A Spray Model for an Adaptive Mesh Refinement Code

by

Sara Dailey Bauman

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

(Special Committee Degree: Mathematics and Computation in Engineering)

at the

University of Wisconsin – Madison

2001

A SPRAY MODEL FOR AN ADAPTIVE MESH REFINEMENT CODE

Sara D. Bauman

Under the Supervision of Professor Christopher J. Rutland

At the University of Wisconsin - Madison

A spray model has been implemented into an existing low Mach number, Navier-Stokes code that employs adaptive mesh refinement. The integration of a spray model with a fluid code possessing the ability to control the resolution of the computational grid attempts to address the problem of insufficient solution resolution when the cell size of fixed grids is compromised in order to achieve reasonable run times in high-pressure spray simulations.

The implementation of the spray model involves the creation and development of a suitable data structure to store spray variables, routines to control the behavior of the parcels of spray droplets, and routines to control the interaction between the ambient fluid and the spray. The spray model includes submodels for aerodynamic drag, droplet oscillation and distortion, turbulence effects, droplet breakup, evaporation, and droplet collision and coalescence. Special provisions for the treatment of spray droplet parcels within the adaptive mesh refinement framework have also been developed.

The performance of individual spray submodels has been validated by comparing single grid results to theory and experimental results from the literature. The performance of the complete spray model has been explored by comparing results calculated using grid adaptation to experimental results from the literature. The effect of adaptation on spray simulations has also been explored and discussed.

Acknowledgements

I would like to thank the folks at the Center for Computational Science and Engineering at the Lawrence Berkeley National Laboratories for all of their work and assistance. I would particularly like to recognize Marc Day, for attempting to answer all of my questions and putting up with my frequent pestering, John Bell, for giving me a jumpstart whenever I needed it, Louis Howell, for getting me started and familiarized with the AMR code, and Scott Stanley, for his work on the k-ε turbulence model.

Prof. Chris Rutland's advice, guidance, and willingness to continue working with me after moving away from Madison is sincerely appreciated. Thanks also go to Prof. Thomas Kurtz for his assistance in working out some details with the turbulence effects model, and to Basanth Roy for his work on displaying spray information using the GMV visualization package and providing some support work for the full spray runs.

Financial support from the following parties is gratefully acknowledged: NSF, through financial support of the MaCE program at the University of Wisconsin-Madison; DOE, 2000 challenge; Caterpillar; US Department of Education, GAANN Fellowship, 1997-1998; AAUW Educational Foundation, Engineering Dissertation Fellowship, 1999-2000.

Table of Contents

Chapter 3

Chapter 4

Chapter 8

Chapter 9

Chapter 10

Appendix I

Appendix J

List of Tables

List of Figures

Chapter 1

Introduction

Modeling the behavior of a liquid fuel spray is an important component of Diesel engine combustion simulations. Spray droplets interact with the surrounding fluid, profoundly affecting the character of the resulting flow. However, spray model performance can be very sensitive to the resolution of the grid used for the fluid calculations. Parcels of spray droplets, commonly modeled as discrete entities, utilize local fluid information to determine how the state of the spray changes. The spray droplets, in turn, affect the state of the fluid locally. As the grid cell size increases, the fluid information interpolated to the droplet locations may suffer and the influence of the spray droplets is spread to affect an increasingly larger region of the fluid.

The size of grid cells used for spray simulations, where the grids are usually defined prior to the calculation, is often determined by balancing between the conflicting concerns of computational cost and accuracy. Given a fixed domain size and a uniform grid, reducing the grid cell spacing greatly increases the number of needed cells to cover the area, thus increasing the computational cost. However, an attempt to control computational cost by using a custom grid with grid cells clustered in specific regions may cause the solution to suffer due to the use of skewed cells and to the needs of the simulation developing beyond the region of clustered grid points.

A number of avenues are being explored in current research efforts to find ways to improve simulations without increasing computational cost. For example, the continued improvement of computer hardware provides researchers with faster computer processors to

perform calculations on finer grids, and further improvements result from the development of better models with more efficient algorithms. Another approach is the development of alternative methods for creating and managing the computational grids that the simulations use. This approach attempts to improve the solution without increasing the computational cost by increasing the number of uniform grid cells in areas of interest and areas of insufficient resolution. Since many simulations of interest have unsteady solutions, the grids respond adaptively, refining grids and focusing computational effort when and where it is most needed.

1.1 Research Objective

Adaptive grid methods have been successful in improving the solutions of a number of problems where grid resolution is key to the success of the calculation. However, to date no one has applied an adaptive grid method to the problem of spray modeling and simulation. The viability of using adaptive grid methods for spray simulations has been explored by developing a spray model for an existing AMR code, and many of the challenges associated with incorporating discrete entities into a finite-volume formulation with adaptive grids have been identified.

The implemented spray model is based on the discrete-droplet spray model used in the KIVA II engine simulation code developed primarily by the Los Alamos National Laboratories. The spray model includes many of KIVA's default physical submodels including aerodynamic drag, droplet distortion and oscillation, evaporation, and droplet collision and coalescence. Numerous additional submodels have been developed for KIVA by researchers with the Engine Research Center at the University of Wisconsin - Madison

(UW-ERC). Some of these improved models, including models for droplet breakup, have been incorporated where deemed appropriate.

The fluid code used in this project is under continued development by the Center for Computational Science and Engineering at the Lawrence Berkeley National Laboratories (LBNL-CCSE). The Navier-Stokes, low Mach number, multi-dimensional fluid solver utilizes the adaptive grid technique called adaptive mesh refinement (AMR) and is built upon the BoxLib C++ template library. The fluid code is approximately of second-order accuracy, which is an improvement upon the first-order accuracy of KIVA's fluid solver. This project was conducted in collaboration with the LBNL-CCSE researchers.

The goals of this research were to implement a spray model into the more accurate LBNL-CCSE fluid code, develop the necessary protocols needed to integrate discrete entities into an AMR framework, identify the effect of adaptive grids on spray simulations, and identify areas where further work needs to be done. In addition, this work provides a platform upon which future work may be accomplished to improve the performance of spray simulations.

Chapter 2

Background

2.1 General Background

2.1.1 Spray Simulations

Sprays have always been a challenge for fluid modelers. Sprays that occur within combustion engines are typically comprised of a very large number of droplets. Each droplet has unique properties and is subject to complex interactions that are a function of those properties. Due to limited computational resources, it is nearly impossible to take into account each individual droplet in a computational simulation. A variety of strategies has been formulated over the years to address this problem. While details vary from model to model, most of these strategies fall into two basic categories: Eulerian-type and Lagrangian-type formulations.

The Eulerian-type formulation represents the spray using continuous fields on the same computational grid as is used for the ambient fluid. This formulation is often chosen for its simplicity and ease of implementation. The gas-jet spray model (Sinnamon et al., 1980, Hallmann et al., 1995) and spray cloud-type model (Chen and Veshagh, 1993) are examples of the Eulerian spray formulation and utilize the analogy of a spray with a turbulent gas jet. Due to the semi-continuous nature of its formulation, spray properties are typically required to remain uniform, such as isothermal droplets and uniform droplet radii, or to follow other simplifying assumptions. Diverse droplet properties can be taken into account by maintaining multiple fields and transport equations. This is equivalent to superimposing multiple sprays with differing sets of droplet properties (Sirignano, 1986). Discrete droplet

behavior, such as droplet breakup and collisions, either is neglected or requires complex models.

The Eulerian-type formulation is most appropriate when concerned about macroscopic behavior of the spray on scales much larger than the average droplet spacing or on scales on the order of the spray penetration length (Sirignano, 1986). Averaged information about the spray may be obtained and general spray behavior can be observed. However, the Eulerian approach suffers from numerical diffusion, particularly on coarse grids (Dukowicz, 1980).

The Lagrangian-type formulation is based on a fluid-particle model introduced by Dukowicz (1980). The spray is represented by a collection of computational particles. Each particle in turn represents a parcel of spray droplets that are assumed to have identical properties such as position, velocity, density, radius, and temperature. Often referred to as the discrete droplet model or stochastic particle model, this formulation is more resistant to the numerical diffusion inherent in a semi-continuous field representation. Though the droplets of a single particle have identical properties, each particle in the spray can have a unique set of properties and interact with the fluid accordingly. If appropriately chosen probability distributions are used to define particle properties, an adequate statistical representation of realistic sprays may be obtained when a sufficiently large number of computational particles are used (Watkins, 1987). In the limit of a single droplet per particle and assuming appropriate initial conditions are known, this type of formulation approaches the ideal conditions for simulating the spray.

2.1.2 Adaptive Grid Methods

The development of adaptive grid methods for finite difference and finite volume calculations has been a relatively recent phenomenon in the literature. With beginnings approximately 25 years ago, there has been an explosion of new applications for these methods in the last several years, from transonic aerodynamics to impaction between solids to magnetohydrodynamics to free surface fluids to cosmology. The common thread is the desire to more accurately model complex processes while controlling computational cost. In some cases, phenomena that previously could not be sufficiently resolved can now be studied (Zeigler, 1998).

Adaptive grid techniques generally have a few common characteristics. Each method requires a procedure for identifying areas that require additional refinement. These methods not only contain procedures for creating the additional grid refinement in the areas that need it, but also procedures for removing the grid refinement when it is no longer needed (Oden, 1989).

Since adaptive grid methods are still in the early stages of development, the terminology to describe them is still somewhat unsettled such that different names are used for the same technique and very similar names are used for very different techniques. In this review, the different approaches will be categorized according to their underlying treatment of the grids.

The moving mesh technique, also called r-refinement or the dynamic grid adaptation technique, uses a single grid that adapts to the solution by moving the nodes of the grid. This technique commonly uses structured grids with quadrahedral or hexahedral cells. The grids are structured in the sense that the cells or grid points are stored contiguously, simplifying the

task of finding neighboring cells or grid points. The location of the grid nodes in space are commonly determined by solving a PDE that is related to the current solution in order to obtain appropriate clustering of the nodes. The main benefit of this approach is that a fixed number of data points are used, so that the data structure used to hold the information can be defined prior to the computation. The primary drawback to this approach is that as the computation progresses, the grid cells gradually become skewed and distorted, degrading the quality of the solution that can be obtained on the grid. Sometimes this is addressed by periodic regridding of the domain or smoothing of the grid, but this can result in additional errors being introduced to the solution. Another drawback is that increasing grid resolution in one area of the domain necessarily means that grid resolution is sacrificed in other areas. This method attempts to provide the maximum accuracy for a fixed cost, while most of the following methods attempt to provide a fixed accuracy for the minimum cost (Oden, 1989, Huang and Russell, 1998, Podber and Bedford, 1998).

Adaptive unstructured grid techniques for finite-volume calculations have close ties to the finite element method. Many of the advancements for adaptive unstructured grids had their beginning in the development of finite element theory. The unstructured grids are composed of triangular (tetrahedral) or quadrilateral (hexahedral) elements, in two (three) dimensions, that occupy disjointed locations in memory and require pointers to their nearest neighbors to define grid connectivity. Adaptation is accomplished by splitting those elements that satisfy some refinement criteria. Benefits of this technique include: relatively simple grid initialization for complex geometries, commonly available unstructured meshing routines, and relatively easy methods for adding new elements within the data structure of an unstructured grid. However, as with the moving meshes, this technique suffers from

degraded accuracy due to the development of skewed cells. Further, the data structure of unstructured grids generally requires more memory and can result in longer run times (Kelmanson and Maunder, 1999, Jayaraman et al., 1997, Biswas and Strawn, 1998).

The domain decomposition technique, also called the domain partition approach or blockwise adaptive grids, typically uses a union of non-overlapping, rectangular, structured grids. Each grid is uniformly refined by some constant factor based on the needs of the solution. This technique benefits from the use of structured grids and predictable interfaces between differing levels of refinement. As might be expected however, uniform refinement of blocks often results in greater portions of the domain being refined than necessary. Using a larger number of blocks can alleviate this problem, but would increase the computational overhead. The blockwise structure of this technique also makes modeling domains with a complex geometry more difficult (Gropp and Keyes, 1992, Ferm and Lötstedt, 1998, Mitchell, 1998).

Grid-embedding techniques, also called local uniform mesh refinement, tree-based grid methods or adaptive hierarchical meshes, utilize semi-structured grids where individual cells of a base grid are refined and the relationship between the parent coarse cell and the child refined cells is kept in a tree-type data structure. The base grid is typically a single rectangular cell or rectangular uniform structured grid. This formulation allows highly localized grid refinement to occur, and is sometimes used to approximate complex geometries. While less memory intensive than a fully unstructured grid, this approach still requires additional overhead to keep track of grid cell relationships. Special handling for modeling complex geometries is still needed as well, though it is accomplished more easily

than with domain decomposition (Davis and Dannenhoffer, 1994, Powell, 1994, Greaves and Borthwick, 1998, Arney and Flaherty, 1989).

Structured adaptive mesh refinement, also called multi-level adaptive grid technique or overset adaptive-grid technique, recursively overlays refined grids in areas that require additional grid refinement. The primary benefit of these methods is the use of structured, uniform grids for all calculations, which simplifies both calculations and the data structures in memory. However, some administrative overhead is generated in order to manage the multiple, overlapping grids. As with many of the above methods, structured adaptive mesh refinement requires special equations to handle fluxes across grid boundaries. Since the grids are uniform and structured, this approach is most natural for rectangular domains and requires special treatment for cases with complex geometries (Berger and Oliger, 1984, Hart and McCormick, 1989, Matsuno et al., 1998).

2.2 KIVA Spray Model

2.2.1 History

The Los Alamos National Labs began developing computational fluid dynamics codes intended to address the need to simulate internal combustion engines in the 1970's. The initial version, called RICE, was a two-dimensional code written in Fortran that used rectangular grids, had a very basic eddy diffusivity model for turbulence, and utilized Arrhenius kinetics. APACHE followed, using arbitrarily shaped grid cells to allow complex geometries to be handled. CONCHAS utilized an arbitrary Lagrangian-Eulerian grid formulation that permitted Lagrangian motion of the grids to match the piston motion. The turbulence model was also improved at this time to include a subgrid scale turbulence model.

A Lagrangian-type spray model was implemented in CONCHAS-SPRAY. The spray model included aerodynamic spray motion and an evaporation model. The Law-of-the-Wall was added to better capture turbulent boundary layers and the chemistry model was improved to handle both kinetic and equilibrium chemistry reactions. KIVA expanded on CONCHAS-SPRAY with the ability to perform two- or three-dimensional computations, the addition of a droplet collisions model, and further improvements to the fluid code for handling low-Mach number problems. KIVA II became the pinnacle of official improvements to the equations and solution algorithms. A droplet breakup model was added, as was a k-ε turbulence model and a library of hydrocarbon thermophysical properties. KIVA 3 involved the move to using block-structured grids. KIVA 3V incorporated a model for moving intake and exhaust valves, a liquid wall film model, and a variety of monitoring routines to simplify the collection of information (Amsden et al., 1989, Amsden, 1993, Amsden, 1997).

KIVA, in all of its various incarnations, is a publicly available code that has broad usage among universities, research labs, and engineering companies. (Amsden and Amsden, 1993). In spray literature that involves Diesel simulations, a large number of papers indicate that KIVA was used, or that submodels developed for KIVA had been implemented into their own code. A number of these users, in turn, have influenced the development of the KIVA code. The Engine Research Center at the University of Wisconsin - Madison has used KIVA for many years, developing a number of improvements and additional/optional routines to improve the simulation of the conditions within the combustion cylinder. As a well-tested code, KIVA's spray model provides an excellent guide for the implementation of a spray model into the framework of a new fluid code. For the remainder of this document,

references to KIVA will imply the KIVA II version of the Los Alamos CFD engine simulation code, unless otherwise stated.

2.2.2 Characteristics

The spray model in KIVA is based on the discrete droplet model of Dukowicz (1980) with Lagrangian, computational particles that represent parcels of spray droplets with uniform properties. Since the scale of a typical grid cell is much larger than an average droplet, each particle is assumed to occupy a single point in space, neglecting the volume occupied by the droplets and simplifying calculations. Thick spray effects are also neglected.

Since the KIVA code was written in Fortran and intended to run on Cray supercomputers, the data structures for holding spray particle properties are designed to take advantage of vectorization. Array size is assigned a priori based on the largest number of particles expected for a given calculation.

The particles are introduced into the computation during the time of injection at locations corresponding to one or more injection nozzles. The properties of each computational particle at the time of injection are assigned using a Monte Carlo sampling technique from appropriate probability distributions. During the course of the calculation, the particles freely interact with the surrounding continuous fluid, described by an Eulerian formulation.

The spray and fluid interaction, involving the exchange of mass, momentum and energy, are primarily governed by a number of submodels. The spray submodels currently available in KIVA include models for droplet aerodynamic drag, turbulence effects, evaporation, droplet oscillation and distortion, droplet breakup, and droplet collision and

coalescence. Source terms generated in the spray submodels are applied during the spray calculations, rather than collecting the source terms and applying them to the fluid later. All spray sources generated by a given particle are applied to the individual cell containing that particle.

The submodel for aerodynamic drag assumes that the drag force on a spray droplet is analogous to the drag force on a rigid sphere of equivalent size. The model accounts for the change in drag forces for both high and low Reynolds numbers. Further details of this model will be provided in Chapter 4.

KIVA uses the two equation k-ε model to account for turbulence in the fluid. The turbulence effects submodel, as detailed in O'Rourke (1989), uses the k and ε fields to generate fluctuating velocities that are then used in all spray model calculations that involve the relative velocity between a particle and the surrounding fluid. Assumed to be piecewise constant functions of time, each fluctuating velocity is effective for the length of time corresponding to the minimum of the time it would take the particle to cross a characteristic eddy or the time it would take the eddy to breakup. If this turbulent timescale is greater than the current timestep, the fluctuating velocity is used until it expires. If the turbulent timescale is less than the current timestep, the particle location and velocity are perturbed to account for the passage of the particle over multiple eddies during the timestep. The effect of the particles on the turbulence fields is determined from the work done by the turbulence to disperse the spray droplets.

The evaporation submodel is based on a combination of the Frossling correlation, detailing the change in droplet radius during evaporation, the Ranz-Marshall correlation, which details the heat conduction rate, and basic energy conservation principles. The

evaporation procedure for a single timestep is subcycled in time. This subcycling results in better evaporation behavior, allowing the evaporation process to respond to the changes in the fluid resulting from the evaporation. Within each subcycle, the change in droplet temperature is first solved for in an implicit fashion. The corresponding change in droplet radius is then determined, as is the resulting changes in the local fluid properties.

The submodel for droplet distortion, oscillation and breakup is based on the TAB model (O'Rourke and Amsden, 1987). This approach uses the analogy between an oscillating liquid droplet and a forced, damped, harmonic oscillator to calculate the distortion of the droplet surface. Droplets are assumed to breakup when the distortion of the droplet surface from its equilibrium position exceeds a threshold proportional to the droplet's radius.

The submodel for droplet collision and coalescence uses a model developed by O'Rourke (1981) that stochastically determines the probability that the droplets within two particles will collide, and the character of the resulting collision. This method checks each pair of particles in the domain for their potential to be a collision pair, dictated by their location in the same cell. When a collision occurs, the collision model determines if the droplets in the two particles coalesce or just graze each other.

2.3 LBNL-CCSE AMR Code

2.3.1 History

The underlying structured adaptive mesh refinement algorithm used by the LBNL-CCSE fluid code was first developed by Berger and Oliger (1984) for the solution of hyperbolic partial differential equations in two dimensions. Their algorithm featured the use of block-structured grids with uniformly refined grids recursively superimposed in areas of

high truncation error. The refined rectangular grids were allowed to rotate to arbitrary orientations in order to reduce the size of the refined grid necessary to cover the desired area and to allow the grid coordinates to align with flow features such as shocks.

The scheme was later improved by Berger and Colella (1989) in order to model shock hydrodynamics. Refined grids were now restricted to remain aligned with the underlying coarse grid and refinement ratios were restricted to be multiples of 2. In addition, the coarse-fine boundary handling was altered to maintain global conservation and an explicit, second-order, Godunov-type method was used for the fluid integration solver. In the literature, the work of Berger and Colella provides the basis for many implementations of the structured adaptive grid refinement technique. It is also the first place that the technique was dubbed AMR, adaptive mesh refinement.

Bell et al. (1994), now at the Lawrence Berkeley National Laboratories, developed a highly generalized code and supporting library, BoxLib (Rendleman et al.), for performing the AMR algorithm. Building on the work of Berger and Colella, the AMR code was extended into the third dimension and improvements were made to the error estimation scheme and to the refined grid generation algorithm. Written in a hybrid of C++ and Fortran, this code made use of the speed of Fortran for computationally intensive routines, as well as the flexibility and power of object-oriented programming techniques and templates in C++.

The LBNL-CCSE AMR fluid code continues to be updated and improved. A second-order projection method (Howell and Bell, 1997, Almgren et al., 1998) was developed to handle the velocity divergence requirement for the solution of the incompressible, Navier-Stokes equations. Algorithms for unsteady, low-Mach number reacting flows and heat transfer (Pember et al., 1998) were then added. Work continues in the areas of turbulence,

compressible flows, complex geometry descriptions, and parallelization, as well as further efforts in the area of combustion and reacting flows.

2.3.2 Characteristics

The basic underlying grid structure is composed of a union of rectangular, uniform, non-overlapping, coarse grids covering the domain. At present, there are no procedures in place to handle complex domains with non-orthogonal boundaries. These base grids, generally referred to as the level 0 grids, remain throughout the calculation while finer grids are added and removed as needed.

Areas where finer grids should be placed are identified using a variety of refinement criteria. The local truncation error, as determined by Richardson's extrapolation, was used by Berger and Oliger (1984) and Berger and Colella (1989), and continues to be a useful criterion. Other common refinement criteria include the presence of large gradients, high concentrations, or particular features, such as a scalar tracer. The choice of refinement criteria, whether a single criterion or a combination of several, can be customized by the user for a particular application.

To determine where refinement is needed on a given level, the state of the solution for each cell is checked and the cell is tagged if the refinement criteria are satisfied. A grid generator then compiles the tagged cells into rectangular patches, compromising between the minimization of the number of patches and the minimization of untagged cells covered by the patches. A grid with a specified level of refinement is then created to cover each patch, and the state of the solution on the new, finer grid is initialized using values from the original,

coarser grid (or previous finer grid, if available). This process is performed recursively over all finer level grids until the desired refinement is attained.

All fine grids are properly nested in the sense that each grid on level L does not overlap, lies completely within the boundaries of one of more grids of level L-1, and does not share a boundary with a level L-1 grid except at the physical boundaries (see Figure 2.1). As grids are refined in space, they are simultaneously refined in time. Thus, if level L+1 has a grid spacing of Δx^{L+1}, a time step of Δt^{L+1}, and a factor of refinement *r*, where *r* is typically 2 or 4, then $r*\Delta x^{L+1} = \Delta x^{L}$ and $r*\Delta t^{L+1} = \Delta t^{L}$. By refining in time using the same refinement factor as the grids, the solution at each level has the same CFL (Courant-Friedrichs-Lewy) number and thus similar stability behavior when using explicit methods such as the default, second-order Godunov solver.

The advancement of the solution occurs in a recursive fashion through the multiple layers of grids, so only the process that occurs on a single generic grid level need be described. At level L, the state of the solution on all level L grids is advanced simultaneously. First, for each grid, a shell of boundary ghost cells is defined to provide Dirichlet boundary values for the grid. These ghost cells use level L information or physical boundary condition information where available. Otherwise, the ghost cell values are interpolated in space and time from the level L-1 grids. The solution on each grid is then advanced by the level timestep, Δt^{L}. The advance procedure for advection, diffusion and chemical reactions is basically performed in a predictor-corrector fashion. A projection is performed in order to enforce the incompressibility constraint.

At this point, the grids at level L+1 are advanced *r* times, where *r* is the factor of refinement for level L+1. Once the level L+1 grids have been advanced to the same time as

the level L grids, fluxes across coarse-fine boundaries need to be balanced. The solution on the L+1 grids is always assumed more accurate, so the level L grid cells surrounding the finer grids are adjusted to maintain flux conservation. The level L timestep is completed by averaging the solution on the level L grids onto the level L-1 grids.

Note that this recursive timestep arrangement embeds the solution development on all of the grids within a single coarse timestep. Thus, if a solution exists at time t^L for the grids at level L, the solution on all finer grids (levels L+k for k>0) are brought up to the same time t^L before the solution at time $t^L + \Delta t^L$ can be found (see Figure 2.2).

The solver used for the fluid in the LBNL-CCSE AMR code utilizes a second-order Godunov method for the convective terms (Berger and Colella, 1989, Bell et al., 1994). Due to the explicit nature of the Godunov methods, the length of each timestep must be restricted in order to attain a certain level of accuracy and stability. The length of a new coarse (level 0) timestep is determined by the state of the solution at the end of the previous coarse timestep. Though there are many factors involved, one such determining factor is that the fluid is restricted to moving (through convection or possible acceleration) no more than a single coarse cell width in distance, thus providing a CFL number typically less than one. Since finer grids are refined in time as well as in space by the same factor, the choice of coarse timestep provides a maximum CFL number, and thus the stability and accuracy characteristics, for the calculation on all levels.

For further information and more details about the LBNL-CCSE AMR code, an extensive list of relevant publications may be found on the Lawrence Berkeley website for the Center for Computational Science and Engineering (http://www.seesar.lbl.gov/ccse).

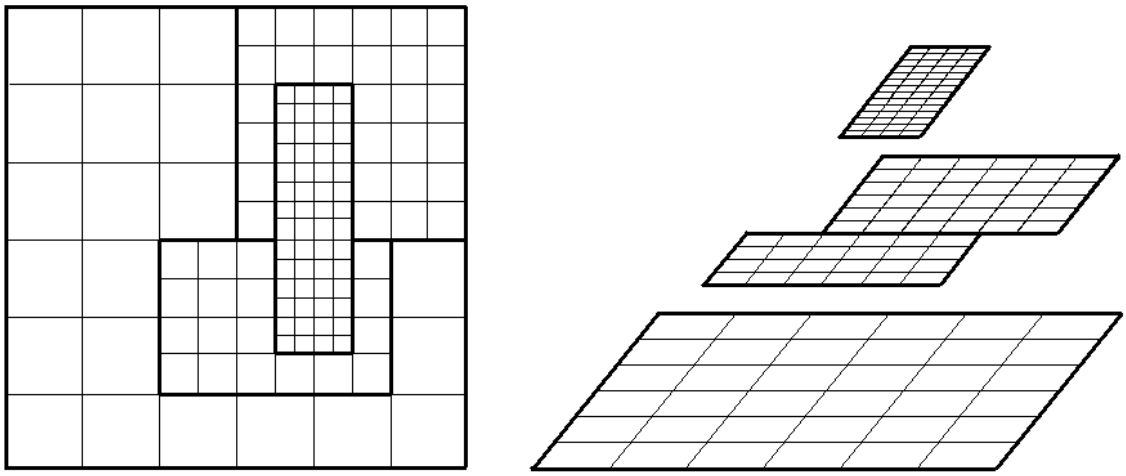

Fig. 2.1 An example AMR grid with two levels of refinement. The finer grids each have a refinement factor of 2.

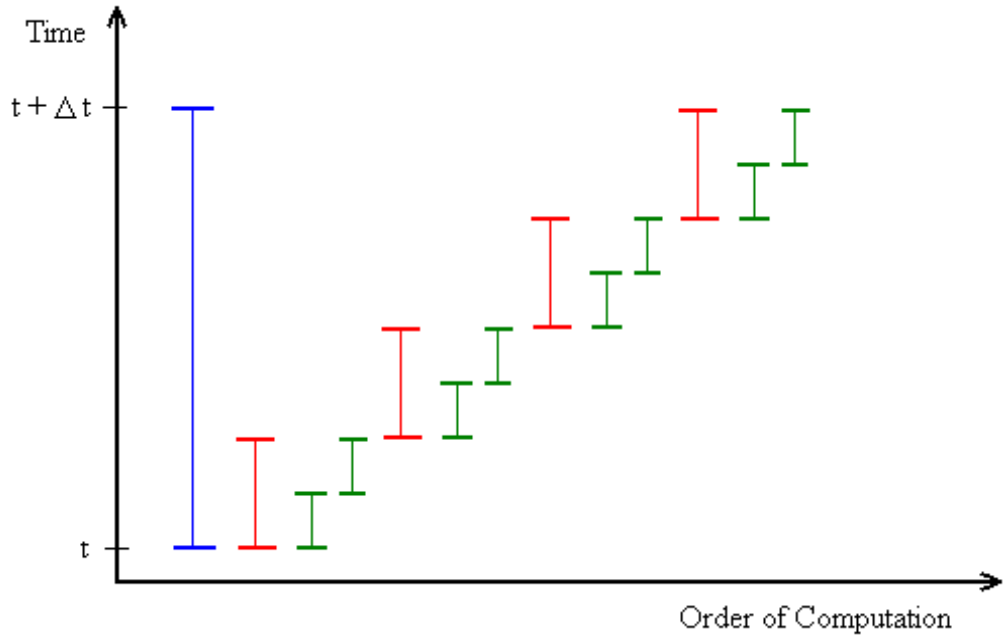

Fig. 2.2 The time step order for three adjacent refinement levels. The level L time step from time t to time $t + \Delta t^L$ is represented by the first and largest cross bar. Level L+1 (the mid-sized cross bar) has a refinement factor of $r = 4$ and level L+2 (the smallest cross bar) has a refinement factor of $r = 2$.

Chapter 3

General Spray Code Information

The spray code developed for this work is based upon the spray model implemented in KIVA. As in KIVA, the spray model uses the discrete droplet model of Dukowicz (1980), where each computational particle represents a parcel of droplets with uniform properties. This chapter presents an overview of the spray code's framework, general support submodels and underlying numerics. More detailed discussions of the spray's physical submodels and other relevant major topics are presented in later chapters and the appendices.

