Applications of 3D Front Tracking to Multi Phase Fluid

A Dissertation Presented

by

Wurigen Bo

to

The Graduate School in Partial Fulfillment of the Requirements for the

Degree of

Doctor of Philosophy

in

Applied Mathematics and Statistics

Stony Brook University

August 2009

State University of New York
at Stony Brook

The Graduate School

Wurigen Bo

We, the dissertation committee for the above candidate for the Doctor of Philosophy degree, hereby recommend acceptance of this dissertation.

James Glimm
Professor of Applied Mathematics and Statistics
Dissertation Advisor

Xiaolin Li
Professor of Applied Mathematics and Statistics
Chairman of Defense

Roman Samulyak
Associate Professor of Applied Mathematics and Statistics
Co-Advisor

Foluso Ladeinde
Professor of Mechanical Engineering
Outside Member

This dissertation is accepted by the Graduate School.

Lawrence Martin
Dean of the Graduate School

Abstract of the Dissertation

Applications of 3D Front Tracking to Multi Phase Fluid

by

Wurigen Bo

Doctor of Philosophy

in

Applied Mathematics and Statistics

Stony Brook University

2009

We make improvements for the 3d front tracking method. First, we apply a 3 component grid based reconstruction method to solve the moving surface-solid wall interaction problem. A moving contact curve on the wall is explicitly tracked. A drop falling problem is studied by the new algorithm. Secondly, we present an improved, robust, locally grid based method for reconstruction of tangled interfaces. This method improves the handling of topological change of the surface mesh in the 3D simulations. The primary breakup of a high speed jet is studied numerically in 3D using the front tracking method. The breakup in the liquid jet is presented in the simulations. The nozzle flow is also studied to determine the cavitation within the nozzle and the level of turbulence occurring at the nozzle exit. The turbulence intensity from the simulations is compared with experimental and theoretical results. We also apply 3D simulations to the hydrodynamic and MHD processes in a liquid mercury target for the Muon Collider/Neutrino Factory, which include a mercury jet interacting with protons in a longitudinal magnetic field. Surface instabilities are observed in the simulations. The stabilizing effect of magnetic field on the growth of filaments are observed. The growth of filaments are compared with experimental results.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Many methods have been proposed to evolve the fluid-gas interface; of these VOF, level set and front tracking methods are the most popular. A complete review is beyond the scope of this thesis. Readers are referred to the papers of Sethian [52], Scardovelli and Zaleski [45], Glimm *et al.* [11, 13], and Tryggvason *et al.* [54].

We use front tracking methods to solve the free surface flow. Front tracking has many advantages for the problems dominated by a geometrically complex and dynamically moving interface. It was used to simulate the Rayleigh-Taylor instability and gave impressive results [33, 34], including agreement with experiments in the overall growth rate as defined by the mixing growth parameter α. Since moving surfaces are tracked by marker particles, front tracking methods are able to model the interface accurately without any numerical diffusion across the interface, in contrast to capturing methods [9]. They also differ from marker particles methods in that particles are located only on the interface, rather than in a volume region near the interface. Geometrical information such as the surface normal and curvature is also easily computed in front tracking methods, as are surface related physical processes such as surface diffusion, surface tension, and surface mediated chemical reactions, because of the explicit representation of the interface by its own mesh.

The front tracking method is implemented in the code *FronTier*. In this chapter, we begin with a background review of interface tracking methods, the front tracking in *FronTier*.

1.1 Front Tracking Method in ***FronTier***

The front tracking method is an adaptive computational method that provides sharp resolution of a wave front by tracking the interfaces between distinct materials. It represents interfaces explicitly as lower dimensional meshes moving through a rectangular grid. The states of fluids are located in the centers of each grid cell. Extensive work on front tracking method and its application in 2D space has been done by J. Glimm and his coworkers [15]. Great effort has been made for its extension to 3D space [12, 13]. The major challenge in 3D front tracking lies in the maintenance of the evolving fluid interface, which requires the ability to detect and resolve changes in the topology of a moving interface. The method has been implemented in the code *FronTier*. We will give a brief introduction of the *FronTier* code in this section.

1.1.1 Equations of the System

In *FronTier* The states in fluid are solved by conservation laws. The basic variables are density, momentum and total energy. The conservation laws for these variables can be written as

$$\mathbf{U}_t + \nabla \cdot \mathbf{F}(\mathbf{U}) = \mathbf{S}\ . \tag{1.1}$$

where

$$\mathbf{U} = \begin{pmatrix} \rho \\ \rho\mathbf{v} \\ \rho E \end{pmatrix} \qquad \mathbf{F}(\mathbf{U}) = \begin{pmatrix} \rho\mathbf{v} \\ \rho\mathbf{v}\mathbf{v} + p\mathbf{I} \\ \rho E + p\mathbf{v} \end{pmatrix}\ , \tag{1.2}$$

are the conservative variables and their corresponding flux. ρ, the density, $\mathbf{v}$, the velocity, $E = e + \mathbf{v} \cdot \mathbf{v}/2$, the specific total energy, e, the specific internal energy. p, the pressure, $\mathbf{I}$, an identity matrix. $\mathbf{S}$ represents the source terms for the equations which are used to model many physical effects such viscosity, gravity, mass diffusion and heat transfer. To close the system of equations, an equation of state $p = p(e, \rho)$ must be given. There can be moving surfaces inside the domain which represent physical waves such as contacts, shocks or the edges of rarefaction waves.

The equations (1.2) need to be written in a coordinate system. Far away from a surface, we write the equations (1.2) in a Cartesian coordinates system and seek a weak solution for the equations. On a surface, by assuming the surface is second order differentiable, we can find a local orthogonal coordinate system which has a normal direction $\mathbf{N}$ and tangential directions $\mathbf{T}$. the equations (1.2) can be rewritten in the coordinate system as

$$\mathbf{U}_t + \nabla_{\mathbf{T}} \cdot \mathbf{F}(\mathbf{U}) + \nabla_{\mathbf{N}} \cdot \mathbf{F}(\mathbf{U}) = \mathbf{S}_{\mathbf{T}} + \mathbf{S}_{\mathbf{N}} \ , \tag{1.3}$$

where the $\nabla_{\mathbf{T}}$ and $\nabla_{\mathbf{N}}$ represents the normal and the tangential components of the divergence operator ∇. $\mathbf{S}_{\mathbf{N}}$ and $\mathbf{S}_{\mathbf{T}}$ represent the normal and the tangential parts of the source term $\mathbf{S}$.

1.1.2 Representation of Interface

In *FronTier*, a package called the interface library is used for the description and manipulation of interfaces. Details about this package can be found in [12–14]. We only give a brief summary of some basic terminology here.

The discontinuity in the numerical solution is described by an interface, which is a set of discrete representations of points, curves and surfaces. The boundaries of surfaces are curves, while boundaries of curves are called nodes. A curve is comprised of connected line segments. Each line segment in the curve is called a bond, which are connectors between two adjacent points. A surface is a connected oriented piecewise linear collection of triangles; each triangle contains 3 points. Both bonds and triangles are linking objects in the sense that they contains pointers to their neighbors. Each bond points to both the previous and following bonds that share its endpoints. Similarly, triangles share a common side with their neighbors and contain pointers to that neighbor's address. During front propagation, front intersections are produced due to wave interactions and require special treatment to resolve the interaction and untangle the interface. Figure 1.1 gives the representation of Interface in *FronTier*.

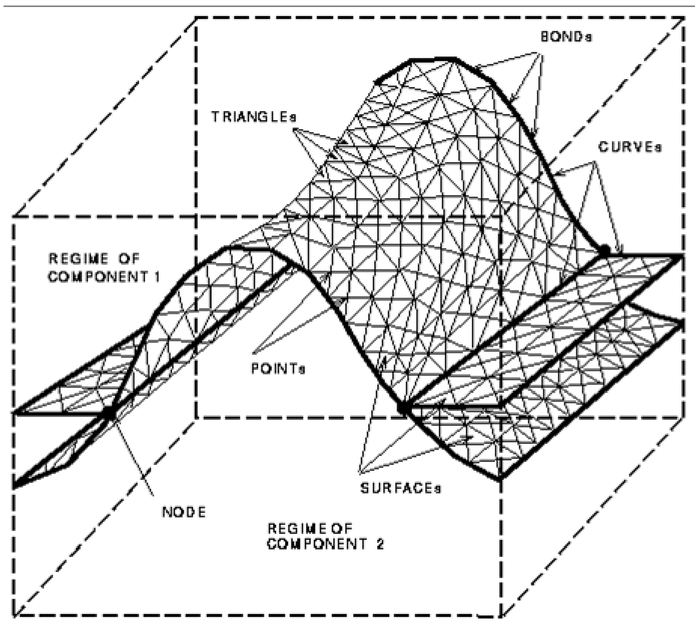

Figure 1.1: The representation of interface in *FronTier*

For the 3D flows considered here, we assume the interface is represented by a triangular mesh which is embedded in a rectangular domain. The interface divides the domain into a set of domains. The interface together with the fixed boundaries of the rectangular domain form the boundaries of domains. Each domain is assigned by a component which is determined from the orientation of the surfaces in the interface. Topological consistency requires that all components be identical for every surface side bounding a given domain.

1.1.3 Front Tracking Method in *FronTier*

Front tracking method is implemented in *FronTier* by the following main steps

1. interface propagation.
2. interpolation reconstruction.
3. interior states update.

In the interface propagation step, a local Riemann problem is solved with initial states interpolated from either side of the interface point. Then, a new position for the interface point is determined by using.

$$\mathbf{x}^{n+1} = \mathbf{x}^n + V\Delta t\mathbf{n}. \tag{1.4}$$

where $\mathbf{x}^n, \mathbf{x}^{n+1}$ are the old and the new positions of the point, V is the wave speed, $\mathbf{n}$ is the normal direction on the point and Δt is the time step size. The left and the right states on the interface are updated by the solution of the Riemann problem. Then the following equations on the tangent plane on each side of the interface are solved.

$$\mathbf{U}_t + \nabla_{\mathbf{T}} \cdot \mathbf{F}(\mathbf{U}) = \mathbf{S}_{\mathbf{T}} \ . \tag{1.5}$$

In the equations 1.5, we update the states on the interface without changing the positions of each point.

After all points in the interface are propagated, we get a new interface at a new time level. The interface may be badly distorted or have self-intersections. Methods are implemented in *FronTier* to resolve the topologically change and to optimize all triangles in the interface: grid free tracking, grid based tracking and locally grid based tracking. The details of the algorithm will be given in the following chapters.

In interpolation reconstruction step, the components in the cell center of all interior points are determined from the propagated interface. First, the crossings between the underlaying Cartesian grid and the interface are calculated. Then, the components in cell centers are determined from these crossings. During this step, a ray casting algorithm in x, y and z directions is applied to find unphysical crossings. After this step, A component is uniquely assigned in each cell center.

In interior states update step, A Strang splitting is applied and three 1D equations are solved consecutively. All the states on cell centers are updated by a finite difference scheme. If the stencil of the finite difference scheme does not cross any interface, the states in the stencil are given by the cell center values. Otherwise, a ghost cell method [16] is used to fill the states on the points on the other side of the interface. A new conservative front tracking scheme [32] has been implemented in *FronTier* recently, which eliminates the need for ghost cells.

1.2 Dissertation Organization

The rest of my thesis is organized as follows: In Chapter 2 We present two improvements for the 3d front tracking methods. We first briefly describe three existing methods: grid free tracking, grid based tracking and locally grid tracking, then we present the improved locally grid based method and prove the newly implemented method ensure a topologically valid interface. We also use a 3 component grid based reconstruction to solve the problems which have a moving surface and a wall. In Chapter 3 we explore the applications of our method in the numerical study of diesel spray. In Chapter 4, we present the study of numerical simulations for proton-mercury jet interaction.

Chapter 2

Improvements for 3D Front Tracking Methods

In this chapter, we first introduce a robust locally grid based method. We prove that the reconstructed interface is always topologically valid. Then we use a three component grid based front tracking method to solve the moving surface and wall interaction problems.

2.1 Robust Locally Grid Based Method

We are using the 3D front tracking method [9] in which the phase boundary between the liquid and gas is tracked by a triangulated mesh. In each time step the code first propagates all the points on the phase boundary. Then the code redistributes the triangular mesh to maintain the mesh quality and resolves all the tangled parts of the mesh.

2.1.1 The Existing 3D Front Tracking Methods

We have three existing 3D interface tracking methods in *FronTier*. The point propagation part is common for all three methods, the equation (1.4) is used to propagate all the points on the interface. *FronTier* has three different ways to optimize the propagated interface.

Grid Free Method

The grid-free tracking method consists of two parts, interface redistribution and interface untangle.

Interface redistribution

Similar to ALE methods, periodic redistribution of the interface is used to maintain uniformity of the triangular mesh. At a frequency specified by users, the triangles with size out of range (either too large or too small) or with a bad aspect ratio with respect to a set of user defined tolerances are put into a queue. Several basic operations (splitting, flipping, deleting) are applied iteratively to the triangles in the queue until no triangles need to be redistributed. During each iteration: (1) large triangles are divided by splitting them along their longest side. At the same time the neighboring triangle (or bond if the split side lies on a surface boundary curve) is also split. The split triangles are removed from the queue, and the resulting new triangles are added to the processing queue. (2) The adjacent pairs of small triangles with a common short side will identified and the indicated side is collapsed and the degenerated triangles are deleted. (3) An additional elementary operation is to flip the interior edge of two adjacent triangles (which thus forms a diamond), so that the diamond with its four vertices is not changed, but its connectivity is reversed. In practice the processing queue will be empty after a relatively few iterations.

Grid-free untangle

The second part of the grid-free algorithm is to resolve any intersections that have been produced during front propagation. It contains three main steps: (1) use a robust and efficient triangle intersection detection algorithm to find the intersecting triangle pairs [18]. A topological grid is introduced to avoid the check for intersecting triangles that are spatially distant from each other. This considerably speeds up the algorithm. (2) retriangulate intersected triangles. For each intersecting triangle on the two intersecting surfaces, the intersection divides it into two polygonal parts each bounded by the bonds of the crossing curve and the original triangle sides. We triangulate these polygonal parts using the constrained planar Delauney triangulation method. For the details of this algorithm please refer to the paper of Chew [6]. (3) delete the unphysical surfaces. After the first two steps, we can get an untangled interface that satisfies all the requirements for a valid interface except for the consistency of its embedding into the computational domain. That is because of the surfaces meeting along the intersection curves, the components of the common side of the surfaces meeting at this curve are inconsistent. It is thus impossible to assign components to the regions of the computational domain defined by this interface. The only thing needed to get a consistent untangled interface is to delete the unphysical surfaces. Methods are designed to identify the unphysical surfaces and delete them [13].

Grid Based Method

Another method to resolve changing interface topology is to reconstruct the interface by using the underlaying rectangular grid block cells. The scheme is divided into three steps: (1) compute the crossings of the interface and the grid block edges. At each crossing of the interface and the grid block edges, we assign components on both sides of the interface. (2) Determine components at the grid block corners and eliminate inconsistent crossings. The crossings divide each edge into a set of subintervals. In regions where the interface is tangled, some subintervals will have different components at their two endpoints. We process each grid block edge to eliminate crossings that produce inconsistent components on the grid corners. (3) Reconstruct a new interface using the remaining interface crossings. The reconstruction process consists of two steps: The reconstruction of an interface segment within a grid cell and the assembling of the single block surface elements into global surfaces. A detailed description of this method can be found in [13].

Locally Grid Based Method

Both the grid-free and the grid-based method described above have advantages as well as deficiencies. The grid-free method produces a high quality distribution of triangle size shapes and accurately controls numerical diffusion. It suffers from being complex and subject to failure when the interface is complex. The grid-based method is over-diffusive, it tends to over-smooth the interface and produces poorly conditioned triangles due to the constraint of reconstructing the surface within a single grid block. On the other hand, it is quite robust and always reconstructs a topologically valid interface.

To reduce the GB interpolation error, locally grid based tracking has been proposed, which combines the advantages of both methods. The LGB algorithm thus identifies some bad triangles of the propagated surface. It isolates these, and preserves the intersections of the surface with the grid cell edges to allow a GB reconstruction locally near the bad region. Triangles neighboring the bad region are removed, and so there is a gap, separating the good part of the interface from the reconstruction of the bad part of the interface. The major step is then to re-seal this gap. The details of the method can be found in [9].

2.1.2 Robust Locally Grid Based Method

The locally grid based method reduces the use of the Eulerian reconstruction to a minimum. It reduces interface interpolation errors and minimizes the unphysical disappearance of consistent components of the material after bifurcation. However, the correctness of the method is not proved and it can also generate a topologically invalid interface when the interface is very complex. When LGB fails to untangle the interface. the GB reconstruction is used to keep the code running robustly. For the primary breakup of jet we are studying, very thin filaments and small droplets are continuously generated from the jet core. The GB reconstruction greatly reduces the resolution of the interface and thus dumps the growth of filaments. We need a robust locally grid based method which always generates a topologically valid interface. In this section, We first describe the topology of a surface mesh. Then we introduce our method.

