problem with the level set method.
Figure 2.6: Comparison of methods for the simple translation problem at $t_{end} = 1$. 

(a) LS  $h = 1/40$  
(b) VOF  $h = 1/40$

(c) LS  $h = 1/160$  
(d) VOF  $h = 1/160$

(e) LS  $h = 1/320$  
(f) VOF  $h = 1/320$
Figure 2.7: Comparison of methods for the solid body rotation problem at $t = 0.5$. 

(a) LS $h = 1/40$  
(b) VOF $h = 1/40$  
(c) LS $h = 1/160$  
(d) VOF $h = 1/160$  
(e) LS $h = 1/320$  
(f) VOF $h = 1/320$
Figure 2.8: Comparison of methods for the single vortex problem at $t = 1$ when $t_{end} = 2$. 
Figure 2.9: Comparison of methods for the single vortex problem at $t = 2$ when $t_{end} = 2$. 

(a) LS $h = 1/40$  
(b) VOF $h = 1/40$  

(c) LS $h = 1/160$  
(d) VOF $h = 1/160$  

(e) LS $h = 1/320$  
(f) VOF $h = 1/320$
Figure 2.10: Comparison of methods for the deformation field problem at $t = 1$ when $t_{end} = 2$. 

(a) LS $h = 1/40$

(b) VOF $h = 1/40$

(c) LS $h = 1/160$

(d) VOF $h = 1/160$

(e) LS $h = 1/320$

(f) VOF $h = 1/320$
Figure 2.11: Comparison of methods for the deformation field problem at \( t = 2 \) when \( t_{\text{end}} = 2 \).
Chapter 3

Particle Level Set Method

The main drawback of the level set interface tracking method with relatively coarse grids is its susceptibility to numerical dissipation, and inability to resolve areas of high curvature. When a level set function is advected, the result is that all sharp edges get smoothed out and mass/volume loss ensues. In [25], the authors propose a hybrid particle level set (PLS) method which is a combination of an Eulerian level set method and a particle-based Lagrangian scheme.

The particle level set method is used to update the position of the interface in a mass preserving manner. The method is essentially a thickened front tracking approach which utilizes particles to assist the level set in accurately tracking flow characteristics in under-resolved regions and consequently preserve mass. This is achieved through the placement of massless marker particles near the interface as an error correction mechanism for the level set function. Detailed implementation of the hybrid particle level set method is presented in the paper by Enright et al. [25].

3.1 Numerical Implementation

3.1.1 Particle Initialization

When the initial interface is defined, two sets of massless marker particles are placed within a band of the interface. This is done by randomly placing particles inside of any
grid cell that is within three cells of the interface. For the particle placed in the cells, both its position and its radius are stored, which later is used to perform error correction on the level set. The radius of a particle changes dynamically throughout the simulation since the particle’s relative location to the interface changes in time. Initially the radius is set so that the boundary of the particle is tangent to the interface whenever possible:

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

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 positive particle radius is always bounded by \( r_{\text{min}} \) and \( r_{\text{max}} \). Also, particles are allowed to overlap in order to get an enhanced reconstruction of the interface.

In our experiment, we first tried to place particles regularly, e.g., put particles equidistantly but not overlap them in each cell. The simulation results showed a not so smooth interface. This is partly because that non-overlapping placement of particles leaves large spaces between each particle, only a small part of the boundary of the particles can be used to represent the interface. For the purpose of good interface reconstruction, we apply random placement of particles and allow particles to overlap. Figures 3.1 and 3.2 demonstrate the effect of using different placement strategies. Figure 3.1 uses three equally spaced non-overlapping circles to represent a straight line while Figure 3.2 uses 6 overlapped circles. It is clear that Figure 3.2 resolve the line much better. In [25], the author shows that overlapping particles does not hinder the scheme because the particles do not represent finite amounts of mass and allowing overlap merely means that some of the characteristic information tracked by particles is represented in duplicated by more than one particle.

### 3.1.2 Time Integration

The particles and the level set function are separately integrated forward in time using a forward Euler time integration scheme. The level set function is integrated forward using equation 2.9, while the particles are advected with the evolution equation
\[ \frac{d\vec{x}_p}{dt} = \vec{u}(\vec{x}_p) \]

where \( \vec{x}_p \) is the position of the particles and \( \vec{u}(\vec{x}_p) \) is its velocity. Then the particles can be updated using standard Lagrangian update as follows:

\[ \vec{x}_p(t) = \vec{x}_p(t - 1) + \Delta t \vec{u}(\vec{x}_p(t - 1)) \]

The time step \( \Delta t \) is chosen as a fraction of the global time step used for the level set to increase accuracy of the particles. The velocity \( \vec{u}(\vec{x}) \) is indirectly interpolated by evaluation in the corresponding point \( \vec{x} \).