3.1 Spray Code Basics

The code for the spray model was written in a combination of C++ and Fortran, following the precedent set by the AMR code. Most of the administrative tasks, data structures, and timestep advancement routines are implemented in C++ while Fortran was used for most of the computationally intensive spray physical submodels. The need to use Fortran for the spray code was also precipitated by the AMR code itself. The data structures used to hold the fluid state were designed such that information on a grid could be manipulated as a whole in C++, but the data for any given cell were never intended to be accessed except in a Fortran routine. In fact, the data is organized in the natural order for Fortran multi-dimensional arrays. Since most of the spray submodels require fluid properties at specific locations, it became necessary to implement the submodels in Fortran in order to avoid breaking the encapsulation of the C++ data structures.

The basic data structure in the spray model is a C++ class, called Particle, which contains all of the information pertaining to a single computational particle and the parcel of droplets it represents. Double-linked lists are used to hold the Particles and organize them in order to minimize searching for those Particles that are relevant to a particular calculation. These lists include a working list to hold Particles appropriate for the "current" calculations, an inlet list to hold Particles waiting to enter through one or more nozzles, and level lists to hold Particles not needed for the "current" calculation on a given refined grid level. Further details about the data structure used for the spray may be found in Appendix A.

3.2 Spray-Fluid Code Interface

The spray code has been developed primarily as a self-contained module with only a thin interface to the AMR code. This arrangement means that only minor changes to the spray code are needed as the AMR code develops and expands. The interface orchestrates the interweaving of the spray calculations with the fluid calculations, providing each with the information needed to perform correctly. In general, the fluid solution being advanced by the AMR code is only aware of the liquid spray via source terms that are provided by the spray calculations. Likewise, the spray is only aware of the fluid via the fluid properties passed through the interface to be used in the spray calculations.

3.3 Timestep Selection

One of the ways that the spray affects the fluid calculations is through the selection of the coarse (level 0) timestep. The new timestep is initially determined from the current state of the fluid velocity field to obtain a defined level of accuracy from the explicit fluid solver.

If heat transfer and/or chemical reactions are involved in the calculation, the timestep is then subject to requirements imposed by the heat transfer and reaction rates to maintain a desired level of stability and accuracy. Once the timestep has been determined to satisfy fluid requirements, the timestep is limited further by the spray, if necessary. No particle is allowed to travel a distance greater than a single grid cell width during a single timestep. This restriction ensures that the particle experiences a relatively continuous fluid field, and that the influence of the particle on the fluid is relatively continuous. Restrictions are also made based on droplet evaporation rates to keep mass and energy exchange rates from becoming too large and possibly producing unnatural results.

3.4 Timestep Advance Procedure

Adding sprays to an AMR computation required some modifications to the procedure for advancing the solution a single timestep. This not only involved interlacing the particle advance routines with the fluid advance routines, but also involved administration of the collection of Particles in order to minimize any searching that must be done to find the Particles that are appropriate for the calculations on a given level.

As the fluid code prepares to perform a timestep advance, the spray also prepares for the timestep advance by reorganizing the Particles to find those particles appropriate for the current level, L. Each Particle in the working list is examined to see if it exists in the level L grids. If not, it is placed in a separate waiting list, a “level list” that holds Particles that are on level L-1 grids but not level L grids, until the next time the level L-1 grids are advanced. The Particles in the L-1 level list are in turn examined to find any particles that have moved onto the level L grids since the last level L timestep, and are transferred to the working list

accordingly. If the advance is for the level 0 grids, any new particles that will enter during the timestep are placed into a temporary list, earmarked for special handling during the advance.

The advance procedure begins with an update of the particles' distortion parameters. The spray aerodynamic drag, turbulence effects, and evaporation submodels are performed in a predictor-corrector fashion and interweaved with the fluid predictor-corrector procedure. The fluid projection operation is then performed and the finer timestep calculations are made. Once the finer timesteps have been completed, the spray breakup and collision submodels are performed.

The advance procedure is completed by enforcing the boundary conditions on the particles and cleaning up the particle lists. Any particle that has a zero mass due to evaporation or collision coalescence is removed from the collection of particles. Any particle located outside the domain is handled according to the boundary that the particle crossed. If the r^{th} iteration of the level L timestep was just performed, where r is the refinement factor for level L, the particles in the level L-1 list are placed in the working particle list in preparation for the next level L-1 timestep advance. Otherwise, the next level L timestep is performed.

3.5 Initialization

All of the spray particles are initialized during the problem setup and initialization. Properties that the particles are to possess at their time entry into the calculation are determined and defined. Any particle not intended to start in the domain is kept in an

ordered list until the calculation has progressed to the time of the particle's entry through one of the nozzles.

Spray properties are defined based on user-supplied parameters. The user may define any number of jets. Each jet may have an arbitrarily unique position, orientation, nozzle diameter, and spray cone shape. The spray entering as part of a jet is given a temperature, a mean droplet radius, an activation time and duration, the total fuel mass to enter through the nozzle, and the desired number of particles to represent the jet. In order to obtain an appropriate statistical distribution for a given property, the user may obtain values for that property by sampling from one of a number of available stochastic distributions (see Appendix J). The user may also provide a data file, for defining the mass flow rate or initial particle velocity for a jet, containing the desired distribution from which to interpolate values.

3.6 Boundary Conditions

Fluid dynamics problems often have a wide variety of boundary conditions. The spray model needed appropriate corresponding boundary conditions for the spray particles. Periodic boundary conditions are handled for particles as it is for the fluid, i.e. a particle exiting through a periodic boundary should re-enter the domain at the corresponding point on the opposing boundary with no change in the particle's properties or velocity. To handle fluid outlets, it is assumed that a particle exiting the domain through one of these boundaries never returns. In this case, a particle is merely removed from the collection of particles making up the spray.

Particles also have solid wall boundary conditions that are used in conjunction with slip and no-slip walls. All walls are assumed essentially smooth from which particles rebound elastically. The complex issues related to wall impingement of the spray are currently being dealt with by the spray modeling community and is not addressed by this project. Symmetric boundary conditions are also handled with the solid wall boundary conditions. As a particle leaves the domain through a symmetric boundary, an identical particle will be entering at the same point with a trajectory equivalent to the original particle trajectory reflected by the symmetry plane.

As we are dealing with multiple overlapping grids, we also need to specify internal boundary conditions corresponding to coarse-fine grid boundaries. Since these internal boundaries do not exist in the physical system being modeled, the internal boundary conditions should be such that any coarse-fine boundary is completely transparent to a particle's motion. The details associated with the internal boundary condition are discussed in Chapter 6.

3.7 Particle-Fluid Interaction

3.7.1 Local Fluid State

Many of the spray submodels require the fluid state local to the particle. In order to obtain this information, bilinear interpolation is used. Fluid properties in the LBNL-CCSE AMR code are located at grid cell centers. Thus, using bilinear interpolation, the fluid properties at the particle location are obtained from the 4 cells (or 8 cells, in three dimensions) closest to the particle. Bilinear interpolation in two dimensions of the fluid's scalar field, w, at the particle location, $\bar{x}$, is given by

$$I(w,\bar{x}) = w_{ur}\left(\Delta x_{low}\Delta x_{left}\right) + w_{ul}\left(\Delta x_{low}\Delta x_{right}\right) + w_{ll}\left(\Delta x_{up}\Delta x_{right}\right) + w_{lr}\left(\Delta x_{up}\Delta x_{left}\right) \qquad (3.1)$$

and in three dimensions

$$\begin{aligned} I(w,\bar{x}) = {} & w_{urf}\left(\Delta x_{low}\Delta x_{left}\Delta x_{back}\right) + w_{urb}\left(\Delta x_{low}\Delta x_{left}\Delta x_{front}\right) + \\ & w_{ulf}\left(\Delta x_{low}\Delta x_{right}\Delta x_{back}\right) + w_{ulb}\left(\Delta x_{low}\Delta x_{right}\Delta x_{front}\right) + \\ & w_{llf}\left(\Delta x_{up}\Delta x_{right}\Delta_{back}\right) + w_{llb}\left(\Delta x_{up}\Delta x_{right}\Delta x_{front}\right) + \\ & w_{lrf}\left(\Delta x_{up}\Delta x_{leftt}\Delta x_{back}\right) + w_{lrb}\left(\Delta x_{up}\Delta x_{leftt}\Delta x_{front}\right) \end{aligned} \qquad (3.2)$$

where

$$\begin{aligned} \Delta x_{low} + \Delta x_{up} &= 1 \\ \Delta x_{left} + \Delta x_{right} &= 1 \\ \Delta x_{back} + \Delta x_{front} &= 1 \end{aligned} \qquad (3.3)$$

and w_{xxx} are the cell-centered fluid properties being interpolated. Illustrations demonstrating the interpolation variables are in Figures 3.1 and 3.2.

3.7.2 Particle Source Distribution

As a particle moves within the fluid and reacts to the fluid properties, it in turn affects the fluid by providing source terms to the fluid's governing equations. Since each particle is essentially modeled as if it occupies a single point in space, the resultant source term must be distributed to nearby fluid grid points. In KIVA, the source terms are distributed to the cell that contains the particle. In this study, source distribution is approximated using a basis function centered on the particle location. The fraction of the basis function that lies in a given fluid cell determines the fraction of the source term assigned to that cell. For all of the tests described in this report, a constant step function the size of a single cell has been used to represent the basis function. See Figure 3.3 for an illustration of source term distribution for a two-dimensional grid.

The use of a basis function that may cover more than one grid cell works well for source term distribution in a grid's interior, but additional considerations become necessary when the source is being distributed near a boundary. Near periodic boundaries, the source distribution is handled in a natural fashion with the source to be distributed to the ghost cells outside of the domain being applied on the cells on the opposite boundary. Near solid wall or symmetric boundaries, the source term is distributed as if the source assigned to the ghost cells outside of the boundary is reflected back into the domain. In other words, the entire source is applied to the cells along the boundary. In the case of source distribution near a coarse-fine boundary, additional provisions must be made. This issue is addressed and discussed in Chapter 6.

3.7.3 Provisions for Sources in Two Dimensions

Two-dimensional computations with the AMR fluid code, as is done in many fluid dynamics codes, assume that the computational cells have a depth of magnitude one normal to the plane. Though most of the AMR fluid code is essentially dimensionless, the use of CHEMKIN (Kee et al., 1980) to obtain thermochemical properties for the fluid precipitated the need to choose a general set of units. Their choice of standard SI units means that the grid cells in two-dimensional calculations have an assumed depth of one meter. If the length scale of a grid cell is on the order of a millimeter or less, then the respective volumes of a two-dimensional and a three-dimensional grid cell are vastly different.

Governing equations for a fluid are usually defined in terms of the fluid's intensive properties. The grid cell depth and volume generally only come into play when external source terms are added to the fluid, such as those added due to the spray. The spray source

terms are a particular problem due to the assumption that the spray droplets are three-dimensional, even when the particle motion is restricted to a two-dimensional plane. Particles provide the same source terms for a two- or three-dimensional simulation, provided the fluid conditions are similar enough. However, a spray source term will have a much greater impact on the fluid in a three-dimensional case due to the difference in grid cell volume.

A rigorous solution to this problem, one that negated the impact of the grid cell depth magnitude and was appropriate for each of the different spray source terms, was not apparent. It was thus decided to permit the user to define the desired depth of the implicit third dimension, preferably a depth of the same order as the two-dimensional grid cell size. By scaling the spray source terms appropriately, the implicit depth of one meter is treated as if it contains multiple identical copies of the fluid-spray system with the desired domain depth. This approach allows the effect of the spray on the fluid to have a similar magnitude in both two- and three-dimensional simulations.

The spray source term scaling factor, α_{grid}, is defined as

$$\alpha_{grid} = 1 \Big/ dx_{3\_eff}$$

where dx_{3_eff} is the desired domain depth. The scaling factor is multiplied directly to the spray source terms as they are being added to the fluid and thus has no direct effect on the spray. For three-dimensional calculations, α_{grid} has a value of one. The desired magnitude of the third dimension, dx_{3_eff}, for the various two-dimensional simulations presented in this document is referred to as the ‘domain “depth”’ in the tables of Chapters 5, 7, 8 and 9.

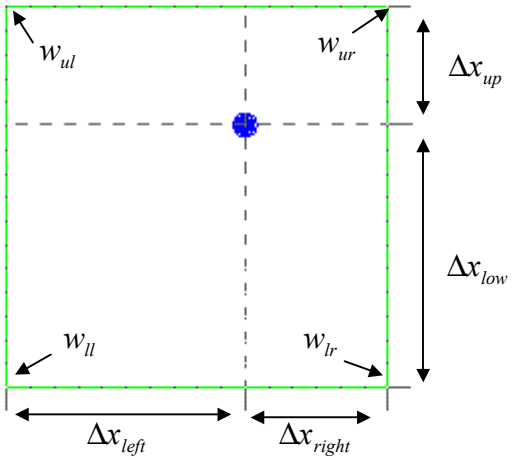

Fig. 3.1 Two-dimensional, bilinear interpolation scheme illustration.

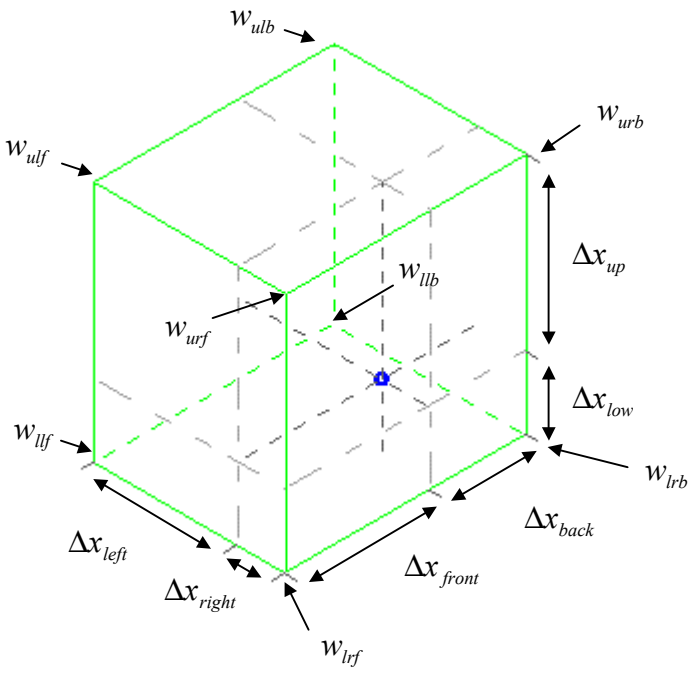

Fig. 3.2 Three-dimensional, bilinear interpolation scheme illustration.

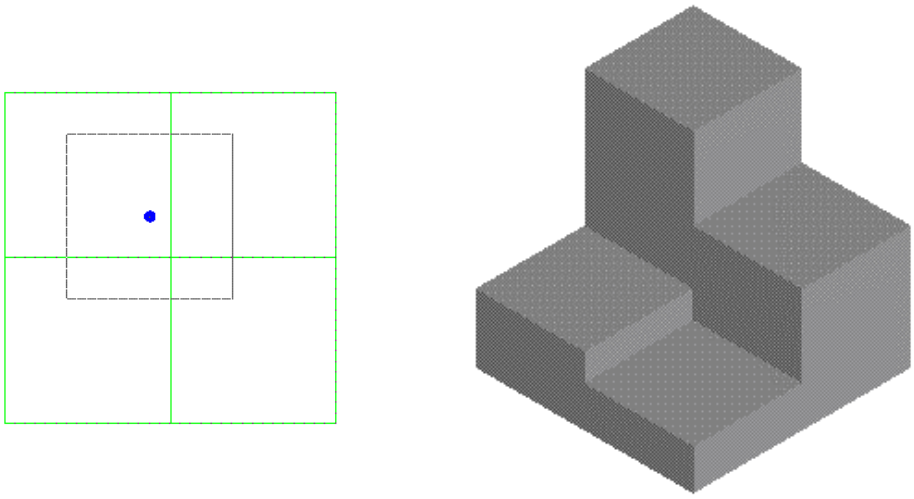

Fig. 3.3 Illustration of the two-dimensional, source distribution template for a particle in the interior of a grid.

Chapter 4

Spray Submodels

A large portion of the spray model is composed of a collection of physical submodels that control different aspects of spray behavior. The spray physical submodels are described here in general terms neglecting the presence of multiple grids. Unless otherwise noted, the submodels in this chapter are based on those used in KIVA (see Amsden et al., 1989). The additional protocols needed to handle grid adaptation in these submodels will be discussed in Chapter 6.

4.1 Aerodynamic Drag

For modeling purposes, each parcel of droplets is assumed to occupy a single point in space. The inherent three-dimensional character of the droplets is accounted for through an aerodynamic drag force. As a particle moves through the fluid, it experiences a force equivalent to the composite drag force of the droplets moving relative to the ambient fluid. Inter-droplet effects on the aerodynamic drag are neglected.

4.1.1 Theory

The basic equations governing the motion of a particle *p* may be expressed as

$$\frac{d\bar{x}_p(t)}{dt} = \bar{v}_p(t) \tag{4.1}$$

$$m_p \frac{d\bar{v}_p(t)}{dt} = \bar{F}_{fp}(t) + m_p \bar{g} \tag{4.2}$$

where $\bar{g}$ is gravity, $\bar{F}_{fp}$ is the drag force applied by the fluid on the particle, and $\bar{x}_p$, $\bar{v}_p$, and m_p are the particle's position, velocity, and total mass respectively. An overbar indicates a vector quantity.

The aerodynamic drag force experienced by a particle *p* moving through a fluid with velocity $\bar{u}_f$ is given by the expression

$$\bar{F}_{fp}(t) = \frac{1}{2}\rho_f(\bar{x}_p,t) C_D A_p \left|\bar{u}_f(\bar{x}_p,t) - \bar{v}_p(t)\right| \left(\bar{u}_f(\bar{x}_p,t) - \bar{v}_p(t)\right) \tag{4.3}$$

where ρ_f is the fluid density, C_D is the coefficient of drag, $A_p = N_d(\pi r_p^2)$ is the composite frontal area of the droplets represented by the particle, assuming a spherical shape, and r_p and N_d are the droplet radius and the number of droplets in the parcel, respectively.

The coefficient of drag used in the expression of the drag force may be defined as a constant or as a function of the fluid's Reynolds number, *Re*. The coefficient of drag and Reynolds number are calculated as follows:

$$C_D(Re) = \begin{cases} \dfrac{24}{Re}\left(1 + \dfrac{1}{6}Re^{2/3}\right) & Re \le 1000 \\ 0.424 & Re > 1000 \end{cases} \tag{4.4}$$

$$Re = \frac{2\rho_f(\bar{x}_p,t)\left|\bar{u}_f(\bar{x}_p,t) - \bar{v}_p(t)\right| r_p}{\mu_f} \tag{4.5}$$

where μ_f is the local viscosity of the fluid. The effect of droplet distortion on the coefficient of drag is discussed in Section 4.4.

By Newton's third law, the ambient fluid experiences a force equal and opposite to that experienced by each of the particles moving through it. Thus, the expression for the force acting on the fluid due to the particles may be written as follows:

$$\overline{F}_{pf}(\bar{x},t) = \sum_{all\ particles\ p} -\overline{F}_{fp}(t)\delta(\bar{x}-\bar{x}_p) \tag{4.6}$$

where $\delta(\bar{x}-\bar{x}_p)$ is the Dirac delta function centered on the particle's location, $\overline{F}_{pf}$ is the force of the particles on the fluid, and $\overline{F}_{fp}$ is the force of the fluid on the particles.

4.1.2 Implementation

The force acting on the fluid due to the particles is calculated as a collection of point forces from the aerodynamic drag force for each of the particles. The point forces are then distributed as a source to the fluid momentum equations as described in Section 3.7.2. For further implementation details of the aerodynamic drag force or the basic particle equations of motion, see Appendix C.

4.2 Spray Turbulence Interaction

The AMR code currently utilizes a k-ε model, as does the KIVA code, to model the turbulence that occurs within the fluid. When turbulence is present in the flow, the effect of the turbulence on the spray and the effect of the spray on the turbulence need to be accounted for. The spray turbulence interaction model utilizes the turbulent kinetic energy field, k, and turbulent energy dissipation field, ε, to account for these effects. The k-ε turbulence model was implemented into the AMR fluid code specifically for use in this research.

4.2.1 Theory

When turbulence is present, a fluctuating component of the fluid velocity exists in addition to the mean fluid velocity affecting the droplets. To account for this fluctuating

velocity, Equations 4.3 and 4.5 must be modified. The particle drag force and the Reynolds number calculation take into account the effect of the fluid's turbulent fluctuating velocities in the following way:

$$\bar{F}_{fp}(t) = \frac{1}{2}\rho_f C_D A_p \left| \bar{u}_f(\bar{x}_p,t) + \bar{u}'_f(\bar{x}_p,t) - \bar{v}_p(t) \right| \left(\bar{u}_f(\bar{x}_p,t) + \bar{u}'_f(\bar{x}_p,t) - \bar{v}_p(t) \right) \tag{4.7}$$

$$Re = \frac{2\rho_f \left| \bar{u}_f(\bar{x}_p,t) + \bar{u}'_f(\bar{x}_p,t) - \bar{v}_p(t) \right| r_p(t)}{\mu_f} \tag{4.8}$$

where $\bar{u}'_f(\bar{x}_p,t)$ is the fluctuating velocity encountered by particle *p*.

Just as the presence of turbulence affects the particles, the presence of the particles affects the turbulence fields. The turbulent eddies perform work on the spray, dispersing the droplets and expending turbulent kinetic energy. The source terms that the particles provide for the *k*-ε model can be expressed as follows:

$$\dot{\mathrm{K}}(\bar{x},t) = \sum_{\text{all particles } p} -\left(\bar{F}_{fp}(\bar{x}_p,t) \cdot \bar{u}'_f(\bar{x}_p,t) \right) \delta(\bar{x} - \bar{x}_p) \tag{4.9}$$

$$\dot{\mathrm{E}}(\bar{x},t) = \sum_{\text{all particles } p} -C_\varepsilon \frac{\varepsilon(\bar{x}_p,t)}{k(\bar{x}_p,t)} \left(\bar{F}_{fp}(\bar{x}_p,t) \cdot \bar{u}'_f(\bar{x}_p,t) \right) \delta(\bar{x} - \bar{x}_p) \tag{4.10}$$

Where $\dot{\mathrm{K}}$ is the source term for the turbulent kinetic energy equation, $\dot{\mathrm{E}}$ is the source term for the turbulent energy dissipation equation, and C_ε is a constant (Amsden et al., 1989).

4.2.2 Implementation

The spray turbulence interaction model accounts for the presence of turbulence by randomly sampling local turbulent fluctuating velocities for each particle. The components

of the fluctuating velocity are assumed to follow, and are sampled from, a normal distribution with a mean square deviation of 2/3 $k\left(\bar{x}_p, t\right)$ (Amsden et al., 1989).

Turbulent fluctuating velocities are assumed to be piecewise constant functions of time. Each fluctuating velocity remains in effect during a period defined by its associated turbulent timescale. The turbulent timescale is the time it would take the particle to traverse the turbulent eddy providing the fluctuating velocity or the time for the eddy itself to breakup, whichever comes first. The turbulent time scale, as determined at some time $\hat{t}$, is calculated using k and ε with the relation

$$t_p^{turb}\left(\hat{t}\right) = \min\left(\frac{k\left(\bar{x}_p, \hat{t}\right)}{\varepsilon\left(\bar{x}_p, \hat{t}\right)}, \quad \frac{C_{turb}\, k^{3/2}\left(\bar{x}_p, \hat{t}\right)}{\varepsilon\left(\bar{x}_p, \hat{t}\right)\left| \bar{u}_f\left(\bar{x}_p, \hat{t}\right) + \bar{u}_f'\left(\bar{x}_p, \hat{t}\right) - \bar{v}_p\left(\hat{t}\right)\right|} \right) \tag{4.11}$$

where C_{turb} is an empirically determined constant (Amsden et al., 1989). While technically fluid properties, the turbulent fluctuating velocities and the associated time scales are defined and assigned to each particle, given the values of the turbulent fields local to the particle.

At the beginning of each coarse timestep, enough fluctuating velocities are sampled for each particle such that the sum of the associated timescales is large enough to cover the entire timestep. The turbulent fluctuating velocities valid for a given timestep are time-averaged, using their associated time scales, to determine an effective fluctuating velocity for the timestep. This effective fluctuating velocity is used to represent the turbulent component of the fluid velocity in all spray calculations requiring the particle's relative velocity.

Further details and discussion regarding the implementation of the turbulence interaction model may be found in Appendix D.

4.3 Evaporation

The droplet evaporation model accounts for the processes that occur due to heat transfer between the droplets and the surrounding fluid. Energy conducted from the fluid to the particles results in a combination of droplet temperature change and fuel phase change from liquid to vapor. Inter-droplet effects on the evaporation process are neglected and the liquid fuel temperature is assumed uniform throughout the droplet.

4.3.1 Theory

The evaporation of a particle results in a change of the particle's droplet radius and droplet temperature. The rate of change of a single droplet's radius is given by

$$\frac{dr_d(t)}{dt} = \frac{-\rho D(T_f)}{2\rho_d r_d(t)} YR_{fv} Sh_d \tag{4.12}$$

where ρ_d is the liquid fuel density, ρD is the fuel vapor diffusivity for a fluid temperature of T_f, and

$$YR_{fv} = \frac{Y_{fv}^* - Y_{fv}(\bar{x}_p, t)}{1 - Y_{fv}^*} \tag{4.13}$$

$$Y_{fv}^* = \frac{MW_{fv}}{MW_{fv} + MW_{ave}(\bar{x}_p, t)\left(\frac{p(\bar{x}_p, t)}{p_{eq}(T_d)} - 1\right)} \tag{4.14}$$

where Y_{fv} is the fuel vapor mass fraction of the fluid, Y_{fv}^* is the fuel vapor mass fraction at the droplet surface, MW_{fv} is the molecular weight of the fuel vapor, MW_{ave} is the local average molecular weight of all other species in the fluid, p is fluid pressure, and p_{eq} is the

equilibrium fuel vapor pressure for a droplet temperature, T_d. Also, the Sherwood number, Sh_d, is calculated using the expression

$$Sh_d = \left(2 + 0.6 Re^{1/2} Sc_d^{1/3}\right) \frac{\ln\left(1 + YR_{fv}\right)}{YR_{fv}} \tag{4.15}$$

using the Schmidt number

$$Sc_d = \frac{\mu_f\left(\breve{T}\right)}{\rho D\left(\breve{T}\right)} \tag{4.16}$$

where $\breve{T} = \left(T_f + 2T_d\right)/3$ is the temperature of the fluid at the surface of the droplet.

The rate at which the temperature of the droplet changes is given by

$$m_d(t) c_{liq} \frac{dT_d(t)}{dt} = 4\pi r_d^2(t) \left(\rho_d \frac{dr_d(t)}{dt} L_{vap}\left(\breve{T}\right) + \frac{K_f\left(\breve{T}\right)\left(T_f(t) - T_d(t)\right)}{2r_d(t)} Nu_d \right) \tag{4.17}$$

where m_d is the droplet mass, c_{liq} is the specific heat of the liquid fuel, L_{vap} is the latent heat of vaporization, and K_f is the heat conduction coefficient of the fluid. The Nusselt number, Nu_d, is calculated with the expression

$$Nu_d = \left(2 + 0.6 Re^{1/2} Pr_d^{1/3}\right) \frac{\ln\left(1 + YR_{fv}\right)}{YR_{fv}} \tag{4.18}$$

and the Prandtl number by

$$Pr_d = \frac{\mu_f\left(\breve{T}\right) c_p\left(\breve{T}\right)}{K_f\left(\breve{T}\right)} \tag{4.19}$$

where c_p is the specific heat of the fluid at constant pressure.

The fluid properties are affected by droplet evaporation as well. Fuel vapor evaporated from the spray increases the fluid's total mass. The momentum of the new fuel

vapor is added to the system, as is the fuel vapor internal energy. The fluid energy also changes due to heat conduction to the spray liquid drops. These source terms to the fluid are all due to conservation principles. Any mass, momentum, or energy change in the liquid spray must be balanced by corresponding changes in the fluid.

Thus, the source terms for the fluid are given by

$$\dot{\mathrm{Y}}_{fv}(\bar{x},t) = \sum_{all\ particles\ p} \frac{dm_p(t)}{dt}\delta(\bar{x}-\bar{x}_p) \tag{4.20}$$

$$\dot{\mathrm{M}}(\bar{x},t) = \dot{\mathrm{Y}}_{fv}(\bar{x},t) = \sum_{all\ particles\ p} \frac{dm_p(t)}{dt}\delta(\bar{x}-\bar{x}_p) \tag{4.21}$$

$$\bar{F}_{evap}(\bar{x},t) = \sum_{all\ particles\ p} \frac{dm_p(t)}{dt}\left(\bar{v}_p(t)-\bar{u}_f(\bar{x}_p,t)-\bar{u}'_f(\bar{x}_p,t)\right)\delta(\bar{x}-\bar{x}_p) \tag{4.22}$$

$$\dot{\mathrm{H}}(\bar{x},t) = \sum_{all\ particles\ p} \frac{d\left(m_p(t)h_p(t)\right)}{dt}\delta(\bar{x}-\bar{x}_p) \tag{4.23}$$

where $\dot{\mathrm{Y}}_{fv}$ is the fuel vapor mass source term, $\dot{\mathrm{M}}$ is the total mass source term, $\bar{F}_{evap}$ is the force on the fluid as a momentum source, $\dot{\mathrm{H}}$ is the source term to the enthalpy of the fluid, and h_p is the particle enthalpy. By definition, the enthalpy source term corresponds to a source for the fluid temperature by the expression

$$c_p\left(T_f(\bar{x},t)\right)\frac{dT_f(\bar{x},t)}{dt} = \frac{dh_f(\bar{x},t)}{dt} \tag{4.24}$$

or

$$\dot{\mathrm{T}}(\bar{x},t) = \sum_{all\ particles\ p} \frac{1}{c_p(T_f)}\frac{d\left(m_p(t)h_p(t)\right)}{dt}\delta(\bar{x}-\bar{x}_p) \tag{4.25}$$

where $c_p(T_f)$ is the specific heat of the fluid at constant pressure.