Topology of a Surface Mesh

We first introduce our notations on a triangular mesh. Let P be a set of points in 3D

$$P = \{\mathbf{p} | \mathbf{p} \in \mathbb{R}^3\} \ . \tag{2.1}$$

A triangle t is defined as a point set comprised of three points for P

$$t = \{\mathbf{p}_k, k = 1, 2, 3 | \mathbf{p}_k \in P\} \ . \tag{2.2}$$

A surface S is represented by a triangular mesh which is defined as a set of triangles. We use T to denote the set of triangles on a surface S.

To study the topology of a surface mesh, we introduce the following two definitions. A triangle $\hat{t}$ is the *neighbor* of another triangle t if and only if they have a common edge: $\exists \mathbf{p}_1, \mathbf{p}_2$, such that $\mathbf{p}_1, \mathbf{p}_2 \in t$ and $\mathbf{p}_1, \mathbf{p}_2 \in \hat{t}$. We use the notation $t \sim \hat{t}$ if $t, \hat{t}$ are neighbors. Given a triangle t on the surface and a point $\mathbf{p}$ on the triangle, *the triangle list around* $\mathbf{p}$ is defined as a set of triangles $T(t, \mathbf{p}) = \{\hat{t}\}$, where $\hat{t} \in T$

satisfies: $\mathbf{p} \in \hat{t}$ and $\exists t_k \in T, k = 1, 2, \cdots n$, such that $t \sim t_1 \sim t_2 \sim \cdots \sim t_n \sim \hat{t}$.

In order to keep the triangular mesh topologically valid, we have three assumptions for the set of triangles T.

(1) For any $t \in T$, there is one and only one neighboring triangle on each side of t.

(2) For any $t \in T$ and $\mathbf{p} \in t$, the triangle set $\{\hat{t} | \hat{t} \in T, \mathbf{p} \in \hat{t}\}$ forms only one triangle list around $\mathbf{p}$.

(3) The surface constructed by T is an orientable surface.

Figure 2.1 shows two examples of the invalid cases. Under the first two assumptions, the surface S constructed by the triangle set T represents a 2D-manifold. We can prove this by checking if all points on S is homeomorphic to an open subset in a 2D plane. For any point $\mathbf{p}$ on S, if $\mathbf{p}$ is in the interior of a triangle t, a neighbor of $\mathbf{p}$ must lie in the 2D plane expanded by t. If $\mathbf{p}$ is on an edge or on a vertex of a triangle, the assumptions (1) and (2) ensure there exists a neighbor of $\mathbf{p}$ which is homeomorphic to an open subset in a 2D plane. With the third assumption, we can define the orientation of each triangle. We then arrange three points in a triangle in an order such that $\mathbf{p}_1\mathbf{p}_2 \times \mathbf{p}_2\mathbf{p}_3$ points to the positive side of the surface.

When some triangles are removed from T, some of the remaining triangles in T may not have neighboring triangles. We define the *boundary of T* to be a set of orientated line segments as follows

$$B(T) = \{\mathbf{p}_a\mathbf{p}_b \mid \mathbf{p}_a, \mathbf{p}_b \in t,\ t \in T,\ t \text{ has no neighboring triangle on side } \mathbf{p}_a\mathbf{p}_b\} \quad (2.3)$$

We consider $B(T)$ as a directed graph [23] where the vertices and the directed line segments in $B(T)$ are the vertices and the edges of the graph. To further study the properties of $B(T)$, we introduce some concepts of a graph. For a vertex, the number

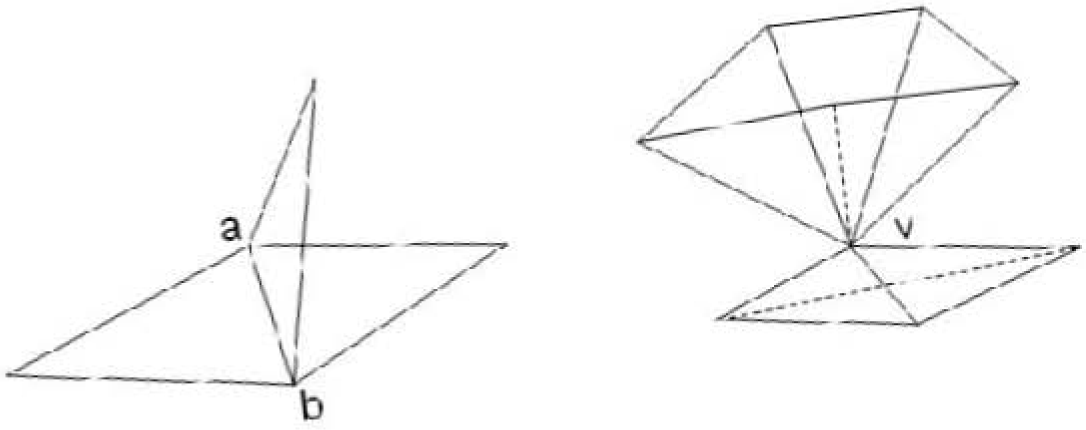

Figure 2.1: Two examples of invalid cases. In left figure, edge *ab* is associated with 3 triangles and assumption (1) is violated. In the right figure, vertex **v** is associated with two triangle lists and assumption (2) is violated.

of head endpoints adjacent to a vertex is called the *indegree* of the vertex and the number of tail endpoints is its *outdegree.* A *simple circle* is a graph with no other repeated vertices than the starting and ending vertices. A *directed simple circle* is a directed Eulerian circuit which has no repeated vertices aside from the start/end vertex. A *directed Eulerian path* is a path in a graph which visits each edge exactly once. A *directed Eulerian circuit* is a directed Eulerian path which starts and ends on the same vertex. Euler's theorem [23] can be used to check if a directed graph has a directed Eulerian circuit: If a directed graph is connected and every vertex has the same indegree and outdegree, then it at least has one Eulerian circuit. Using Euler's theorem, we can prove the following theorem.

Theorem 2.1.1 *T is a set of triangle which satisfies the assumptions (1)-(3). T_1 is a subset of T which contains a finite number of triangles. Then, $B(T \setminus T_1)$ has a directed Eulerian circuit.*

Proof: We use mathematical induction. The number of triangles in set T_1 is n. If

$n = 1$, the conclusion is true. Let t be a triangle in T_1 and T_2 be a set of triangle defined as $T_2 = T \setminus (T_1 \setminus t)$. Then, $B(T \setminus T_1) = B(T_2 \setminus t)$. Assuming $B(T_2)$ has a directed Eulerian circuit, we will show that $B(T_2 \setminus t)$ also has a directed Eulerian circuit. We count the indegree and the outdegree of a vertex $\mathbf{p}_2 \in t$. There are three cases:

(1) If t has two neighboring triangles on the edge $\mathbf{p}_1\mathbf{p}_2$ and $\mathbf{p}_2\mathbf{p}_3$, both the indegree and the outdegree of $\mathbf{p}_2$ increase by one after t is removed from T_2. Thus $\mathbf{p}_2$ has the same indegree and outdegree after t is removed from T_2.

(2) If the edge $\mathbf{p}_1\mathbf{p}_2$ ($\mathbf{p}_2\mathbf{p}_3$) has a neighboring triangle but the edge $\mathbf{p}_2\mathbf{p}_3$ ($\mathbf{p}_1\mathbf{p}_2$) does not, the indegree and the outdegree of $\mathbf{p}_2$ is unchanged after t is removed from T_2.

(3) If t has no neighboring triangles on the edge $\mathbf{p}_1\mathbf{p}_2$ and the edge $\mathbf{p}_2\mathbf{p}_3$, the indegree and the outdegree of $\mathbf{p}_2$ is decreased by one after t is removed from T_2. Thus $\mathbf{p}_2$ has the same indegree and outdegree after t is removed from T_2.

Similarly, we can show that the indegree and the outdegree of $\mathbf{p}_1$ and $\mathbf{p}_3$ are equal after t is removed from T_2. From Euler's theorem, $B(T_2 \setminus t)$ has a directed Eulerian circuit.

Robust Locally Grid Based Method

We introduce an improved LGB reconstruction of the interface to handle topological changes. The essential feature of the improvement is to reduce the size of the GB region, which previously was the smallest rectangular solid containing a tangled set; and in the case of overlaps, the smallest rectangular solid containing the overlap, etc. This improvement is required in our study of the atomization process, as it turned out that the recursively defined bounding rectangular solids were typically too large.

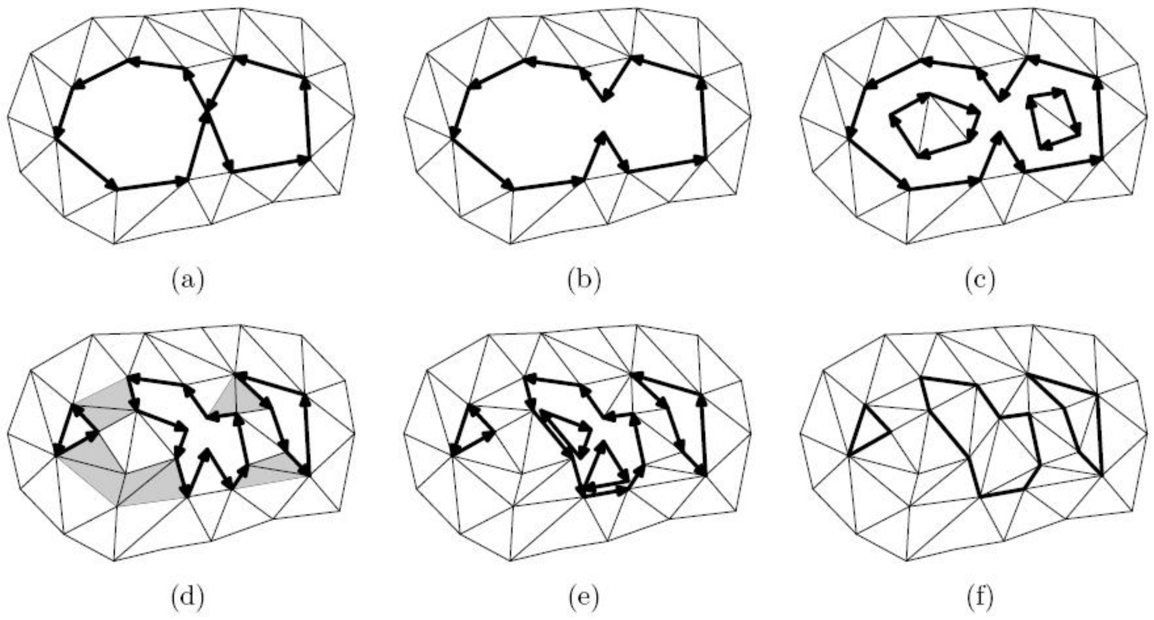

Figure 2.2: The substeps of the LGB resealing algorithm.

We describe the main procedures for the robust LGB reconstruction algorithm. See Figure 2.2 for a schematic description. The moving interface is represented by a surface S comprised of triangles. B is a 3D rectangular box. S is tangled inside box B. Let S_1 be a subsurface of S which is comprised of all the triangles inside or intersecting B.

(a) S_1 is removed from S. We rename S to stand for $S \setminus S_1$. Polygonal holes form on S after this step.

(b) If a vertex is common for two or more polygonal holes of S, the vertex is separated so that the polygonal holes are merged. This step is repeated until there are no common vertices for the polygonal holes. We use B_1 to denote the set of the sides of the polygonal holes after this step.

(c) S_1 is reconstructed in B using the GB algorithm. We use S_2 to denote the reconstructed surface. The boundary of S_2 consists of polygons due to the GB reconstruction [13]. The set of the sides of the polygons are denoted as B_2.

(d) A new surface S_3 is constructed to connect S and S_2 using the following method: For any two sides $\mathbf{p}_1\mathbf{p}_2 \in B_1$ and $\mathbf{p}'_1\mathbf{p}'_2 \in B_2$, we make a pair of triangles $\mathbf{p}_1\mathbf{p}_2\mathbf{p}'_1$ and $\mathbf{p}'_2\mathbf{p}'_1\mathbf{p}_2$ and add them to S_3 if the following two constraints are satisfied.

   (i) There exists three vertices out of $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}'_1, \mathbf{p}'_2$ which lie in the same triangle from S_1.

   (ii) If triangles $\mathbf{p}_1\mathbf{p}_2\mathbf{p}'_1$ and $\mathbf{p}'_2\mathbf{p}'_1\mathbf{p}_2$ are added to S_3, the connected surface $S \bigcup S_2 \bigcup S_3$ satisfies the constraints (1) and (2).

   This step is repeated until no new pair of triangles can be added to S_3.

(e) S, S_2 and S_3 are connected to form a new surface S_4. For each polygonal hole on S_4, if two consecutive sides are two sides of a triangle, the triangle is removed from S_4.

(f) Each polygonal hole on S_4 is sealed by a constrained triangulation considering the constraints (1) and (2). The triangulation only adds new triangle edges and does not add new vertices. We use S_5 to denote the resulting surface.

Let T_1 be the set of triangles on the surface S_1 and T_5 be the set of triangles on the surface S_5. We will show that if T_1 satisfies the assumptions (1)-(3), T_5 also satisfies the assumptions (1)-(3). In other word, if the input surface is a 2D manifold, the output surface is also 2D manifold.

We first study the surface S_2. Let T_2 be the set of triangles in surface S_2, we will show $B(T_2)$ only consists of directed simple circles. If S_2 is constructed inside only one grid cell, there are only 13 cases [13]. For each case, it is easy to check $B(T_2)$ only consists of directed simple circles. If S_2 is constructed inside two adjacent grid cells, we count the indegree and the outdegree on the common edges of the two grid cells. There are only 4 different configurations on the common face of the two grid cells, see Figure 2.3. In each case, the indgree and the outdegree are one on all the vertices on the common edge of the two grid cells. Therefore $B(T_2)$ only consists of directed simple circles. If T_2 is constructed from more than two grid cells, we can show that $B(T_2)$ only consists of directed simple circles by using the same method.

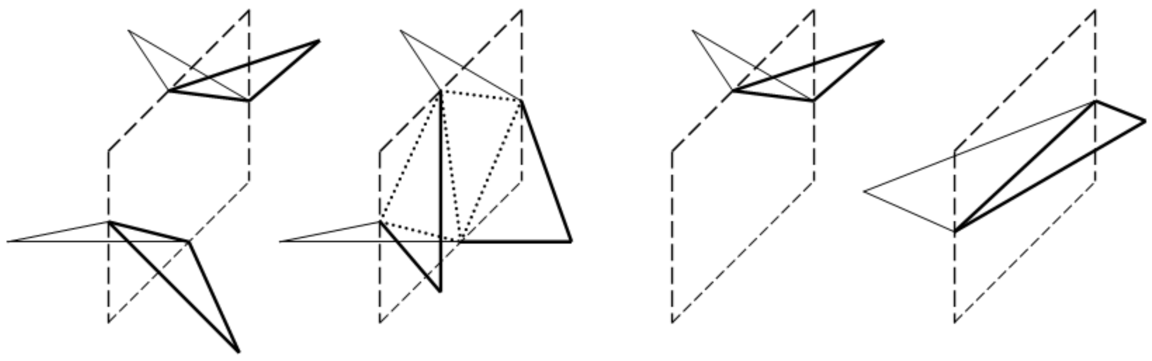

Figure 2.3: Four configurations on the common face of two adjacent grid cells from grid based reconstruction. The dashed lines are the edges of the the common face. The thin solid lines are the edges of the triangles on the first grid cell. The thick solid lines are the edges of the triangles on the second grid cell. The dotted lines are the edges of the triangles on the common face.

The robustness of the algorithm means all the operations in the algorithm are realizable and the resulting surface is topologically valid. The following theorem shows the improved LGB algorithm ensures the robustness of the algorithm.

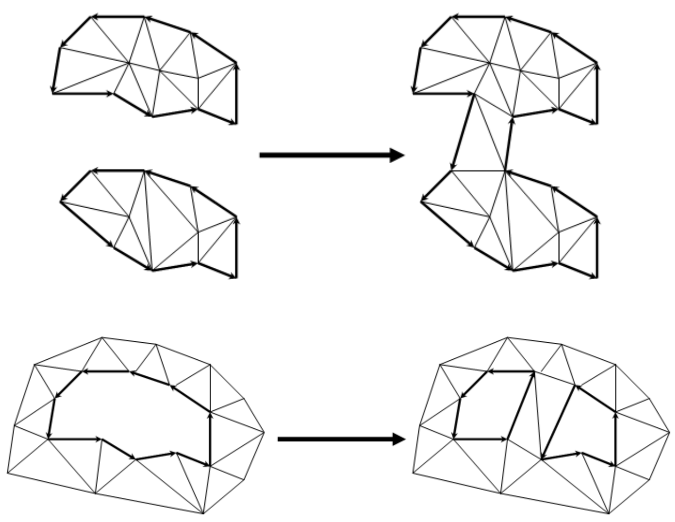
Figure 2.4: The two cases in the step (d) of the LGB reconstruction algorithm, Upper: After a pair of triangles is added, two directed simple circles are merged into one. Lower: After a pair of triangles is added, one directed simple circle becomes two.