### 3.1.3 Error Correction

To enable error correction, we define a level set value for each particle as follows:

\[ \phi_p(\vec{x}) = s_p(\vec{r}_p - |\vec{x} - \vec{x}_p|) \]

Whenever a particle is on the wrong side of the interface by more than its radius, it is considered to have escaped and can be used to perform error correction on the interface. Error correction is performed using the positive particles to create a temporary \( \phi^+ \) and the negative particles to create a temporary \( \phi^- \). For each positive escaped particle, we find \( \phi_p \) for each node of the cell that contains the particle. The value for each node is then set to
\[ \phi^+ = \max_{\forall p \in E^+} (\phi_p, \phi^+) \]

For each negative escaped particle, we similarly find \( \phi_p \) for each node of the cell that contains the particle. The value for each node is then set to

\[ \phi^- = \min_{\forall p \in E^-} (\phi_p, \phi^-) \]

The complete corrected level set function is then reconstructed using \( \phi^+ \) and \( \phi^- \) by choosing the value with minimum magnitude at each grid node:

\[
\phi = \begin{cases} 
  \phi^+, & \text{if } |\phi^+| \leq |\phi^-| \\
  \phi^-, & \text{if } |\phi^+| > |\phi^-| 
\end{cases}
\]

### 3.1.4 Re-initialization

The level set function \( \phi \) is maintained to be a signed distance function through a re-initialization process using the fast marching method. Since re-initialization may cause the zero level set to move, we use the particles to identify and correct these errors.

The order of operation is:

- evolve both the particles and the level set function forward in time
- correct errors in the level set function using particles
- apply re-initialization
- again correct errors in the level set function using particles
- adjust the particle radii.

### 3.1.5 Particle Reseeding

In order to accurately resolve the interface for all times, we periodically readapt the particle distribution to the deformed interface. This includes particle addition and deletion. Particles will periodically be added to the cell which is near the interface and has fewer particles than a previously defined maximum. And we also delete particles which are too
far from the interface to provide any useful information. More details can be found in the paper [25].

### 3.2 Results for the Test Cases

We here use the same four test cases discussed in section 2.4.1 to test the hybrid particle level set method’s capability of tracking an interface undergoing deformation. For the four test cases, we tried to place 8, 16 or 32 particles in each cell respectively.

The results for the test cases computed with the modified TP2D code are presented here. All computations were performed on rectangular tensor product grids with cell sizes $h = 1/\{40, 80, 160\}$ as before. The time step was again fixed to $\Delta t = h/16$.

Tables 3.1, 3.2 and 3.3 show the relative error norms and convergence rates for the simple translation problem with the particle level set method with 8, 16 and 32 particles in each cell respectively. For the simple translation problem we found that the density error norm calculated by the PLS on coarser grids is close to or even better than the results calculated by the standard level set method on finer grids. Thus, the additional computational cost of placing particles near the interface is potentially offset by the ability to use much coarser grids without sacrificing a faithful representation of the interface. The relative area change is somewhat better for the LS method than for the PLS method, but the PLS method still converges nicely. Also the level set results for the $h = 1/80$ grid is a little too good, by chance perhaps. The circularity for the PLS is roughly equal or better than for LS.

Regarding the solid body rotation problem, we can find from the Tables 3.4, 3.5 and 3.6 that the density error is better for the PLS method while the relative area change for the LS method is somewhat better than the PLS method. The circularity for the PLS method is a little better or equal to the LS method. All in all one can hardly say which method is better since the PLS method behaves very close to the standard LS method for these simple problems.
For the single vortex problem, which errors can be seen in Tables 3.7, 3.8 and 3.9, the ability of the particle level set method to preserve mass is good. The density errors computed by particle level set method compare favorably with those computed by the standard level set method. For the purpose of error analysis, the velocity field is time reversed by multiplying it with \( \cos(\pi t / T) \) where \( t_{end} \) is the time at which the flow returns to its initial state. Figure 3.4 shows the interface resolution solved by the standard level set method and the particle level set method on a \( 40 \times 40 \) cell computational grid at time \( t = 1 \) when \( t_{end} = 2 \). We can hardly make out any difference between these two methods. Figure 3.5 shows the final interface resolution solved by the standard level set method and the particle level set method on a \( 40 \times 40 \) cell computational grid at time \( t = 2 \) when \( t_{end} = 2 \). It is evident that the hybrid particle level set method returns to the initial circular interface much better than the standard level set method. This is due to its ability of conserving mass to an accuracy comparable with the Volume of Fluid method.

When the final time is increased to \( t_{end} = 4 \), the differences between both methods become much clearer. The exact solution of the single vortex problem at \( t = 2 \) when \( t_{end} = 4 \) is shown in Figure 3.3.