An additional source term must be accounted for in order to maintain the incompressible, velocity divergence constraint required by the AMR code. The divergence constraint has the form

$$\nabla \cdot \bar{u}_f = \frac{1}{\rho_f}\frac{D\rho_f}{Dt} + \frac{1}{T_f}\frac{DT_f}{Dt} + MW \sum_{all\ species\ k} \frac{1}{MW_k}\frac{DY_k}{Dt} \tag{4.26}$$

where MW is the local average molecular weight of the fluid, and MW_k and Y_k are the molecular weight and local mass fraction, respectively, of species k (Bell, 1999). Thus, the source to the divergence constraint due to droplet evaporation is

$$\dot{\Psi}(\bar{x},t) = \frac{1}{T_f(\bar{x},t)}\dot{\mathrm{T}}(\bar{x},t) + \frac{1}{\rho_f(\bar{x},t)}\dot{\mathrm{M}}(\bar{x},t)\frac{MW(\bar{x},t)}{MW_{fv}} \tag{4.27}$$

or

$$\dot{\Psi}(\bar{x},t) = \sum_{all\ particles\ p}\left[\frac{1}{T_f(t)c_p(T_f(t))}\frac{d(m_p(t)h_p(t))}{dt} + \frac{MW(\bar{x}_p,t)}{\rho_f(\bar{x}_p,t)MW_{fv}}\frac{dm_p(t)}{dt}\right]\delta(\bar{x}-\bar{x}_p) \tag{4.28}$$

4.3.2 Implementation

Unlike the other spray submodels where the spray calculations are performed for an entire timestep, the evaporation routine splits the evaporation of each particle into a series of subtimesteps. Subcycling the evaporation calculations for each particle keeps large heat and mass transfer rates from producing unphysical results (Amsden et al., 1989). Evaporation is also the only submodel where the results for one particle may affect the calculation for the following particles in a single timestep.

At the beginning of the evaporation calculation, the necessary fluid properties for the entire domain are placed in temporary registers. The fluid properties at a given particle's location are interpolated from these temporary registers just prior to the particle's first evaporation subtimestep and are used to define the properties of a cell-sized region of fluid centered on the particle. As the evaporation subcycling progresses, the sources from the particle only affect this cell-sized region of fluid. Once the particle's evaporation is complete, the sources to the fluid collected during the subcycling are then distributed to the temporary fluid property registers as described in Section 3.7.2. The collected source terms are also stored for later distribution to the AMR fluid fields. The updated temporary fluid property registers are then used to determine the local fluid properties for the next particle.

The calculations performed for each subcycle are essentially identical. The particle temperature is first determined in an implicit fashion using the Secant method. The particle temperature iterations assume that the ambient fluid properties remain constant, but the thermodynamic properties that depend on particle temperature are allowed to change. Once the droplet temperature converges, the change in temperature is used to determine the new droplet radius. The changes in droplet temperature and radius are in turn used to determine the fluid property source terms.

A lower threshold on the particle mass and droplet radius was instituted to remove particles with tiny droplets from the spray. These particles have a negligible effect on the further development of the system, but require the same computational effort as particles with larger droplets. Further, particles with very low mass have little momentum to prevent them from being swept into the recirculating fluid. By removing these particles, the

computational workload is reduced and the general development of the spray is less likely to be obscured by a cloud of particles with tiny droplets.

Implementation of the evaporation submodel also required some basic changes to the AMR fluid code. The AMR code, as currently implemented, allows the total density and the species concentration of the fluid to vary through time and from cell to cell. However, the code's underlying conservation equations assume that the total fluid mass is conserved with no source terms, ie.

$$\frac{\partial \rho_f}{\partial t} + \nabla \cdot \rho_f \bar{u}_f = 0 \qquad (4.29)$$

The necessary alterations needed to include the fuel vapor source terms provided by the evaporation model were incorporated into the spray-fluid code interface. Similarly, changes were incorporated to allow the addition of source terms to the velocity divergence calculations (Equation 4.28).

The AMR code also assumes that the ambient pressure in the domain will remain constant. With the addition of mass source terms, it became desirable to introduce the possibility of a variable ambient pressure to handle those cases where the spray was evaporating inside a closed domain. This objective was never fully realized due to various extenuating circumstances. See Appendix I for discussion and implementation details about the variable ambient pressure effects and the additions to the velocity divergence calculations. Additional details about the rest of the evaporation submodel may be found in Appendix E. It should be noted that all tests that include evaporation, presented in later chapters, were performed with at least one outflow boundary, ensuring constant pressure in the domain.

4.4 Droplet Distortion and Oscillation

The droplets being represented by the computational particles are generally not rigid spheres moving through the ambient fluid, but liquid drops with a deformable surface. The shape of each droplet is affected by a combination of external aerodynamic forces, surface tension forces, and internal viscous forces. The resulting droplet behavior is analogous to a forced, damped, harmonic oscillator. The distortion and oscillation of the droplets are determined using the TAB model (O'Rourke and Amsden, 1987).

4.4.1 Theory

Droplet distortion is a non-dimensional parameter proportional to the ratio between the displacement of the droplet surface from equilibrium and the droplet radius. The acceleration of the droplet distortion, $\ddot{y}_p$, for a given particle is given by following expression:

$$\ddot{y}_p(t) = \frac{2}{3}\frac{\rho_f(\bar{x}_p,t)}{\rho_p}\frac{\left|\bar{u}_f(\bar{x}_p,t)+\bar{u}_f'(\bar{x}_p,t)-\bar{v}_p(t)\right|^2}{r_p^2(t)} - \frac{8\sigma_p(T_p(t))}{\rho_p r_p^3(t)}y_p(t) \qquad (4.30)$$

$$-\frac{5\mu_p(T_p(t))}{\rho_p r_p^2(t)}\dot{y}_p(t)$$

where ρ_p is the density of the particle's liquid fuel, T_p is the temperature of the particle, μ_p is the viscosity of the particle's liquid fuel, σ_p is the droplet surface tension. From the equation for the distortion acceleration, expressions for the distortion, y_p, and the distortion rate of change, $\dot{y}_p$, can be derived.

$$y_p(t) = e^{-\frac{t}{t_{vd}}}\left[\left(y_p(0) - \frac{We}{12}\right)\cos(\omega t) - \frac{1}{\omega}\left(\dot{y}_p(0) + \frac{1}{t_{vd}}\left(y_p(0) - \frac{We}{12}\right)\right)\sin(\omega t)\right] \quad (4.31)$$

$$+ \frac{We}{12}$$

$$\dot{y}_p(t) = e^{-\frac{t}{t_{vd}}}\left[\left(\dot{y}_p(0) + \frac{1}{t_{vd}}\left(y_p(0) - \frac{We}{12}\right)\right)\cos(\omega t) - \omega\left(y_p(0) - \frac{We}{12}\right)\sin(\omega t)\right] \quad (4.32)$$

$$+ \frac{1}{t_{vd}}\left(\frac{We}{12} - y_p(t)\right)$$

where

$$We = \frac{\rho_f(\bar{x}_p, t)\left|\bar{u}_f(\bar{x}_p, t) + \bar{u}'_f(\bar{x}_p, t) - \bar{v}_p(t)\right|^2 r_p(t)}{\sigma_p(T_d)} \quad (4.33)$$

$$t_{vd} = \frac{2}{5}\frac{\rho_p r_p^2(t)}{\mu_p(T_d)} \quad (4.34)$$

$$\omega^2 = 8\frac{\sigma_p(T_p)}{\rho_p r_p^3(t)} - \frac{1}{t_{vd}^2} \quad (4.35)$$

We is the Weber number, t_{vd} is the time scale associated with the viscous damping within the drops, and ω is the oscillation frequency (Amsden et al., 1989).

As the droplet distorts, the shape of the droplet varies from its originally assumed spherical shape to a flattened disk. This variation in shape affects the drag forces that occur between the droplet and the fluid. To account for this variation, the coefficient of drag is redefined as

$$C_D = C_{D,sphere}(1 + 2.632 y_p) \quad (4.36)$$

where $C_{D,sphere}$ is the coefficient of drag defined in Equation 4.4 (Liu et al., 1993).

4.4.2 Implementation

Implementation of the droplet distortion and oscillation mirrors the theory closely. Each particle keeps track of the droplet distortion and distortion rate of change parameters. The modification of the coefficient of drag due to droplet distortion is implemented as an option for the aerodynamic drag submodel.

4.5 Droplet Breakup

The droplet breakup model controls the atomization process within the spray. As liquid fuel enters the domain and interacts with the surrounding fluid, it breaks up into progressively smaller droplets. Although the TAB model (O'Rourke and Amsden, 1989) is used as the default breakup model in KIVA, the Reitz wave model (Reitz, 1987) and the Rayleigh-Taylor model (Patterson, 1997) have been implemented in the AMR spray code instead.

The Reitz wave breakup model has been used in place of the TAB model in the UW-ERC's modified KIVA code for several years. A comparison of the performance of the TAB model and the Reitz wave model may be found in Liu et al. (1993). The Rayleigh-Taylor model has been added to the UW-ERC's modified KIVA code in recent years. The superposition of the Rayleigh-Taylor and Reitz wave models has been shown to improve the simulation of spray breakup behavior (Patterson and Reitz, 1998).

4.5.1 Theory

4.5.1.1 Reitz Wave Model

The Reitz wave model determines how and when droplets breakup by calculating the wavelength of the fastest growing disturbances on the surface of a liquid fuel "blob" due to aerodynamic instabilities. This wavelength corresponds to the surface disturbance most likely to result in breakup, and is given by the expression

$$\Lambda_{wave} = \frac{9.02 r_p \left(1 + 0.45\sqrt{\mathrm{Z}}\right)\left(1 + 0.4 T_p^{0.7}\right)}{\left(1 + 0.865 We^{1.67}\right)^{0.6}} \tag{4.37}$$

where

$$Z = \frac{\sqrt{We_d}}{Re_d} \tag{4.38}$$

$$T = Z\sqrt{We} \tag{4.39}$$

The drop Reynolds number and the gaseous and drop Weber numbers are given by

$$Re_d = \frac{\rho_p \left| \bar{u}_f\left(\bar{x}_p, t\right) + \bar{u}'_f\left(\bar{x}_p, t\right) - \bar{v}_p\left(t\right)\right| r_p\left(t\right)}{\mu_p\left(T_p\right)} \tag{4.40}$$

$$We = \frac{\rho_f\left(\bar{x}_p, t\right)\left| \bar{u}_f\left(\bar{x}_p, t\right) + \bar{u}'_f\left(\bar{x}_p, t\right) - \bar{v}_p\left(t\right)\right|^2 r_p\left(t\right)}{\sigma_p\left(T_p\right)} \tag{4.41}$$

$$We_d = \frac{\rho_p \left| \bar{u}_f\left(\bar{x}_p, t\right) + \bar{u}'_f\left(\bar{x}_p, t\right) - \bar{v}_p\left(t\right)\right|^2 r_p\left(t\right)}{\sigma_p\left(T_p\right)} \tag{4.42}$$

To determine when breakup will occur, the wavelength's growth rate, given by the expression

$$\Omega_{wave} = \frac{0.34 + 0.385 We^{1.5}}{\left(1 + \mathrm{Z}\right)\left(1 + 1.4 T_p^{0.6}\right)}\sqrt{\frac{\sigma_p}{\rho_p r_p^3}} \tag{4.43}$$

is calculated and combined with the wavelength to provide a breakup timescale, τ_{wave}. This timescale determines the time that should pass between breakup events. The breakup timescale is given by

$$\tau_{wave} = \frac{3.788 C_{wave\_time} r_p}{\Omega_{wave} \Lambda_{wave}} \tag{4.44}$$

where C_{wave_time} is an adjustable constant that should be defined based on the characteristics of the spray nozzle being simulated (Patterson and Reitz, 1998).

The radius of the particle's droplets is determined by the relationship between the calculated wavelength and the original droplet radius. If the particle's drop radius is smaller than some fraction of the breakup wavelength, the drops in the parcel are assumed to be breaking off from the liquid core in the center of the spray. In this case, the new droplet radius for the parcel is allowed to have a radius larger than the nozzle radius and is given by

$$r_p^{new} = \min \begin{cases} \left(\dfrac{3\pi r_p^2 \left| \bar{u}_f\left(\bar{x}_p, t\right) + \bar{u}'_f\left(\bar{x}_p, t\right) - \bar{v}_p\left(t\right) \right|}{2\Omega_{wave}} \right)^{0.33} \\ \left(\dfrac{3}{4} r_p^2 \Lambda_{wave} \right)^{0.33} \end{cases} \tag{4.45}$$

Once a parcel of droplets has broken in this fashion, it is not allowed to do so again.

If the particle's drop radius is larger than some fraction of the breakup wavelength, the disturbance results in new droplets being shed from the original parcel. The radius of the new droplets is assumed proportional to the calculated wavelength, given by

$$r_p^{shed} = C_{wave\_rad} \Lambda_{wave} \tag{4.46}$$

where C_{wave_rad} is a constant (Reitz, 1987). The radius of the parent droplets is then redefined using the expression

$$r_p^{new} = \frac{\tau_{wave} r_p + dt\left(C_{wave\_rad}\Lambda_{wave}\right)}{\tau_{wave} + dt} \tag{4.47}$$

where *dt* is the current timestep.

4.5.1.2 Rayleigh-Taylor Model

The Rayleigh-Taylor breakup model also determines how and when droplets will breakup by predicting the wavelength of the fastest growing disturbances. However, the disturbances for the Rayleigh-Taylor model are due to acceleration instabilities on the droplet surface rather than aerodynamic instabilities (Patterson and Reitz, 1998). The fastest growing wavelength is given by

$$\Lambda_{RT} = 2\pi C_{RT\_rad}\sqrt{\frac{3\sigma_p}{a_p\left(\rho_p - \rho_f\right)}} \tag{4.48}$$

where $a_p = \left|\overline{F}_{fp}\right| / m_p$ is the acceleration of the particle due to aerodynamic drag and C_{RT_rad} is an adjustable constant that should be modified to account for nozzle conditions (Patterson and Reitz, 1998).

The associated breakup timescale, calculated from the frequency of the fastest growing wavelength, is given by

$$\tau_{RT} = C_{RT\_time}\sqrt{\frac{\sqrt{\sigma_p}\left(\rho_p + \rho_f\right)}{2}\left(\frac{3}{a_p\left(\rho_p - \rho_f\right)}\right)^{1.5}} \tag{4.49}$$

where C_{RT_time} is usually defined as 1.0. This timescale determines the time that should pass between breakup events. When enough time has passed for breakup to occur, the expression

$$r_p^{new} = \frac{1}{2}\Lambda_{RT} \tag{4.50}$$

is used to define the new droplet radius for the parcel.

4.5.2 Implementation

When using a breakup model, the user may use the Reitz wave submodel alone, or in conjunction with the Rayleigh-Taylor submodel. When the two models are used together, the calculations for the Rayleigh-Taylor model are performed first. If the Rayleigh-Taylor model does not result in a breakup, the particle is then manipulated by the Reitz wave model.

The implementation of the Rayleigh-Taylor breakup model has been modified from Patterson's original formulation, where breakup is only allowed to occur if the particle is a given distance from the nozzle. In a fashion similar to that introduced by Beale (1999), the Rayleigh-Taylor model is allowed to also influence those particles within the breakup distance, but uses a larger breakup time constant of $C_{RT\_time} = 9.0$.

The Reitz wave model allows a particle to break up and create smaller droplets at every timestep in a nearly continuous fashion. However, instead of creating new particles at each timestep to hold the child droplets, the accumulating amount of mass broken off from the parent particle is tracked by a placeholder. A new particle is introduced once a sufficient amount of mass, 3% of the average particle mass at the time of injection, has been shed from the parent particle (Reitz, 1987). The newly created particle is given the mass shed from the parent particle, with the droplet radius defined by Equation 4.46, and a velocity defined as the parent particle velocity with a slight perturbation. This velocity perturbation has a random orientation in the plane perpendicular to the parent particle trajectory, and has a magnitude given by the expression

$$v_{pert} = C_{wave\_vel} \Lambda_{wave} \Omega_{wave} \tag{4.51}$$

where C_{wave_vel} is a constant. It should also be noted that the radius of the parent particle's droplets are defined according to Equation 4.47, regardless of whether a new particle has been created or not.

For further implementation details of the breakup models, refer to Appendix F.

4.6 Droplet Collision

When two particles approach each other, there exists a possibility that the droplets they represent may collide. Depending on the each particle's trajectory and speed, the droplets may coalesce in a purely inelastic collision, graze each other in a nearly elastic collision, or miss each other altogether. In the spray collision models, stochastic principles are used to determine the probability of each outcome. The droplets within the same particle are assumed to be moving in the same direction with the same speed and are thus not allowed to collide.

Three different collision models have been implemented into the current spray code. The KIVA collision model, a slight variation of the collision model used in the KIVA code, checks every pair of particles that exist in a single cell to determine if a collision occurs. The NTC collision model, recently developed by Schmidt and Rutland (2000), predicts how many collisions should occur in a given cell then randomly samples collision pairs from the particles within the cell. The Proximity collision model, developed for this project in particular, examines each pair of particles within some distance of each other to determine if a collision occurs. All three models use the same logic to determine the outcome of a collision (coalescent or grazing) once the collision event has been established.

4.6.1 Theory

4.6.1.1 KIVA Collision Model

In the default KIVA collision model, the droplets of each particle are assumed to be uniformly distributed in the cell containing the particle. If two particles occupy the same cell, the probability that no collision occurs between the droplets of this particle pair is calculated.

$$P_{no\_collsion} = e^{-\nu\Delta t} \tag{4.52}$$

where Δt is the timestep length, and the collision frequency, ν , is given by

$$\nu = \frac{N_{r\_small}}{\forall}\pi\left(r_{r\_big} + r_{r\_small}\right)^2\left|\bar{v}_{r\_big} - \bar{v}_{r\_small}\right| \tag{4.53}$$

where N_{r_small} is the number of droplets in the particle with the smaller radius drops, $\forall$ is the volume of the cell that the particles occupy, and the subscripts *r_big* and *r_small* indicate whether the property is for the particle with the larger or smaller radius drops, respectively.

If the droplets collide, the character of the collision is determined by calculating a critical impact parameter based on the particle properties.

$$b_{cr}^2 = \left(r_{r\_big} + r_{r\_small}\right)^2 \min\left(1.0, \quad \frac{2.4f(\gamma)}{We_L}\right) \tag{4.54}$$

where

$$f(\gamma) = \gamma^3 - 2.4\gamma^2 + 2.7\gamma \tag{4.55}$$

$$\gamma = \frac{r_{r\_big}}{r_{r\_small}} \tag{4.56}$$

$$We_L = \frac{\rho_d\left|\bar{v}_{r\_small} - \bar{v}_{r\_big}\right|^2 r_{r\_small}}{\sigma\left(\bar{T}_d\right)} \tag{4.57}$$

$$\overline{T}_d = \frac{r_{r\_big}^3 T_{r\_big} + r_{r\_small}^3 T_{r\_small}}{r_{r\_big}^3 + r_{r\_small}^3} \tag{4.58}$$

If the impact parameter for the pair, *b* which is proportional to $\left(r_{r\_small} + r_{r\_big}\right)$, is greater than the critical impact parameter, b_{cr}, then the droplets in the particle pair just graze. In this case, each particle maintains its properties, but undergoes velocity changes in the resulting semi-elastic collision.

$$\overline{v}_{r\_big} = \frac{r_{r\_big}^3 \overline{v}_{r\_big} + r_{r\_small}^3 \overline{v}_{r\_small} + r_{r\_small}^3 \left(\overline{v}_{r\_big} - \overline{v}_{r\_small}\right)\varsigma_{graze}}{r_{r\_big}^3 + r_{r\_small}^3} \tag{4.59}$$

$$\overline{v}_{r\_small} = \frac{r_{r\_big}^3 \overline{v}_{r\_big} + r_{r\_small}^3 \overline{v}_{r\_small} + r_{r\_big}^3 \left(\overline{v}_{r\_small} - \overline{v}_{r\_big}\right)\varsigma_{graze}}{r_{r\_big}^3 + r_{r\_small}^3} \tag{4.60}$$

where ς_{graze} is an inelasticity factor given by

$$\varsigma_{graze} = \frac{b - b_{cr}}{r_{p1} + r_{p2} - b_{cr}} \tag{4.61}$$

If the pair's impact parameter is less than the critical impact parameter, then some number of droplets from the more populous particle, call it *p1*, coalesce with each droplet in the other particle, *p2*. The number of coalescences, n_c, per drop in particle *p2* is the largest integer that satisfies

$$R_{coll} < \sum_{k=0}^{n_c} \frac{\left(\nu\,\Delta t\right)^k}{k!} e^{-\nu\Delta t} \tag{4.62}$$

where R_{coll} is a randomly sampled parameter for the collision pair that was used to determine if collision occurred. The droplets in *p1* that do not coalesce maintain their properties. The properties of the coalesced droplets are mass averaged in the following fashion:

$$T_{p2} = \frac{n_c T_{p1} r_{p1}^3 + T_{p2} r_{p2}^3}{r_{p1}^3 + r_{p2}^3} \tag{4.63}$$

$$\bar{v}_{p2} = \frac{n_c \bar{v}_{p1} r_{p1}^3 + \bar{v}_{p2} r_{p2}^3}{r_{p1}^3 + r_{p2}^3} \tag{4.64}$$

$$r_{p2} = \sqrt[3]{r_{p1}^3 + r_{p2}^3} \tag{4.65}$$

4.6.1.2 NTC Collision Model

Similar to the KIVA collision model, the NTC collision model assumes that the droplets of each particle are uniformly distributed in the cell of a collision grid. This collision grid is generated independently of the fluid computational grid. Each cell of the collision grid is examined and the number of collisions that should occur, M_{cand}, is predicted using the expression

$$M_{cand} = \frac{N_p^2 \theta_{\max} \Delta t}{2\forall} \tag{4.66}$$

where N_p is the number of particles in the collision cell and $\theta_{\max}$ is given by

$$\theta_{\max} = \max_{all\ particles\ p\ in\ cell} \left((N_d)_p \right) \cdot \max_{all\ particles\ p\ in\ cell} \left(\left| \bar{v}_p \right| \right) \cdot \max_{all\ particles\ p\ in\ cell} \left(4\pi r_p^2 \right) \tag{4.67}$$

where N_d is the number of droplets in particle *p*.

M_{cand} pairs are then chosen at random with replacement as candidates for collision. For each pair, the probability of a collision occurring is given by

$$P_{coll} = \frac{\max\left((N_d)_{p1}, (N_d)_{p2} \right) \left| \bar{v}_{p1} - \bar{v}_{p2} \right| \pi \left(r_{p1} + r_{p2} \right)^2}{\theta_{\max}} \tag{4.68}$$

where the subscripts *p1* and *p2* indicate the two particles in the potential collision pair. If a collision occurs, then the collision outcome is determined in a fashion identical to that described in the previous section, with one exception. In cases with coalescence, only one droplet in each particle is allowed to coalesce and thus $n_c = 1$. Since collision pair candidates are chosen with replacement, the possibility of multiple coalescences occurring between a given collision pair does not need to be accounted for by Equation 4.62.

4.6.1.3 Proximity Collision Model

The Proximity collision model was designed to address the concern that the KIVA collision model introduces computational grid artifacts into the spray simulation. By using the computational grid cell to determine possible collision pairs, a pair of particles on opposite sides of a large grid cell may collide but neighboring particles on opposite sides of a cell boundary are not allowed to collide.

To counter this problem, possible collision pairs were chosen for the new submodel based on the proximity of the two particles from each other, irrespective of their placement relative to the computational grid. A user-defined collision radius was introduced to define the maximum distance that two particles could be from each other to be considered for collision. The equations for determining whether collision occurs, and the outcome of the collision, are nearly identical to those found in the KIVA collision model. The computational grid cell volume, $\forall$, used in Equation 4.53 is replaced by a spherical collision volume, $\forall_{coll}$, defined by

$$\forall_{coll} = \frac{4}{3}\pi r_{coll}^3 \tag{4.69}$$

where r_{coll} is the user-defined collision radius.

4.6.2 Implementation

Each of the three collision submodels begins by sorting the particles by cell with respect to a collision grid. This collision grid is a structured, Cartesian grid that covers the entire domain and does not necessarily have any connection to the fluid computational grids. For the NTC submodel, the collision grid cell size, r_{coll}, is user-defined and the location of the collision grid is perturbed randomly, every timestep, relative to the fluid computational grid. The perturbation of the collision grid location is done in an effort to reduce the possibility of collision grid effects entering the solution. The Proximity submodel also randomly perturbs its collision grid, but the cell size is defined to be $2r_{coll}$. The collision grid for the KIVA submodel corresponds directly to the fluid computational grid, both in cell size and in cell alignment. A discussion and demonstration of the effects of choosing different values for the user-defined r_{coll} is included in Chapter 7. Once the particles have been sorted, the appropriate collision submodel is applied to the particles in each collision grid cell to determine which particle pairs should collide.

The KIVA collision model iterates through every possible pair of particles within a given collision grid cell. For each pair, the probability of no collision (Equation 4.52) is calculated and a random number is sampled from a uniform distribution to determine if the pair will collide. If a collision occurs, the outcome of the collision is determined by sampling an additional random number from a uniform distribution to calculate the pair's impact parameter given by

$$b = \sqrt{Random}\left(r_{r\_big} + r_{r\_small}\right) \tag{4.70}$$

The particle properties are then adjusted as outlined in Section 4.6.1.1.

The NTC collision submodel first inspects the particles within a given collision grid cell to predict the number of collisions that will occur. If the number of collisions is larger than the possible number of unique collision pairs in the cell, the KIVA collision model is performed using the NTC collision grid. Otherwise, two uniformly distributed random numbers are then sampled for each predicted collision. These random numbers are used to select the particles for the potential collision pair. The probability of a collision between the chosen particles is then calculated (Equation 4.68) and an additional uniformly distributed random number is sampled to determine if the collision will occur. If so, the pair's impact parameter is calculated (Equation 4.70), using a fourth uniformly distributed random number, to determine the outcome of the collision. The properties of the particles are then modified as described in Section 4.6.1.1.

The Proximity submodel considers each collision cell in turn, looking for pairs of particles that lie within a distance of $2r_{coll}$ of each other. These particle pairs may consist of two particles in the current collision cell or one particle in the current collision cell and the other in a neighboring collision cell. Once a pair of particles has been identified as being in close proximity, the rest of the collision calculation proceeds as described for a potential collision pair with the KIVA collision model.

See Appendix G for more details on the implementation of the collision submodels.

Chapter 5

Submodel Validation Tests

In order to ensure that the AMR spray model would perform as expected, a suite of test problems was compiled to validate the spray model's individual physical submodels. A subset of these submodel validation tests are presented in this chapter to demonstrate the physically appropriate behavior of the submodels. The majority of the tests described here were performed without adaptation and in two dimensions, for ease of interpretation. For every test, the ambient fluid in the domain begins at rest and a summary of all the other fluid, spray, domain, and grid properties are provided for reference in Tables 5.1 - 5.4.

5.1 Equations of Motion and Aerodynamic Drag Tests

5.1.1 Conservation of Momentum

In this basic test, the viscosity of the fluid was set to zero (to inhibit viscous diffusion of the fluid velocity), a constant coefficient of drag ($C_d = 1.2$) was used, and periodic boundary conditions were used in a three-dimensional domain. A single particle was then introduced into the fluid, as illustrated in Figure 5.1. The particle interacts with the fluid through drag forces, accelerating the fluid as it slows down. The momentum of the particle, the total momentum of the fluid in the domain, and the total momentum of the system were calculated at each timestep. As can be seen in Figure 5.2, the total momentum for the system remained relatively constant.

There is a slight degradation of the total momentum believed to be due to the incomplete removal of viscous effects within the fluid. While the fluid viscosity was set to

zero for this test, the fluid does not use an inviscid form of the equations of motion and the acceleration of the fluid throughout the domain is apparent, though not shown here. This test, together with the success of the conservation of mass and energy tests discussed in Section 5.3.1, indicates that the particle-fluid interaction succeeds in conserving the momentum source terms.

5.1.2 Equations of Motion with Drag and Gravity

This test was performed in order to check the implementation of the particle's basic equations of motion with both aerodynamic drag and gravitational forces. The particle-fluid interaction was restricted such that a particle experienced forces due to the presence of the fluid, but the fluid was not affected by the passage of the particle. In this way, the behavior of the particles could be examined more closely without the intricacies involved in mutual fluid-particle interaction. Hereafter, particles will be referred to as being "non-influential" when this one-way interaction is used.

A single particle was introduced to a two-dimensional domain, as illustrated in Figure 5.3, moving perpendicular to the direction of gravity. The path of the particle computed using the AMR spray model was compared to the path obtained from solving the two-dimensional, coupled, partial differential equations using a high-order Runge-Kutta solver available in Matlab (The Math Works, Inc.). For simplicity, a constant coefficient of drag ($C_d = 1.2$) was used and gravity was given a value of $g = 9.8x10^{-2}\ m/s^2$. The results of this test, shown in Figure 5.4, suggest that the equations of motion for the particle due to drag and gravity are working correctly. Note that the particle path from the spray model shows the particle bouncing off the lower wall boundary.

5.1.3 Fluid-Particle Interaction through Drag

Given that the fluid obtains the correct magnitude of the drag force and that the particles react to drag forces correctly, it must now be shown that the fluid reacts correctly to forces it experiences due to the passage of a particle. This is very difficult to do beyond qualitative estimations. One such test captured some of the finer detail that occurs within the fluid motion due to the presence of a moving particle.

Two particles were introduced on opposite sides of a two-dimensional domain, slightly offset on opposing sides of center by $1\,cm$, and moving toward each other with parallel trajectories. See Figure 5.5 for an illustration of this test. All particle-fluid interaction was utilized so that the moving particles push the fluid ahead of them and off to the sides. A constant coefficient of drag ($C_d = 1.2$) was again used for simplicity. As the particles get closer, each particle encounters the fluid motion induced by the opposing particle. Due to their offset position, this causes the particles to deviate from their original trajectory and away from each other. Figure 5.6 shows the magnitude of the deviation with respect to the relative position of the particles. Note that the scale of the deviation is much smaller than that of the domain.