Theorem 2.1.2 *Let T be the set of the triangles in the surface S which satisfies the assumptions (1)-(3). The LGB algorithm (a)-(f) reaches the step (f). Moreover, let T_5 be the set of the triangles of the surface S_5. T_5 also satisfies the assumptions (1)-(3).*

Proof: Let T_1 be the set of the triangles in the surface S_1. After step (a), $B_1 = B(T \setminus T_1)$ which contains a directed Eulerian circuit from theorem 2.1.1. For each vertex $\mathbf{p}$ in B_1, $\mathbf{p}$ has the same indegree and outdegree. Let the indegree of $\mathbf{p}$ be n. If $n > 1$, $\mathbf{p}$ is separated into n different vertices $\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_n$ after step (b) and the indegree and the outdegree of $\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_n$ are one. Therefore, the indegree and the outdegree of all vertices in B_1 are one. It means B_1 contains only directed simple circles. We now consider step (c). Since B_2 is the boundary of the set of the triangles of the grid based surface S_2. B_2 only contains directed simple circles from the above discussion. In step (d), let the set of the triangles of $S \bigcup S_2 \bigcup S_3$ be

T_3. At the beginning of step (d), $T_3 = \emptyset$, thus $B(T_3)$ contains only directed simple circles. After a pair of triangles t_1 and t_2 are added into T_3 at step (d), it connects two directed simple circles into one simple circle or separates one directed simple circle into two directed simple circles, see Figure 2.4. The procedure is repeated in step (d). Therefore $B(T_3)$ consists only directed simple circles after step (d). Step (e) only changes the number of edges in each directed simple circle. The boundary of S_4 consists only directed simple circles, we can perform a constrained triangulation on these directed simple circles. Since no edge is connected with more than 2 triangles in each substep and no vertex is connected with more than 2 triangle lists after step (b), T_5 also satisfies the assumptions (1)-(3).

2.2 Three-Component Grid Based Front Tracking Method for Moving Interface

In this chapter, we use a three component grid based reconstruction algorithm to solve the moving surface and wall interaction. The construction of 3-component problems is more complex than that of 2-component problems, where there are at most 3 surfaces and two curves in a grid cell.

In physics, when 3 different materials meet, the three different materials exist around a single curve. We call this curve a triple curve. From the viewpoint of computational geometry, when we use 2 functions $f(x, y)$ and $g(x, y)$ to divide three materials into three regions. The triple curve is the curve on the grid face which satisfies the functions of both $f(x, y)$ and $g(x, y)$. The template has been found by Jia *et.al.* [26]. He implemented a method to solve a problem which contains one fixed triple curve [26].

In this section, we first give an introduction to the 3 component reconstruction algorithm. Then, we apply the method for moving surface and wall interaction where a moving triple curve exists on a solid wall.

2.2.1 Three-Component Algorithm

Similar to 2-component grid based reconstruction algorithm, the 3-component algorithm uses the crossing points between the interface and the grid edges to reconstruct the interface. The interface is first reconstructed in each grid cell, which are then connected to the adjacent grid cells to form the entire interface. We use three colors to represent 3 different materials. Each corner of a grid cell has 3 color choices and there are totally $3^8 = 6561$ different kinds of grid cells. Considering rotation and reflction symmetry of a grid cell, we have 57 different kinds of grid cells. If two end points on an edge have different components, there is at least one crossing point on that edge. Assuming there is at most one crossing point on each edge of one grid cell, we can construct the interface in the grid cell by using these crossing points. See Figure 2.5 for an example.

The difference from the 2-component reconstruction is that a triple curve may appear after the reconstruction. We call a crossing point between a triple curve and the face of a grid cell a curve crossing point. We assume there exists at most one curve crossing on each face. Moreover we assume we get a curve crossing if and only if the following assumptions are satisfied:

(1) Exactly three distinct components are found.

(2) Two components at the diagonal positions must be different.

Under the above assumptions, the curve crossings appear in only one case: See Figure 2.6 for details.

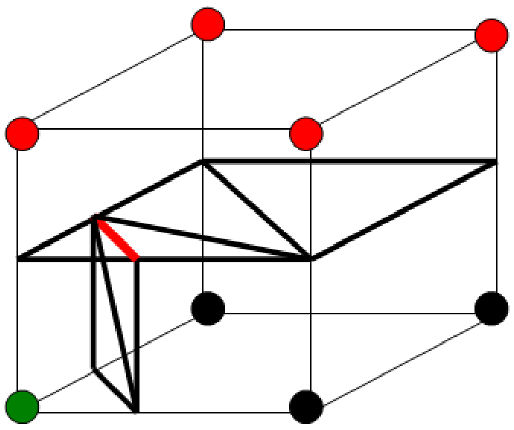

Figure 2.5: An example for the 3 component grid based reconstruction on a grid cell. The black lines are the edges of the triangles on the interface and the red line is a bond on a triple curve.

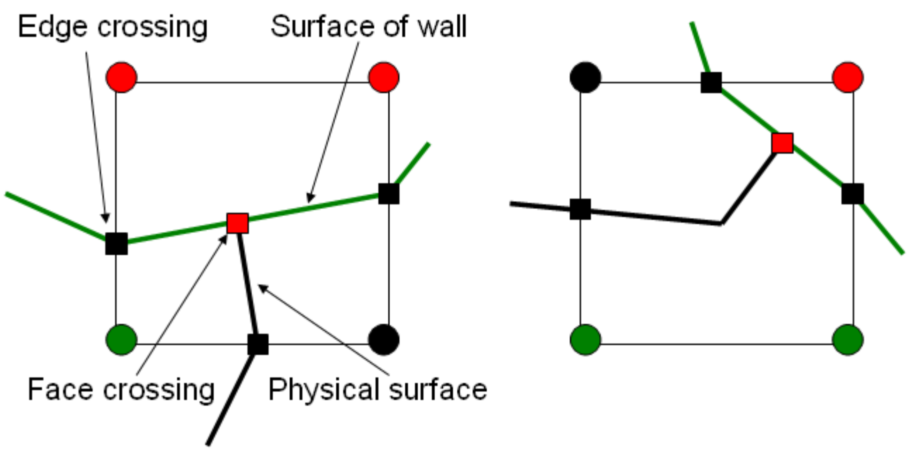

Figure 2.6: Two cases of a curve crossing point on the face of a grid cell. The green line represents the surface of a wall, the black line represents the physical surface, the red square represents a curve crossing point, the black square represents edge crossing points.

2.2.2 The Algorithm for Moving Surface-Wall Interaction

We simulate the surface-wall interaction problem. The computational domain is rectangular and there are some solid walls in the computational domain. These walls separate the computational domain into several subdomains. One subdomain contains two materials which are separated by a moving surface. The intersection between the moving surface and the solid wall forms a triple curve. When the surface is moving, the triple curve is also moving along the wall.

The 3 component algorithm can be used to handle the moving surface-wall interaction. Since there is a triple curve on the solid wall, the triple curve must be propagated along the wall. In an operator splitting scheme, the equations in the normal direction is first solved. Since the triple curve is moving along the wall, the propagating direction must be tangential to the wall. A 5-point stencil for the point propagation algorithm is schematically shown in Figure 2.7. All 5 points are on the solid wall and the 5 states $s_{-2}, s_{-1}, \cdots s_2$ come from a linear interpolation from the states on the wall. There are two tangential directions; the first is normal to the wall. The tangential point propagation in this direction is equivalent to a normal point propagation on the wall. Another direction is along the triple curve, the tangential point propagation gets states on a stencil from those on the triple curve.

The propagated interface must be reconstructed in order to resolve tangles. The summary of the algorithm implementation is as follows.

(1) All the edges on the computational domain are classified as 3 types: INSIDEWALL, ONWALL and OUTSIDEWALL. INSIDEWALL means the edge is completely on the domain occupied by the two materials. ONWALL means the edge has an intersection with the wall. OUTSIDEWALL means the edge is completely behind the wall.

(2) The crossing points between the surface which separates the two materials (we call it the physical surface) and the grid edges are computed. The components for the grid corners lying within the two materials are determined from the physical surface. Multiple crossing points are removed on each edge to ensure one edge contains only one crossing points. A ray casting algorithm is used here which is the same as that used in 2 component grid based algorithm [13] except that the algorithm is performed for all the edges labeled as INSIDEWALL.

(3) For the edges labeled as ONWALL, all the crossing points generated from the physical surface are removed, thus only the crossing points generated from the interaction between the wall and the grid remain. After this step, each grid edge has at most one crossing point on it.

(4) For each face of a grid cell, we check the 4 components on the 4 corners of the face. If the conditions (1) and (2) in Section 2.2.1 are satisfied, there must be one curve crossing point on the face. There are only two configurations for a face crossing point as is shown in Figure 2.6. The face crossing point is the point of a intersection between the triple curve and the face of a grid cell. If more than one point is found on a face, only one is picked as the curve crossing point.

(5) On each grid cell, the 3 component algorithm is used to reconstruct the triangles and the bonds on it. At common faces of the grid blocks, the reconstructed triangles are connected to form the wall surfaces and the physical surface; the bonds are also connected at the common point to form the triple curve.

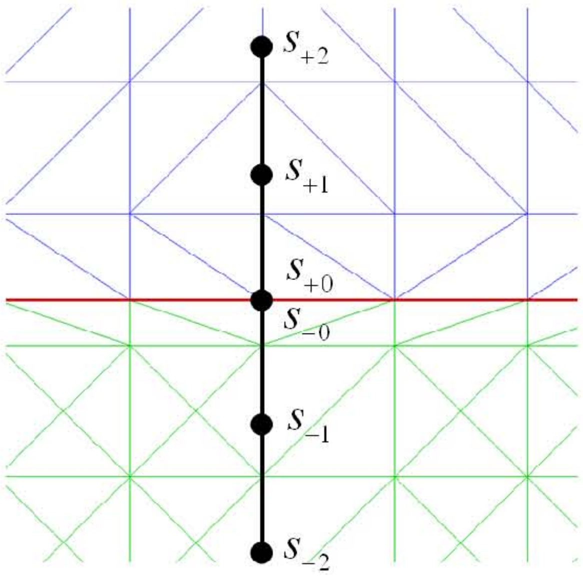

Figure 2.7: A schematic graph of a stencil for the normal point propagation algorithm for the triple curve. The red line represents the triple curve, the blue and the green triangles are the surfaces of the solid wall.

2.2.3 Numerical Test: Falling Droplet Problem

To test our algorithm for the moving surface-wall interaction. We solve a falling droplet problem. The setup of the problem is as follows. A spherical drop with a diameter $0.4cm$ is put in a rectangular container in the initial time. The system is solved in Cartesian coordinates. The dimension of the container is $[-0.5cm, 0.5cm] \times [-0.5cm, 0.5cm] \times [0, 2cm]$. The center of the droplet is $(0, 0, 1cm)$. The material of the droplet is heavy gas. The container is filled with light gas. The density ratio is $3:1$. Polytropic equation of states [37] are used to model the gas properties for both the light and the heavy gas. A constant gravity which has a magnitude $200m/s^2$ and is directed in the negative z direction is imposed on the system. The mesh is $40 \times 40 \times 80$. Figure 2.8 shows the snapshots when the droplet falls down. As the droplet hits the bottom of the container, it splashes and then hits the four vertical walls of the container. During the procedure, the light gas is trapped between the heavy gas and the container and produces several triple curves on the wall of the container. After all the kinetic energy are converted into heat, the heavy gas lies on the bottom of the container. The test problem shows our algorithm not only handles the topological change of the surface but also that of the triple curves.

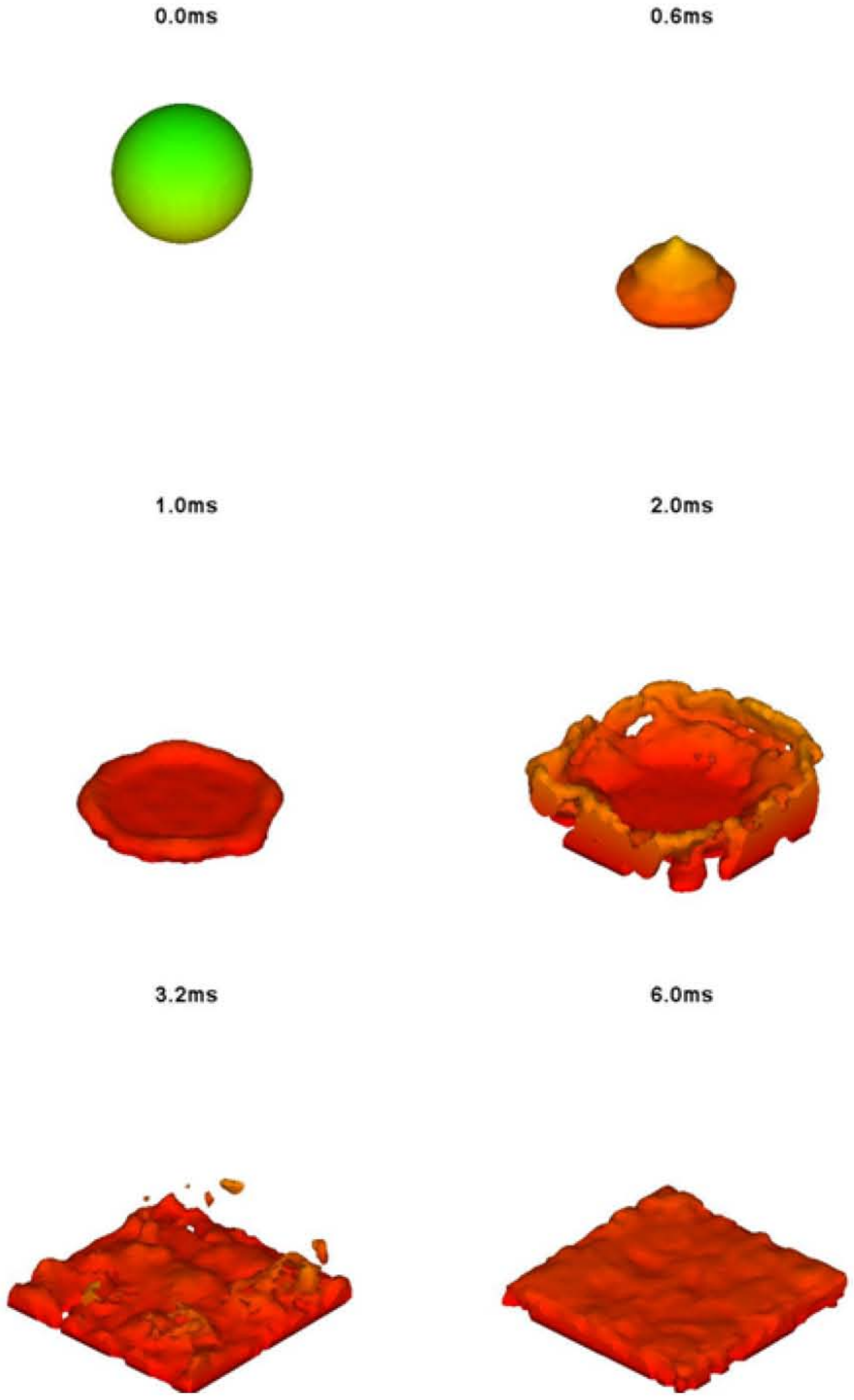

Figure 2.8: The snapshots of the falling droplet problem.

Chapter 3

Diesel Spray Simulations

3.1 Introduction

The mechanism of the atomization process of a high speed jet has been studied extensively in theory [30, 43, 44] and experiments [31, 35]. Primary jet breakup remains a challenging research topic due to the large range of scales (spatial and temporal), and the complex flow regimes involved. Experimental observations are difficult because the droplets in the spray from the breakup obscure the spray interior and the liquid core.

Numerical simulation appears to be a promising method to study the details of the breakup process of the liquid jet. However, the physical process of atomization poses difficulties for numerical algorithms. First, as the fuel has a much higher density than the ambient air, an accurate algorithm to deal with the interface between these two is required. Secondly, since the growth of perturbations on the interface is one of the important mechanisms in atomization [43], the algorithm should have small numerical diffusion so as not to suppress this process. Since one of the characteristics of atomization is topological change, such as breakup and merging of the interface, the algorithm must also resolve topological change robustly. During atomization, droplets whose diameter is much less than that of the nozzle are generated, and consequently, a high resolution algorithm is required to resolve small droplets. Third, liquid cavitation occurs inside the nozzle, mitigating pressure disturbances and influencing turbulence in the flow, as well as atomization [59].

From linear stability theory, the flow leading to liquid jet breakup is divided roughly into four parameter regions, according to the flow Reynolds number and Weber number [29]. Here we are concerned with high speed jets, well within the fourth of these regimes, called the second wind driven regime. Generally, jet breakup and atomization in this regime is caused by the growth of waves originated at the nozzle exit, which eventually break up the jet core when the amplitude reaches a certain value [30]. Since the initial disturbance of the jet is provided by the flow in the nozzle. The initial flow condition inside the nozzle also has great contribution to jet breakup [21], which include the nozzle geometry, cavitation and turbulence within the nozzle, and relaxation of the boundary layer as the fluid flows out of the nozzle.