![Figure 3.3: Exact solution for the single vortex problem at \( t = 2 \) when \( t_{end} = 4 \).](image)

Figures 3.6 and 3.7 are comparisons of the standard level set method and the particle level
set method on a $40 \times 40$ cell computational grid for the single vortex problem at $t = 2$ and $t = 4$ respectively when $t_{end} = 4$. The particle level set method clearly outperforms the level set method. The tail evolution is captured for much longer times using the particle level set method when compared to the level set solution. The level set method only left a short “tail” while the particle level set still maintain thin, elongated filaments at $t = 2$. Note that in Figure 3.6(c), in under-resolved regions, the particles are not close enough together to accurately represent the interface and thin filament structures break apart. However, the particles still track the interface motion with high accuracy, and thus the resulting pieces are accurate in their location. Figure 3.7 depicts the results as the flow returns to its initial state at $t = 4$. The level set method loses all mass on a $40 \times 40$ cell computational grid and the interface disappears. Therefore the error of the relative area change is 1. However, the particle level set method can still return to its initial state. From Tables 3.10, 3.11 and 3.12, we find that the circularity calculated by the particle level set with 8 particles in each cell is better than that with 16 and 32 particles in each cell initially on a $40 \times 40$ cell computational grid. That means, the result calculated by placing 16 or 32 particles in each cell initially (Fig.3.7) is not improved as expected. Figure 3.8 shows the comparison results of the standard level set method and the particle level set method with a grid size $h = 1/80$ for the single vortex problem at $t = 2$ when $t_{end} = 4$. Compared with Figure 3.6, the solution quality of both the level set method and the particle level set method has been improved as the grid is refined. We also notice that placing more particles in each cell initially has better results on relative area change and circularity for the particle level set method. This confirmed our assumption that particles played a role in conserving mass and providing useful information to maintain the interface.

For the deformation field problem, the particle level set method performed equally well in conserving the mass. From the error Tables 3.13, 3.14 and 3.15, the relative area changes calculated by the particle level set method are much smaller than those calculated by the standard level set method. And the particle level set method also had a higher convergence rate for this error. Figure 3.9 displays the deformation results at a time when the largest deformation of the initial body has taken place. Figure 3.9(a) shows the result calculated by the standard level set method. Mass loss is clearly evident. Figures 3.9(b), 3.9(c) and 3.9(d) show the results calculated by the particle level set method with 8, 16
and 32 particles in each cell respectively. As shown in these Figures, the interface crosses
the top boundary of the domain to reappear on the bottom at $t = 1$ using a time reversed
flow field with a period of $t_{\text{end}} = 2$. Figure 3.10 displays the interface resolution of the
flow after deformation and return of its initial state on a $40 \times 40$ cell computational grid
for the deformation problem. Figure 3.10(a) presents the solution of the standard level
set method. Because of the severe mass loss, the flow cannot return to its initial circular
shape. Figures 3.10(b), 3.10(c) and 3.10(d) present the solution of the particle level set
method. The general shape of the circle has been retained but the interface is not smooth
and exhibits some “bloppy” structure. And this leads to non-satisfactory results of the
circularity.

From the comparison and analysis above, we conclude that level set methods are good at
smoothly capturing the interface, but suffer an excessive amount of mass loss in under-
resolved regions of the flow. This prevents the resolution of thin interfacial filaments and
regions of high curvature. The particle level set method uses massless marker particles to
provide characteristic information and thus counteracts this problem. This method main-
tains the nice geometric properties of level set methods along with the ease and simplicity
of implementation. Surprisingly here adding more particles generally caused a reduction
in accuracy. This can possibly be explained by the difficult velocity field, that is adding
more particles gives more chances for them to stray away from the interface.
<table>
<thead>
<tr>
<th>$1/h$</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.00238</td>
<td>–</td>
<td>0.05303</td>
<td>–</td>
<td>0.99425</td>
</tr>
<tr>
<td>80</td>
<td>0.00030</td>
<td>2.9879</td>
<td>0.02203</td>
<td>1.2673</td>
<td>0.99636</td>
</tr>
<tr>
<td>160</td>
<td>0.00027</td>
<td>0.1512</td>
<td>0.01112</td>
<td>0.9863</td>
<td>0.99933</td>
</tr>
</tbody>
</table>

**Table 3.1**: Relative error norms and convergence rates for simple translation problem with the particle level set method with 8 particles in each cell.