5.2 Turbulence Effects Test

The spray turbulence effects model was tested by placing a large number of particles at a single point within a two-dimensional domain where the fluid has constant, non-zero, turbulent kinetic energy and dissipation fields ($k = 1x10^{-5}\ m^2/s^2$, $\varepsilon = 3.75x10^{-5}\ m^3/s^2$). The particles disperse from their initial position due to the presence of the turbulent

fluctuating velocity components. The particles in this test were again set to be non-influential to remove inter-particle effects via the fluid and a constant coefficient of drag ($C_d = 1.2$) was used. Since the turbulent fluctuating velocities are sampled from a normal distribution, the average distance of the particles from their original position should initially grow quadratically in time, then continue to grow linearly with respect to time (Monin and Yaglom, 1971). The spread of the particles due to turbulence is demonstrated in Figure 5.7 and the mean distance of the particles from their initial position versus time is plotted in Figure 5.8.

5.3 Droplet Evaporation Tests

5.3.1 Conservation of Mass and Energy

A single particle was placed in the center of a three-dimensional domain and allowed to evaporate in order to determine if mass and energy source terms are conserved. No turbulence was present and complete fluid-particle interaction was used. One wall of the domain was defined as an outflow boundary while the rest were solid, adiabatic walls. At each timestep during the evaporation, the mass and energy of the particle, the mass and energy of the fluid and the total mass and energy of the system were calculated. Since the single outflow boundary allows a constant ambient pressure to exist, the fluid and particle enthalpies were used for the conservation of energy tests.

As can be seen in Figures 5.9 and 5.10, the total mass and total energy of the system both remain constant during the evaporation, indicating that conservation of these quantities is achieved. The sudden drop in the particle mass and enthalpy corresponds to the point where the remainder of the particle vaporizes. It should also be noted that since mass,

momentum and energy source terms are distributed using the same routines, that the success of the mass and energy conservation test partially validates the hypothesis that the slight drop in the total system momentum seen in Figure 5.2 is due to viscosity effects and not faulty source term distribution.

5.3.2 D^2 Law Test

As with the mass and energy conservation test, a particle containing a single droplet was placed in the center of a domain and allowed to evaporate. The particle was set to be non-influential to remove the effects of a changing local fluid composition due to evaporation, and the particle temperature was forced to remain constant to remove liquid fuel temperature effects. By the so-called "D^2 Law", the surface area of a single evaporating droplet is expected to decrease linearly with time (Williams, 1990). Figure 5.11 shows the square of the droplet diameter through time as it evaporates, which indeed shows linear behavior until the droplet radius becomes small enough that it essentially evaporates away. The curve at the very end, for very small droplet radii, is an artifact of the very low thresholds chosen for removing particles with small radii and little mass.

5.4 Particle Collision Test

To test the performance of the three different collision submodels, a modified version of a test used by Schmidt and Rutland (1999) was performed. This test compares the number of collisions predicted by a specified collision model with the number of collisions obtained from the analytical expression for the integral of the collision probability.

The domain for this test consisted of a two-dimensional, 4x4 grid with solid walls. The spray consisted of N_p particles uniformly distributed throughout the domain. The radii of the drops represented by the particles were sampled uniformly from the interval $[0, r_{\max})$ where $r_{\max} = 5x10^{-7}\, m$. Likewise, the x-component of the particle velocities were sampled uniformly from the interval $[0, v_{\max})$ where $v_{\max} = 1\, m/s$, and the y-component was set to zero. A single timestep of 10^{-4} s and a collision grid cell volume, $\forall_{IJK} = 1.56x10^{-8}\, m^3$, were also used. The particles represented approximately 10^8 total droplets, but the number of droplets assigned to each particle was determined by random sampling from a uniform distribution.

Over a single timestep, the number of predicted droplet collisions was counted in each collision cell and totaled. For this test, the number of droplet collisions between two colliding particles was defined as the number of drops in the less populous particle, and the outcomes of the collisions were neglected. The number of predicted droplet collisions was then compared to the expected number of collisions in each cell, calculated from the following analytical expression (Schmidt and Rutland, 1999):

$$M_{theory} = \sum_{IJK} \frac{7\pi\, \Delta t\, v_{\max} r_{\max}^2 \left(N_d\right)_{IJK}}{36 \forall_{IJK}} \tag{5.1}$$

where the subscript *IJK* indicates a single collision cell, and $\left(N_d\right)_{IJK}$ is the number of droplets in the collision cell. The relative error was then calculated using

$$\varepsilon_{rel} = \frac{\left| M_{theory} - M_{predicted} \right|}{M_{theory}} \tag{5.2}$$

Since the collision model is highly stochastic in nature, this relative error was averaged over fifty independent runs in order to minimize random effects.

As can be seen in Figure 5.12, the three versions of the collision submodel have comparable performance. Also, note that the performance of the submodels improves as the number of particles representing the droplets increases. The $O\left(1/\sqrt{N_p}\right)$ trend of the relative error matches the trend reported by Schmidt and Rutland (2000).

5.5 Droplet Breakup Tests

The performance of the breakup submodels is difficult to demonstrate in a direct and deterministic manner. Thus, the performance of the breakup models is not demonstrated here, but may be ascertained from the non-evaporative spray tests for which experimental data is available. Since these tests required the use of the turbulence effects and collision submodels to obtain reasonable results, as well as the use of adaptation to keep the runtimes low while maintaining adequate grid resolution, the results of these non-evaporating spray tests are presented in Chapter 8.

Section	Species	Pressure (*MPa*)	Temperature (*K*)
5.1.1	N_2	1.0	300
5.1.2	Air	0.1	300
5.1.3	Air	0.1	300
5.2	Air	0.1	300
5.3.1	N_2	1.0	900
5.3.2	N_2	1.0	900
5.4	N_2	0.1	300

Table 5.1 The primary properties of the ambient fluid for the tests associated with each section.

Section	Species	Liquid Density (kg/m^3)	Drop Radius (m)	Initial Temp. (K)	Initial Speed (m/s)	Initial Number of Particles	Total Mass (kg)
5.1.1	$C_{14}H_{30}$	760	$5.0x10^{-4}$	300	10.0	1	$3.979x10^{-10}$
5.1.2	$C_{14}H_{30}$	720	$5.0x10^{-5}$	300	0.01	1	$3.77x10^{-10}$
5.1.3	$C_{14}H_{30}$	720	$5.0x10^{-5}$	300	0.01	1	$3.77x10^{-10}$
5.2	$C_{14}H_{30}$	720	$5.0x10^{-5}$	300	0.0	1000	$3.77x10^{-7}$
5.3.1	$C_{14}H_{30}$	760	$2.0x10^{-5}$	400	0.0	1	$2.547x10^{-9}$
5.3.2	$C_{14}H_{30}$	760	$2.0x10^{-5}$	400	0.0	1	$2.547x10^{-9}$
5.4	$C_{14}H_{30}$	760	variable	300	variable	variable	$5.0x10^{-7}$

Table 5.2 The primary properties of the spray for the tests associated with each section.

Section	Domain Size (*m* x *m* x *m*)	Boundary Conditions		
		x-dir (lower/upper)	y-dir (lower/upper)	z-dir (lower/upper)
5.1.1	0.096x0.096x0.384	no-slip/no-slip	no-slip/no-slip	periodic/periodic
5.1.2	0.02x0.02	no-slip/no-slip	no-slip/no-slip	
5.1.3	0.04x0.02	no-slip/no-slip	no-slip/no-slip	
5.2	0.02x0.02	no-slip/no-slip	no-slip/no-slip	
5.3.1	0.064x0.064x0.064	no-slip/no-slip	no-slip/no-slip	outlet/no-slip
5.3.2	0.064x0.064x0.064	no-slip/no-slip	no-slip/no-slip	no-slip/no-slip
5.4	0.04x0.04	no-slip/no-slip	no-slip/no-slip	

Table 5.3 The properties of the physical domain and boundary conditions for the tests associated with each section.

Section	Number of Grid Cells	Domain "depth" (*m*)
5.1.1	8x8x32	
5.1.2	64x64	1
5.1.3	128x64	1
5.2	64x64	1
5.3.1	16x16x16	
5.3.2	16x16x16	
5.4	4x4	0.001

Table 5.4 The properties of the computational grid for the tests associated with each section.

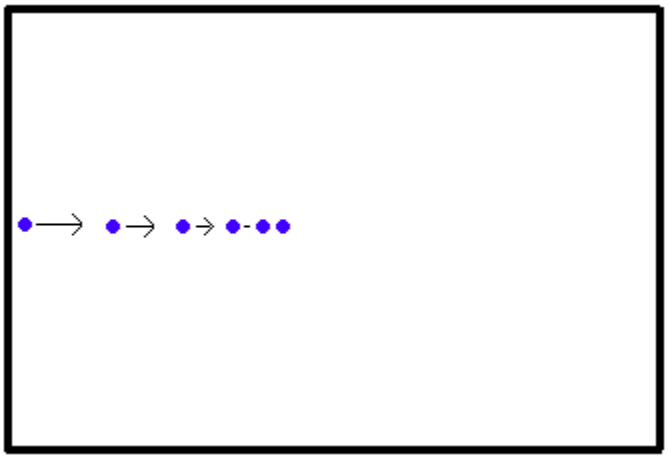

Fig. 5.1 An illustration of the conservation of momentum test. A single particle is introduced into a box with zero viscosity.

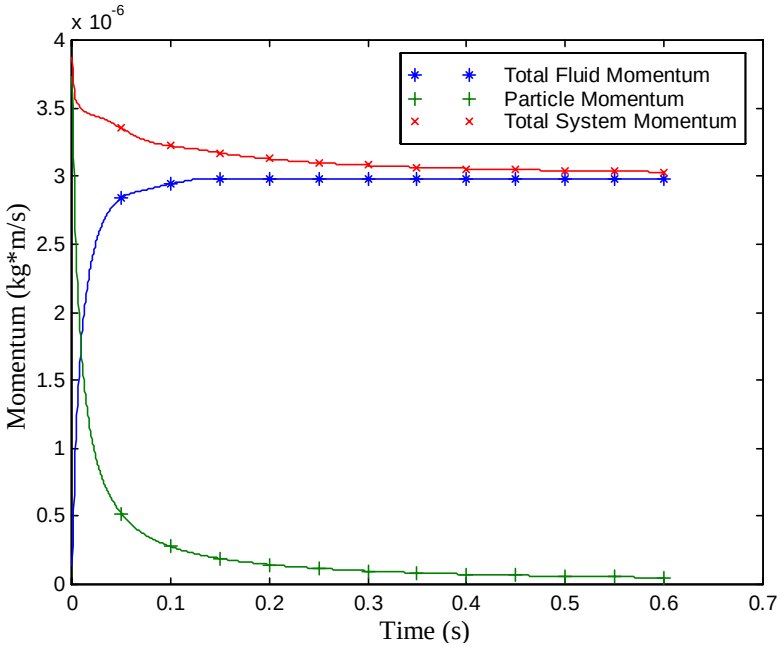

Fig. 5.2 Results for the conservation of momentum test. The total system momentum is the sum of the particle and fluid momentum.

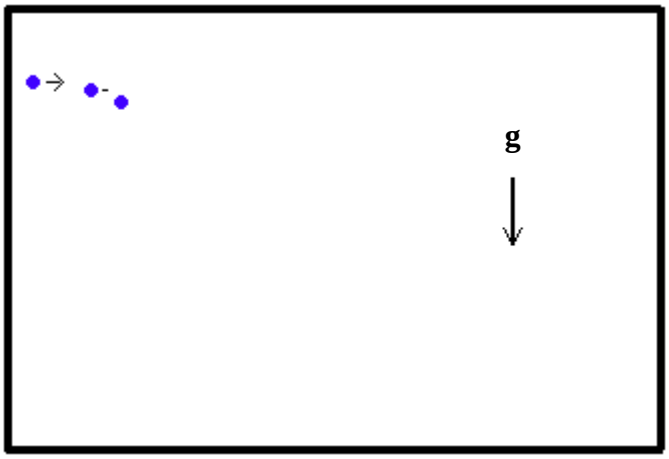

Fig. 5.3 Illustration of the test for drag and gravity forces. A single particle is introduced into a domain perpendicular to the direction of a gravitational force.

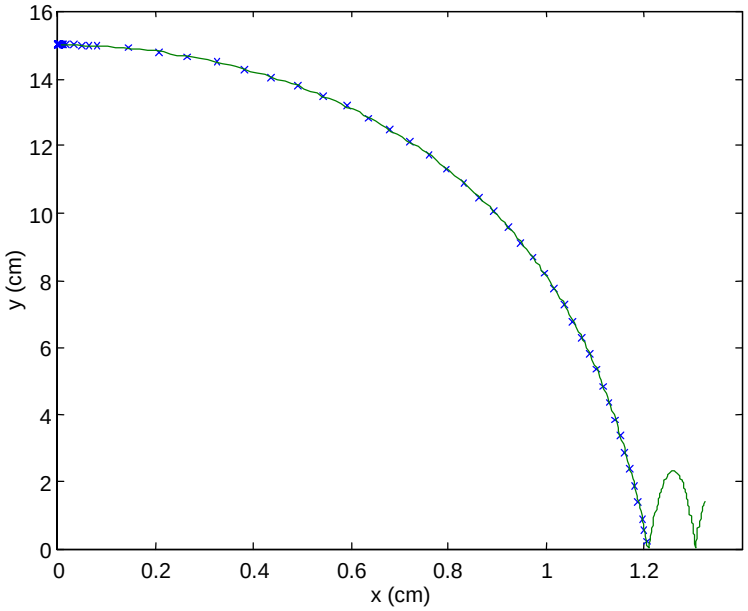

Fig. 5.4 Comparison of the particle paths calculated for the test of drag and gravity forces. The solid line indicates the path of the particle calculated by the AMR spray code. The crosses show the particle path calculated directly from the differential equations of motion using the same gravitational force. The lower boundary is a solid wall.

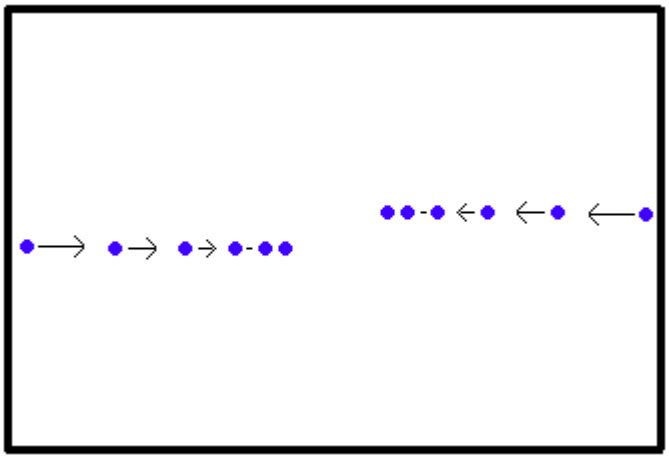

Fig. 5.5 Illustration of the fluid-particle interaction test. Two particles are introduced at opposite ends of a box of fluid at rest, vertically offset symmetrically from the domain's centerline.

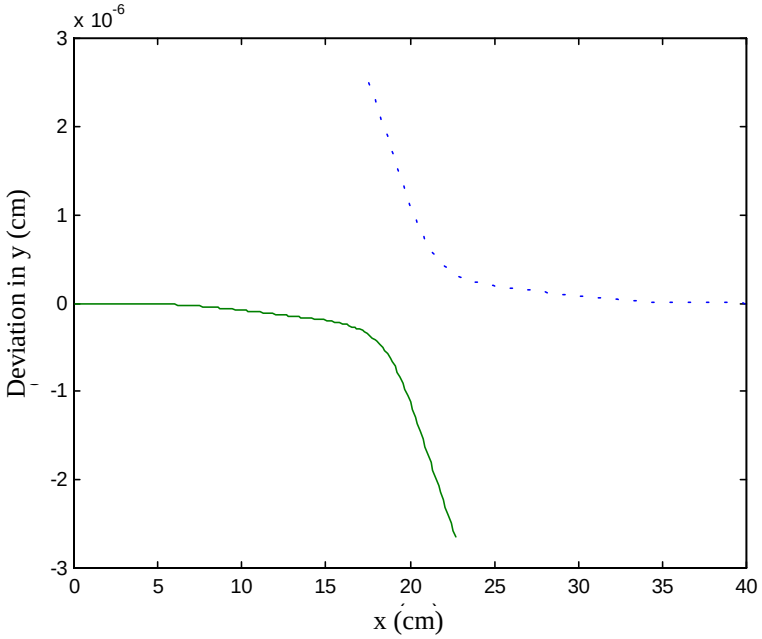

Fig. 5.6 Results for the fluid-particle interaction test. The deviations of the particles in the y-direction are plotted vs. their x-position in the domain. Note the difference in scale of the two axes.

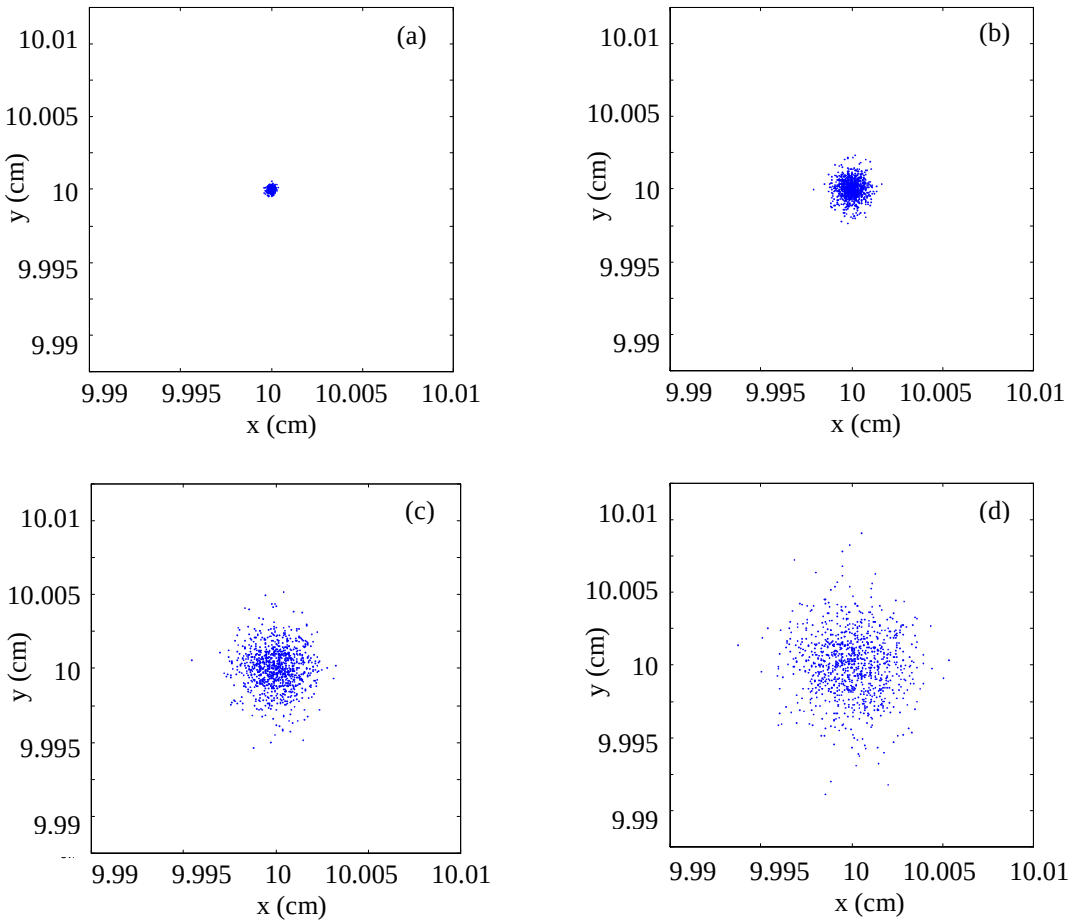

Fig. 5.7 Location of the particles for the turbulence effects test. 1000 non-influential particles are placed at the center of a box and allowed to diffuse due to turbulent fluctuating velocities. Constant values of k and ε were used. The above graphs, (a)-(d), show the position of the particles after times of 1, 3, 5 and 7 seconds, respectively.

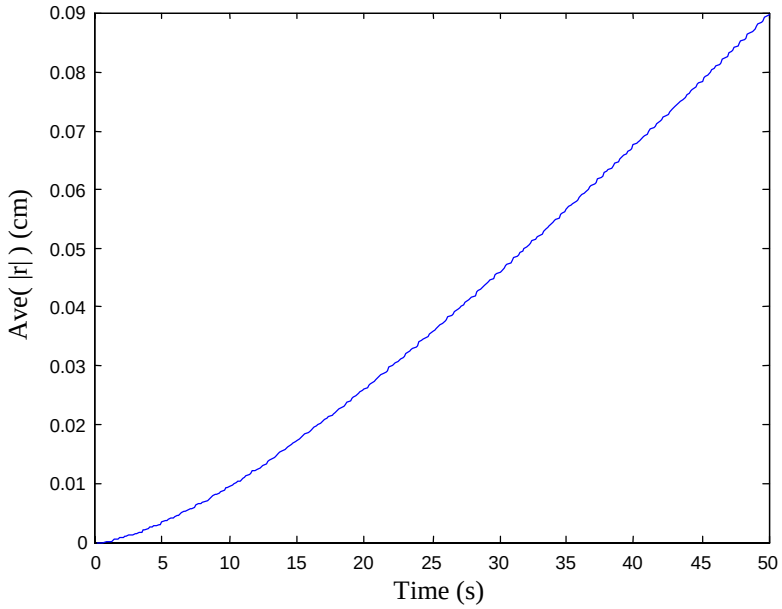

Fig. 5.8 The average particle displacement from the center, in the turbulence effects test, as a function of time.

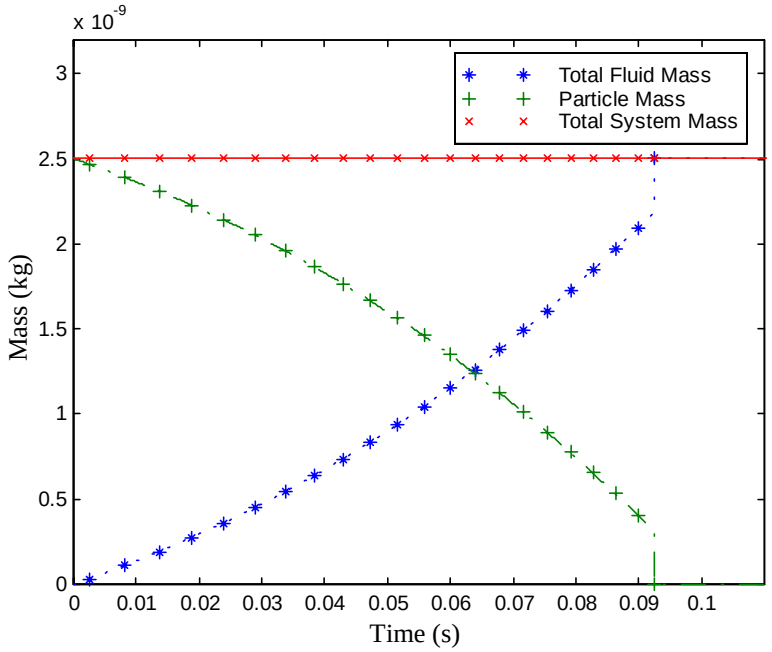

Fig. 5.9 Results for the conservation of mass test.

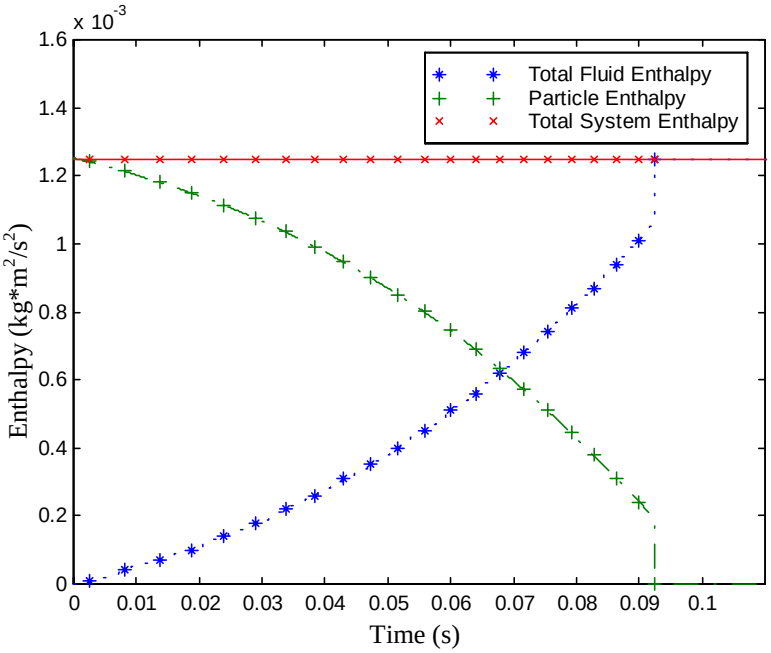

Fig. 5.10 Results for the conservation of energy test. The fluid enthalpy used to obtain the 'total fluid enthalpy' and the 'total system enthalpy' is the enthalpy added/removed from the fluid since the beginning of the computational run.

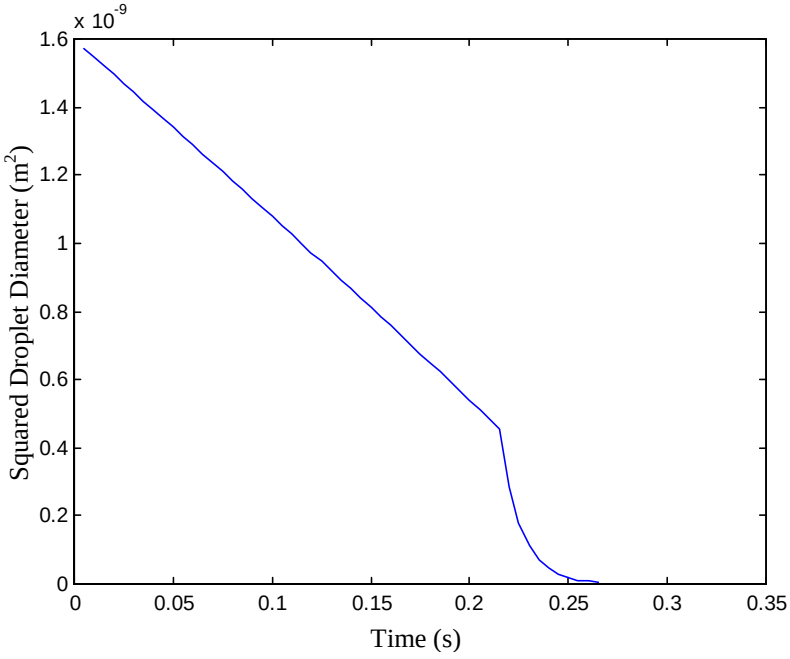

Fig. 5.11 Results for the D^2 Law test.

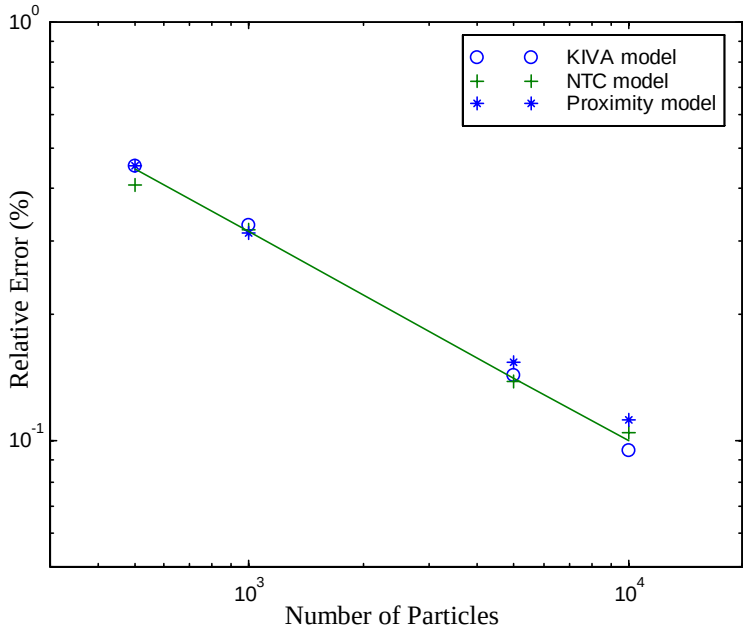

Fig. 5.12 Results for the collision incidence test. The solid line indicates a slope of $1/\sqrt{N_p}$.

Chapter 6

Adaptive Grid Issues

Adaptive mesh refinement is characterized by its use of multiple grids of varying refinement. Single grid protocols for the spray model, discussed in Chapters 3 and 4, generally work well for adaptive cases where the particles are far from coarse-fine grid boundaries. When a particle is adjacent to or crossing a coarse-fine grid boundary, additional protocols must be in place to avoid introducing grid related artifacts into the solution. While most of the spray submodels are essentially independent of a particle's proximity to a boundary, some of the spray models require specific modifications. This chapter is devoted to discussing the spray model issues associated with the use of adaptation and the special provisions and modifications needed in the spray model implementation to handle multiple grids.

6.1 Basic Assumptions

As mentioned at the beginning of Section 4.1, the volume of fluid displaced by the spray droplets is commonly neglected by using the assumption that the particles occupy a single point in space. This assumption is appropriate as long as the size of the cell containing the particle has dimensions much larger than the dimensions associated with the spray droplets. In the past, for single, fixed grids, this has not been an issue since computational cost requirements limited the grid resolution that could be used. With the introduction of adaptive refinement of the grids, we run into the possibility of refining a grid to the point where the validity of this assumption is called into question.

The problem of whether neglecting droplet volume is a valid assumption can be addressed by restricting the smallest cell dimension allowed for the finest grid or by accounting for the displacement of the fluid by the spray droplets in the fluid calculations. The latter option would be preferable in the long run, as well as more accurate, but is not included in the current implementation of the spray model. Further, the code does not contain any explicit restrictions on the cell size, but leaves the choice of smallest cell size to the user's good judgment. The smallest dimension of the cell size used in the simulations presented here is no smaller than ten times the largest droplet radius.

6.2 Spray Refinement Criteria

Routines providing criteria for grid refinement based on the spray have been added to collection of refinement criteria for fluid properties. Refinement criteria that have been added include: the presence of one or more particles in a given cell (as determined by non-zero particle source terms in a grid cell), and the magnitude of any desired spray source term.