In this chapter, we first study the flow inside the nozzle. The development of the turbulence and the cavitation inside the nozzle is investigated numerically. We then study the primary breakup of the high speed liquid jet numerically. The breakup of the jet tip is observed in our simulations.

3.2 The Nozzle Flow Simulations

We first study the nozzle flow in a 3D space. Besides the fluid properties such as density and viscosity, the flow in the nozzle is significantly affected by the geometry of nozzle. The first parameter we consider is the nozzle length L, more specially, the length and diameter ratio L/d. The second parameter we consider is the shape of the nozzle inlet. A sharp inlet corner usually produce a larger cavitation zone than does a round one [57]. If the inlet corner of our nozzle is sufficiently sharp, the flow detaches from the nozzle wall and cavitation appears in the low pressure region at the inlet corner. When the pressure difference between the nozzle inlet and outlet increases the cavitation region extends and enhances the turbulence level at the nozzle outlet. Experimental studies suggest that cavitation enhances spray breakup unless super cavitation occurs in the nozzle [21].

We model cavitations by homogeneous nucleation theory [4]. Vapor bubbles form in a region where the pressure of the liquid falls sufficiently below its vapor pressure, a value given by the tensile strength of the liquid. Two phases (liquid and vapor) are involved in the nozzle cavitation, as well as two equations of state (EOS), or two branches of a common EOS. The free surface between vapor and liquid as well as that between the liquid and the ambient gas (with its own EOS) are modeled in *FronTier*. We refer to the cavitation model proposed in [59]. This model involves two numerical parameters, the vapor bubble size and the bubble spacing at the time of bubble insertion, and one physical parameter, the critical pressure below which a vapor bubble will be inserted.

Homogeneous nucleation theory [4] is applied to determine the critical pressure $P_c(T)$. Assuming the probability for fluid to cavitate within a given time interval is one half under the critical pressure, we can compute $P_c(T)$ by using the following formula

$$P_c(T) \doteq -\left(\frac{16\pi\sigma^3}{3k_B T \ln(\Gamma_0 V t/\ln 2)}\right)^{1/2}, \tag{3.1}$$

where σ is the surface tension of the liquid, k_B is the Boltzmann's constant, T is the absolute liquid temperature, $V^{1/3}$ is the averaged distance between the centers of two bubbles, t is the duration of time considered, and Γ_0 is a factor of proportionality

Table 3.1: Summary of parameters for jet simulation.

fluid density	0.66 g/cm^3	ambient density	0.004 g/cm^3
density ratio	165	nozzle diameter	0.0178 cm
nozzle length	0.1 cm	Inflow pressure	500 bar
mesh size	2.225 μm	surface tension	$23 g/s^2$
fluid viscosity	0.013 Poise	ambient pressure	1.0 bar
Reynolds number (Re)	30,000	Weber number (We)	5.5×10^4
Ohnesoge number (Oh)	0.02		

defined as

$$\Gamma_0 = N \left(\frac{2\sigma}{\pi m} \right)^{1/2} , \tag{3.2}$$

where N is the number density of the liquid which has a unit molecules/cm^3 and m is the mass of a molecule. We consider n-heptane in our simulations and the bubble spacing is $6\mu m$, formula 3.1 gives $P_c = -50$ bar.

3.2.1 Problem Setup

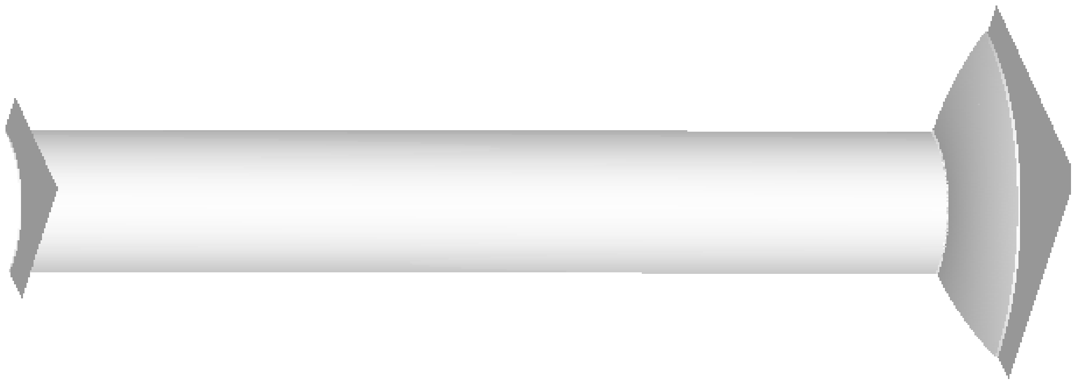

Figure 3.1: The configuration for the 3D nozzle flow simulation. A inflow boundary condition is imposed in the right side.

In our nozzle flow simulations, only a quarter of the nozzle is simulated due to

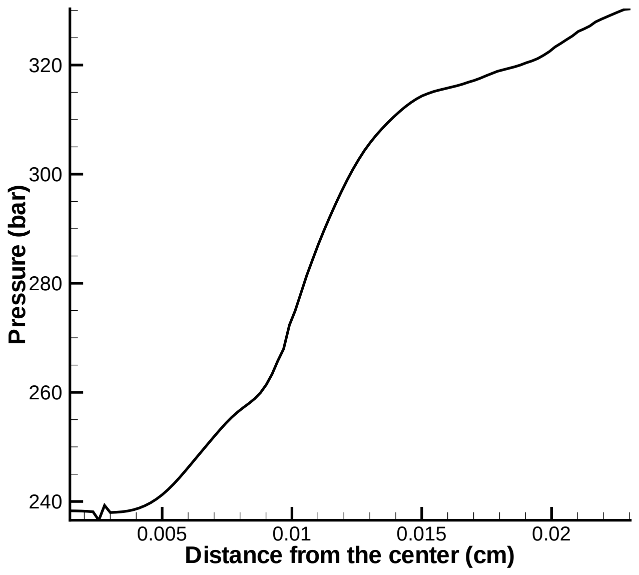

Figure 3.2: The inflow pressure profile from the 2D simulation in steady state.

the rotation symmetry of the nozzle. Moreover, the states from the 2D simulations are used as an inflow boundary conditions at the nozzle inlet. Since we want to study the nozzle flow in steady state, we use inflow data in a steady state from the 2D simulation. See Figure 3.2 for the inflow pressure profile. A no-slip Neumann boundary condition is used along the nozzle wall. Figure 3.1 shows the geometry of the nozzle. Physical parameters are given in Table 3.1. In the table, the dimensionless parameters are defined as

$$We = \frac{\rho_L U_L^2 d}{\sigma}, \quad Re = \frac{\rho_L U_L d}{\mu}, \quad Oh = \frac{\mu}{\sqrt{\rho_L d \sigma}}, \tag{3.3}$$

where ρ_L is the density of the liquid, U_L is the averaged velocity of the liquid, μ is the the liquid dynamic viscosity, σ is the surface tension coefficient between n-heptane and vapor and d is the diameter of the nozzle. The mesh in our simulations is $110 \times 110 \times 500$. We performs two simulations. In the first simulation, the flow in the nozzle is assumed to be purely liquid (the bubble insertion algorithm is turned off). In the second simulation, the discrete vapor bubble insertion algorithm is used. Usually, cavitation parameters such as the nuclei density for the bubble growth need to be determined through experiments. For the present problem we study, these data are not available. Previous simulations for 2D flow show that the flow pattern in the nozzle is insensitive to the critical bubble radius r and the bubble spacing parameter h [59]. A similar study was also performed in [49] for the case of cavitation of mercury under large external energies. The results of [49] are insensitive to the two parameters. We use a fixed critical bubble radius $r = 6$ microns and bubble spacing $h = 6$ microns which are used in the 2D simulations [55, 59].

3.2.2 Simulation Results

We first simulate the nozzle flow without the bubble insertion algorithm. We compare the streamwise velocity profiles along a radial direction just upsteam of the nozzle outlet when both 2D [55] and 3D flows achieve the steady state, see Figure 3.3. The Reynolds number $Re = 3 \times 10^4$ greatly exceeds the critical value for transition to turbulence. The boundary layer for the 3D simulation is thinner than that for the 2D simulation, and there are small perturbations in the 3D velocity profile, which implies that the 3D flow has stronger fluctuations than the 2D flow does. Two mechanisms affect the turbulence level in the nozzle outlet: The first is the velocity fluctuation produced at the sharp corner of the nozzle inlet. Since $L/d = 5$ for our nozzle, the turbulence in the nozzle outlet is greatly influenced by the sharp corner upstream [57]. The second is the development of boundary layer. The distance from the nozzle inlet to the location where transition to turbulent flow is first fully developed constitutes the theoretical initial length of the laminar flow and its magnitude is approximately

$$l = 0.03d \times Re, \tag{3.4}$$

where d is the nozzle diameter [51]. For $Re \doteq 10^4$, it is about 300 nozzle diameters which is much larger than L/d for our nozzle, transition to fully turbulent flow is not expected. The turbulence in our simulations is mainly produced by the sharp corner of the nozzle inlet.

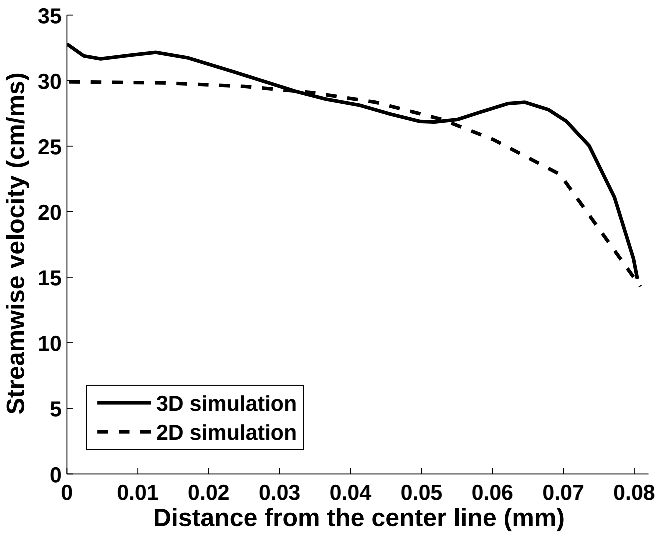

Figure 3.3: The comparison of the streamwise velocity profiles along a radial direction just upsteam the exit of the nozzle.

The difference between 2D and 3D results is expected by turbulence theory [25]. Consider the equation for vorticity,

$$\frac{\partial \mathbf{\Omega}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{\Omega} = \mathbf{\Omega} \cdot \nabla \mathbf{u} + \nu \nabla^2 \mathbf{\Omega}, \tag{3.5}$$

where $\mathbf{\Omega}$ denotes vorticity, $\mathbf{u}$, velocity and ν, viscosity. The left side of the equality represents the convection of vorticity. The first term on the right hand side represents a stretching effect which is responsible for changes in both the magnitude and direction of the vorticity vector. The second term in the right hand side represents the viscous diffusion of vorticity due to viscosity.

For the 2d flow, voriticity is directed in the z direction and the vortex stretching term is zero. The stretching effect does not act in a 2D flow. Since the viscosity is very small in our simulations, the vorticity is convected and is not changed by the flow. The initial voritcity is simply deformed by the flow in the 2D plane. This fact can be noticed in the vorticity plot in Figure 3.4. For the 3D flow, a ring of vorticity is generated at the nozzle inlet. Such a vortical ring is unstable and breaks down. Further downstream, smaller filaments of vorticities are generated due to the stretching effect, see Figure 3.5. Figure 3.6 shows the cross-sectional velocity fields on a upstream cross section and a downstream cross section. It shows that smaller structures are produced downstream which leads to intense random vorticity.

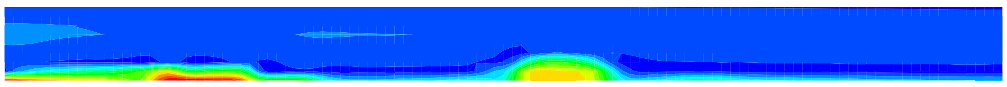

Figure 3.4: The vorticity contour in a 2D nozzle flow simulation [55].

Without the cavitation model, strong rarefaction waves are observed in the numerical simulations. This is unphysical since fluid will evaporate and produce cavitation. Figure 3.7 shows the cavitations inside the nozzle at steady state. The cavitation bubbles are first produced at the sharp corner. Further downstream, a cavitation film forms an annulus structure that extends in flow direction along nozzle walls. This is consistent with the experiments for nozzle flow [5]. Some bubbles are produced near the center line of the nozzle, but most of them disappear further downstream due

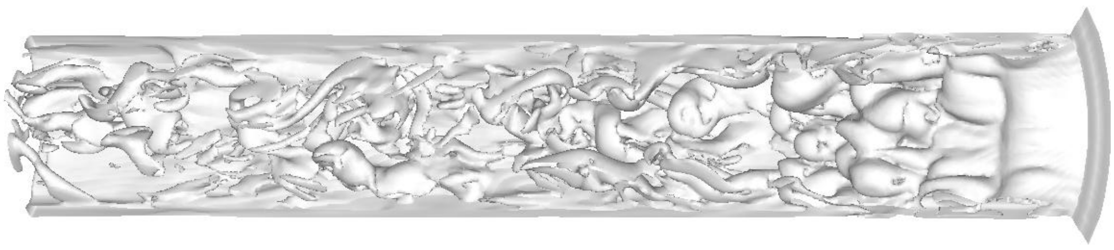

Figure 3.5: $|\mathbf{\Omega}| = 10s^{-1}$ Isosurface for vorticity in the 3D nozzle flow simulation. The nozzle inlet is on the right side.

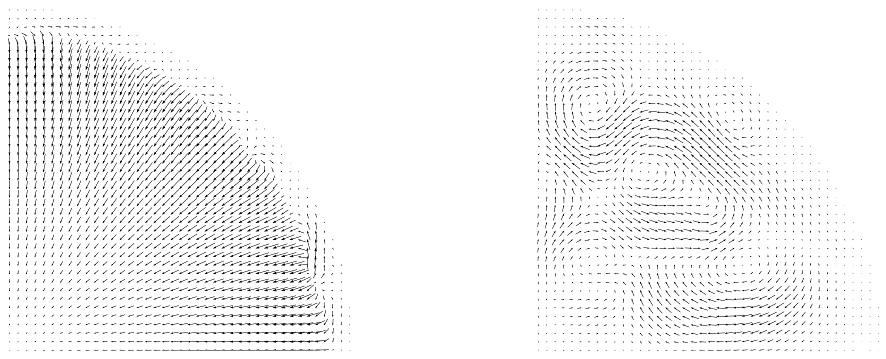

Figure 3.6: The radical and angular velocity fields in an upstream and downstream cross sections in the 3D simulation.

to the condensation of vapor bubbles. Figure 3.8 shows the pressure profile near the nozzle wall. The pressure rises up to the ambient pressure in the nozzle outlet after it drops below the vapor pressure of the fluid at the nozzle inlet.

To calculate the averaged streamwise velocity profile and the streamwise turbulence intensity, we first collect data in a cylindrical coordinate system (r, φ, z) where the z-axis is the center line of the nozzle. Then we average the results over φ direction so that all the data are functions of r, z and the time t. The averaged streamwise velocity $\overline{u}(r, z)$ over a time interval T_1 and T_2 is defined as

$$\overline{u}(r,z) = \frac{1}{T_2 - T_1} \int_{T_1}^{T_2} u(r,z,t)dt \ . \tag{3.6}$$

The streamwise turbulence intensity is defined as

$$I(r,z) = \frac{u'(r,z)}{\overline{u}(r,z)} \ , \tag{3.7}$$

where $u'(r, z)$ is the streamwise velocity fluctuation

$$u'(r,z) = \left[\frac{1}{T_2 - T_1} \int_{T_1}^{T_2} [u(r,z,t) - \overline{u}(r,z)]^2 dt \right]^{1/2} \ . \tag{3.8}$$

Figure 3.9 shows the averaged streamwise velocity profile in in the nozzle outlet. The results has been averaged over 100 time steps after the flow reaches a steady state. Figure 3.10 shows the turbulence intensity plotted vs. radical distance for a fixed z at the nozzle outlet. The simulation without the cavitation model is influenced by strong pressure waves. When the cavitation bubbles appear in the nozzle, the large tensile force induced by strong rarefaction waves are relaxed [49], thus the turbulence intensity is smaller than that without the cavitation model. Heukelbach *et.al.* [20] measured the turbulence intensity with $Re = 8500$ in experiments. They found the

the turbulence intensity is between 1% and 2% near the center of nozzle and is about 3 near the nozzle wall. Koo *et.al.* [28] performed some similar measurement with $Re = 15,000$ and $L/d = 4$. They found the turbulence intensity is between 2% to 4%. For fully developed channel flow, the turbulence intensity for $Re = 30,000$ is 7% [56]. The turbulence intensity in our simulations is in a reasonable range considering the small L/d ratio of our nozzle.