<table>
<thead>
<tr>
<th>$1/h$</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.00238</td>
<td>–</td>
<td>0.04135</td>
<td>–</td>
<td>0.99428</td>
</tr>
<tr>
<td>80</td>
<td>0.00046</td>
<td>2.3713</td>
<td>0.02001</td>
<td>1.0472</td>
<td>0.99871</td>
</tr>
<tr>
<td>160</td>
<td>0.00015</td>
<td>1.6167</td>
<td>0.00964</td>
<td>1.0536</td>
<td>0.99954</td>
</tr>
</tbody>
</table>

**Table 3.2**: Relative error norms and convergence rates for simple translation problem with the particle level set method with 16 particles in each cell.

<table>
<thead>
<tr>
<th>$1/h$</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.00178</td>
<td>–</td>
<td>0.03605</td>
<td>–</td>
<td>0.99468</td>
</tr>
<tr>
<td>80</td>
<td>0.00030</td>
<td>2.5688</td>
<td>0.01602</td>
<td>1.1701</td>
<td>0.99909</td>
</tr>
<tr>
<td>160</td>
<td>0.00012</td>
<td>1.3219</td>
<td>0.00815</td>
<td>0.9750</td>
<td>0.99947</td>
</tr>
</tbody>
</table>

**Table 3.3**: Relative error norms and convergence rates for simple translation problem with the particle level set method with 32 particles in each cell.
<table>
<thead>
<tr>
<th>$1/h$</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.00119</td>
<td>–</td>
<td>0.03320</td>
<td>–</td>
<td>0.99811</td>
</tr>
<tr>
<td>80</td>
<td>0.00030</td>
<td>1.9879</td>
<td>0.01771</td>
<td>0.9066</td>
<td>0.99908</td>
</tr>
<tr>
<td>160</td>
<td>0.00012</td>
<td>1.3219</td>
<td>0.00831</td>
<td>1.0916</td>
<td>0.99948</td>
</tr>
</tbody>
</table>

*Table 3.5:* Relative error norms and convergence rates for solid body rotation problem with the particle level set method with 16 particles in each cell.

<table>
<thead>
<tr>
<th>$1/h$</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.00238</td>
<td>–</td>
<td>0.04836</td>
<td>–</td>
<td>0.99839</td>
</tr>
<tr>
<td>80</td>
<td>0.00030</td>
<td>2.9879</td>
<td>0.02006</td>
<td>1.2695</td>
<td>0.99901</td>
</tr>
<tr>
<td>160</td>
<td>0.00019</td>
<td>0.6590</td>
<td>0.00968</td>
<td>1.0512</td>
<td>0.99947</td>
</tr>
</tbody>
</table>

*Table 3.4:* Relative error norms and convergence rates for solid body rotation problem with the particle level set method with 8 particles in each cell.

<table>
<thead>
<tr>
<th>$1/h$</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.00178</td>
<td>–</td>
<td>0.03053</td>
<td>–</td>
<td>0.99833</td>
</tr>
<tr>
<td>80</td>
<td>0.00046</td>
<td>1.9522</td>
<td>0.01414</td>
<td>1.1104</td>
<td>0.99934</td>
</tr>
<tr>
<td>160</td>
<td>0.00008</td>
<td>2.5236</td>
<td>0.00675</td>
<td>1.0668</td>
<td>0.99962</td>
</tr>
</tbody>
</table>

*Table 3.6:* Relative error norms and convergence rates for solid body rotation problem with the particle level set method with 32 particles in each cell.
<table>
<thead>
<tr>
<th>$1/h$</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.00297</td>
<td>–</td>
<td>0.05408</td>
<td>–</td>
<td>0.99815</td>
</tr>
<tr>
<td>80</td>
<td>0.00046</td>
<td>2.6908</td>
<td>0.02419</td>
<td>1.1607</td>
<td>0.99877</td>
</tr>
<tr>
<td>160</td>
<td>0.00019</td>
<td>1.2756</td>
<td>0.01144</td>
<td>1.0803</td>
<td>0.99942</td>
</tr>
</tbody>
</table>

**Table 3.7:** Relative error norms and convergence rates for the single vortex problem with the particle level set method with 8 particles in each cell when $t_{end} = 2$.

<table>
<thead>
<tr>
<th>$1/h$</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.00238</td>
<td>–</td>
<td>0.04156</td>
<td>–</td>
<td>0.99770</td>
</tr>
<tr>
<td>80</td>
<td>0.00015</td>
<td>3.9879</td>
<td>0.02025</td>
<td>1.0373</td>
<td>0.99877</td>
</tr>
<tr>
<td>160</td>
<td>0.00019</td>
<td>-0.3410</td>
<td>0.00979</td>
<td>1.0485</td>
<td>0.99941</td>
</tr>
</tbody>
</table>

**Table 3.8:** Relative error norms and convergence rates for the single vortex problem with the particle level set method with 16 particles in each cell when $t_{end} = 2$.