Currently, all data used for determining if refinement criteria are satisfied, are fluid property fields already in use by the AMR code for other purposes. It is not clear how to implement refinement criteria based on data that is not in a form that corresponds to the AMR grids in use. Although not currently implemented, a user could create additional criteria based directly on particle properties by creating empty fields with the grid structure of the current level, then filling these fields with spray data. Examples of spray data that could be put in each cell include the maximum particle velocity, maximum (or minimum) droplet radius, number of particles (or droplets), etc.

6.3 Coarse-Fine Boundary Conditions

Internal boundary conditions, corresponding to coarse-fine boundaries, should be defined such that the boundary is essentially transparent to a particle in its vicinity. For example, a particle crossing such a boundary should not experience any deviation of trajectory or velocity magnitude. However, these boundaries introduce issues regarding which level information should be used to update the state of a particle as it passes from a coarse grid to a fine grid or vice versa.

As a particle passes from a coarse grid to a fine grid, the particle is introduced into the fine grid calculations at the point that the particle is predicted to be located within a fine grid at the start a fine timestep. In this case, the coarse level particle information is linearly interpolated in time and space to determine the particle state at the beginning of the fine timestep. When a particle passes off a fine grid onto a coarser grid, the state of the particle reverts to the state computed by the coarser level timestep. This simple scheme ensures that particles do not need to be included in fine level calculations until they are needed, though they may still affect the fluid on the finer level grid through source terms calculated on the coarser level, as discussed in Section 6.4. Further, it precludes the need to interpolate coarse grid fluid data or extrapolate fine grid fluid data when advancing the state of a particle that has passed off a fine grid during a fine grid timestep.

6.4 Particle Source Distribution

When distributing source terms to the fluid, you ideally want to approximate the localized nature of the particle source as best as you can. Particles within the interior of a grid can use the default source distribution template that utilizes the local grid cell size as

described in Section 3.6.2, irrespective of refinement level. Near coarse-fine boundaries, source distribution needs to take into consideration the difference in refinement between the adjacent levels, whether the particle is near one or more fine grid corners, and how to ensure that all grids get the appropriate amount of the source term.

The template that was chosen to distribute sources near coarse-fine grid boundaries is similar to the template used for a single grid. The source is uniformly distributed over an area equivalent to a coarse grid cell centered over the particle, regardless of whether the particle is located on the fine grid or just adjacent to it. The coarse grid distributes the source the same way as in the single grid case, but the fine grid distributes the source according to the fine grid cells that fall under the particle's coarse cell sized distribution basis function. Source terms for the fine grid from particles on the coarse grid are interpolated in time and space from the coarser grid timestep information. See Figures 6.1 and 6.2 for illustrations of this distribution template. This scheme provides a continuity of source distribution as a particle passes between coarse and fine grids and does not require special logic to determine distribution near a fine grid corner.

6.5 Spray Turbulence Interaction

As discussed in Section 4.2, the spray accounts for the effect of turbulence on the spray droplets by sampling a series of turbulent fluctuating velocities (and their associated timescales) to obtain an effective fluctuating velocity to use in all spray calculations for the given timestep. With the introduction of adaptation, multiple timesteps are performed on a fine grid for a single timestep on a coarse grid. This brings up a few issues regarding which

fluctuating velocities to use to calculate the effective fluctuating velocity for the finer timesteps.

For a generic level L, we sample the number of fluctuating velocities needed such that the sum of the associated timescales occupy the level L timestep, dt^L. For the first timestep for level L+1, the fluid turbulence fields of k and ε are the same as was used for the level L timestep, so we can use a subset of the same fluctuating velocities such that the associated timescales occupy the shorter level L+1 timestep, dt^{L+1}. The turbulence fields are updated during the first level L+1 timestep, and we'd like to update our fluctuating velocities, as necessary, to reflect the change in the local turbulence. In this case, only the last fluctuating velocity from the previous fine level timestep is kept, and additional fluctuating velocities are sampled to fill the remainder of the timestep. The effective fluctuating velocity is then recalculated using the new values.

6.6 Particle Collision

The use of adaptation provides the opportunity to refine the search for colliding particle pairs. For each of the three collision models, the collision grid cell size is refined using the same refinement ratios as are used for the computational grids.

For the KIVA model, where the collision grid corresponds directly to the computational grids, collisions only occur between particles in the same collision cell on the same grid level. The collision grid for the NTC and Proximity collision models is randomly perturbed relative to the computational grid, so collision cells may cover portions of both a coarse grid and a fine grid. Since the collision model is performed at the end of the timestep

for a given level, particles on finer grids collide with each other before particles on the coarser grids are allowed to collide (recall the timestep ordering discussed in Section 2.3.2).

For a given level L, the particles on both the level L and L+1 grids are sorted according to the level L collision grid. The level L particles in a collision grid cell are allowed to collide with each other first. The level L particles are then allowed to collide with the particles on the level L+1 grids in the same collision cell. The Proximity model also checks for collisions between particles in a collision cell entirely on the level L grid and particles on level L+1 grids that lie in neighboring collision cells. In this way, the coarse-fine grid boundaries should be transparent to the collision model results while still taking advantage of the finer resolution where possible.

6.7 Creation and Destruction of Particles

The addition of adaptation required some rethinking on the order in which various models were performed. In the KIVA implementation of the spray model, droplet breakup and collision were performed at the very beginning of each timestep. New particles, resulting from breakup, were created and used in the remainder of the timestep calculation and particles with zero mass due to droplet coalescence were removed from the calculation. Within the AMR framework, the creation and removal of particles at the beginning of the timestep becomes problematic. Fine level timesteps essentially recalculate the processes that occur during a portion of the coarser level timestep. It is not evident what should be done with newly created particles or particles with zero mass when recalculating the results of breakup and collision on a finer level. By placing the collision and breakup calculations at the end of the timestep, after fine level computations have been completed, collision and

breakup are performed only once on the finest level for each particle over a given time interval.

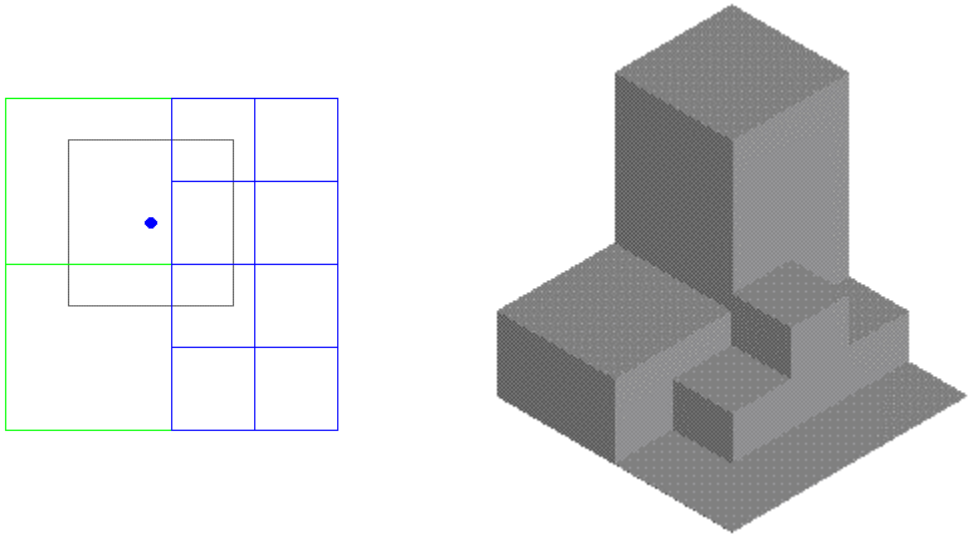

Fig. 6.1 Illustration of the two-dimensional, source distribution template for a particle on a coarse grid near a coarse-fine boundary.

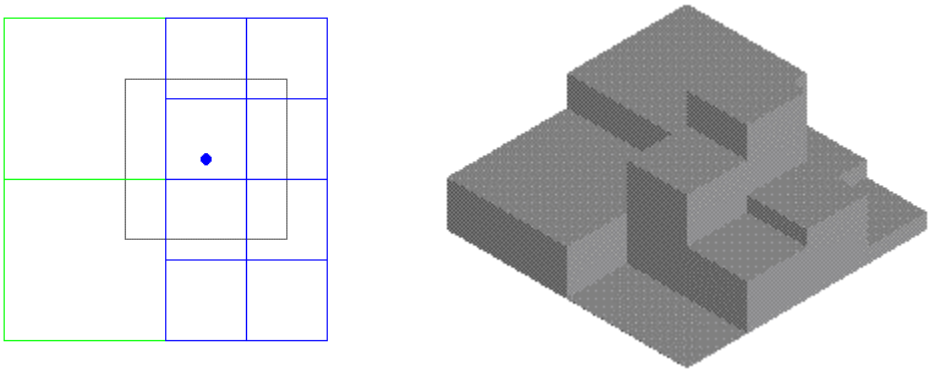

Fig. 6.2 Illustration of the two-dimensional, source distribution template for a particle on a fine grid near a coarse-fine boundary.

Chapter 7

Adaptivity Validation Tests

The addition of adaptivity and the protocols for handling particles near coarse-fine grid boundaries required that additional testing be performed to validate the code. Beyond the prerequisite tests to ensure that the spray's administration routines were working correctly within the adaptive grid framework, tests where adaptation could affect the behavior of the physical submodels were also needed. This chapter contains a subset of the tests performed to ensure that the use of adaptation did not negatively affect the spray behavior. A summary of the input parameters for each test in this chapter is provided in Tables 7.1 - 7.4.

7.1 Coarse-Fine Boundary Traversal

One of the basic requirements of adaptive grid methods is that the presence of internal grid boundaries should not adversely affect the resultant solution. Thus, a coarse-fine boundary should be transparent to a particle traveling across the grids. To determine if this requirement is satisfied, a single, non-influential particle was introduced to a two-dimensional domain with solid walls. The domain had one refined grid ($r = 2$) in the upper right quadrant of the domain and the particle path was defined to intersect with the coarse-fine grid boundary. As shown in Figure 7.1, the path of the particle did not deviate as it crossed the coarse-fine boundary. Further, the solid wall boundary condition, described in section 3.6, was enforced correctly, unaffected by the particle's presence on the finer grid. The magnitude of the velocity of the particle with respect to time, shown in Figure 7.2, was

also unaffected by the coarse-fine grid boundary traversal and the contact with the domain wall.

7.2 Conservation Tests

The distribution of source terms when adaptation is present, as described in Section 6.4, should preserve the conservation of mass, momentum, and energy. The source distribution template for particles in the interior of a fine grid is identical to the template for single grids; so, conservation tests with adaptive grids should naturally concentrate on cases with particles near coarse-fine boundaries. Though not shown here, mass, momentum, and energy are conserved for particles remaining in the interior of fine grids.

7.2.1 Conservation of Momentum

The basic conditions of this test are identical to those used in the single grid version of the conservation of momentum test described in Section 5.1.1, with the exception of the presence of a refined grid. A single grid, of refinement factor 2, was placed along the particle path such that half of the momentum source term at each timestep would fall onto the grid cells covered by the finer grid. The fluid and particle momentums are recorded with respect to the coarsest level grid at the end of each complete timestep, after the fine grid results have been averaged back down to the coarser grid. The results of this test are shown in Figure 7.3. As with the single grid version of this test, there is a slight loss of momentum assumed to be due to viscous effects. However, the presence of finer grids does not appear to affect the conservation of momentum.

7.2.2 Conservation of Mass and Energy

The conditions of this test are identical to those used in the single grid version of the conservation of mass and energy test described in Section 5.3.1, with the exception of a refined grid being added. As with the conservation of momentum test, a single refined grid ($r = 2$) was placed such that half of the source terms resulting from evaporation would fall onto the grid cells covered by the finer grid. Again, the quantities in question were recorded at the end of each complete timestep on the coarser grid. The plots of mass and energy, as seen in Figures 7.4 and 7.5, respectively, demonstrate that the presence of finer grids does not affect the conservation of mass or energy.

7.3 Turbulence Effects Test

The turbulence parameters are resampled for particles on finer grids as the fluid turbulence fields are updated, as described in Section 6.5. To ensure that updating effective fluctuating velocities for particles during fine timesteps does not significantly affect the behavior of the spray due to turbulence effects, the single grid turbulence effects test of Section 5.2 was repeated with a refined grid ($r = 2$) placed in the upper right quadrant of the domain. As shown in Figures 7.6 and 7.7, the presence of the refined grid did not appear to affect the dispersion of the particles. The difference between the average displacement curves for the single grid and adapted cases, not shown here, falls within the bounds of random variation obtainable for the single grid case.

Section	Species	Pressure (*MPa*)	Temperature (*K*)
7.1	Air	0.1	300
7.2.1	N_2	1.0	300
7.2.2	N_2	1.0	900
7.3	Air	0.1	300

Table 7.1 The primary properties of the ambient fluid for the tests associated with each section.

Section	Species	Liquid Density (kg/m^3)	Drop/Nozzle Radius (m)	Initial Temp. (K)	Initial Speed (m/s)	Initial Number of Particles	Total Mass (kg)
7.1	$C_{14}H_{30}$	720	$5.0\text{x}10^{-5}$	300	0.01	1	$3.77\text{x}10^{-10}$
7.2.1	$C_{14}H_{30}$	760	$5.0\text{x}10^{-4}$	300	10.0	1	$3.979\text{x}10^{-10}$
7.2.2	$C_{14}H_{30}$	760	$2.0\text{x}10^{-5}$	400	0.0	1	$2.547\text{x}10^{-9}$
7.3	$C_{14}H_{30}$	720	$5.0\text{x}10^{-5}$	300	0.0	1000	$3.77\text{x}10^{-7}$

Table 7.2 The primary properties of the spray for the tests associated with each section.

Section	Domain Size (m x m x m)	Boundary Conditions		
		x-dir (lower/upper)	y-dir (lower/upper)	z-dir (lower/upper)
7.1	0.05x0.05	no-slip/no-slip	no-slip/no-slip	
7.2.1	0.096x0.096x0.384	no-slip/no-slip	no-slip/no-slip	periodic/periodic
7.2.2	0.064x0.064x0.064	no-slip/no-slip	no-slip/no-slip	outlet/no-slip
7.3	0.02x0.02	no-slip/no-slip	no-slip/no-slip	

Table 7.3 The properties of the physical domain and boundary conditions for the tests associated with each section.

Section	Level 0 Grid	Number of Fine Grid Cells (Refinement Ratio)	Domain "depth" (*m*)
7.1	64x64	64x64 (2)	1
7.2.1	8x8x32	4x8x64 (2)	
7.2.2	8x8x8	8x8x4 (2)	
7.3	64x64	64x64 (2)	1

Table 7.4 The properties of the computational grids for the tests associated with each section.

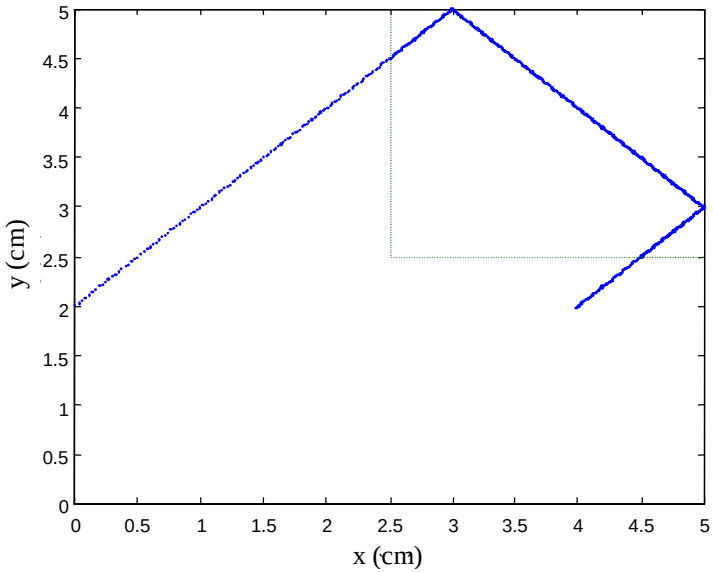

Fig. 7.1 The path of the particle for the coarse-fine grid boundary traversal test. A single non-influential particle is introduced at $(x, y) = (0, 2)$. The upper right quadrant is refined by a factor of 2.

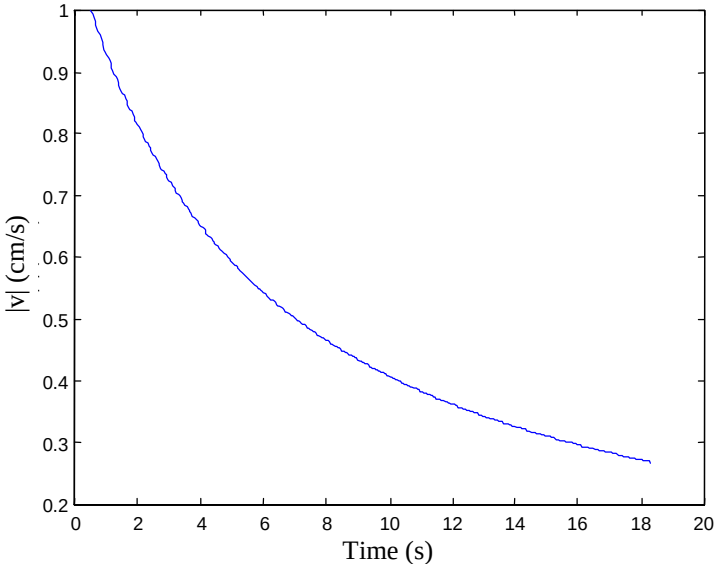

Fig. 7.2 The magnitude of the particle's velocity with respect to time for the coarse-fine grid boundary traversal test. The particle crosses the coarse-fine boundary at times of 5.25 and 15.5 seconds, and encounters a wall at times of 6.5 and 13.25 seconds.

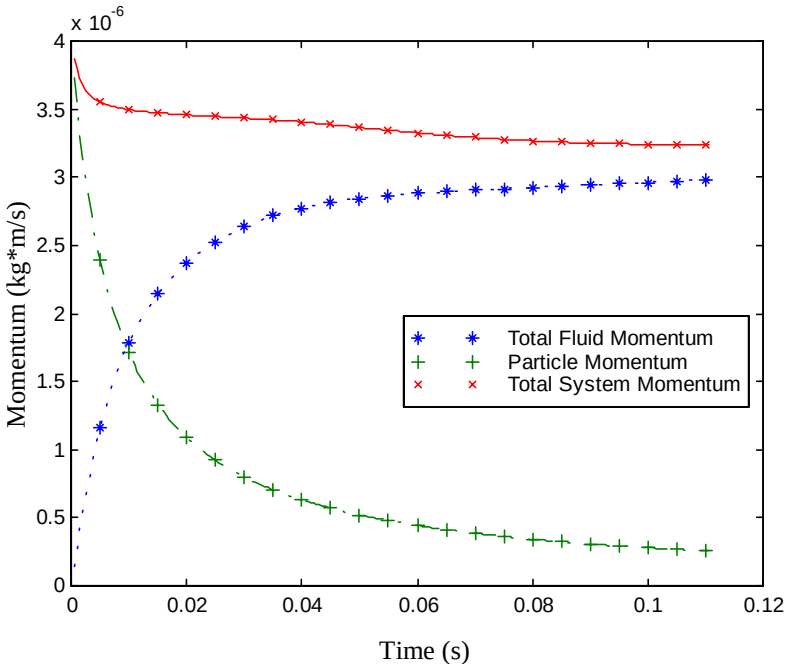

Fig. 7.3 The conservation of momentum test with partial grid adaptation. The total fluid momentum and total system momentum sum the fluid momentum over the entire domain.

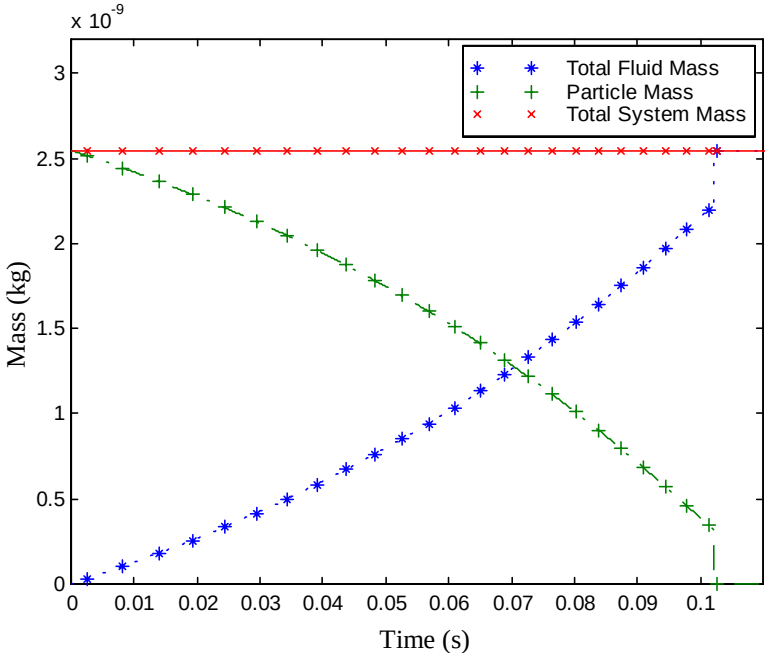

Fig. 7.4 The conservation of mass test with partial grid adaptation.

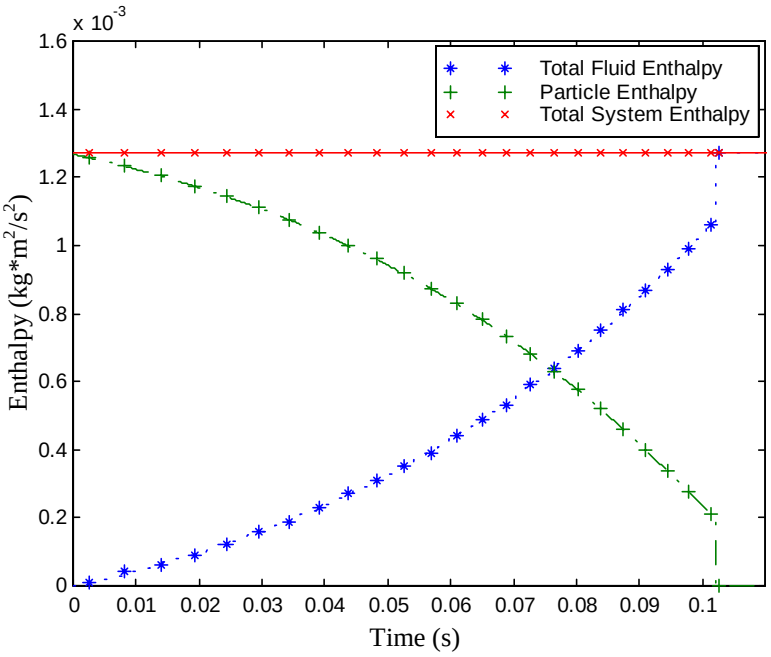

Fig. 7.5 The conservation of energy test with partial grid adaptation. The total fluid enthalpy and total system enthalpy sum the fluid enthalpy over the entire domain, and remove the initial total fluid enthalpy.

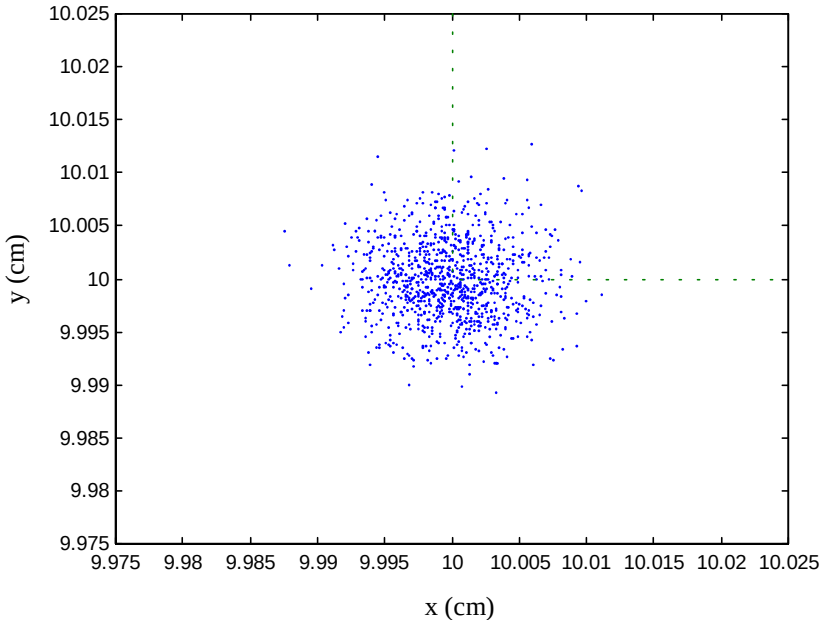

Fig. 7.6 Location of the particles for the adaptive turbulence effects test. One thousand non-influential particles are placed at the center of a box and allowed to diffuse due to turbulent fluctuating velocities. The above graph shows the position of the particles after 7 seconds. Constant values of k and ε were used. The upper right quadrant is refined by a factor of 2.

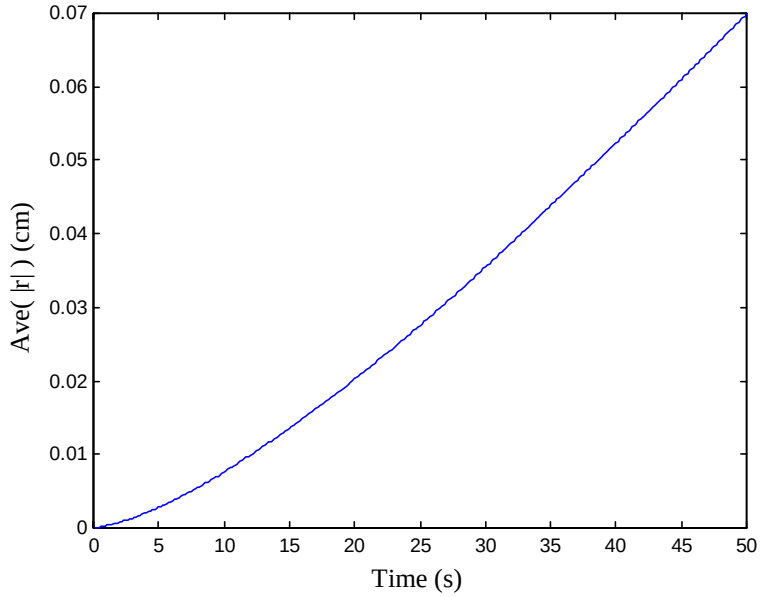

Fig. 7.7 The average particle displacement from the center, in the adaptive turbulence effects test, as a function of time.

Chapter 8

Complete Spray Model Tests

In order to be useful when simulating sprays for which there is no prior information, a spray model must be able to correctly reproduce spray results for which there is experimental data. Previous chapters have established that individual components of the AMR spray model are working correctly, but not necessarily that the spray models work in concert as expected. In this chapter, results obtained from numerically simulated sprays with and without evaporation are compared to selected experimental results from the spray literature.

8.1 General Simulation Information

Each numerical run presented here utilized all of the available spray submodels and the code's full adaptive grid capabilities. To reduce computational time, the majority of the calculations were performed in two dimensions. One three-dimensional, non-evaporative case was simulated for the purposes of comparison.

The simulations described in this chapter have a few characteristics in common. Each spray originates in the center of the upper boundary, with the centerline of the spray aligned with the gravitational vector. The size of each fluid domain was defined to be sufficiently large enough to minimize the effect of the walls on the spray development. The simulations used three levels of refinement, each with a refinement factor of two. Further, the initial grids were defined such that the finest level grids had a cell size of $1mm$ in each direction (a typical cell size for similar KIVA simulations). An upper threshold for the fluid vorticity field and the presence of non-zero spray source terms were used for the adaptive

grid refinement criteria. The NTC collision model was used with a collision grid cell size on the finest level of $dx_{coll} = 6.25x10^{-5} m$, and the breakup model consisted of the combined Reitz Wave and Rayleigh-Taylor models. Finally, and consistent with all of the experimental sprays described in this chapter, the fluid in each spray simulation began at rest. Configuration details about the varying spray, fluid and grid properties of each simulation are summarized and provided for reference in Tables 8.1 - 8.5.

The spray physical submodels have a complex interaction. The changes in the spray due the modification of a single submodel parameter can affect the subsequent performance of the other submodels. A full optimization of the spray submodel parameters to match simulation results with experimental data is beyond the scope of the present work. Instead, only a partial adjustment of the parameters was undertaken to demonstrate that the AMR spray code implementation is working appropriately.

8.2 Non-evaporating Sprays

The evaporation submodel was turned off for all non-evaporating spray test cases. These tests not only demonstrate the performance of the AMR spray code for cold spray bomb cases, but also validate the performance of the spray breakup models. For the purposes of comparison with experimental data, liquid fuel penetrations of the simulated, non-evaporating sprays were defined as the distance from the nozzle to the particle furthest from the nozzle in the domain.

8.2.1 Alloca et al.

In the study done by Alloca et al. (1992), the spray tip penetration and Sauter Mean Diameter (SMD) of non-evaporating, high-pressure sprays were measured then compared to numerical results obtained from KIVA. The three cases studied were differentiated by changes in fluid pressure and temperature. The initial conditions of the simulated spray and fluid were obtained from the Alloca et al. paper, where available, and from the input files used for Patterson's thesis (1997). The parameters for the breakup model (described in Appendix F) were tuned to obtain the best match between the experimentally and numerically obtained penetration curves for case B, the base case from which the other two vary. These breakup parameters were then used for all three cases.

The penetrations simulated in two dimensions and the experimentally measured penetrations are plotted as functions of time in Figure 8.1. In spite of attempts to more closely match the penetration curves for case B, the numerically calculated penetration curve was consistently low. After fixing the various tunable breakup parameters, it appears that the AMR spray code gave a reasonable approximation for case C as fluid temperature was increased, but overestimated the penetration as fluid pressure decreased for case A. It is reassuring to note, perhaps, that the error between penetrations obtained with experiment and the AMR spray code are no larger than the error between experimental penetrations and the KIVA simulated penetrations, reported in Alloca et al. Further, the calculated penetrations appear to follow the basic trends indicated by the experimental data as pressure and temperature are altered.