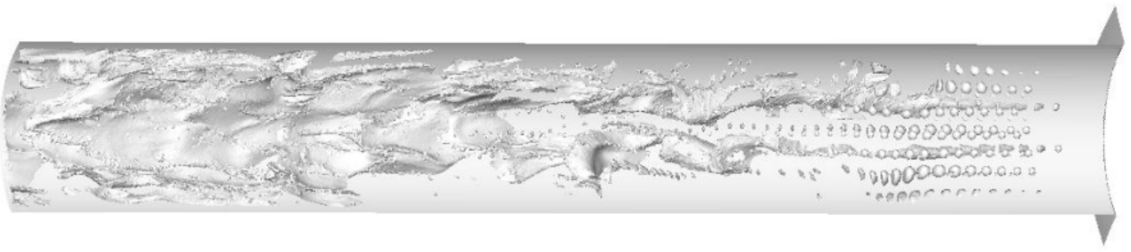

Figure 3.7: The surface of cavitation zone inside the nozzle at steady state. The nozzle inlet is on the right side.

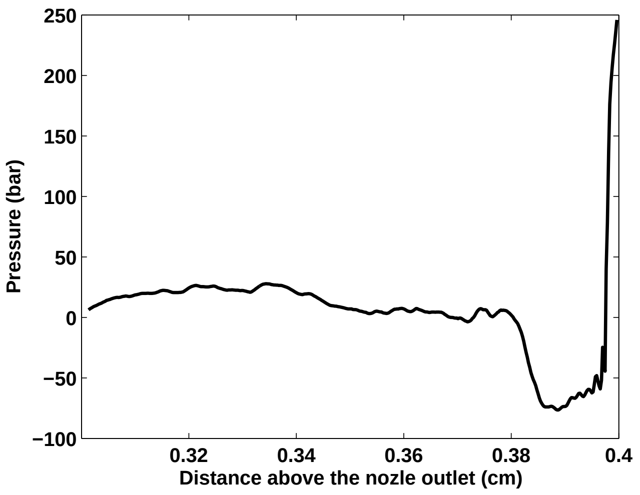

Figure 3.8: The pressure profile near the nozzle wall at steady state. The nozzle inlet is on the right side.

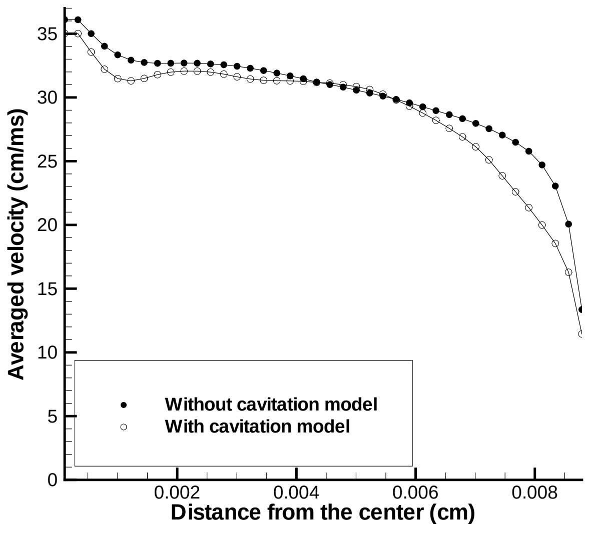

Figure 3.9: The average streamwise velocity vs. radical distance at the nozzle outlet.

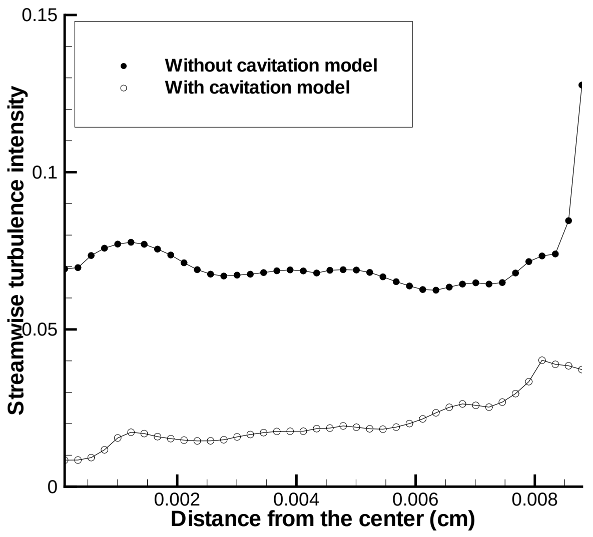

Figure 3.10: The streamwise turbulence intensity vs. radical distance at the nozzle outlet.

3.3 Simulations for Primary Jet Breakup

Primary breakup plays an important role in the performance of a diesel engine. Fluid mechanisms leading to primary breakup are still debated [10]. Aerodynamic theory [42, 53] was first proposed to explain primary breakup. This theory postulated that primary break is due to aerodynamic interaction between the liquid and the gas leading to unstable wave growth on the liquid jet surface. The drop size is proportional to the the wave length of the unstable surface waves. Reitz *et al.* [43, 44] reported several experimental investigations where the spray angle was measured. The effect of ambient pressure, density ratio, and viscosity ratio were studied. They found results that were consistent with aerodynamic theory. Wu *et al.* [58] and Sallam *et al.* [46] proposed that the primary breakup is initialized by the turbulence of the liquid jet. They assume that the onset ligament diameter is equal to the onset eddy size and droplets are formed due to Rayleigh breakup of the corresponding ligaments. They derived the correlation

$$\frac{D}{\Lambda} = 133\left(We_L\frac{\Lambda}{d}\right)^{-0.74} \tag{3.9}$$

where D is the Sauter mean diameter of the droplet at the point of breakup initiation, Λ the integral radial spatial scale of turbulence, We_L the Weber number and d the nozzle diameter. Parker and Rainaldi [40] measured the droplet diameters near the injector tip using infrared laser diagnostics. Under atmospheric pressure and room temperature, the Sauter mean diameter ranged from 5.4 to 7.8 μm, which is in the range of Wu's model based on equation (3.9). Arcoumanis *et al.* [1] found that cavitation induced in the nozzle enhances the turbulence level near the nozzle exit and thus helps the breakup of the liquid core. Dumouchel [10] summarized many experiments. He concluded that the internal flow has a strong influence on the primary breakup in the second wind-induced and atomization regimes. Aerodynamic forces are unimportant in primary breakup for high density ratios. The exact influence of cavitation on primary breakup is not fully identified and conclusions on this point differ from one investigation to another.

Numerical simulations are performed to study the details of primary breakup. Due to the small time and length scales, these simulations still remain very challenging. Only in recent years, some results are achieved. Using VOF-LES simulations, Bianchi *et al.* [3] performed simulations under both laminar and turbulent conditions with grid width 4 μm. The results conform the role of turbulence on determining the onset of jet surface breakup. However, the mean droplet size is 12 μm, larger than that predicted by the turbulence theory of Sallam [46]. Using a refined level set grid method, Herrmann [19] simulated primary breakup with several levels of grid. Although the resulting drop-size distributions were of log normal type for different grid sizes, the distributions are quite different, which means that the grid independence of drop-size distribution was not achieved. Desjardins *et al.* [7] reported some results for turbulent jet atomization using a conservative level-set method coupled with a ghost-fluid method. Menard *et al.* [36] employed the CLSVOF method to study the breakup of turbulent jet. Unfortunately, neither of them gave quantitative comparison to experimental data.

3.3.1 Problem Setup

We perform a simulation with the 3D computational domain $4d \times 4d \times 24d$, where d is the nozzle diameter. All the parameters come from the experiments performed at Argonne National Laboratory [35], summarized in Table 3.1. The whole region is discretized by using a uniform cartesian mesh 160 × 160 × 1280. As a result, a cell width is 4.2 μm. At the beginning of the simulation, the domain is filled with air at a constant state specified in Table 3.1. On the top of the region is a turbulence inflow boundary condition where the fully developed pipe flow states are given by a filter based generator [27]. All other five faces are flow out boundaries.

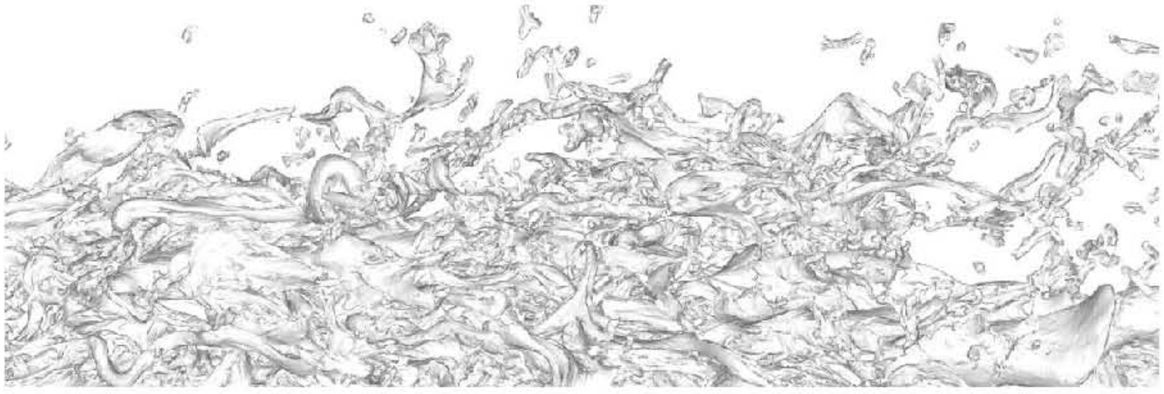

Figure 3.11: The jet interface at the tip of the jet at $We = 5 \times 10^4$.

3.3.2 Simulation Results

Figure 3.12 and 3.13 shows the snapshots of the jet simulations. The jet surface is smoother for large Weber number. Linear stability theory predicts that surface tension always tends to stabilize the jet interface in the second wind induced breakup regime [30, 44]. The jet has an intact core near the nozzle exit, which is noticed in the X-ray image [41].

Surface waves appear along the jet surface and the jet breaks up at the tip. Filament growth on the jet tip is observed in our simulation which is also observed in Sallam and Wu *et.al.*'s experiments [46, 58]. Many droplets detach from the filaments, see Figure 3.11 for details. The mean droplet diameter is larger than that from Parker *et.al.*'s experiments [40]. This is mainly due to the low resolution of our simulations.

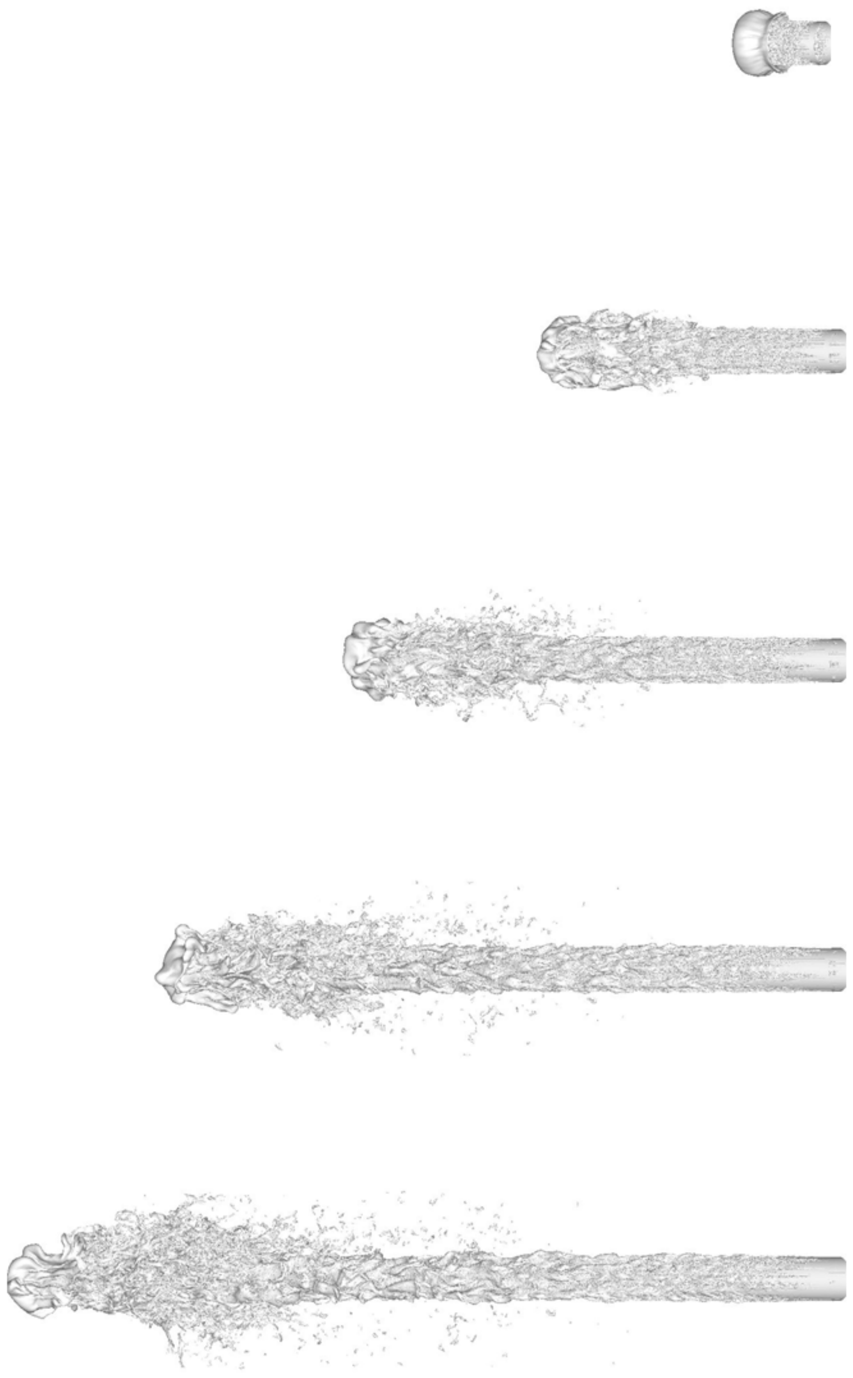

Figure 3.12: Snapshots of jet interface at $We = 5 \times 10^4$.

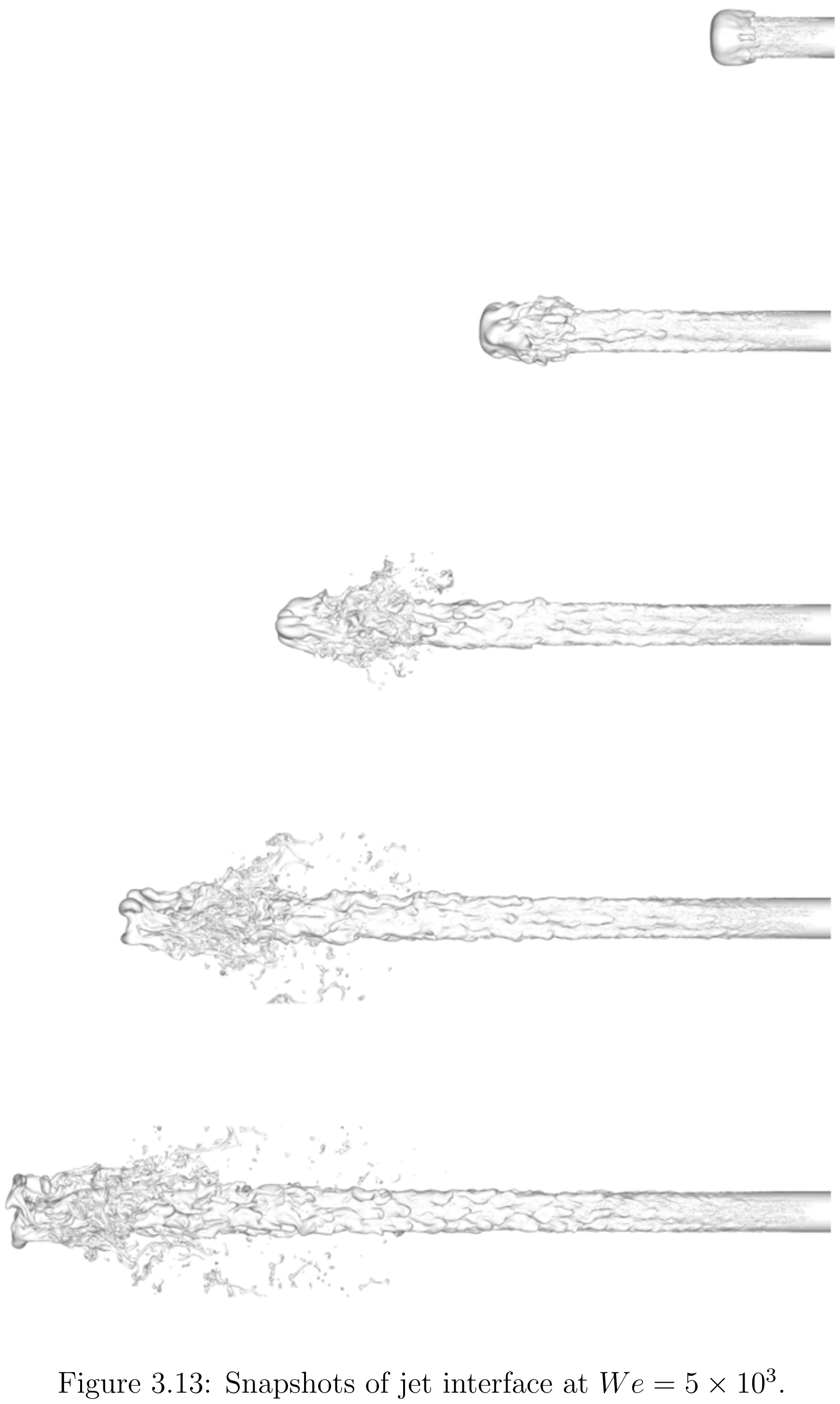

Figure 3.13: Snapshots of jet interface at $We = 5 \times 10^3$.