<table>
<thead>
<tr>
<th>$1/h$</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.00119</td>
<td>–</td>
<td>0.03122</td>
<td>–</td>
<td>0.99668</td>
</tr>
<tr>
<td>80</td>
<td>0.00030</td>
<td>1.9879</td>
<td>0.01806</td>
<td>0.78967</td>
<td>0.99935</td>
</tr>
<tr>
<td>160</td>
<td>0.00023</td>
<td>0.3833</td>
<td>0.00845</td>
<td>1.0958</td>
<td>0.99961</td>
</tr>
</tbody>
</table>

**Table 3.9:** Relative error norms and convergence rates for the single vortex problem with the particle level set method with 32 particles in each cell when $t_{end} = 2$. 
<table>
<thead>
<tr>
<th>$1/h$</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.00357</td>
<td>–</td>
<td>0.06099</td>
<td>–</td>
<td>0.99795</td>
</tr>
<tr>
<td>80</td>
<td>0.00015</td>
<td>4.5729</td>
<td>0.02235</td>
<td>1.4483</td>
<td>0.99751</td>
</tr>
<tr>
<td>160</td>
<td>0.00023</td>
<td>-0.6167</td>
<td>0.00986</td>
<td>1.1806</td>
<td>0.99752</td>
</tr>
</tbody>
</table>

**Table 3.10:** Relative error norms and convergence rates for the single vortex problem with the particle level set method with 8 particles in each cell when $t_{end} = 4$.

<table>
<thead>
<tr>
<th>$1/h$</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.00178</td>
<td>–</td>
<td>0.04322</td>
<td>–</td>
<td>0.99771</td>
</tr>
<tr>
<td>80</td>
<td>0.00015</td>
<td>3.5496</td>
<td>0.01876</td>
<td>1.204</td>
<td>0.99806</td>
</tr>
<tr>
<td>160</td>
<td>0.00019</td>
<td>-0.3398</td>
<td>0.00968</td>
<td>0.9546</td>
<td>0.99922</td>
</tr>
</tbody>
</table>

**Table 3.11:** Relative error norms and convergence rates for the single vortex problem with the particle level set method with 16 particles in each cell when $t_{end} = 4$.

<table>
<thead>
<tr>
<th>$1/h$</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.00178</td>
<td>–</td>
<td>0.03093</td>
<td>–</td>
<td>0.99686</td>
</tr>
<tr>
<td>80</td>
<td>0.00015</td>
<td>3.5496</td>
<td>0.01776</td>
<td>0.8004</td>
<td>0.99917</td>
</tr>
</tbody>
</table>

**Table 3.12:** Relative error norms and convergence rates for the single vortex problem with the particle level set method with 32 particles in each cell when $t_{end} = 4$. 
<table>
<thead>
<tr>
<th>$1/h$</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.00833</td>
<td>–</td>
<td>0.04757</td>
<td>–</td>
<td>0.93378</td>
</tr>
<tr>
<td>80</td>
<td>0.00229</td>
<td>1.863</td>
<td>0.01992</td>
<td>1.2558</td>
<td>0.95618</td>
</tr>
<tr>
<td>160</td>
<td>0.00158</td>
<td>0.5354</td>
<td>0.00826</td>
<td>1.27</td>
<td>0.92932</td>
</tr>
</tbody>
</table>

**Table 3.13:** Relative error norms and convergence rates for the deformation field problem with the particle level set method with 8 particles in each cell.

<table>
<thead>
<tr>
<th>$1/h$</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.00892</td>
<td>–</td>
<td>0.02875</td>
<td>–</td>
<td>0.98126</td>
</tr>
<tr>
<td>80</td>
<td>0.00457</td>
<td>0.9649</td>
<td>0.01555</td>
<td>0.8867</td>
<td>0.83855</td>
</tr>
<tr>
<td>160</td>
<td>0.00212</td>
<td>1.1081</td>
<td>0.00656</td>
<td>1.2451</td>
<td>0.86830</td>
</tr>
</tbody>
</table>

**Table 3.14:** Relative error norms and convergence rates for the deformation field problem with the particle level set method with 16 particles in each cell.

<table>
<thead>
<tr>
<th>$1/h$</th>
<th>Density error</th>
<th>Order</th>
<th>Relative area change</th>
<th>Order</th>
<th>Circularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>0.00952</td>
<td>–</td>
<td>0.02625</td>
<td>–</td>
<td>0.81540</td>
</tr>
<tr>
<td>80</td>
<td>0.00396</td>
<td>1.2655</td>
<td>0.01297</td>
<td>1.0171</td>
<td>0.87054</td>
</tr>
</tbody>
</table>

**Table 3.15:** Relative error norms and convergence rates for the deformation field problem with the particle level set method with 32 particles in each cell.
Figure 3.4: Comparison of methods on a $40 \times 40$ cell computational grid for the single vortex problem at $t = 1$ when $t_{end} = 2$. 