Case B was also simulated in three dimensions for comparative purposes. A comparison of the penetrations obtained with two-dimensional and three-dimensional

calculations, shown in Figure 8.2, substantiates the use of two-dimensional calculations to demonstrate AMR spray code performance when using spray penetration as the primary measure. Despite the similarity of the penetration profiles, it should be noted that the side view of the two and three-dimensional sprays, as can be seen in Figure 8.3, appear only vaguely similar. Most apparent is the difference in the number of particles present, in spite of the same number of particles injected. The apparent profusion of particles for the three-dimensional case is probably due to a reduced number of particle collisions. The spray at a given distance from the nozzle is split among many more collision grid cells for the three-dimensional simulation, thus decreasing the likelihood of particle collisions.

8.2.2 Farrell et al.

The work by Farrell, Chang and Su (1996) involved the measurement of spray penetration, spray cone angle, and SMD of non-evaporating, high-pressure sprays with single and multiple injections under varying initial conditions. For the purposes of comparing the AMR spray code results to experiment, only measured penetrations for single injection cases were considered. Of the six single injection cases performed by Farrell et al., four were simulated with the AMR spray code. Three cases (Farrell's cases 4, 5, and 6) varied fluid pressure over a range of 0.827 *MPa* to 1.654 *MPa*. Farrell's case 2 was also simulated to provide a comparison with case 5 in which the injection pressure, and thus the initial spray velocity, was varied. The initial conditions of the fluid and spray were primarily obtained from the Farrell et al. paper, where available, and from the input files for Patterson's thesis (1997). The breakup parameters were the same as those used for the Alloca et al. cases in the previous section.

Figure 8.4 shows the experimentally measured and numerically calculated penetrations of the spray, as functions of time, for the three cases where only the fluid pressure has been changed (cases 4-6). The AMR spray code appears to capture the approximate magnitude of the spray penetration for case 4, but does a poor job of replicating the shape of the penetration curve as a function of time, particularly at later times. The penetration curve shapes are better represented for the two cases with higher fluid pressures, but the magnitudes of the spray penetration are underestimated.

The spray penetration comparison for those cases where only the initial spray velocity was altered (cases 2 and 5) is shown in Figure 8.5. The drop in spray penetration for the slower spray (case 2) is captured by the AMR spray code, but the magnitude of the drop is not as great as indicated by the experimental results.

As with the Alloca et al. simulations, the penetrations obtained from simulating the sprays do not closely match the experimentally measured penetrations, but demonstrate the correct trends as the fluid pressure and injection velocity are varied. It should be noted that the average difference between numerically and experimentally obtained penetration profiles is on the same order as the Alloca et al. cases.

8.3 Evaporating Sprays

For the vaporizing spray cases, the liquid fuel penetrations of the simulated sprays were defined as the distance from the nozzle to a particle such that the area within an arc passing through the particle and centered on the nozzle contained 90% of the liquid fuel in the domain. Likewise, the penetration of the simulated spray's fuel vapor is defined as being

the distance equivalent to the radius of the arc centered on the nozzle demarking an area containing 90% of the fuel vapor in the domain.

8.3.1 Naber and Siebers

In the study reported by Naber and Siebers (1996), the effects of ambient gas density and fuel evaporation on the penetration of the spray were explored. The fuel vapor penetrations measured for vaporizing spray cases with five different ambient gas densities, varying from 3.3 kg/m^3 to 58.6 kg/m^3, were used for comparison with AMR spray code results.

The parameters used to initialize the AMR spray simulations were obtained primarily from the Naber and Siebers paper. The remaining physical and computational parameters were obtained from the input files used for Beale's thesis (1999). As gas density is not a definable parameter for the AMR code, the fluid pressures corresponding to the given densities and temperature were calculated for use. The breakup parameters used were a combination of those used for the non-evaporative cases and the breakup parameters used by Beale.

Figure 8.6 shows the numerically and experimentally obtained fuel vapor penetration profiles for the five different fluid densities. The AMR spray code captures the trend of increasing penetration rate as the fluid density decreases, but consistently overestimates the penetration as a function of time. The calculated fuel vapor penetration profiles also successfully captures the appropriate shape, indicating that the rate of penetration through time is approximately correct.

8.3.2 Kamimoto et al.

Kamimoto et al. (1987) explored the effects of modifying the injection pressure, and thus the initial spray velocity, on a number of different measurable spray features of vaporizing and non-vaporizing sprays. For this study, only the data from the evaporation cases were considered and the experimentally measured liquid fuel and fuel vapor penetrations were used for comparison with the AMR spray code results.

Most of the initial conditions for the simulated evaporating sprays were obtained from the paper. The breakup parameters and spray cone angle were adjusted as needed to obtain the desired penetration profiles for the entire set of cases (see Table 8.3). Specifically, the spray cone angle had the greatest effect on the fuel vapor penetration rate, increasing the penetration as the cone angle decreased, and the time parameter for the Reitz Wave model had the largest influence on the liquid fuel penetrations, dropping the average liquid penetration as the breakup time parameter was reduced.

A comparison of the experimental and calculated fuel vapor penetrations may be found in Figure 8.7. As with the Naber and Siebers cases, vapor penetrations calculated by the AMR spray code generally overestimate the measured penetrations as functions of time. However, the vapor penetrations of the simulated sprays exhibit the appropriate trend of increasing penetration as the spray injection velocity increases.

The experimental liquid fuel penetrations, as can be seen in Figure 8.8, appear to oscillate slightly about some constant distance from the nozzle, regardless of the initial spray velocity. The liquid fuel penetrations calculated by the AMR spray code initially level off at a distance slightly less than the average of the measured penetrations, but then begin to show oscillations of increasing magnitude. Not only are these simulated oscillations much greater

than those displayed in the experimental results, but the oscillations appear to begin earlier as the initial spray velocity increases. It is believed that these oscillations are connected to the Rayleigh-Taylor breakup model. Though not shown here, increasing the CRT_rad constant of the breakup model (described in Appendix F) generally increases the magnitude and decreases the period of the oscillations in the calculated liquid fuel penetrations.

8.3.3 Siebers

Siebers (1998) investigated the effects of altering a variety of different parameters on the liquid fuel penetration for evaporating sprays. One of the tests reported in the paper involved varying both the fluid temperature and fluid density, using five values for each. Of the twenty-five experimental cases reported, nine of them, resulting from choosing three temperatures and three densities, were simulated with the AMR spray code for comparison purposes. As with the other experimental cases mentioned in this chapter, most of the information used to initialize the spray simulations was obtained directly from the paper. The remaining parameters were obtained from the input files for Beale's thesis (1999). Given the oscillating behavior of the calculated liquid penetration observed in the Kamimoto et al. cases, the Siebers simulations were only run long enough to establish an average liquid penetration length.

The comparison of the estimated liquid penetrations calculated with the AMR spray code with the penetrations reported by Siebers is shown in Figure 8.9. The set of calculated liquid penetrations succeeds in reflecting the changes due to varying fluid density, but the AMR spray code did not capture the variations in liquid fuel penetration due to fluid temperature changes. This can also be seen in the profiles of the nine calculated liquid fuel

penetrations, plotted in Figure 8.10. The irresponsiveness of the spray code to the fluid temperature may be due to interplay between the evaporation and breakup models; however, attempts to attain the desired effect on liquid penetration due to temperature by modifying the breakup parameters were unsuccessful.

8.4 Additional Comments

8.4.1 Runtimes

The runtimes associated with the full spray simulations were disappointingly long. For example, on an SGI Origin 2000 with 300 MHz processors, the average run for a two-dimensional Alloca et al. case took approximately two days to complete and a Kamimoto et al. case took an average of approximately seven days. The three-dimensional run of the Alloca et al. case took nearly three weeks to get the presented results. Similar KIVA runs generally take much less that 24 hours, and complex AMR runs without sprays also typically take much less time per timestep.

Changing the number of particles in the simulation, say increasing the initial number of particles from 1000 to 5000, had little noticeable effect on the overall run time. Conversely, increasing the number of coarse grid cells by refining the grids by a factor of two significantly increased runtimes. It should be noted that the use of adaptation was still highly beneficial in keeping runtimes relatively low for a desired grid density in the region of the spray. For example, the two-dimensional Alloca et al. case took approximately five days to complete when using a single fine grid as opposed to the two days when using adaptation.

Some known factors are partially responsible for the unusually long runtimes. The AMR fluid code can be compiled with or without additional information that facilitates

debugging. All of the runs performed for this work used executables that contained this additional information. Including the debugging information increases the runtimes by about a factor of two. Another factor is the output file format for the fluid information. In order to facilitate displaying spray and fluid information simultaneously, the output files were written in ASCII. Though typically not a large factor for two-dimensional cases, the difference in file creation time when using ASCII versus a binary format can be significant for three-dimensional cases. Finally, the integration of the spray model with AMR is itself partially responsible for the long runtimes when compared with similar KIVA spray simulations. Since the spray is always located on the finest level of grids, submodel routines that would only be performed once per fine grid timestep for a KIVA spray simulation are performed additional times for the coarser level timesteps in an AMR spray simulation.

Other additional factors are only hypothesized to negatively impact the runtimes. For example, midway through this project, the AMR fluid code was optimized for use with parallel processors. The spray part of the code does not include the capability for parallel processing, and so all simulations were performed with a single processor. It is not clear how the changes in the AMR code from serial to parallel have affected the runtimes, but it is possible that additional overhead was introduced. A second possibility is the choice of solvers from those available for the AMR fluid code. It is conceivable that the solvers used for the spray simulations were not the most appropriate choice given the character of the resulting fluid flows. This could result in additional time spent in iterative solvers that may not otherwise be needed.

8.4.2 Instabilities

There were two basic instabilities most likely to interrupt the full spray simulations. One of these instabilities typically manifested itself as a warning about an “unstable criterion” for the conjugate gradient solver. In these cases, modifying the tolerances for the timestep routines such that the average timestep was decreased usually solved the problem.

The second, and more common, occurrence involved regions of the turbulent diffusion field increasing beyond the maximum limit. As with the conjugant gradient instability, this problem was alleviated by reducing the size of the timesteps used in the simulations. Another source of the turbulent diffusion instability was the presence of coarse-fine boundaries crossing areas of steep gradients in the turbulence fields. This provided an additional reason to tag cells for refinement such that the entire spray, and the immediate area around it, lay within the same level of grids.

Section	Case	Species	Pressure (*MPa*)	Temperature (*K*)
8.1.1	A	N_2	0.1	300
	B		1.7	300
	C		1.7	480
8.1.2	2	Ar	1.378	300
	4		0.827	
	5		1.378	
	6		1.654	
8.2.1	1	N_2	0.989	1000
	2		2.018	
	3		4.126	
	4		8.488	
	5		17.392	
8.2.2	All	N_2	3.0	900
8.2.3	1_1	N_2	0.748	700
	1_3		1.068	1000
	1_5		1.389	1300
	3_1		3.075	700
	3_3		4.393	1000
	3_5		5.710	1300
	5_1		12.258	700
	5_3		17.511	1000
	5_5		22.645	1300

Table 8.1 The primary properties of the ambient fluid for the tests of each section.

Section	Case	Species	Liquid Density (kg/m^3)	Nozzle Radius (m)	Initial Temp. (K)	Initial Speed (m/s)	Total Mass (kg)
8.1.1	All	$C_{14}H_{30}$	760	7.96×10^{-5}	298	377.72	1.6×10^{-5}
8.1.2	2	$C_{14}H_{30}$	868	1.163×10^{-4}	293	141.58	2.667×10^{-5}
	4					190.73	
	5					190.73	
	6					190.73	
8.2.1	All	$C_{14}H_{30}$	703	1.23×10^{-4}	452	419	5.6×10^{-5}
8.2.2	30	$C_{13}H_{28}$	786.5	8.0×10^{-5}	433	198.56	12.8×10^{-6}
	50					246.62	13.0×10^{-6}
	80					311.13	13.0×10^{-6}
	110					349.70	11.0×10^{-6}
8.2.3	All	$C_{14}H_{30}$	710	1.23×10^{-4}	438	500	8.44×10^{-5}

Table 8.2 The primary properties of the spray for the tests of each section.

Section	Case	Initial Number of Particles	Reitz Wave Breakup		Rayleigh-Taylor Breakup		Cone Angle (*deg*)
			bw_time	bw_rad	CRT_dist	CRT_rad	
8.1.1	All	1000	70	0.6	0.2	3.0	9
8.1.2	All	1000	70	0.6	0.2	3.0	13
8.2.1	1	3000	70	0.6	0.02	0.1	7.3
	2						10.38
	3						14.74
	4						20.93
	5						28.96
8.2.2	All	3000	30	0.6	0.03	0.1	50
8.2.3	1_5	3000	70	0.6	0.085	0.1	7.3
	1_3						
	1_5						
	3_1				0.050		14.74
	3_3						
	3_5						
	5_1				0.015		28.96
	5_3						
	5_5						

Table 8.3 The spray's computational parameters for the tests of each section.

Section	Domain Size (*m* x *m* x *m*)	Boundary Conditions		
		x-dir (lower/upper)	y-dir (lower/upper)	z-dir (lower/upper)
8.1.1	0.064x0.128	no-slip/no-slip	outlet/no-slip	
8.1.1 (3-D)	0.064x0.064x0.128	no-slip/no-slip	no-slip/no-slip	outlet/no-slip
8.1.2	0.064x0.128	no-slip/no-slip	outlet/no-slip	
8.2.1	0.064x0.128	no-slip/no-slip	outlet/no-slip	
8.2.2	0.064x0.192	no-slip/no-slip	outlet/no-slip	
8.2.3	0.064x0.128	no-slip/no-slip	outlet/no-slip	

Table 8.4 The properties of the physical domain and boundary conditions for tests associated with each section.

Section	Level 0 Grid	Refinement Ratios			Domain "depth" (*m*)
		Level 1	Level 2	Level 3	
8.1.1	8x16	2	2	2	0.001
8.1.1 (3-D)	8x8x16	2	2	2	
8.1.2	8x16	2	2	2	0.001
8.2.1	8x16	2	2	2	0.002
8.2.2	8x24	2	2	2	0.002
8.2.3	8x16	2	2	2	0.002

Table 8.5 The properties of the computational grids for the tests associated with each section.

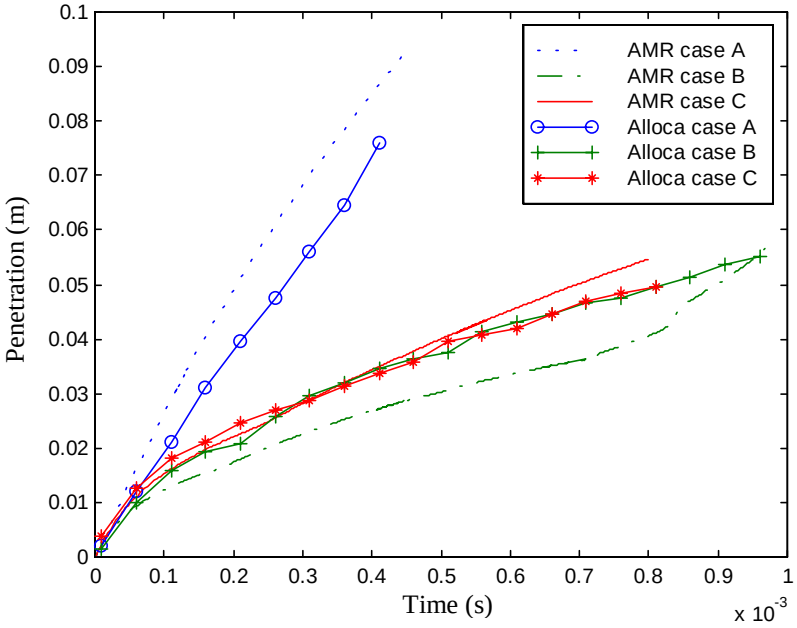

Fig. 8.1 Spray penetration comparison with Alloca et al. experimental measurements. Cases A and B are differentiated by fluid pressure. Cases B and C are differentiated by fluid temperature.

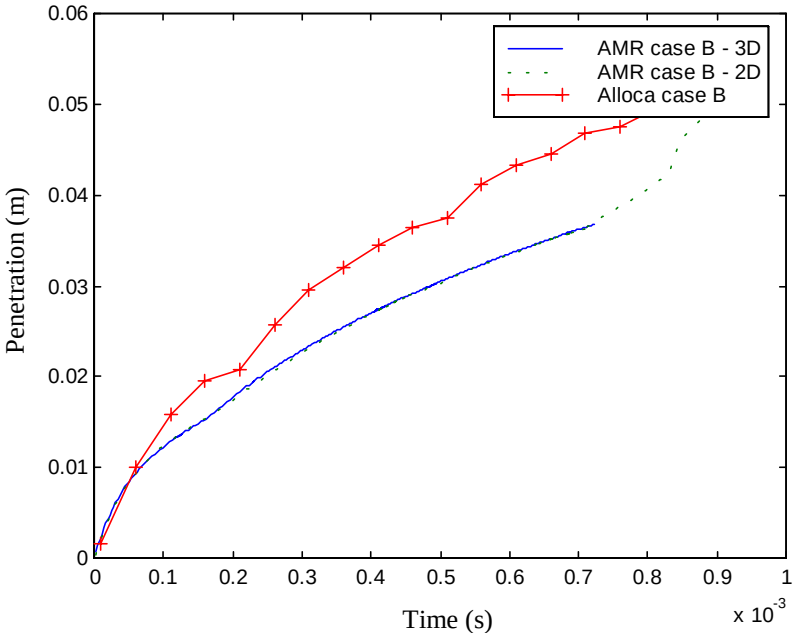

Fig. 8.2 Spray penetration comparison of numerically simulated Alloca case B in two and three dimensions. The experimentally measured penetration is also included.

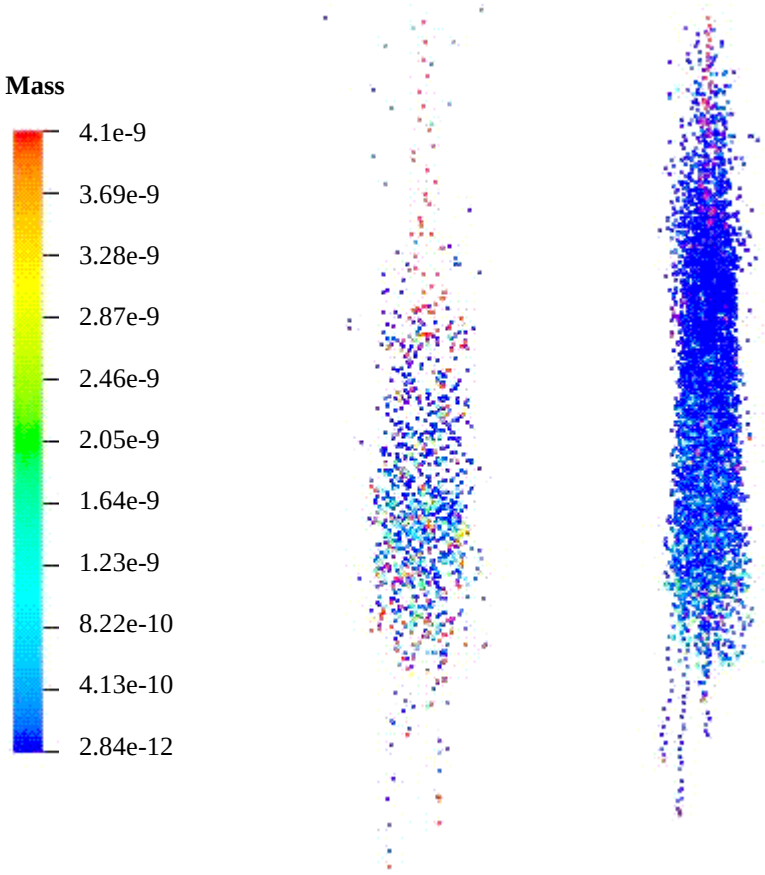

Fig. 8.3 Comparison of the spray for the numerically simulated Alloca case B in two and three dimensions. The three-dimensional spray is on the right and the color key indicates the mass of the spray particles in *kg*.

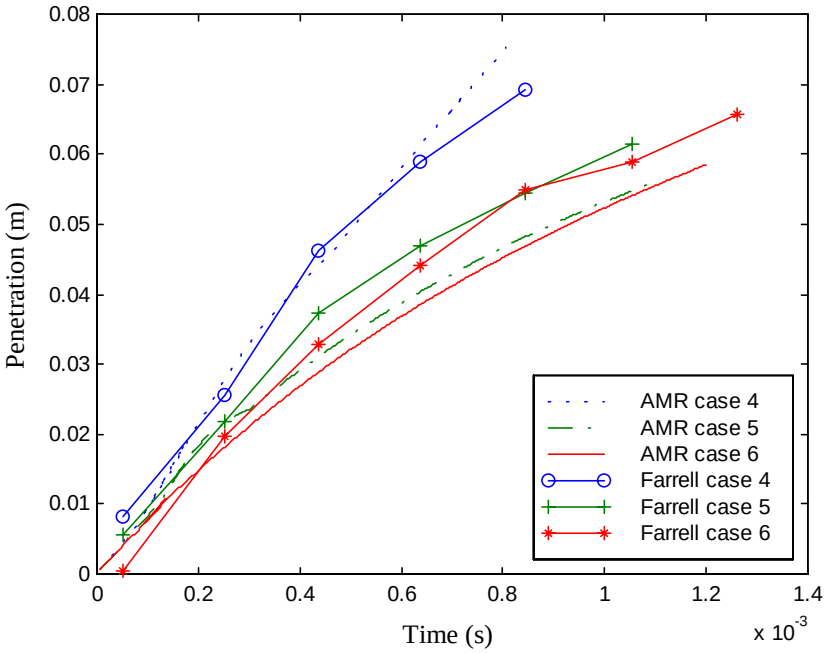

Fig. 8.4 Spray penetration comparison with Farrell et al. experimental measurements. The fluid pressure increases from case 4 to case 6.

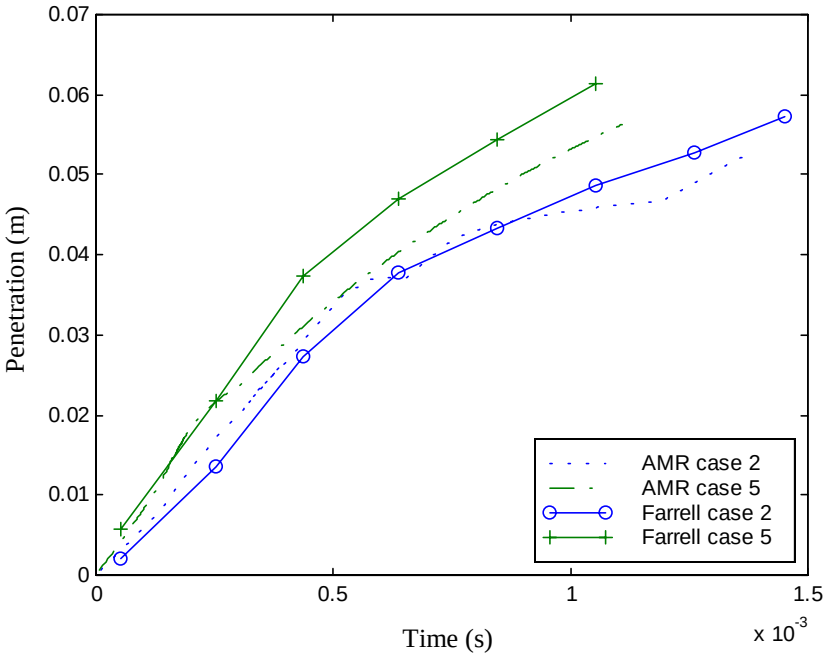

Fig. 8.5 Spray penetration comparison with Farrell et al. experimental measurements. The initial spray velocity in case 5 is greater than in case 2.

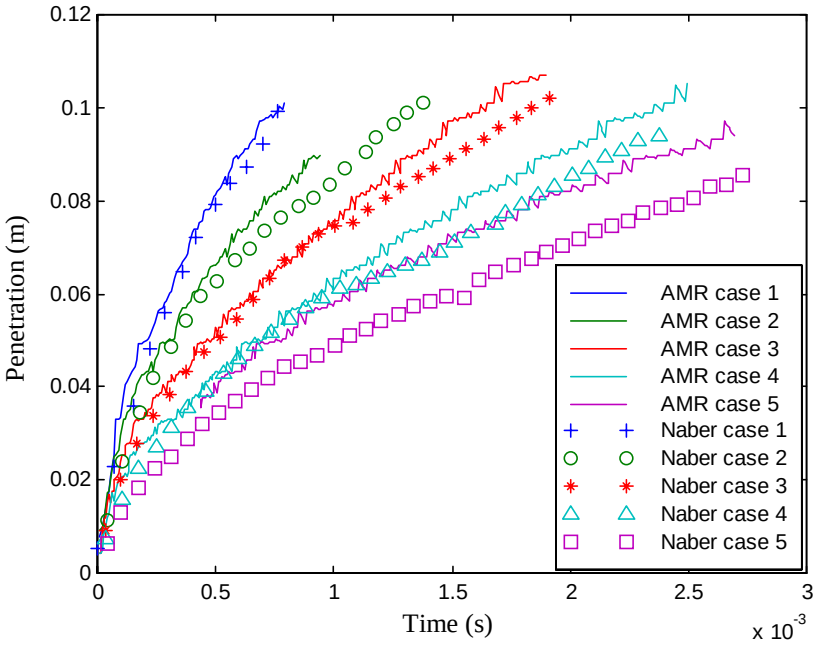

Fig. 8.6 Spray fuel vapor penetration comparison with Naber and Siebers experimental measurements. Fluid density increases from case 1 to case 5.

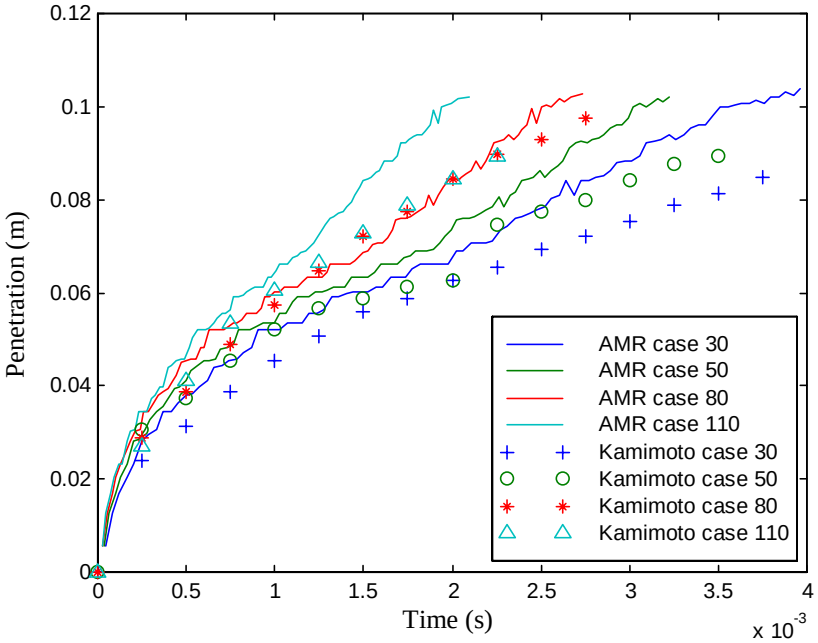

Fig. 8.7 Spray fuel vapor penetration comparison with Kamimoto et al. experimental measurements. Initial spray velocity increases from case 30 to case 110.

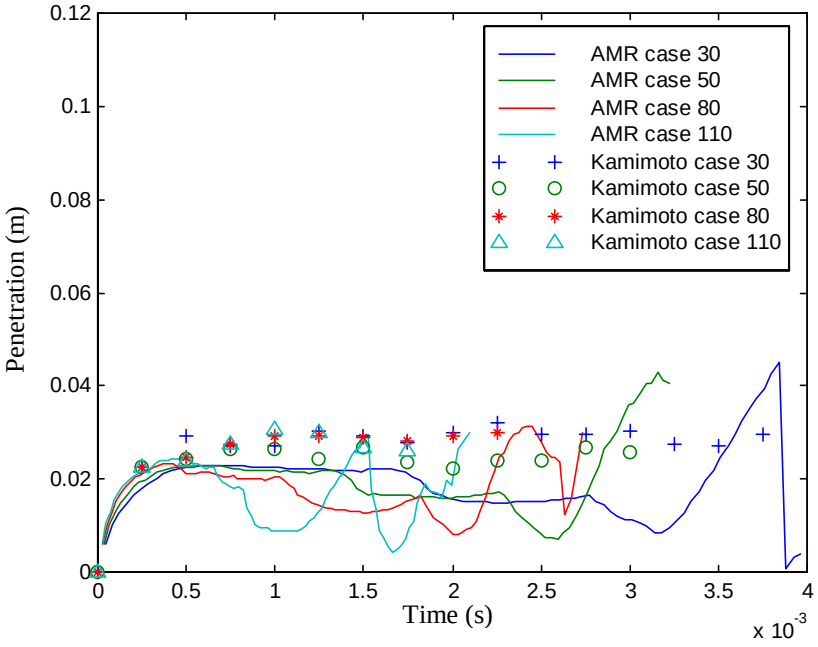

Fig. 8.8 Spray liquid fuel penetration comparison with Kamimoto et al. experimental measurements. Initial spray velocity increases from case 30 to case 110.