Chapter 4

Muon Collider Targetry Simulations

4.1 Introduction

Neutrino factories require a large number of muons, which are obtained from the decay of pions. Efficient production of pions can be achieved ay colliding an intense proton beam with a mercury jet. The mercury jet contains a series of mercury jet pulses of about 1 cm in diameter and 30 cm in length. Each pulse is shot at a velocity of 20 m/s into a magnetic field up to 15T. When the jet reaches the center of the magnet, it is hit by a 2 ns proton pulse. Every pulse will deposit about 100 J/g energy in the mercury jet. The schematic setup of the experiment is shown in Figure 4.1 from [17]. The large tensile force developed within the target induced by the proton pulse greatly exceeds the vapor pressure of mercury and cavitation occurs. Without a magnetic field, the mercury jet quickly breaks up after interacting with proton pulse. Figure 4.2 from [39] shows the the mercury target after interaction with a proton beam. Numerical simulation can reduce the amount of costly experiments and help to optimize the target parameters. To simulate the target-proton interaction, cavitation should be considered in our front tracking method. When MHD process is simulated, the numerical methods to solve MHD equations in moving domains with complex geometries are required.

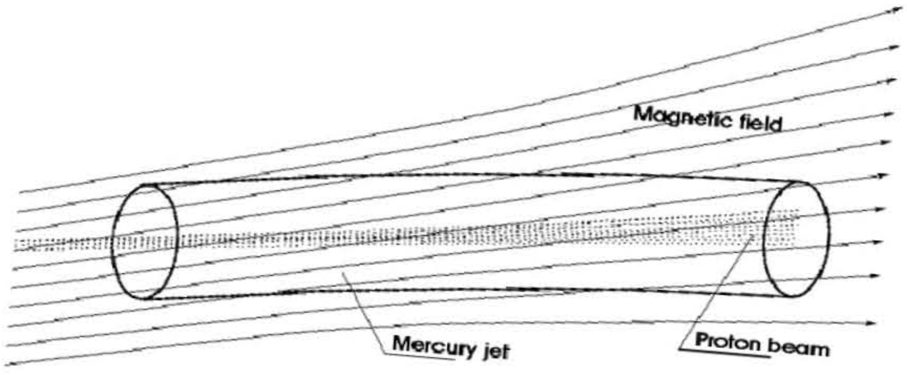

Figure 4.1: Schematic picture of muon collider target setup [17].

Two models are proposed to handle cavitations [49] in our front tracking method. The first is the homogenized model [50]. Suitable averaging is performed over the length scale which is large compared to the distance between bubbles and the mixture is treated as a pseudofluid that obeys an equation of state of a single component flow. The homogenized model is easy to implement and its computational cost is low. However, it is unable to resolve spatial scales comparable to the distance between bubbles. The second is the heterogeneous model (or discrete bubble insertion algorithm) we introduced in Chapter 3, which models liquid-vapor a system of one phase domains separated by free interfaces. Although computationally intensive, the heterogeneous model is very accurate. It allows to handle drag, surface tension and phase transitions on the interfaces. The following system of MHD equations [24] are used to model MHD effects in the mercury target with a constant external magnetic

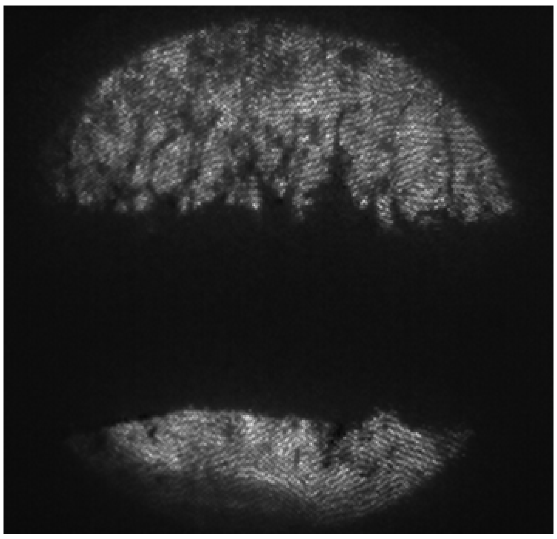

Figure 4.2: Mercury jet breakup after interaction with proton pulse [39].

field.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) \ , \tag{4.1}$$

$$\rho \left(\frac{\partial}{\partial t} + \mathbf{u} \cdot \nabla \right) \mathbf{u} = -\nabla P + \rho \mathbf{g} + \frac{1}{c} (\mathbf{J} \times \mathbf{B}) \ , \tag{4.2}$$

$$\rho \left(\frac{\partial}{\partial t} + \mathbf{u} \cdot \nabla \right) e = -P \nabla \cdot \mathbf{u} + \rho \mathbf{u} \cdot \mathbf{g} + \frac{1}{c} \mathbf{J}^2 \ , \tag{4.3}$$

$$\nabla \cdot \mathbf{B} = 0 \ . \tag{4.4}$$

$$P = P(\rho, e) \ . \tag{4.5}$$

where $\mathbf{u}, \rho, e$ are the velocity, density and the specific internal energy of the fluid, P is the pressure, $\mathbf{g}$ is the gravitational acceleration, $\mathbf{B}$ is the magnetic field induction, $\mathbf{J}$ is the current density distribution, σ is the fluid conductivity, and c is the speed of light. Equation (4.5) is the equation of state. $\mathbf{J}$ can be obtained from Ohm's law

$$\mathbf{J} = \sigma \left(-\nabla \varphi + \frac{1}{c} \mathbf{u} \times \mathbf{B} \right) \ , \tag{4.6}$$

where φ is the electric field potential and satisfies the Poisson equation

$$\nabla \cdot (\sigma \nabla \varphi) = \frac{1}{c} \nabla \cdot \sigma (\mathbf{u} \times \mathbf{B}) \ . \tag{4.7}$$

The electric potential must satisfy the following Neumann boundary condition

$$\frac{\partial \varphi}{\partial \mathbf{n}} = \frac{1}{c} (\mathbf{u} \times \mathbf{B}) \cdot \mathbf{n} \ . \tag{4.8}$$

where $\mathbf{n}$ is a normal vector at the fluid surface.

For a 2D axis symmetric flow, $\nabla \cdot (\mathbf{u} \times \mathbf{B}) = 0$ and $(\mathbf{u} \times \mathbf{B}) \cdot \mathbf{n} = 0$, which implies $\varphi =$ const and we do not need to solve the Poisson equation (4.7). For a 3D flow, an embedded boundary method for the elliptic problem is applied to solve the electric field potential [47].

In [48], numerical methods for free surface MHD flows have been developed in a 2D space. A Stiffened gamma law was used to model the equation of state of mercury. The instability of the jet surface is caused by the multiple reflections of pressure waves from the mercury-ambient interface. Magnetic fields up to 20T are used in the simulations. It was found that a 10T magnetic field is able to stabilize the jet during the period of times typical for the jet breakup at zero magnetic field. In [50] and [49], cavitation bubbles are modeled by homogenized and heterogeneous models after interaction with the proton pulse. Strong rarefaction waves are mitigated by expanding cavitation bubbles. The calculated jet velocities of expansion are in the range of experimentally measured values. In [8], 3D simulations with MHD effect and the homogenized model for cavitations are performed. All the 3D simulations give the same expansion velocities as the corresponding 2D simulations. However, no surface instability is observed in these simulations.

We solve the same problem in a 3D space. The equations (4.1)-(4.8) are solved by using the embedded boundary method proposed in [47]. Cavitation bubbles are modeled by the heterogeneous model. Magnetic field with **B** ranging from 0T to 15T cases were simulated and compared with experimental results from [39].

4.2 Problem Setup

To avoid unnecessary cost, a segment of the jet is simulated. The segment is long enough to avoid an artificial boundary effect. In the first simulations, the length of the jet is 8 cm and the radius is 0.795 cm. The dimension of the computational domain is [3.5 cm, 3.5 cm, 10.5 cm]. The mesh is $100 \times 100 \times 300$. Since the photos from all experiments only provide the shape of the jet in a projected plane, we do not know the shape of the jet in a 3D space. We also simulate an elliptic jet. The length of the jet is 8 cm, the major radius is 0.795 cm and the minor radius is 0.375 cm. The dimension of the computational domain is [1.5 cm, 3 cm, 9 cm]. The mesh is $60 \times 120 \times 360$.

After interaction with the proton pulse, energy is deposited in the mercury jet as its own internal energy. The actual energy deposition is calculated using a Monte-Carlo code MARS [38]. In [8], the energy deposition profile is approximated by a 2D Gaussian function in cylindrical coordinates. To model the energy deposition accurately, we use the the results from Striganov's calculation. The characteristics of the proton beam in his simulations are summarized in Table 4.1. We use a stiffened gamma law [37] to model the equation of state of mercury,

$$p + \gamma p_\infty = (\gamma - 1)\rho(e + e_\infty) \ , \tag{4.9}$$

where the adiabatic coefficient $\gamma = 2.866$, the stiffening constant $p_\infty = 9.8 \cdot 10^4 bar$ and $e_\infty = 7300 cm^2/ms^2$. The polytropic gamma law [37] gas is used for the ambient

Table 4.1: Characteristics of the proton beam in Striganov's calculation.

Cases	Beam energy (GeV)	Horizontal length (mm)	Vertical length (mm)	Energy deposition density (J/g)
Circular jet	24	1.34	0.78	100
Elliptic jet	24	3.3	2.7	100

gas. We can calculate the change of pressure with the change of internal energy by using

$$\Delta p = (\gamma - 1)\rho\Delta e \ . \tag{4.10}$$

At the initial time, we can get $\Delta p = 35,000bar$ with (4.10) and the energy deposition density given in Table 4.1. Such a large pressure produces strong rarefaction waves and cavitates the mercury jet. The critical pressure calculated from homogeneous nucleation theory [4] is $-100bar$. Figure 4.3 shows the contour plot of the initial pressure distribution inside the jet at $x = 0$ plane.

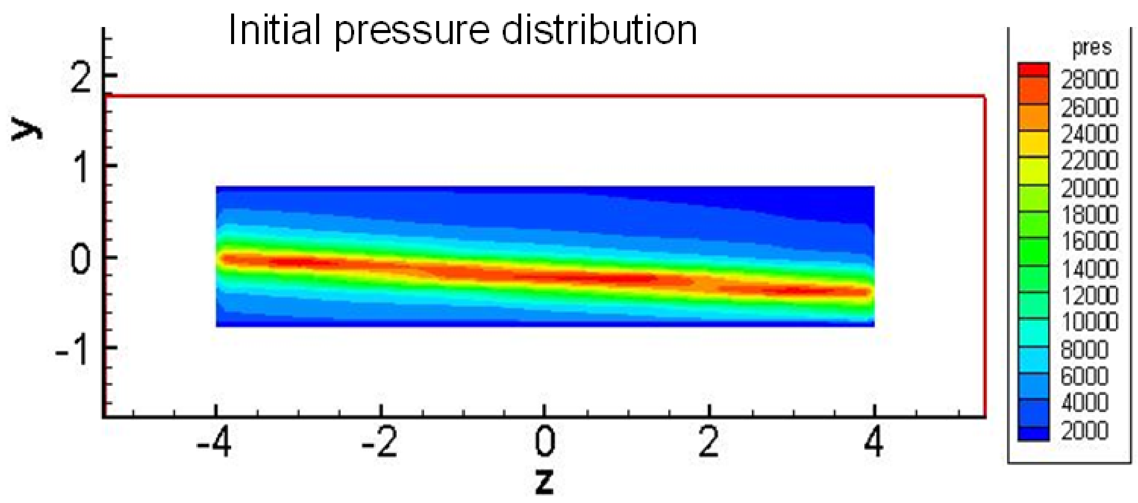

Figure 4.3: The contour plot of the initial pressure distribution inside the jet at $x = 0$ plane.

4.3 Simulation Results

We first consider the simulations without the magnetic field. Figure 4.4 shows the surface filaments at 160 μs after the proton-jet interaction. Figure 4.5 shows the snapshots of the density at $x = 0$ plane at different times. The expansion of the jet and the growth of the filaments on jet surface are noticed in the simulations. This is consistent with experimental observation. The filaments on the left side of the jet surface are longer than those on the right side (Figure 4.5), which means that the surface instabilities have the memory of the initial energy deposition. We evaluate the length and the velocity of the fastest growing filament, see Figure 4.6. The filaments reach their maximal velocity when they first protrude out of the jet surface. The average filament velocity evaluated from the simulation is 35 m/s, less than Park's measurement [39] for the corresponding case (45 m/s). There can be several reasons: First, the simulation has 40 cells across the jet diameter, the relatively coarse grid can introduce numerical diffusion which dumps the growth of the filaments. Second, the unknown parameters such as beam spot size and the energy deposition at the viewpoint may affect the observed velocity in the experiments. Last, there can be discrepancy between Striganov's results and the real energy deposition in the experiments. The delay of filament growth found in experiments is not observed in the simulations. Since the physics of the observed delay of the jet disruption is unknown [39], we may need to develop a new numerical method to model it.

Figuue 4.7 shows the surface filaments at 140 μs after the proton-jet interaction for the elliptic jet. Figure 4.8 shows the length of the fastest growing filament. The velocity is smaller compared to the cylindrical case as the initial energy profile is much flatter. We also found the filament velocity along minor axis is much larger than that along the major axis. Figure 4.9 shows the distribution of the angles between the

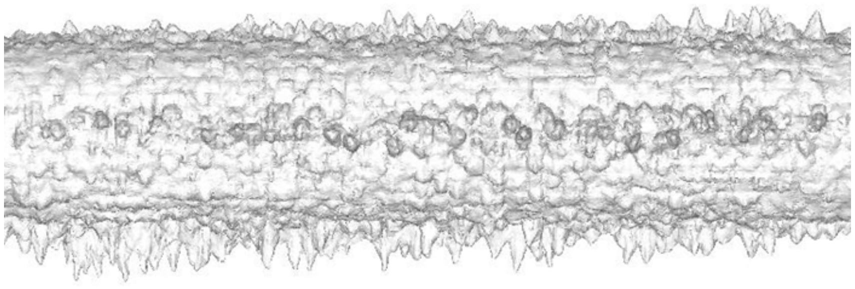

Figure 4.4: The surface filaments for the circular jet at 160 μs after the proton-jet interaction.

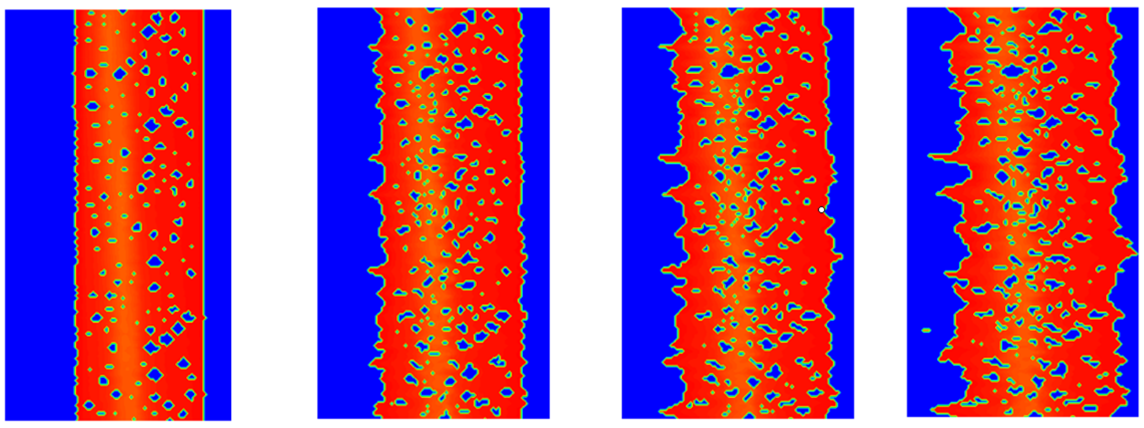

Figure 4.5: The snapshots of the contour plot of density at $x = 0$ plane at time 30, 130, 200, 250 μs.