(a) Level Set method
(b) PLS with 8 particles in each cell
(c) PLS with 16 particles in each cell
(d) PLS with 32 particles in each cell
Figure 3.5: Comparison of methods on a $40 \times 40$ cell computational grid for the single vortex problem at $t = 2$ when $t_{\text{end}} = 2$. 
Figure 3.6: Comparison of methods on a $40 \times 40$ cell computational grid for the single vortex problem at $t = 2$ when $t_{end} = 4$. 
(a) Level Set method (fails to conserve mass and the interface disappears)  

(b) PLS with 8 particles in each cell  

(c) PLS with 16 particles in each cell  

(d) PLS with 32 particles in each cell  

**Figure 3.7:** Comparison of methods on a $40 \times 40$ cell computational grid for the single vortex problem at $t = 4$ when $t_{end} = 4$. 
Figure 3.8: Comparison of methods on a $80 \times 80$ cell computational grid for the single vortex problem at $t = 2$ when $t_{end} = 4$. 

(a) Level Set method

(b) PLS with 8 particles in each cell

(c) PLS with 16 particles in each cell

(d) PLS with 32 particles in each cell
Figure 3.9: Comparison of methods on a $40 \times 40$ cell computational grid for the deformation problem at $t = 1$ when $t_{end} = 2$. 

(a) Level Set method

(b) PLS with 8 particles in each cell

(c) PLS with 16 particles in each cell

(d) PLS with 32 particles in each cell
Figure 3.10: Comparison of methods on a $40 \times 40$ cell computational grid for the deformation problem at $t = 2$ when $t_{end} = 2$. 
Chapter 4

Practical Application of the Particle Level Set Method

4.1 Rising Bubble Problem

The initial configuration consists of a circular bubble of radius \( r_0 = 0.25 \) centered at \([0.5, 0.5]\) in a \([1 \times 2]\) rectangular domain. The density of bubble is lighter than that of the surrounding fluid \( (\rho_2 < \rho_1) \). No-slip boundary conditions are used at the horizontal top and bottom boundaries \( (u = 0) \) while slip conditions are used on the vertical ones \( (u = 0) \). Figure 4.1 shows a sketch of the initial configuration together with boundary conditions. Table 4.1 specifies the physical parameters of the test case. It models a rising bubble at \( Re = 35, Eo = 10 \), with both density and viscosity ratios equal to 10. According to Clift et. al this bubble falls somewhere in the ellipsoidal regime which would mean that surface tension effects should be able to hold the bubble together, thus we do not expect any breakup to occur for this test case [26].

4.2 Particle Level Set Solver

We use the same numerical implementation as described in Chapter 3. But in order to meet the requirement of the test case, we made some modifications.

Firstly, in the step of particle initialization, we place particles randomly in the band which
is three grid cells on each side of the interface and take the exact distance from the particles to the interface as its radius. But once the radius of the particle is less than $r_{\text{min}}$, the particle is deleted and we place a new particle in the cell until the radius of all particles are larger than or equal to $r_{\text{min}}$. In this way, we do not need to limit the minimum radius of the particles.

Secondly, in the time integration step, we choose a much smaller time step for the convection of the particles in order to achieve more accurate and smoother interface resolution.

Thirdly, in the error correction step, escaped particles which are more than 1.5 times their radius from the appropriate side of the interface are deleted in order to retain a smooth interface. In addition, another modification to the error correction procedure was made according to the paper [28]. Given a level-set function $\phi$ and a set of escaped positive particles $E^+$, we initialize the corrected distance function values at the four surrounding grid points $\phi^+$ with $\phi$, the values obtained by advection of the grid-based level set, and then calculate:
\begin{align*}
\phi^+ &= \max_{\forall p \in E^+} (|\phi_p|, |\phi^+|) \quad \text{if } \phi^+ > 0, \\
\phi^+ &= -\min_{\forall p \in E^+} (|\phi_p|, |\phi^+|) \quad \text{if } \phi^+ < 0,
\end{align*}

\begin{align*}
\phi^- &= \min_{\forall p \in E^-} (|\phi_p|, |\phi^-|) \quad \text{if } \phi^- > 0, \\
\phi^- &= -\max_{\forall p \in E^-} (|\phi_p|, |\phi^-|) \quad \text{if } \phi^- < 0,
\end{align*}