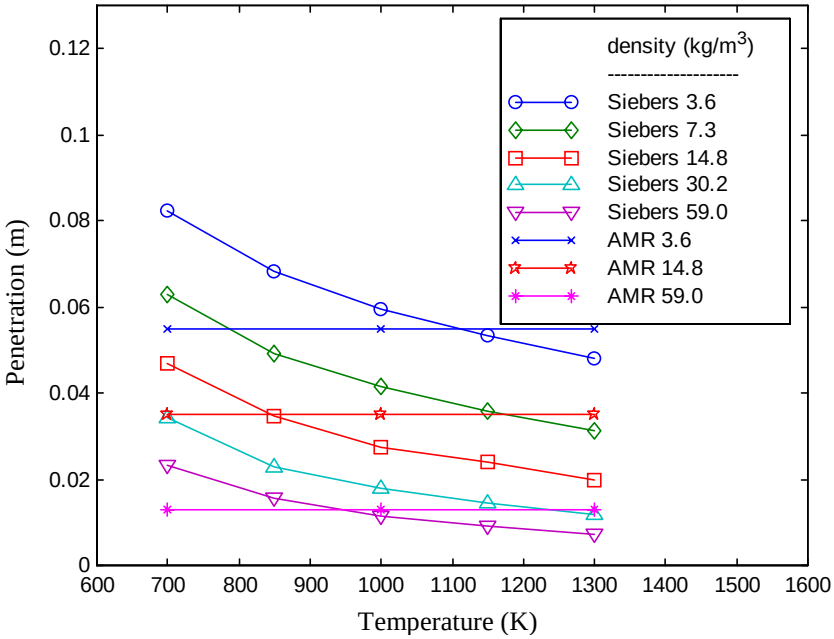

Fig. 8.9 Spray penetration comparison with Siebers experimental measurements, with varying fluid density and temperature.

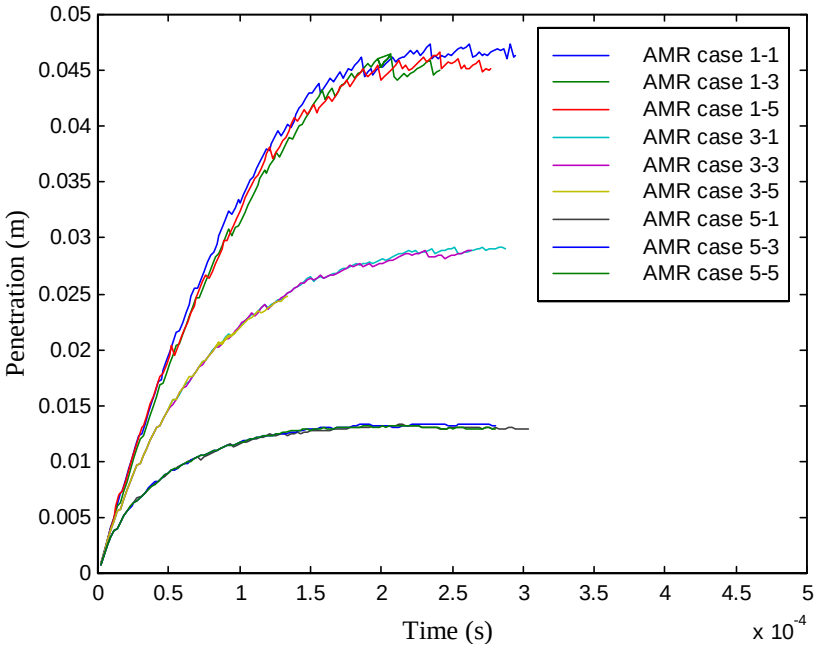

Fig. 8.10 Spray penetrations calculated from the Siebers simulations. The fluid temperature and fluid density were varied.

Chapter 9

Grid and Refinement Effects

The simultaneous use of grids with multiple length scales is inherent in simulations using adaptive grid methods. When using AMR, multiple timescales are also present since the timestep is refined with the grids. Thus, the effects of the timestep and the grid cell size on the models being used should be well understood. This chapter is devoted to a discussion and demonstration of the effects of refinement and grid choices on the behavior of the spray model. As in previous chapters, a summary of the input parameters used for the test cases of each section are provided for reference and may be found in Tables 9.1 - 9.5.

9.1 Collision Grid Effects

The collision grid in KIVA is the same as the grid used for the fluid computations. For this reason, it is very difficult to extricate the affect grid cell size has on the collision model from the effect it has on the spray as a whole. The liberation of the collision grid from the fluid grid for the NTC and Proximity collision models permits the examination of the effect of collision grid cell size on spray behavior as an independent factor. This becomes particularly desirable with the addition of adaptation, since the grid cell size may vary widely from one region of the domain to another.

Case B of the non-evaporative Alloca et al. simulations was chosen to demonstrate the effects of collision grid cell size on the overall spray development. The sample case, performed on a single coarse grid, was repeated several times using the NTC collision model allowing only the collision grid cell sizes, r_{coll}, to change.

A couple factors were found to vary widely depending on the collision grid chosen. The time it took the run to complete increased at both ends of the spectrum (Figure 9.1) with a minimum for a collision cell size around $1x10^{-4}m$. The increase in runtime as cell size increases is most likely due to the increase in the number of particles in the spray to be collided. Likewise, the increase in runtime as cell size decreases probably results from the increase in the number of collision cells. Since each of the runs used identical fluid grids and each took the same number of timesteps, the variance in run time is due directly to the spray and the computational cost of the collision model for the given collision grid.

The number of particles and Sauter Mean Radius (SMR) of the whole spray at the end of the run were also affected, as may be seen in Figure 9.2. These factors are influenced due to the relationship between the logic used by the collision model and collision grid size. As the collision grid cells get smaller, the number of particles per cell that may collide decreases. However, the volume in which the associated droplets are contained is smaller, increasing the probability that two particles in the same cell will collide. The cell size also affects the likelihood of finding two colliding particles with sufficiently different trajectories, influencing whether the resultant collision is coalescent or grazing. The result of these effects is that the SMR for the spray increases and the number of particles in the spray generally decreases as the collision cell size decreases.

Figure 9.3 provides a visual demonstration of the effects of different collision grid sizes. Sprays resulting from using three different collision grid sizes are pictured, with the particles colored to indicate the droplet radius. The increase in spray penetration as the grid size decreases is believed to be due to the increased SMR and mass per particle, resulting in a higher momentum that increases the spray travel.

9.2 Refinement Effects on Spray

The addition of grid adaptation to the simulation of liquid sprays introduces the possibility that different regions of the spray may lay on grids of differing refinement. Every effort was made to reduce any negative impact that may result from spray particles existing on different grids and crossing grid boundaries. However, though it is hoped that a simulation would improve with increased resolution, the very presence of differing grid refinements will have an impact on the simulation. To demonstrate how the development of the spray may be affected by adaptation, variations of two of the full spray simulation cases (described in Chapter 8) were performed where the grid refinement was altered.

9.2.1 Whole Spray Refinement

All of the runs performed for Chapter 8 used adaptive grids such that the entire spray lay within the finest level of grids and the grid cell size on the finest level were of the same order as is commonly used for similar KIVA spray simulations. To explore the effect of choosing this generic grid cell size for the simulations, case 3 of the Naber and Siebers runs was repeated using grids that were further refined by a factor of 2. To remove the effect of a different collision grid on the spray, the collision grid cell size on the finest level was kept the same as the original run, $dx_{coll} = 6.25x10^{-5} m$.

Figure 9.4 shows a comparison of the resulting fuel vapor penetrations. Although Naber and Siebers did not report experimental measurements of the liquid fuel penetrations in their paper, Figure 9.4 also includes a comparison of these penetrations as calculated by the AMR spray code. The vapor penetration profile calculated with the finer grid more closely matches the experimental penetration profile than that obtained with the coarser grid.

However, the simulation with the finer grid still overestimates the fuel vapor penetration. It is interesting to note that the liquid fuel penetrations calculated with the coarser and finer grids initially agree only to diverge later. It is not clear which effects, such as breakup, collision, evaporation, or some combination thereof, are responsible for the later difference.

9.2.2 Partial Spray Refinement

Perhaps the most vivid demonstration of the possible ramifications of using adaptation is a picture of a spray that developed over grids of differing refinement. Case B of the Alloca et al. simulations was performed with grids of fixed refinement such that the grids to the right of the spray centerline were refined by a factor of two. For this case, the turbulence model was omitted for both the fluid and the spray in order to avoid instabilities that tend to appear around the coarse-fine grid boundaries crossing regions of high gradients in the turbulent diffusion field. To remove any effects caused by differing collision grids, the collision grid cell size for the finest level was restricted to be identical to that for the next coarser level, $dx_{coll} = 6.25x10^{-5} m$. As can be seen in Figure 9.5, the presence of the finer grids has a noticeable effect on the resulting appearance of the spray. The boundary between the two grid refinement regions is visually apparent, with many more particles present in the mid-region of the spray on the side with the additional refinement.

9.3 Rate of Convergence Tests

Since the fluid code is known to have second-order convergence (Almgren et al., 1998) when used alone, it is desirable to know what order of convergence may be obtained with the combined AMR spray code. The rate of convergence of the primary spray

submodels was explored by determining the penetration of one or more non-influential particles through a quiescent fluid field over a given amount of time for a range of grid and timestep refinements. The effect of the spray on the fluid code's rate of convergence was then explored using a single case involving a continuous jet of particles in a channel with progressively refined grids and timesteps.

9.3.1 Spray Submodel Convergence

The rate of convergence was determined for four cases: the aerodynamic drag model alone, the drag model plus evaporation, the drag model plus breakup, and the drag model plus breakup and collision. In each case, the particles were non-influential, the ambient fluid was quiescent, the aerodynamic drag model utilized a constant coefficient of drag ($C_D = 1.2$), and the particles were allowed to travel for a given time period ($time_{tot} = 1x10^{-4} s$). A single particle was introduced to the domain for all cases except for the case with collision, where four particles traveling the same trajectory were introduced at the same point and time with slightly varying initial velocities.

The penetration of the particles, defined as the distance of the particle furthest from the starting point, was noted at the end of the specified time for each case and used as the measure for determining convergence rate. The use of non-influential particles and a uniformly quiescent fluid field rendered the relevance of grid refinement as essentially meaningless, so the grids were kept fixed and relatively coarse. The difference between refinement cases was instead differentiated by the constant timesteps, progressively refined by constant factors of two.

The convergence rate was estimated by comparing three cases with successive timestep refinements using the expression

$$\Theta = \frac{\log\left(\frac{\left|P_{dt} - P_{dt/2}\right|}{\left|P_{dt/2} - P_{dt/4}\right|}\right)}{\log(2)} \tag{9.1}$$

where P is the measured particle penetration at the end of the time period, the subscript indicates the timestep used, and Θ is the calculated order of convergence (Ferziger and Peric, 1996). The calculated particle penetrations and the resultant estimated convergence rates from the spray submodel tests are shown in Table 9.6.

The rate of convergence of the spray's aerodynamic drag model alone exhibits second-order convergence, as expected since the model uses a second-order predictor-corrector scheme to update the particle's location and velocity. The evaporation model also uses a second-order predictor-corrector scheme to update the particle properties, which is reflected in the second-order convergence of the penetration when evaporation is present.

To test the order of convergence of the breakup model, only the Reitz Wave model was used with the usual aerodynamic drag model. The addition of the breakup model appears to reduce the apparent order of convergence to approximately 1.5 or less. The use of alternate definitions of the penetration, e.g. the radius of an arc containing 90% of the spray mass, did not appear to improve this estimation of the convergence rate.

The convergence rate of the collision model was tested with the Reitz Wave breakup model in order to obtain a broad enough array of particles with differing velocities such that the collision model became a factor in the spray behavior. Further, the same collision grid size was used for each case, regardless of refinement to remove collision grid effects. When

considering the convergence rate results, it should be recalled that when grid and timestep refinement occurs due to adaptation, the collision grid size is usually likewise refined. As with the breakup model, the convergence rate appears to be approximately 1.5 or less. Although it is unclear whether the collision model would have exhibited a higher order of convergence if it could have been tested alone, a collision model is rarely used without a breakup model and this test shows that the collision model does not further degrade the obtainable convergence rate.

9.3.2 Fluid Convergence with Spray

In order to estimate the effect of the spray on the fluid code's convergence rate, an attempt was made to create a test case that would result in a fluid motion that was nearly steady state and had relatively smooth features. A steady jet of particles was introduced to a channel of fluid and the system allowed to develop sufficiently to a near steady state.

The domain used slip wall boundary conditions on the channel sides and inlet/outlet boundary conditions on the other two boundaries. To encourage smooth features in the fluid motion, the spray used a large number of particles, at a rate of $1x10^8 \frac{particles}{s}$, with initial trajectories parallel to the sides of the channel. The jet of particles itself used a normal distribution, with a standard deviation of 5% of the channel width, to determine the initial location of each particle at the inlet such that the majority of the particles were concentrated along the channel's centerline. Full particle-fluid interaction was used and all spray submodels were turned off except for the aerodynamic drag model, which used a constant coefficient of drag ($C_D = 1.2$). Each particle in the jet was given the same initial velocity. See Figure 9.6 for an illustration.

The convergence rate was estimated by examining the velocity field of the fluid after a fixed amount of time had been allowed to pass. The y-component of the velocity was used, as it was generally the larger of the two components. Comparing the solution on two grids of differing refinement was accomplished by averaging the solution on the fine grid cells corresponding to each coarse grid cell and taking the difference. The rate of convergence was then calculated in both a pointwise and a spanwise fashion. The pointwise comparison found the difference between the coarse grid cell velocity and the averaged velocity from the corresponding fine grid cells, and then calculated the norm over the entire domain. The spanwise comparison first averaged the coarse grid cell velocities by row and likewise averaged the corresponding rows of fine grid cell velocities. The difference was then taken between the row-averaged coarse and fine grid cell velocities and the norm calculated over the entire domain. Both the pointwise and the spanwise comparisons calculated the rate of convergence according to the expression

$$\Theta = \frac{\log\left(\dfrac{L_p\left(\left|v_{dx,dt} - v_{dx/2,dt/2}\right|\right)}{L_p\left(\left|v_{dx/2,dt/2} - v_{dx/4,dt/4}\right|\right)}\right)}{\log(2)} \tag{9.2}$$

where v is the (averaged) calculated fluid velocity at the end of the time period, the subscripts indicate the refinement used for the timestep and grid size, L_p indicates the use a norm over the entire grid and Θ is the order of convergence. The norm of the differences between the calculated particle velocities and the resultant estimated convergence rates are in Tables 9.7 and 9.8.

The pointwise comparison demonstrates the difficulty of attempting to determine a convergence rate when the problem involves small-scale phenomena, such as the effect of the spray particles on the fluid. When considering the convergence rates estimated using the infinity norm, it appears that the fluid exhibits a convergence order of approximately 0.5 while the grids are relatively coarse, but degrades with refinement. The results obtained using the L_2 norm do not suggest a reliable convergence rate, but shows the convergence degrading more rapidly and the solutions diverging. Figures 9.7 and 9.8, pictures of the fluid's y-component of the velocity for a coarse grid and a fine grid, respectively, suggest that the apparent convergence rate degrades as a result of the finer mesh grids resolving fluid flow details of increasingly smaller length scales. In essence, the inherent discontinuity of a spray's influence, when modeled with discrete particles, interferes with the attempts to numerically determine the fluid's convergence rate.

The rate of convergence determined from using the span-averaged velocities provides a much more consistent view of the effect of the spray on the fluid calculations as the grids are refined. As can be seen in Table 9.8, the rate of convergence calculated with both the infinity norm and the L_2 norm is shown to be approximately second-order. Thus, when using spray submodels that demonstrate at least second-order convergence, such as the aerodynamic drag model, the AMR fluid code's second-order rate of convergence is preserved.

Section	Case	Species	Pressure (*MPa*)	Temperature (*K*)
9.1		N_2	1.7	300
9.2.1		N_2	4.126	1000
9.2.2		N_2	0.1	300
9.3.1	Drag Only	N_2	1.7	300
	Evaporation			1000
	Breakup			300
	Breakup & Collision			300
9.3.2		N_2	1.7	300

Table 9.1 The primary properties of the ambient fluid for the tests associated with each section.

Section	Species	Liquid Density (kg/m^3)	Drop/Nozzle Radius (m)	Initial Temp. (K)	Initial Speed (m/s)	Total Mass (kg)
9.1	$C_{14}H_{30}$	760	7.96×10^{-5}	298	377.72	1.6×10^{-5}
9.2.1	$C_{14}H_{30}$	703	1.23×10^{-4}	452	419	5.6×10^{-5}
9.2.2	$C_{14}H_{30}$	760	7.96×10^{-5}	298	377.72	1.6×10^{-5}
9.3.1	$C_{14}H_{30}$	760	7.96×10^{-5}	300	400.0	1.6×10^{-8}
9.3.2	$C_{14}H_{30}$	760	5.0×10^{-5}	298	400	7.96×10^{-6}

Table 9.2 The primary properties of the spray for the tests associated with each section.

Section	Case	Initial Number of Particles	Reitz Wave Breakup		Rayleigh-Taylor Breakup		Cone Angle (*deg*)
			bw_time	bw_rad	crt_dist	crt_rad	
9.1		1000	70	0.6	0.2	3.0	9
9.2.1		3000	70	0.6	0.02	0.1	50
9.2.2		1000	70	0.6	0.2	3.0	9
9.3.1	Drag Only	1					0
	Evaporation	1					
	Breakup	1	70	0.6			
	Breakup & Collision	4	70	0.6			
9.3.2		5000					

Table 9.3 The computational parameters of the spray for the tests associated with each section.

Section	Domain Size (*m* x *m* x *m*)	Boundary Conditions	
		x-dir (lower/upper)	y-dir (lower/upper)
9.1	0.064x0.128	no-slip/no-slip	outlet/no-slip
9.2.1	0.064x0.128	no-slip/no-slip	outlet/no-slip
9.2.2	0.064x0.128	no-slip/no-slip	outlet/no-slip
9.3.1	0.064x0.128	no-slip/no-slip	outlet/no-slip
9.3.2	0.006x0.006	slip/slip	inlet/outlet

Table 9.4 The properties of the physical domain and boundary conditions for the tests associated with each section.

Section	Level 0 Grid	Number of Fine Grid Cells (Refinement Ratio)				Domain "depth" (*m*)
		Level 1	Level 2	Level 3	Level 4	
9.1	8x16					0.001
9.2.1	16x32	variable (2)	variable (2)	variable (2)		0.001
9.2.2	8x16	8x8 (2)	16x15 (2)	32x29 (2)	32x57 (2)	0.001
9.3.1	4x8					0.001
9.3.2	variable					0.001

Table 9.5 The properties of the computational grids for the tests associated with each section.

Case		dt_o	$^1/_2\ dt_o$	$^1/_4\ dt_o$	$^1/_8\ dt_o$	$^1/_{16}\ dt_o$	$^1/_{32}\ dt_o$
Aerodynamic Drag	Penetration (10^{-2} *m*)	1.399	1.353	1.341	1.3382	1.3375	1.3374
	Convergence Rate	1.87			---	---	---
		---	2.055			---	---
		---	---	2.046			---
		---	---	---	2.028		
Evaporation	Penetration (10^{-2} *m*)	2.347	2.338	2.3356	2.3351	2.33495	2.33492
	Convergence Rate	2.04			---	---	---
		---	2.024			---	---
		---		2.015			---
		---	---	---	1.99		
Breakup	Penetration (10^{-2} *m*)	2.292	2.274	2.267	2.2637	2.2623	2.2617
	Convergence Rate	1.593			---	---	---
		---	1.560			---	---
		---	---	1.372			---
		---	---	---	1.217		
Breakup & Collision	Penetration (10^{-2} *m*)	1.353	1.298	1.280	1.274	1.2716	1.2706
	Convergence Rate	1.593			---	---	---
		---	1.559			---	---
		---	---	1.379			---
		---	---	---	1.202		

Table 9.6 Convergence rate results for the spray submodels.

Case		dt_o, dx_o	$^1/_2\,dt_o$, $^1/_2\,dx_o$	$^1/_4\,dt_o$, $^1/_4\,dx_o$	$^1/_8\,dt_o$, $^1/_8\,dx_o$	$^1/_{16}\,dt_o$, $^1/_{16}\,dx_o$
L_∞	Velocity Difference (m/s)	163.76		---	---	---
		---	109.87		---	---
		---	---	72.38		---
		---	---	---	72.28	
	Convergence Rate	0.57			---	---
		---	0.60			---
		---	---	0.002		
L_2	Velocity Difference (m/s)	649.08		---	---	---
		---	313.85		---	---
		---	---	349.24		---
		---	---	---	763.05	
	Convergence Rate	1.048			---	---
		---	-0.15			---
		---	---	-1.12		

Table 9.7 Convergence rate results for the fluid with spray, calculated pointwise.

Case		dt_o, dx_o	$^1/_2\,dt_o$, $^1/_2\,dx_o$	$^1/_4\,dt_o$, $^1/_4\,dx_o$	$^1/_8\,dt_o$, $^1/_8\,dx_o$	$^1/_{16}\,dt_o$, $^1/_{16}\,dx_o$
L_∞	Velocity Difference (m/s)	0.251		---	---	---
		---	0.057		---	---
		---	---	0.014		---
		---	---	---	0.0034	
	Convergence Rate	2.13			---	---
		---	2.02			---
		---	---	2.04		
L_2	Velocity Difference (m/s)	0.251		---	---	---
		---	0.057		---	---
		---	---	0.014		---
		---	---	---	0.0034	
	Convergence Rate	2.13			---	---
		---	2.02			---
		---	---	2.04		

Table 9.8 Convergence rate results for the fluid with spray, calculated using span-averaged velocities.

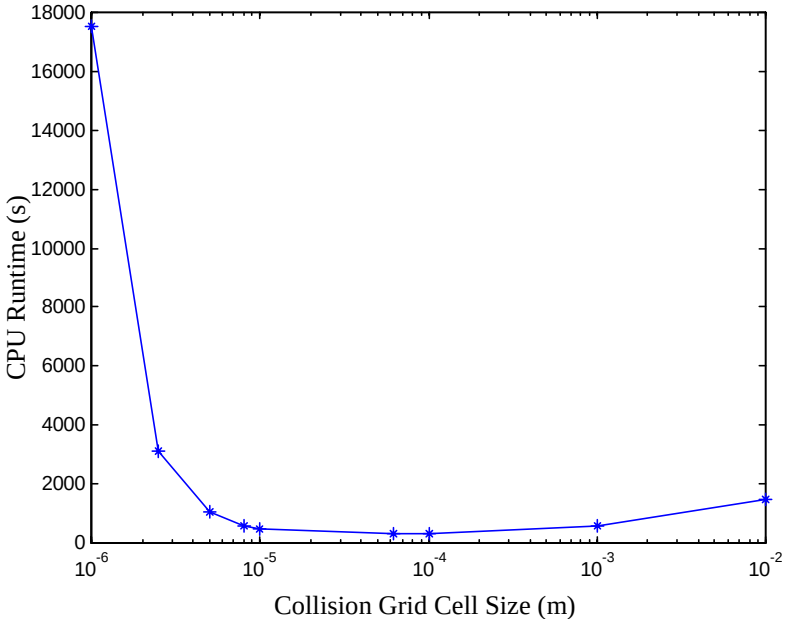

Fig. 9.1 The cpu runtimes recorded for a non-evaporative spray case using different collision grid cell sizes, r_{coll}, for the NTC collision model.

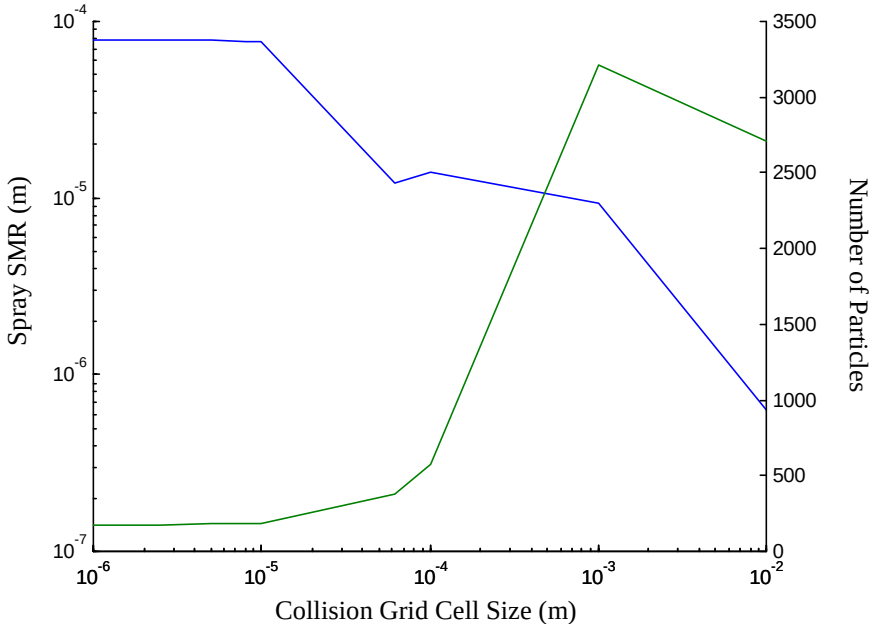

Fig. 9.2 The resultant number of particles (right axis, blue) and the total spray SMR (left axis, green) for a non-evaporative spray case using different collision grid cell sizes, r_{coll}, for the NTC collision model.

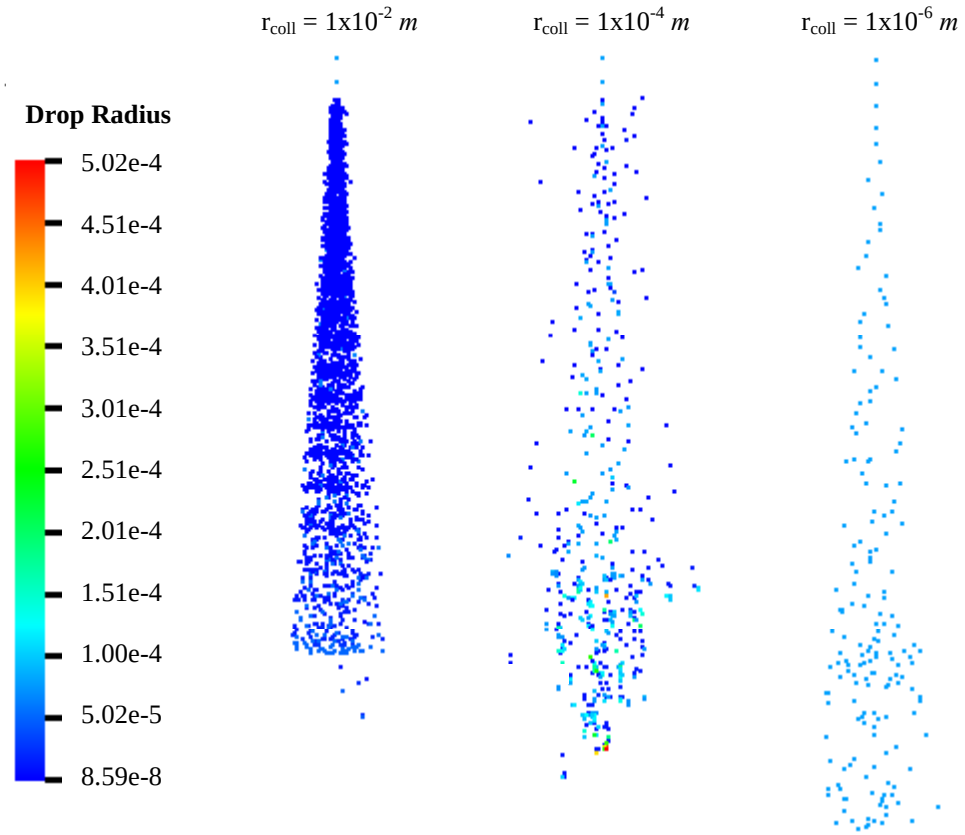

Fig. 9.3 Pictures of a spray calculated using three different collision grid sizes. The color key indicates the magnitude of the droplet radius (in *m*) for the particles in each case.

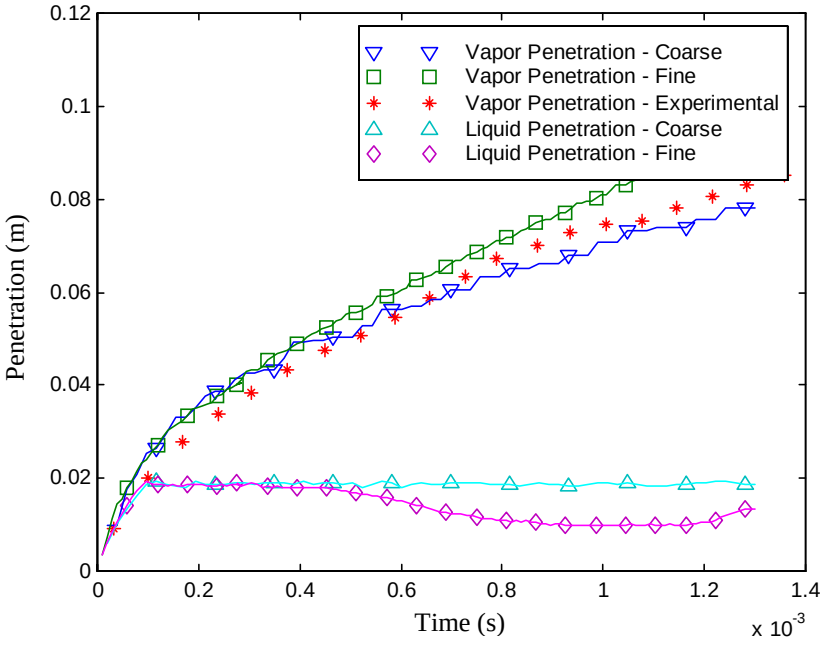

Fig. 9.4 Spray penetration comparison of Naber and Siebers case 3 between runs with the finest grid cell size differing by a factor of two.

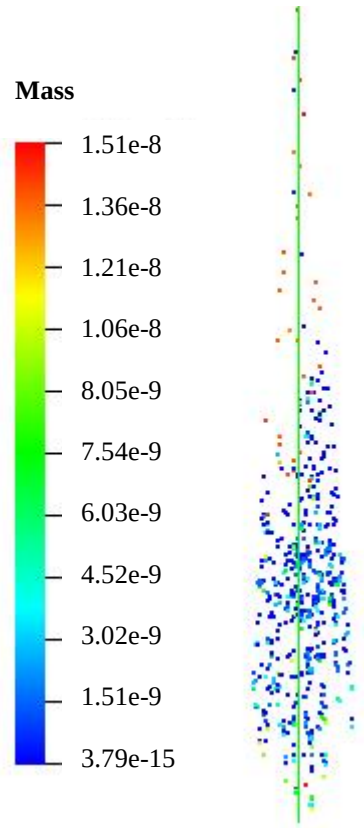

Fig. 9.5 The spray of Alloca et al. case B where the right half of the domain has been refined by a factor of two. The color key indicates the particle mass in *kg*.