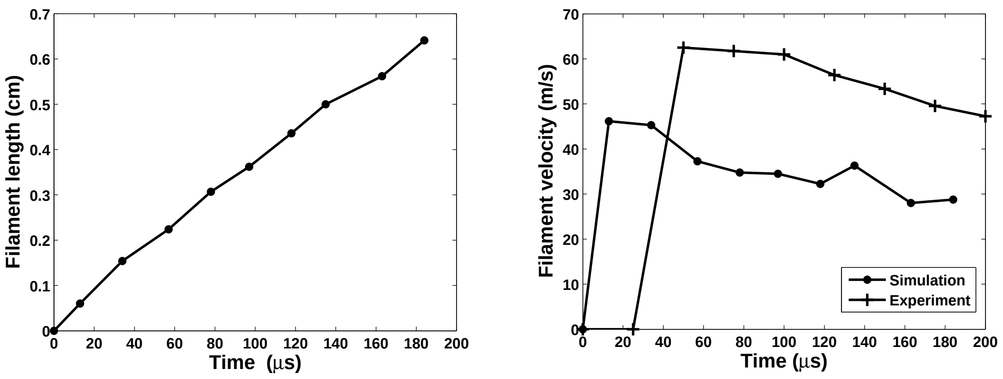

Figure 4.6: The growth of the fastest growing filament. The left figure is the length. The right figure is the velocity from the simulation and the corresponding experiment.

Table 4.2: The filament velocities from the simulations and the experiments.

Magnetic field	5T	10T	15T
Experiments	54 m/s	50 m/s	35 m/s
Simulations	36 m/s	27 m/s	22 m/s

filaments and the jet surface. Most filaments grow along the normal direction of the jet surface.

Figure 4.10 shows the jet surface at 150 μs after adding longitudinal magnetic fields. Both the interior velocity and the surface velocity of the jet are decreasing with the increasing magnetic field. The MHD stabilizing effect is weaker than the corresponding 2D simulations where circular current exists in filaments [8]. The length and the velocity of the fastest growing filaments are evaluated from the simulations, see Figures 4.11 and 4.12. We found that a 10T magnetic field is able to stabilize the jet during period of times typical for the jet breakup at zero magnetic field, which is found in experiments [39]. The filament velocity in the simulations is about 25% smaller than the experimental value (see Table 4.2). It is possible due to the low resolution of the 3D simulations.

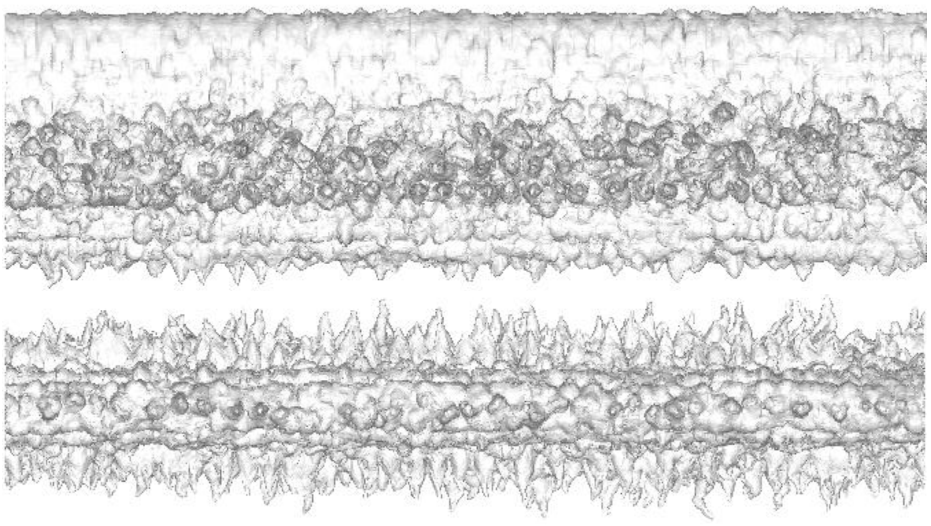

Figure 4.7: The surface filaments of the elliptic jet at 160 μs. Left: viewed from the major axis, Right viewed from the minor axis.

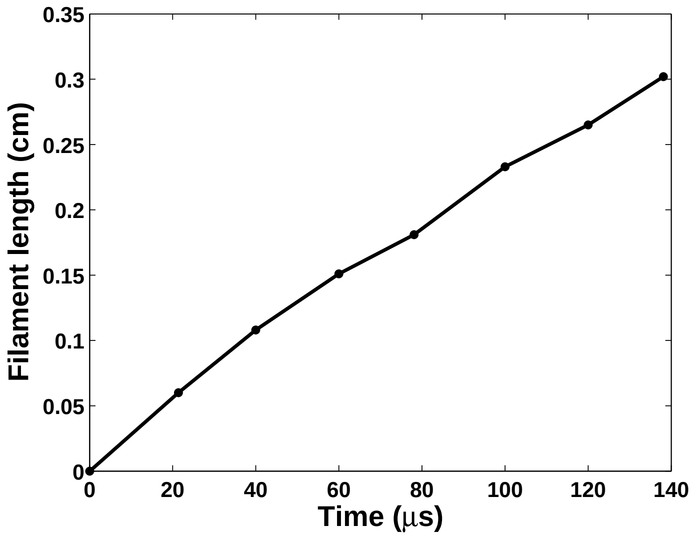

Figure 4.8: The length of the fastest growing filament on the elliptic jet.

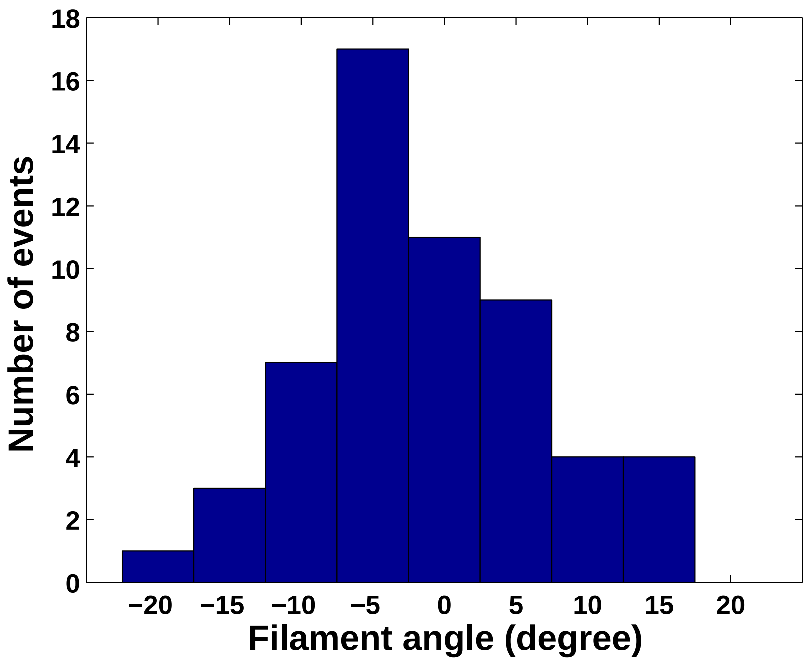

Figure 4.9: The distribution of the angles between the filaments and the jet surface.

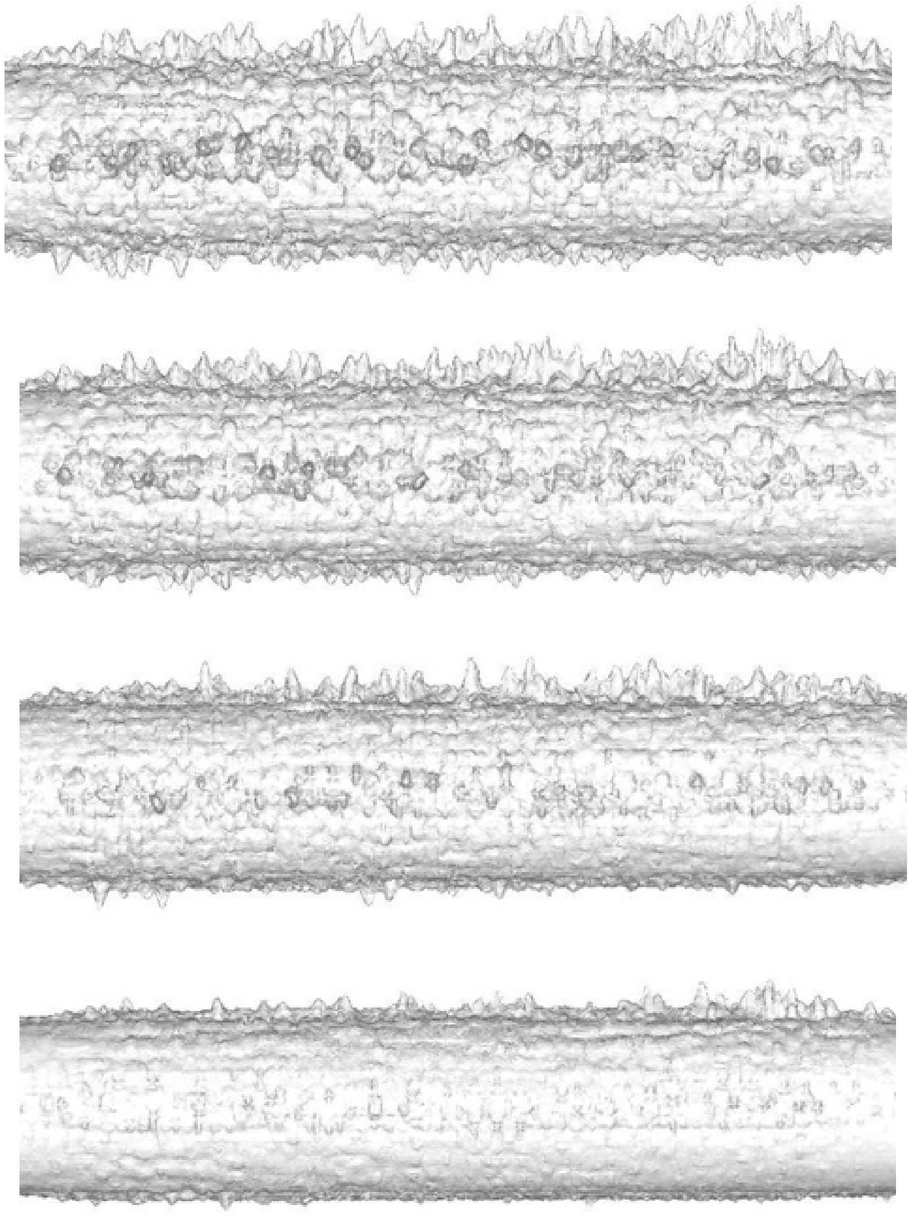

Figure 4.10: The jet surface at 150 μs under the longitudinal magnetic field B=0, 5T, 10T, 15T.

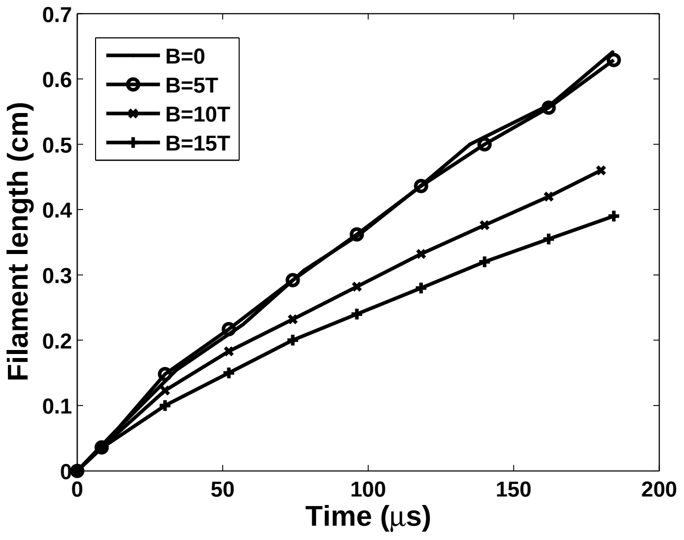

Figure 4.11: The length of the fastest growing filaments under different magnetic fields.

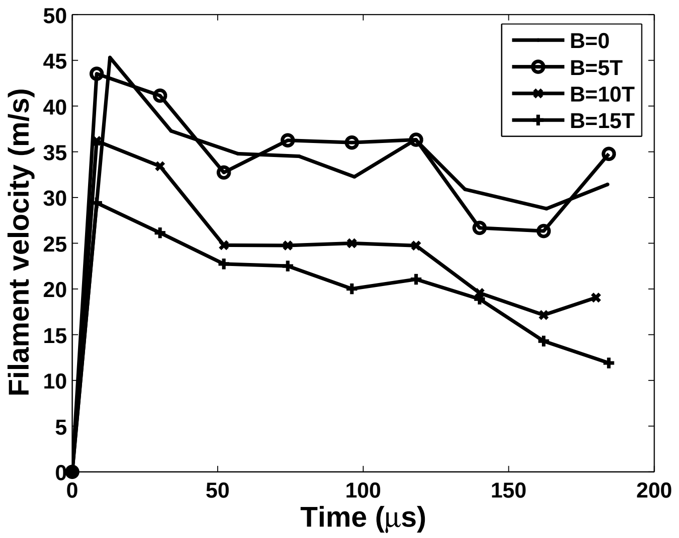

Figure 4.12: The velocity of the fastest growing filaments under different magnetic fields.

Chapter 5

Conclusions

We make improvements for the 3d front tracking method. First, we apply a 3 component grid based reconstruction method to solve the moving surface-solid wall interaction problem. The moving contact curve on the wall are explicitly tracked. A drop falling problem is studied by the new algorithm. Secondly, we present an improved, robust, locally grid based method for reconstruction of tangled interfaces. This method improves the handling of topological change of the surface mesh in the 3D simulations. We prove the method produces a topologically valid interface, thus it is suitable for large scale simulations.

The primary breakup of a high speed jet is studied numerically in a 3D space using the front tracking method. The breakup in the liquid jet is presented in the simulations. The sizes of droplets and ligaments are studied and compared with experimental and theoretical results. The nozzle flow is also studied to determine the cavitation within the nozzle and the level of turbulence occurring at the nozzle exit. The results are in the range of experimental results. We found the cavitation model is crucial in the simulations.

We also apply 3D simulations to the hydrodynamic and MHD process in liquid mercury target for muon collider/neutrino factory, which includes mercury jet interacting with protons in a longitudinal magnetic field. Surface instabilities are observed in the simulations. The stabilizing effect of magnetic field on the growth of filaments is observed. The growth of filaments is compared with experimental results.

Bibliography

[1] C. Arcoumanis, H. Flora, M. Gavaises, N. Kampanis, and R. Horrocks. Invectigation of cavitation in a vertical multi-hole injector. *SAE Technical Papers 1999-01-0524*, 1999.

[2] T. J. Baker. Mesh movement and metamorphosis. *Engineering with Computers*, 18:188–198, 2002.

[3] G. M. Bianchi, F. Minelli, R. Scardovelli, and S. Zaleski. 3d large scale simulation of the high-speed liquid jet atomization. *SAE Technical Papers 2007-01-0244*, 2007.

[4] C. E. Brennen. *Calculation and Bubble Dyanamics.* Oxford University Press, 1995.

[5] C.Baumgarten, Y.Shi, R.Busch, and G.P.Merker. Numerical and experimental investigations of cavitating flow in high pressure diesel nozzles. In *ILASS-Europe 2001, Switzerland*, September 2001.

[6] L. P. Chew. Constrained delaunay triangulations. *Algorithmica*, 4:97–108, 1989.

[7] O. Desjardins, V. Moureau, and H. Pitsch. An accurate conservative level set/ghost fluid method for simulating turbulent atomization. *Journal of Computational Physics*, 227:8395–8416, September 2008.

[8] Jian Du. *Simulation of Magnetohydrodynamic Multiphase Flow.* SUNY at Stony Brooks, August 2007.

[9] Jian Du, Brian Fix, James Glimm, Xicheng Jia, Xiaolin Li, Yunhua Li, and Lingling Wu. A simple package for front tracking. *J. Comput. Phys.*, 213:613–628, 2006. Stony Brook University preprint SUNYSB-AMS-05-02.

[10] C. Dumouchel. On the experimental investigation on primary atomization of liquid streams. *Experiments in Fluids*, 45:371–422, September 2008.

[11] J. Glimm, M. J. Graham, J. W. Grove, X.-L. Li, T. M. Smith, D. Tan, F. Tangerman, and Q. Zhang. Front tracking in two and three dimensions. *Comput. Math. Appl.*, 35(7):1–11, 1998.

[12] J. Glimm, J. W. Grove, X.-L. Li, K.-M. Shyue, Q. Zhang, and Y. Zeng. Three dimensional front tracking. *SIAM J. Sci. Comp.*, 19:703–727, 1998.

[13] J. Glimm, J. W. Grove, X.-L. Li, and D. C. Tan. Robust computational algorithms for dynamic interface tracking in three dimensions. *SIAM J. Sci. Comp.*, 21:2240–2256, 2000.

[14] J. Glimm, J. W. Grove, X.-L. Li, and N. Zhao. Simple front tracking. In G.-Q. Chen and E. DiBenedetto, editors, *Contemporary Mathematics*, volume 238, pages 133–149. Amer. Math. Soc., Providence, RI, 1999.

[15] J. Glimm, J. W. Grove, W. B. Lindquist, O. McBryan, and G. Tryggvason. The bifurcation of tracked scalar waves. *SIAM journal on scientific computing*, 9:61–79, 1988.

[16] J. Glimm, E. Isaacson, D. Marchesin, and O. McBryan. Front tracking for hyperbolic systems. *Adv. Appl. Math.*, 2:91–119, 1981.