Table 4.1: Physical parameters used in the test case.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\rho_1) (liquid)</td>
<td>1000</td>
</tr>
<tr>
<td>(\rho_2) (gas)</td>
<td>100</td>
</tr>
<tr>
<td>(\mu_1) (liquid)</td>
<td>10</td>
</tr>
<tr>
<td>(\mu_2) (gas)</td>
<td>1</td>
</tr>
<tr>
<td>(g_y)</td>
<td>-0.98</td>
</tr>
<tr>
<td>(\sigma)</td>
<td>24.5</td>
</tr>
<tr>
<td>(Re)</td>
<td>35</td>
</tr>
<tr>
<td>(Eo)</td>
<td>10</td>
</tr>
<tr>
<td>(\rho_1/\rho_2)</td>
<td>10</td>
</tr>
<tr>
<td>(\mu_1/\mu_2)</td>
<td>10</td>
</tr>
</tbody>
</table>

For all the escaped positive particles, equations 4.1 and 4.2 attempt to correct the interface location by computing new level set values at the four nodes. This in effect moves the grid-based interface (i.e., the \(\phi = 0\) level) so that any escaped positive particle would be returned to the correct distance on the positive side of the interface. Similarly, a correction of the grid values can be applied in the correction procedure for the negatively escaped particles. Thus, for a set of escaped negative particles \(E^-\), we initialize \(E^-\) with \(\phi\) and then calculate

\begin{align*}
\phi^- &= \min_{\forall p \in E^-} (|\phi_p|, |\phi^-|) \quad \text{if } \phi^- > 0, \\
\phi^- &= -\max_{\forall p \in E^-} (|\phi_p|, |\phi^-|) \quad \text{if } \phi^- < 0,
\end{align*}

The \(\phi^+\) and \(\phi^-\) values are then merged back to obtain the distance function field \(\phi\) on the grid as described above.
4.3 **Analysis of Result**

The results for the rising bubble problem computed with the basic/standard level set method and particle level set method are presented in the following. All computations were performed on rectangular tensor product grids with cell sizes \( h = 1/[20, 40, 80] \). The time step was fixed to \( \Delta t = h/16 \). Table 4.2 shows the final results for the circularity, center of mass and rise velocity calculated by the standard level set method. Table 4.3 shows the final results calculated by the particle level set method at time \( t_{end} = 3 \).

<table>
<thead>
<tr>
<th>1/h</th>
<th>Circularity</th>
<th>Center of mass</th>
<th>Rise velocity</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>0.89235</td>
<td>1.03700012</td>
<td>0.225899867</td>
</tr>
<tr>
<td>40</td>
<td>0.91920</td>
<td>1.0678412</td>
<td>0.211351528</td>
</tr>
<tr>
<td>80</td>
<td>0.92080</td>
<td>1.07637569</td>
<td>0.202887881</td>
</tr>
</tbody>
</table>

**Table 4.2:** Final results for the rising bubble problem with the standard level set method.

<table>
<thead>
<tr>
<th>1/h</th>
<th>Circularity</th>
<th>Center of mass</th>
<th>Rise velocity</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>0.84805</td>
<td>1.02413585</td>
<td>0.233138327</td>
</tr>
<tr>
<td>40</td>
<td>0.91896</td>
<td>1.06749676</td>
<td>0.211101561</td>
</tr>
<tr>
<td>80</td>
<td>0.91833</td>
<td>1.07495245</td>
<td>0.202411565</td>
</tr>
</tbody>
</table>

**Table 4.3:** Final results for the rising bubble problem with the particle level set method.

Figure 4.2 shows a comparison of the time evolution of circularity calculated by the basic/standard level set method and the particle level set method with cell sizes \( h = 1/40 \). In the figure, “LS-L8-Ref” is the circularity calculated by TP2D code with grid size \( h = 1/640 \). It is taken as the reference solution and compared with the results calculated by the PLS method. From Figure 4.2, we observe that circularity calculated by PLS is very close to that calculated by LS and both of them approach to “LS-L8-Ref”. That means, both methods approach the accuracy comparable to the \( h = 1/640 \) results.

The following Figure 4.3 shows the comparison of the time evolution of circularity calculated by the basic/standard level set method and the particle level set method with cell
Figure 4.2: Circularity for the rising bubble problem with $h = 1/40$.

sizes $h = 1/80$. The circularity calculated by level set only method is improved obviously and converges towards the $h = 1/640$ results. The circularity calculated by the particle level set method agrees with the $h = 1/640$ result in the beginning. That means, the particle level set method is much more accurate than the level set method till $t = 2$. But after $t = 2.7$ oscillation appears. Although it seems to converge at the end, the results are still not as satisfying as we expected.

To analyze the failure of the application of the particle level set method to the rising bubble problem, we tried to observe the convection of the particles which were used to track the moving and deforming interface. In order to do that, we initially placed positive particles equidistantly on four circles outside the interface in bands between a distance of $b_{min} = r_{min}$ (the minimum particle radius) and $b_{max} = 3 \max(\Delta x, \Delta y)$. The distance between the four concentric circles was equal to the distance between the particles on the circumference which was nearest to the interface. The negative particles were placed inside the interface in the same way. The initial distribution of the particles are shown in Figure 4.4(a), in which positive particles are shown in red solid points and negative parti-
Figure 4.3: Circularity for the rising bubble problem with $h = 1/80$.

circles are shown in blue circles. The real solution of the interface is represented as a solid line.