[17] J. Glimm, H. Kirk, X. L. Li, J. Pinezich, R. Samulyak, and N. Simos. Simulation of 3D fluid jets with application to the muon collider target design. In M. Rahman and C. A. Brebbia, editors, *Advances in Fluid Mechanics III*, volume 26, pages 191–200. WIT Press, Southampton, Boston, 2000.

[18] Martin Held. A collection of efficient and reliable intersection tools. *J. Graphics Tools*, 2:25–47, 1997.

[19] M. Herrmann. On simulating primary atomization using the refined level set grid method. In *Proceedings of ILASS-Americas 21th. annual Conference on Liquid Atomization and Spray.* 2008.

[20] K. Heukelbach and C. Tropea. Influence of the inner flowfield of flat fan pressure atomizers on the disintegration of the liquid sheet. In *ILASS-Europe 2001, Switzerland*, September 2001.

[21] H.Hiroyasu, M. Arai, and M. Shimizu. *"Effect of flow conditions inside injector nozzles on jet breakup processes", Recent advances in spray conmbustion: Spray atomization and drop burning phenomena, Volume 1.* American Institute of Aeronautics and Astronautics Inc., 1996.

[22] R. Mencl J. Vollmer and H. Mller. Improved laplacian smoothing of noisy surface meshes. In *Proc. 20th Conf. Eur. Assoc. for Computer Graphics (EuroGraphics '99)*, 1999.

[23] J.A.Bondy. *Graduate texts in mathematics, 244, Graph theory.* Springer, 2008.

[24] J. D. Jackson. *Classical Electrodynamics.* John Willy & Sons, New York, 1975.

[25] Julian Scott Jean Mathieu. *An Introduction to Turbulent Flow.* Cambridge University Press, 2000.

[26] Xicheng Jia. *Applications of Front Tracking to Multiple Scientific Problems.* SUNY at Stony Brook, August 2006.

[27] M. Klein, A. Sadiki, and J. Janicka. A digital filter based generation of inflow data for spatially developing direct numerical or large eddy simulations. *Journal of Computational Physics*, 186:652–665, April 2003.

[28] J. Koo, S.T. Hong, J.S. Shakal, and S. Goto. *Influence of Fuel Injector Nozzle Geometry on Internal and External Flow Characteristics.* SAE technical papers, 1997.

[29] A. Lefebure. *Atomization and Sprays.* Hemisphere Publishing Corp., New York, 1989.

[30] S. P. Lin and R. D. Reitz. Drop and spray formation from a liquid jet. *Annu. Rev. Fluid Mech.*, 30:85–105, 1998.

[31] Mark Linne, Megan Paciaroni, Tyler Hall, and Terry Parker. Ballistic imaging of the near field in a diesel spray. *Experiments in Fluids*, 40(6):836–846, June 2006.

[32] J.-J. Liu, J. Glimm, and X.-L. Li. A conservative front tracking method. In F. Asakura, H. Aiso, S. Kawashima, Matsumura A, S. Nishibata, and K. Nishihara, editors, *Hyperbolic Problems: Theory, Numerics, and Applications*, pages 57–62. Yokohama Publishers, Osaka, Japan, 2006.

[33] X. F. Liu, E. George, W. Bo, and J. Glimm. Turbulent mixing with physical mass diffusion. *Phys. Rev. E*, 73:1–8, 2006.

[34] X. F. Liu, Y. H. Li, J. Glimm, and X. L. Li. A front tracking algorithm for limited mass diffusion. *J. of Comp. Phys.*, 222:644–653, 2007. Stony Brook University preprint number SUNYSB-AMS-06-01.

[35] Andrew G. MacPhee, Mark W. Tate, and Christopher F. Powell. X-ray imaging of shock waves generated by high-pressure fuel sprays. *Science*, 295:1261–1263, 2002.

[36] T. Menard, S. Tanguy, and A. Berlemont. Coupling level set/VOF/ghost fluid methods: Validation and application to 3D simulation of the primary break-up of a liquid jet. *International Journal of Multiphase Flow*, 33:510–524, May 2007.

[37] R. Menikoff and B. Plohr. The Riemann problem for fluid flow of real materials. *Rev. Mod. Phys.*, 61:75–130, 1989.

[38] N.V. Mokhov. Mars code developments, benchmarking and applications. In *Proc. of ICRS-9 International Conference on Radiation Shielding*, 2000.

[39] Hee Jin Park. *Experimental Investigation of Magnetohydrodynamic Flow For An Intense Proton Target.* SUNY at Stony Brooks, December 2008.

[40] T. E. Parker and L. R. Rainaldi. A comparative study of room-temperature and combusting fuel sprays near the injector tip using infrared laser diagnostics. *Atomization Spray*, 8:565–600, 1998.

[41] Powel, C. F., Y. Yue, Poola, J. R., Wang, M. Lai, and J. Schaller. Quantitative x-ray measurements of a diesel spray core. In *In Proc. 14th Annual Conference on Liquid Atomization and Spray Systems (ILASS)*. Dearborn, MI, 2001.

[42] L. Rayleigh. On the stability of jets. *Proc Lond Math Soc*, 10:4–13, 1878.

[43] R. D. Reitz and F. V. Bracco. On the dependence of spray angle and other spray parameters on nozzle design and operating conditions. In *SAE paper 790494*, 1979.

[44] R. D. Reitz and F. V. Bracco. Mechanism of atomization of a liquid jet. *Phys. Fluids*, 25:1730–1742, 1982.

[45] R.Scardovelli and S.Zaleski. Direct numerical simulaion of free-surface and interfacial flow. *Ann. Rev. Fluid Mech.*, 31:567–603, 1999.

[46] K. A. Sallam and G. M. Faeth. Surface Properties During Primary Breakup of Turbulent Liquid Jets in Still Air. *AIAA Journal*, 41:1514–1524, August 2003.

[47] R. Samulyak, J. Du, J. Glimm, and Z. Xu. A numerical algorithm for MHD of free surface flows at low magnetic reynolds numbers. *J. Comput. Phys.*, 226:1532–1546, 2007.

[48] R. Samulyak, J. Glimm, W. Oh, H. Kirk, and K. McDonald. Simulation of free surface MHD flows: Richtmyer - meshkov instability and applications. *Lecture Notes Comput. Sci.*, 2667:558–567, 2003.

[49] R. Samulyak, T. Lu, Y. Prykarpatskyy, J. Glimm, Z. Xu, and M. N. Kim. Comparison of heterogeneous and homogenized numerical models of cavitation. *Int. J. Multiscale Comp. Eng.*, 4:377–389, 2006.

[50] R. Samulyak and Y. Prykarpatskyy. Richtmyer-Meshkov instability in liquid metal flows: influence of cavitation and magnetic fields. *Mathematics and Computers in Simulations*, 65:431–446, 2004.

[51] H. Schlichting. *Boundary Layer Theory.* Mcgraw-Hill Book Company Inc., 1960.

[52] J. A. Sethian. *Level Set Methods.* Cambridge University Press, 1996.

[53] M. Sterling and C. A. Sleicher. The instability of capillary jets. *Journal of Fluid Mechanics*, 68:477–495, 1975.

[54] G. Tryggvason, B. Bunner, A. Esmaeeli, D. Juric, N. Al-Rawahi, W. Tauber, J. Han, S. Nas, and Y.-J. Jan. A front-tracking method for the computations of multiphase flow. *J. Comput. Phys.*, 169:708–759, 2001.

[55] W.Bo, X. Liu, J. Glimm, and X. Li. Primary breakup of a liquid jet. To appear.

[56] T. Wei and W.W.Willmarth. Reynolds-number effects on the structure of a turbulent channel flow. *J.Fluid Mech.*, 204:57–95, 1989.

[57] E. Winklhofer, E. Kull, and A. Morozov. Comprehensive hydraulic and flow field documentation in model throttle experiments under cavitaion condition. In *ILASS-Europe 2001, Switzerland*, September 2001.

[58] P.-K. Wu, L.-K. Tseng, G. M. Faeth, J. W. Usry, J. F. Meyers, and L. S. Miller, editors. *Primary breakup in gas/liquid mixing layers for turbulent liquids*, January 1992.

[59] Z. Xu, M. Kim, T. Lu, W. Oh, J. Glimm, R. Samulyak, X. Li, and C. Tzanos. Discrete bubble modeling of unsteady cavitating flow. *J. Multiscale Comp. Eng.*, 4:601–616, 2006.

Appendix A

Redistribution Algorithm for a 3D Surface Mesh

When a interface is moving, some triangles can be elongated and some may be contracted. To keep the regularity of the interface, Dynamic adaption for the interface is needed. In the previous *FronTier* code, redistribution is periodically applied to the 3D triangular mesh. If a triangle is too large, we split the longest edge into two and replace both this triangle and one sharing the long edge by four new elements. Similarly, triangles are deleted two at a time by collapsing the shortest edge into a point. Sometimes we also reconnect the points by swapping edges to make the triangles better shaped. The method improves the quality of the surface mesh, but it can not resolve all the bad cases which appear in the interface. For example, sharp edges or corners may appear in the interface. Also, there are forbidden cases for the edge collapsing algorithm.

In this Section, a improved redistribution algorithm is described. Since there is no forbidden cases for the algorithm, the quality of the surface mesh is always ensured. We follow the idea from T. Baker's work [2] with some modifications. Baker implemented a cycle of mesh adaption by the following steps: First, each point on the mesh is mapped to a new position and each component of the 3D displacement of vector satisfies the Laplacian equation. In this step, the interface is smoothed and the connectivity of each point is unchanged. Then, mesh coarsening is carried out to remove small triangles. Finally, mesh enrichment serves to create a mesh whose triangles are comparable to those of the mesh at the initial time.

Instead of solving Laplacian equations, we use a revised Laplacian smoothing in the first step. The position $\mathbf{p}_i$ of a vertex i is replaced with the average of the positions of adjacent vertices. We have

$$\mathbf{p}_i = \begin{cases} \alpha \frac{1}{|adj(i)|} \sum_{j \in adj(i)} \mathbf{q}_j + (1-\alpha)\mathbf{q}_i & i \in V_{var} \\ \mathbf{q}_i & i \in V_{fix} \end{cases} \tag{A.1}$$

where $\mathbf{q}_i$ is the original position of $\mathbf{p}_i$, $adj(i)$ is the set of the adjacent vertices of vertex i. V_{var} is the set of moveable vertices and V_{fix} is the set of fixed vertices. We construct V_{var} as follows. We first compute the angle between two adjacent triangles on their common edge. If the angle is less than 20°, two ending points of the edge is added to V_{var}. $0 < \alpha < 1$ is a irrational number. The convergence for the algorithm is proved in [22].

In the second step, we consider two cases. Let the set of triangles on the surface is T. Supposing we want to collapse the edge PQ, all triangles associated with points P and Q are denoted as a set T_1. The bounding points for T_1 are $P_1, P_2, \cdots, P_n$. Case (1): $P_1, P_2, \cdots, P_n$ are distinct. We collapse PQ into a point. Case (2): There are common points in $P_1, P_2, \cdots, P_n$. T_1 forms a pipe and collapsing PQ will lead to a topological change, see step A in figure A.1. We remove T_1 from the interface. From theorem 2.1.2, $B(T \setminus T_1)$ must be a directed Eulerian circuite. If a vertex in $B(T \setminus T_1)$ has a indegree greater than 1, after we separate the vertex into several vertices, $B(T \setminus T_1)$ becomes many directed simple circles. We then perform a constrained triangulation on each simple circle. The operation leads to a bifurcation in the surface mesh, see figure A.1 for the substeps.

The third step remains the same as before. We split the longest edge of large triangles. This operation is always realizable and the resulting surface is topologically valid.

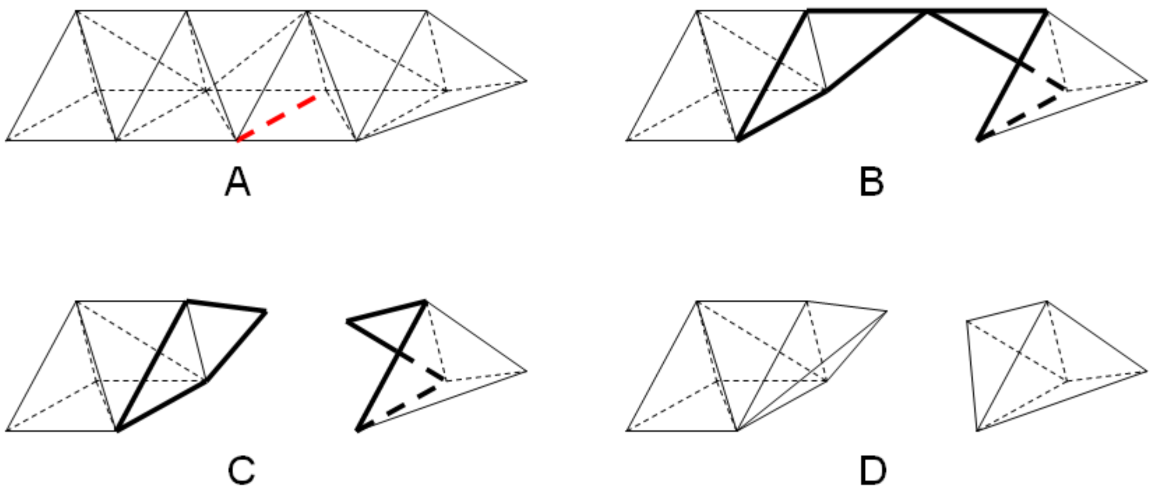

Figure A.1: Topological change happens when collapsing a edge. A: The red edge is going to collapse. B: All the triangles associated with two ending points of the edge are removed, where the thick lines represent the boundary. C: The point with indegree greater than one is separated. D: the directed simple circles are sealed by the triangulation.

Appendix B

Reflection Boundary for the Front Tracking Method

The 3D simulation is computational expensive. When the flow has reflective symmetry, the total computational cost can be decreased by half when a reflection boundary condition is applied. It is found to be very useful in our numerical study for diesel spray simulations. Without any moving surface on the reflective plane, we simply reflect all the states on cell centers from the computational domain to the buffer zone across the reflective plane. When a moving surface intersects with the reflective plane. We reflect the surface and reconnect it to the original one. Assuming we need to reflect an interface I across the plane $x_i = X_i$ ($i = 1, 2, 3$ represent x, y, z directions) plane, we perform the reflection by the following steps, see figure B.1.

(1) compute the intersections between all the edges on the interface I and the plane $x_i = X_i - h_i/2$, where h_i is the grid size in x_i direction.

(2) Retriangulate all the triangles which have intersections on their edges.

(3) Remove all triangles which lie on the other side of the plane $x_i = X_i - h_i/2$. After this step, the boundary of all surfaces contains several directed Eulerian circuite from theorem 2.1.1. We separate all vertices on the boundary which have a indegree greater than one. Then, the boundary only contains simple circles.

(4) Reflect all the boundary points on the plane $x_i = X_i - h_i/2$ across the plane $x_i = X_i$. Construct a strip of triangles by using the reflected points and the original points on the boundary and add the strip into I,

(5) Reflect I across the plane $x_i = X_i$ and reconnect the reflected interface with I along the common strip of triangles.

If the procedure is applied every time step, many small triangles may be produced after step (1). To avoid the problem, the algorithm is used every 20 steps and a redistribution of interface is performed each time the interface is reflected.

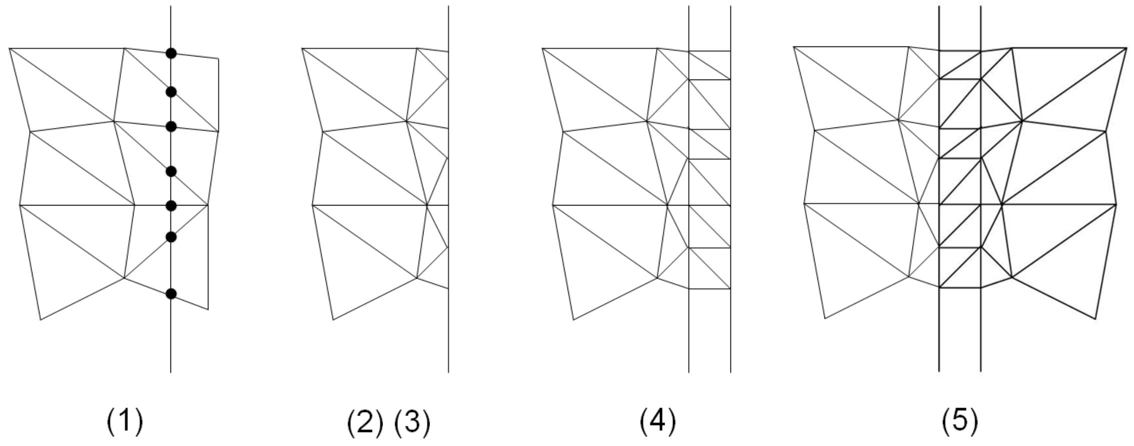

Figure B.1: The substeps to reflect a interface in the front tracking method. The vertical line in (1) represents the plane $x_i = X_i - h_i/2$