From Figure 4.4, we can observe that after some time, positive particles and negative particles move in different directions. Especially negative particles concentrating on the center part of the interface had a tendency to move upwards. They formed a different distribution shape as the positive particles. Obviously the particles did not follow the interface very well and thus they could not track it correctly. Then after the error correction procedure, a number of particles which were far away from the correct interface were deleted, which potentially also affected the accuracy of the solution.

It is clear from this little experiment that for “real” two-phase flow problems the convecive velocity field does not lead to nice particle distributions as for the test problems described in Chapter 3. The main reason for this difficulty is that the velocity field varies quite significantly as particles move away from the interface. One possible remedy to this problem is to construct a so called extension velocity by computing
\[ \nabla \phi \cdot \nabla F_{ext} = 0 \]

with boundary condition

\[ F_{ext} = \vec{u} \cdot \vec{N} \]

at \( \phi = 0 \). The extension velocity \( F_{ext} \) is thus equal to \( \vec{u} \) at the interface and is equal to the normal velocity at the closest point of the interface everywhere else. In this way the particles close to the interface will move at the same speed as the interface and thus stay close to it allowing for more accurate corrections. \( F_{ext} \) can be calculated as a substep in the fast marching procedure or directly evaluating the normal velocity at the closest point on the interface.
Figure 4.4: Convection of the particles for the rising bubble problem.
Chapter 5

Conclusions

This section summarizes the results obtained from the numerical simulations solved by different interface tracking methods.

In order to compare the capability of different methods tracking a moving and deforming interface, four test cases are presented and served as benchmark problems. The computations are carried out using three different methods, i.e., VOF, the level set method and the particle level set method.

From the error results and solution figures, we can conclude that the level set method performed better and obtained a more accurate solution than the VOF method for simple translation and solid body rotation problems. The results exhibited good mass conservation for both methods because of the lack of topology changes. For the single vortex problem, the level set method still kept its advantage over the VOF method although it began to lose mass due to numerical diffusion in the distance function solution. For the deformation field problem, the performance of the level set method fell short of the VOF method because of the severe mass loss.

Compared with the VOF method, the level set method is accurate, robust and treats topology changes (tearing or merging of interface) naturally. One difficulty with the use of the level set method is the need to control the numerical diffusion (mass loss) present in the method, especially in areas of high curvature and long, thin filamentary regions.
For the four test cases, the hybrid particle level set method has been shown to provide superior results. Our results indicate that the particle-based method is the most accurate of those tested. The robustness of the level set method is maintained by the particle level set method since the marker particles locally capture the location of the interface through the $\phi_p$ function, and the level set function itself is used to automatically treat connectivity (merging and pinching of fronts). The ease-of-implementation of the level set method is also maintained since the particles are disconnected and communicate with the level set function only during the error correction stage described in Chapter 3. Since the particles are placed within a band about the $\phi = 0$ isocontour, the interface is resolved on multiple scales by the particles. This multi-resolution approach is successful in preserving the volume of the level set when the interface undergoes large amounts of stretching induced by an incompressible flow field. With respect to computational cost, the most expensive is the particle-based method, followed by the level set method and the VOF method. However, considering the accuracy that the particle level set method can achieve, the additional cost of utilizing particles is offset by the ability to use much coarser grid sizes without sacrificing a faithful representation of the interface, as seen in the single vortex and deformation field test cases.

Although the particle level set method performed well for the four benchmark problems, we found some problems with this method that could prohibit its practical application if they are not resolved. For example, in the four test cases, we found that increasing the number of initial particles did not increase the accuracy correspondingly. The other open issues, such as the manner in which particles are introduced as well as the number of particles necessary to obtain a prescribed accuracy, also need to be resolved. In Chapter 4, we tried to apply the particle level set method in the practical rising bubble problem. The particles failed to follow the interface correctly and the simulation results were not satisfying. However, since there is no related practical fluid flow example has rigorously been investigated with the particle level set method in the literature to date, our test was still meaningful to highlight this problem. We also provided some suggestions to construct a velocity field for which particles do not stray from the interface significantly.
Declaration

I here by declare that the Master Thesis titled “Evaluation of Interface Tracking Schemes with Finite Element Discretizations” contains no material that has been submitted previously, in whole or in part. I have written this thesis without any illegal assistance of others and did not use any sources and/ or aids other than those cited in the bibliography section.

(Jing Xu)
Bibliography