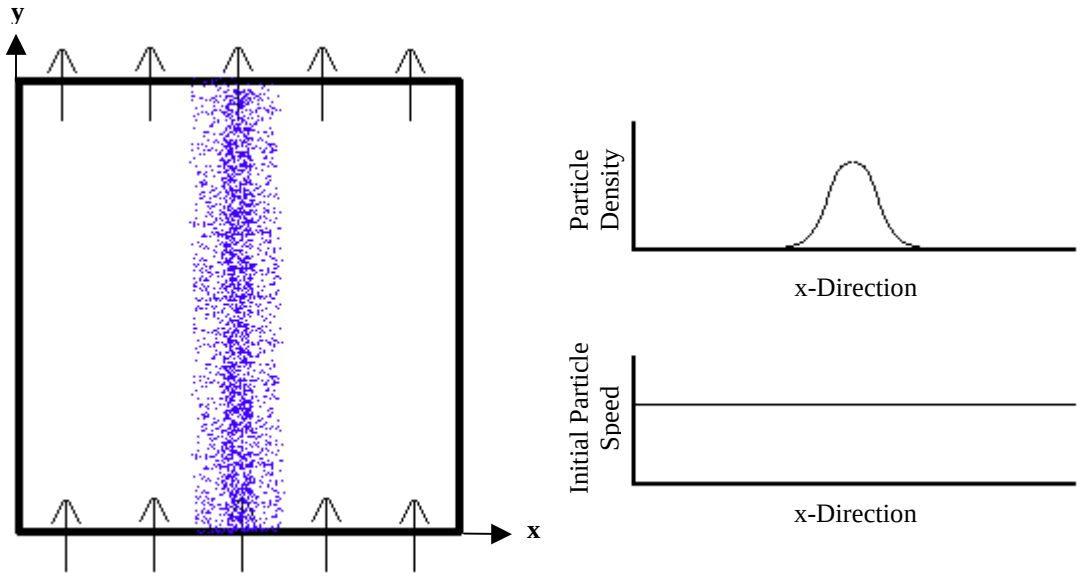

Fig. 9.6 An illustration of the rate of convergence test for the fluid with a spray.

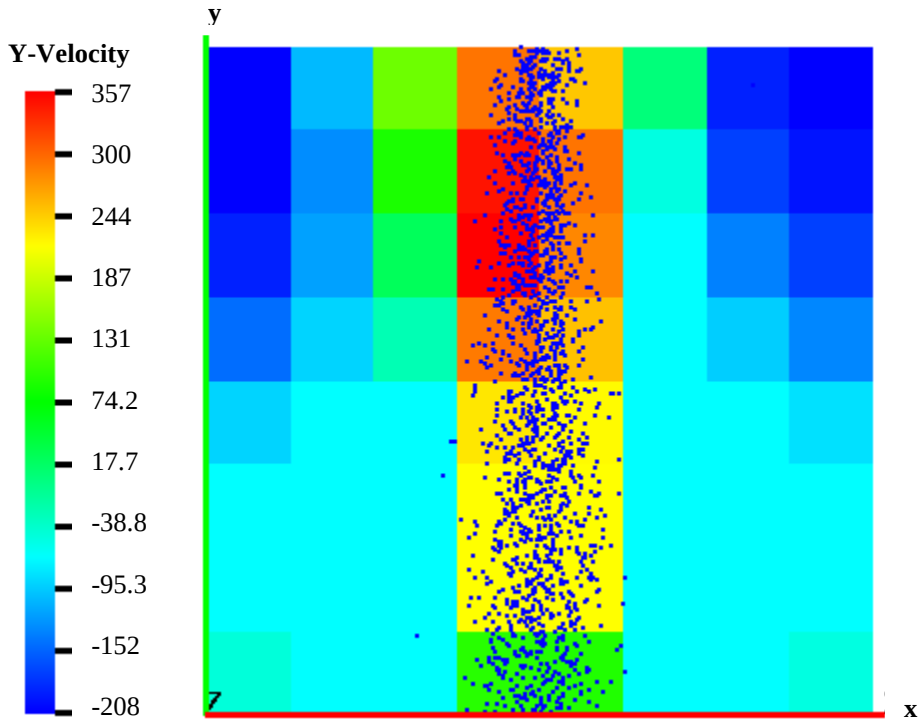

Fig. 9.7 Convergence rate test case for the fluid with a spray, calculated using a very coarse grid of 8x8 cells. The color key indicates the fluid's y-component of velocity (in *m/s*).

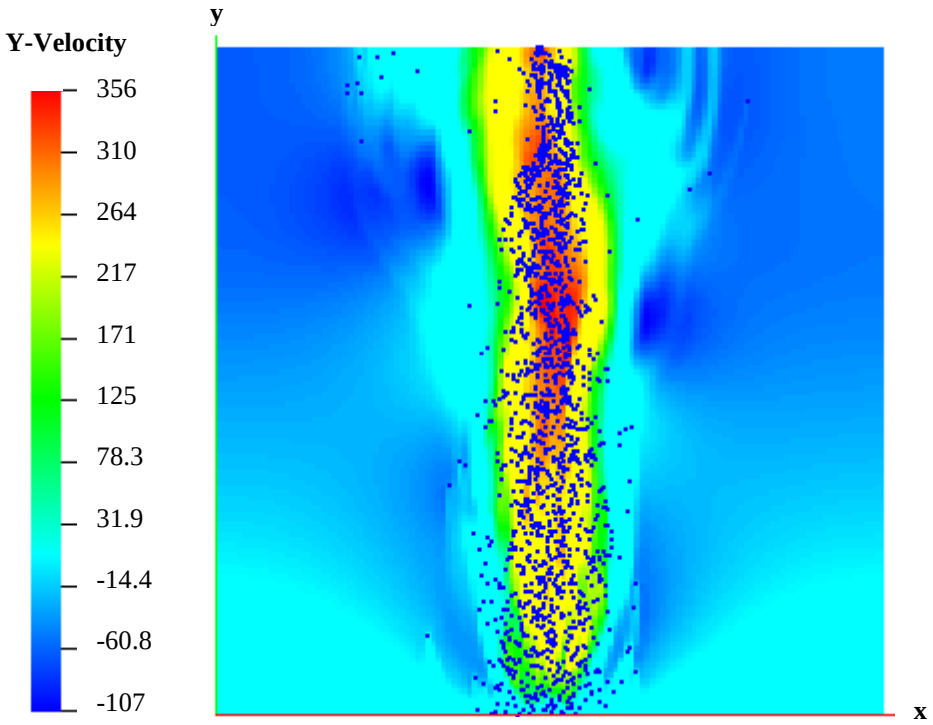

Fig. 9.8 Convergence rate test case for the fluid with a spray, calculated using a relatively fine grid of 128x128 cells. The color key indicates the fluid's y-component of velocity (in *m/s*).

Chapter 10

Conclusion and Recommendations

10.1 Conclusion

A discrete droplet spray model, based on the spray model found in KIVA, has been implemented and incorporated into an existing fluid code that employs adaptive mesh refinement. The spray code consists of a collection of physical submodels and a variety of administrative routines for both the management of the spray within an adaptive grid framework and the spray data structure itself. The routines necessary for providing an interface between the spray code and AMR fluid code, as well as alternate versions of a few fluid code routines, were also created.

The resultant AMR spray code was tested to determine the validity of the implementation and appropriateness of the simulation results. Each physical submodel was tested individually on single grids to verify that the implementation was correct. Tests were also performed to ensure that the particle-fluid interaction did not create any grid-related artifacts near coarse-fine boundaries. The full AMR spray code was then tested by simulating spray bombs for which there existed published experimental data and comparing the spray penetrations. The AMR spray code was successful in capturing basic trends in liquid fuel and fuel vapor penetrations as various parameters were altered. The only exceptions were the Siebers vaporization cases of Section 8.3.3 where changes in fluid temperature did not appear to affect the calculated liquid penetration.

Additional tests were performed to determine what effects grids of differing refinement and their associated timesteps might have on the performance of the spray model.

The development of the spray was shown to be dependent on the cell size of the finest grid containing part of the spray. This dependency is potentially a serious problem if grid refinement is allowed to vary over the region containing the spray. Further, the choice of collision grid for the collision submodels was found to significantly affect the character of the spray, resulting in larger droplets and fewer particles as the collision grid cell size decreased.

During the implementation of the spray model, every effort was made to preserve the second-order convergence of the AMR fluid code. However, some of the spray submodels, such as the breakup model, were found to exhibit a convergence rate of only 1.5. Investigations into the combined AMR spray code demonstrated the difficulty of creating suitable tests to reveal the attainable convergence rate when highly localized source terms, such as those associated with the discrete particles of the spray, are present. However, it was determined that when only spray submodels exhibiting second-order convergence were used, such as the aerodynamic drag model, the second-order convergence rate of the fluid is preserved.

Though the combined AMR spray code was disappointingly slow, the use of adaptive grids helped reduce the runtimes of full spray simulations over the runtimes attainable using single fine grids. Some factors affecting the length of runtimes include: inclusion of debugging information in the code executable, choice of output file format, additional applications of the spray subroutines for coarser level timesteps, and possibly the choice of iterative solvers for the fluid code and the use of a serial spray code with a fluid code optimized for parallel computations.

10.2 Recommendations for Future Work

The work presented in this document merely represents an initial foray into the simulation of sprays in conjunction with fluids modeled with adaptive grid techniques. There are a number of areas where future work may be pursued:

I The AMR fluid code is currently fully capable of parallel computations. It would be highly advantageous to modify the spray code so that it would also perform its computations in parallel. While the necessary modifications would most likely be straightforward, given the AMR code as an example, special care should be taken when devising a way to split the spray into manageable chunks and ensuring that all of the necessary fluid information is available without passing more information than is needed to each processor.

II Further work needs to be done to address the issue of handling mass sources in a closed domain correctly. As mentioned in Chapter 4 and Appendix I, this will primarily involve deciding how to handle a variable ambient pressure in the context of adaptive grids and implementing the necessary changes.

III There are several additions and alterations to the AMR code that would be desirable for improving the simulation of sprays within a combustion cylinder. The ability to model complex geometries would allow the shape of the combustion cylinder to be represented more accurately. The capacity to modify the level 0 grids, particularly the shape of the domain, during the computation would make it

possible to model a moving piston. Provisions for different boundary conditions to exist along the same boundary would allow the effect of intake and exhaust valves to be accounted for. Some of these are currently being addressed by LBNL-CCSE and their collaborators.

IV The ability of the AMR code to handle combustion is currently being pursued by LBNL-CCSE. Once this is complete, the AMR spray code should be examined for suitability for full combustion simulations.

V To address the problem of how far the grid cells can be refined before the particle volume can no longer be assumed to be negligible, it is recommended that the fluid volume displaced by the particles be accounted for in the fluid code.

VI There are a number of physical submodels available for the KIVA code not included in the present work that can be added. Some examples are models for wall impingement, liquid wall films, multi-component fuels, and soot.

VII It is recommended that a set of higher quality random number generators be implemented.

VIII In recent years, there has been a great deal of work done in the area of turbulence modeling. The LBNL-CCSE has implemented an LES model into the AMR code to more accurately model the effects of turbulence on the fluid. It would be

beneficial to develop a spray submodel capable of utilizing the LES model information.

IX It would be beneficial to improve the way transport properties for the fuel vapor are handled. This could be accomplished with a transport property database that uses assumptions that are valid fuel vapor, or by implementing a supplemental database for the fuel vapor.

X The KIVA spray submodels have a number of assumptions embedded in their formulation regarding the typical timesteps and grids that will be used. These assumptions cause some difficulties when attempting to integrate it with a code that uses multiple time and length scales. If further development of the AMR spray code is pursued, additional work should be done to make the spray physical submodels as grid and timestep independent as possible.

XI Once the spray model is reasonably grid and timestep independent, so that the spray may exist on different grid levels without complication, additional refinement criteria based on the spray should be implemented. Further, the effects of choosing different refinement criteria should be explored.

XII The spray model in KIVA may not be the most appropriate for simulating sprays in an adaptive grid framework. In this case, investigations into other spray models should be considered.

XIII In this work, spray penetration was the only measurable spray characteristic that was compared with experimental data. The performance of the spray model with respect to other spray characteristics available in the experimental spray literature should also be examined.

XIV The handling of spray source terms with respect to the depth of the grid cells in two-dimensional simulations should be handled more rigorously. The current method of using a scaling factor and leaving the domain "depth" to be defined by the user adds an additional arbitrary factor that affects the spray behavior.

XV Utilizing symmetry would help reduce computational cost, particularly for three-dimensional computations. Thus, it would be beneficial to improve the performance of the spray model when a symmetry boundary is present along the axis of the spray.

XVI The current implementation of the spray physical submodels has some code redundancies that could be removed. Most of the routines currently have two versions: one for two-dimensional computations and another for three-dimensional computations. To reduce the workload when debugging, it would help to reformulate most of the routines in C++, only using Fortran to extract the necessary local fluid data for each particle. However, it should first be determined

if the use of Fortran for the spray physical submodel routines results in sufficient computational cost savings to justify the additional code maintenance costs.

References

1. Almgren, A.S., Bell, J.B., Colella, P., Howell, L.H., and Welcome, M.L., "A Conservative Adaptive Projection Method for the Variable Density Incompressible Navier-Stokes Equations", *J. of Computational Physics*, **142**, (1998), pp. 1-46.
2. Amsden, A.A., "KIVA-3: A KIVA Program with Block-Structured Mesh for Complex Geometries", *Los Alamos Scientific Laboratory Report* LA-12503-MS (1993).
3. Amsden, A.A., "KIVA-3V: A Block-Structured KIVA Program for Engines with Vertical or Canted Valves", *Los Alamos Scientific Laboratory Report* LA-13313-MS (1997).
4. Amsden, A.A., O'Rourke, P.J., and Butler, T.D., "KIVA-II: A Computer Program for Chemically Reactive Flows with Sprays", *Los Alamos Scientific Laboratory Report* LA-11560-MS (1989).
5. Amsden, D.C. and Amsden, A.A., "The KIVA Story: A Paradigm of Technology Transfer", *IEEE Transactions on Professional Communication*, **36**, (1993), pp. 190-195.
6. Arney, D.C. and Flaherty, J.E., "An Adaptive Local Mesh Refinement Method for Time-Dependent Partial Differential Equations", *Applied Numerical Mathematics*, **5**, (1989), pp. 257-274.
7. Beale, J., "Modeling Fuel Injection Using the Kelvin-Helmholtz / Rayleigh-Taylor Hybrid Atomization Model in KIVA-3V", Masters Thesis, University of Wisconsin-Madison, (1999).
8. Bell, J.B., private correspondence, (1999).

9. Bell, J.B., Berger, M.J., Saltzman, J., and Welcome, M.L., "Three-Dimensional Adaptive Mesh Refinement for Hyperbolic Conservation Laws", *SIAM J. of Scientific Computing*, **15**, (1994), pp. 127-138.

10. Bell, J.B. and Marcus, D.L., "A Second-Order Projection Method for Variable-Density Flows", *J. of Computational Physics*, **101**, (1992), pp. 334-348.

11. Berger, M.J. and Colella, P., "Local Adaptive Mesh Refinement for Shock Hydrodynamics", *J. of Computational Physics*, **82**, (1989), pp. 64-84.

12. Berger, M.J. and Oliger, J., "Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations", *J. Computational Physics*, **53**, (1984), pp. 484-512.

13. Biswas, R. and Strawn, R.C., "Tetrahedral and Hexahedral Mesh Adaptation for CFD Problems", *Applied Numerical Mathematics*, **26**, (1998), pp. 135-151.

14. Chen, C. and Veshagh, A., "A Simple Unified Fuel Spray Model", *SAE Paper* 930923, (1993).

15. Davis, R.L. and Dannenhoffer III, J.F., "Three-Dimensional Adaptive Grid-Embedding Euler Technique", *AIAA Journal*, **32**, no. 6, (1994), pp. 1167-1173.

16. Dukowicz, J.K., "A Particle-Fluid Numerical Model for Liquid Sprays", *J. of Computational Physics*, **35**, (1980), pp. 229-253.

17. Ferm, L. and Lötstedt, P., "Blockwise Adaptive Grids with Multigrid Acceleration for Compressible Flow", *AIAA Journal*, **37**, no. 1, (1998), pp. 121-123.

18. Ferziger, J.H. and Peric, M., *Computational Methods for Fluid Dynamics*, Springer-Verlag, Berlin Heidelberg, (1996).

19. Filliben, J.J., *User's Guide to Datapac*, NBS, Gaithersburg, MD, (1977).

20. Greaves, D.M. and Borthwick, A.G.L., "On the Use of Adaptive Hierarchical Meshes for Numerical Simulation of Separated Flows", *International J. for Numerical Methods in Fluids*, **26**, (1998), pp. 303-322.

21. Gropp, W.D. and Keyes, D.E., "Domain Decomposition with Local Mesh Refinement", *SIAM J. on Scientific and Statistical Computing*, **13**, no. 4, (1992), pp. 967-993.

22. Hallmann, M., Scheurlen, M., and Wittig, S., "Computation of Turbulent Evaporating Sprays: Eulerian Versus Lagrangian Approach", *J. of Engineering for Gas Turbines and Power*, **117**, (1995), pp. 112-119.

23. Hart, L. and McCormick, S., "Asynchronous Multilevel Adaptive Methods for Solving Partial Differential Equations on Multiprocessors: Basic Ideas", *Parallel Computing*, **12**, (1989), pp. 131-144.

24. Huang, W. and Russell, R.D., "A High Dimensional Moving Mesh Strategy", *Applied Numerical Mathematics*, **26**, (1998), pp. 63-76.

25. Jayaraman, V., Udaykumar, H.S., and Shyy, W.S., "Adaptive Unstructured Grid for Three-Dimensional Interface Representation", *Numerical Heat Transfer, Part B*, **32**, (1997), pp. 247-265.

26. Kee, R.J., Dixon-Lewis, G., Warnatz, J., Coltrin, M.E., and Miller, J.A., "A Fortran Computer Package for the Evaluation of Gas-Phase, Multicomponent Transport Properties", *Sandia National Laboratories Report* SAND86-8246, (1986).

27. Kee, R.J., Miller, J.A., and Jefferson, T.H., "CHEMKIN: A General-Purpose, Problem-Independent, Transportable, Fortran Chemical Kinetics Code Package", *Sandia National Laboratories Report* SAND80-8003, (1980).

28. Kelmanson, M.A. and Maunder, S.B., "Modeling High-Velocity Impact Phenomena Using Unstructured Dynamically-Adaptive Eulerian Meshes", *J. of the Mechanics and Physics of Solids*, **47**, (1999), pp. 731-762.

29. Liu, A.B., Mather, D., and Reitz, R.D., "Modeling the Effects of Drop Drag and Breakup on Fuel Sprays", *SAE paper* 930072, (1993).

30. Matsuno, K., Kamakawa, M., and Satofuka, N., "Overset Adaptive-Grid Method with Applications to Compressible Flows", *Computers and Fluids*, **27**, nos. 5-6, (1998), pp. 599-610.

31. Mitchell, W.F., "A Domain Partition Approach to Distributing Adaptive Grids", *Applied Numerical Mathematics*, **26**, (1998), pp. 265-275.

32. O'Rourke, P.J., "Statistical Properties and Numerical Implementation of a Model for Droplet Dispersion in a Turbulent Gas", *J. of Computational Physics*, **83**, (1989), pp. 345-360.

33. O'Rourke, P.J. and Amsden, A.A., "The TAB Method for Numerical Calculation of Spray Droplet Breakup", *SAE paper* 872089, (1987).

34. Oden, J.T., "Progress in Adaptive Methods in Computational Fluid Dynamics", *Adaptive Methods for Partial Differential Equations*, Eds. Flaherty, J.E., Paslow, P.J., Shephard, M.S., Vasilakis, J.D., Society for Industrial and Applied Mathematics, Philadelphia, (1989), pp. 206-251.

35. Patterson, M.A., "Modeling the Effects of Fuel Injection Characteristics on Diesel Combustion and Emissions", Ph.D. Thesis, University of Wisconsin-Madison, (1997).

36. Patterson, M.A. and Reitz, R.D., "Modeling the Effects of Fuel Spray Characteristics on Diesel Engine Combustion and Emissions", *SAE paper* 980131, (1998).

37. Pember, R.B., Howell, L.H., Bell, J.B., Colella, P., and Crutchfield, W.Y., "An Adaptive Projection Method for Unsteady, Low-Mach Number Combustion", *Combustion Science and Technology*, **140**, (1998), pp. 123-168.

38. Podber, D.P. and Bedford, K.W., "An Analysis of Grid Convergence for the Dynamic Grid Adaptation Technique Applied to the Propagation of Internal Waves", *Estuarine and Coastal Modeling: Proceedings of the 5th International Conference*, American Society of Civil Engineers, Alexandria, VA, (1998), pp. 834-848.

39. Powell, K.G., "A Tree-Based Adaptive Scheme for Solution of the Equations of Gas Dynamics and Magnetohydrodynamics", *Applied Numerical Mathematics*, **14**, (1994), pp. 327-352.

40. Reitz, R.D., "Modeling Atomization Processes in High-Pressure Vaporizing Sprays", *Atomisation and Spray Technology*, **3**, (1987), pp. 309-337.

41. Rendleman, C., Beckner, V., Bell, J.B., Crutchfield, B., Howell, L.H., and Welcome, M., *Boxlib User's Guide and Manual: A Library for Managing Rectangular Domains*, Center for Computational Science and Engineering, Lawrence Berkeley National Laboratory, edition 2.0, for Boxlib version 2.0.

42. Robson, R., *Using the STL: The C++ Standard Template Library, Second Edition*, Springer-Verlag, New York, Inc., (1999).

43. Schmidt, D.P. and Rutland, C.J., "A New Droplet Collision Algorithm", *J. of Computational Physics*, **164**, (2000), pp. 62-80.

44. Sinnamon, J.F., Lancaster, D.R., Steiner, J.C, “An Experimental and Analytical Study of Engine Fuel Spray Trajectories”, *SAE paper* 800135, (1980).

45. Sirignano, W.A., “The Formulation of Spray Combustion Models: Resolution Compared to Droplet Spacing”, *J. of Heat Transfer*, **108**, (1986), pp. 633-639.

46. Watkins, A.P., “Computation of Diesel Sprays by a Non-Iterative Implicit Solution Scheme”, *Atomisation and Spray Technology*, **3**, (1987), pp. 261-290.

47. Williams, A., *Combustion of Liquid Fuel Sprays*, Butterworths, London, Boston, (1990), pp. 75-105.

48. Ziegler, U., “NIRVANA+: An Adaptive Mesh Refinement Code for Gas Dynamics and MHD”, *Computer Physics Communications*, **109**, (1998), pp. 111-134.

Appendix A

Data Structure

The data structure for holding spray information is composed of a collection of C++ classes. These classes not only store the spray variables, but also provide an organization for the spray's computational particles as they are manipulated during the calculation. The classes also provide a natural division of the administrative routines for the data structure.

A.1 Particle

The Particle class contains all of the information and operations pertaining to a single computational particle representing a parcel of droplets. The variables that define a Particle include the following:

- Location coordinates of the Particle.
- Velocity of the Particle.
- Total mass of the droplets in the Particle
- Number of droplets represented by the Particle.
- Radius of the droplets in the Particle.
- Droplet distortion parameter for the droplets in the Particle.
- Rate of change of the droplet distortion for the Particle's droplets.
- Temperature of the droplets represented by the Particle.

Each parameter is represented twice in the Particle class for each level on which the particle rests. This permits both the "old" (time t^L) information and the "new" or advanced (time $t^L + \Delta t^L$) information to be stored and referenced during the timestep calculations for

each level. Since the number of levels that a Particle may lie on is allowed to vary during a simulation, the information for each parameter is kept in a variable length array.

The Particle class also stores additional information pertaining to an individual particle that are not particle properties but assist in the timestep calculations. For cases where a particle encounters a grid boundary, additional registers are in place to assist in calculations.

- The time the boundary is encountered.
- The location where the particle encounters the boundary.
- The velocity of the particle when the boundary is encountered.
- The momentum sources due to drag and evaporation at the time the boundary is encountered.

These parameters are defined to be identical to the corresponding "old" Particle parameters by default. When a grid boundary is encountered, these parameters are interpolated based on the particle's "new" and "old" location information and the location of the grid boundary.

Additional information stored to assist calculations include the following:

- An array of turbulent fluctuating velocities affecting the Particle during the current timestep.
- An array of turbulent timescales that correspond to the turbulent fluctuating velocities.
- The times associated with the "old" (time t^L) and "new" (time $t^L + \Delta t^L$) Particle information.
- Predicted and corrected aerodynamic drag forces.

- Predicted and corrected evaporation sources.
- Location of the Particle's originating nozzle.
- The time the Particle enters the calculation through a nozzle.
- A collection of placeholders for the breakup models (see Appendix F).

A.2 ParticleList

The ParticleList class contains all of the information and operations pertaining to a single list of Particles. The ParticleList is a double-linked list derived from the List class in BoxLib. To ease navigation through the list, ParticleList utilizes the helper class PLIter, derived from the ListIterator class in BoxLib, which makes accessing the list relatively similar to indexing an array. The ParticleList class provides the primary organizing structure for the Particles.

The ParticleList class also contains routines that orchestrate the spray submodel calculations that occur on a single ParticleList. These routines collect all needed information from the ParticleList, and passes on the particle and fluid information to the appropriate Fortran submodel routine.

A.3 PartColl

The PartColl class contains all of the information and operations pertaining to the entire collection of Particles that represents the spray. PartColl utilizes multiple ParticleLists to organize the Particles. The use of multiple lists provides a way of sorting the Particles, reducing the amount of searching needed during a timestep calculation. PartColl is made up of the following elements:

wrklst (working list) - A ParticleList to hold all of the Particles pertinent to the calculation on the current level.

inlst (injection list) - A ParticleList to hold all of the Particles that have not yet entered the calculation. This list is sorted during spray initialization in order of time of entry.

bndlst (boundary list) - A ParticleList to hold Particles that are entering the calculation during the current timestep. Particles move from the inlst to the bndlst before moving to the wrklst since particles entering the domain in mid-timestep require special treatment during the calculation.

lvlst (level lists) - An array of ParticleLists to hold Particles that are in the domain, but not relevant to the calculation on the current level. Particles are kept in a level list, the ParticleList associated with the finest level on which the Particle can be found, until it is again needed in the calculation.

PartColl also contains a collection of flags for the calculation. These include flags for indicating which submodels are to be used, which type of drag coefficient is to be used, the type of liquid fuel in the droplets, and whether the particles are non-influential.

Appendix B

Spray Definition

Particles that are to be injected into the domain through a nozzle need to be assigned an initial velocity to be used at the time of entry. The magnitude of this particle velocity, v_p, is defined from either the mass flow rate profile for the jet or the velocity profile for the jet, depending on which was defined. The initial trajectory of the particle is defined using the user-defined spray cone parameters, $(\bar{p}, \Phi, \Psi)$.

The primary jet direction, $\bar{p}$, is a vector, in the coordinate system of the domain, which lies on the conical line of symmetry and defines the direction the spray would follow if the spray cone were injected in to the domain as a single stream of droplets. The cone angle thickness, Ψ, is the angle that delimits the cross-section of the spray on one side of the primary jet direction line. The mean cone angle, Φ, is the angle from the mean jet direction to the center of the spray cross-section. The combination of these injection parameters uniquely defines the spray cone. See Figure B.1 for an illustration.

The initial particle trajectories need to be assigned such that the particles will be uniformly distributed throughout the spray cone. Thus, if we look at a planar cross-section of the three-dimensional spray cone (orthogonal to the primary jet direction vector), the points where the particles will pass through this surface should be uniformly distributed through the resultant area (see Figure B.2). To obtain this desired distribution, we sample two random numbers for each particle from uniform distributions. One of these distributions is scaled to cover the range $\left[-\frac{\Psi}{2}, \frac{\Psi}{2}\right]$, representing the particle position within the cone angle thickness

relative to the mean cone angle, and the other is scaled to cover the range $[-180°,180°]$, representing the azimuthal position of the particle in the cone.

B.1 Two Dimensions

In two dimensions, we can obtain the initial trajectory of a particle using the following scheme. Assume the base of the cone is located at the origin. Let ψ be the sampled position within the cone angle thickness for the particle, ξ be the sampled azimuthal position of the particle, and θ be the angle of the primary jet direction vector, as shown in Figure B.1. We use the azimuthal position of the particle to determine in which half of the two-dimensional spray cone the particle will be found by letting:

$$\tilde{\xi} = sign(\xi) \tag{B.1}$$

We define the resultant angle of the particle trajectory to be:

$$\gamma = \theta + \tilde{\xi}\,\Phi + \psi \tag{B.2}$$

We can then define the resultant particle trajectory, $\bar{t}$, in terms of the spray cone parameters in the following manner:

$$\frac{t_y}{t_x} = \tan(\gamma) \tag{B.3}$$

and

$$\frac{p_y}{p_x} = \tan(\theta) \tag{B.4}$$

by definition, so if we let $t_x = p_x$ we can write:

$$t_y = p_x \tan(\gamma\ ) \tag{B.5}$$

$$t_y = p_x \tan\left(\theta + \tilde{\xi}\,\Phi + \psi\right) \tag{B.6}$$

$$t_y = p_x \tan\left(\tan^{-1}\left(\frac{p_y}{p_x}\right) + \tilde{\xi}\,\Phi + \psi\right) \tag{B.7}$$

In the final step, $\bar{t}$ is normalized and multiplied by v_p to obtain the particle's initial velocity.

B.2 Three Dimensions

Again, assume the base of the cone is located at the origin. The spray cone shape can be defined in terms of the spray cone parameters naturally in spherical coordinates (assuming a unit length). The resultant cone, when transformed into Cartesian coordinates is symmetric about the z-axis. This cone can then be transformed such that it lies symmetrically about the primary direction vector, $\bar{p}$.

As before, let ψ be the sampled position of a particle within the cone angle thickness relative to the mean cone angle, ξ be the sampled azimuthal position of the particle, and define φ to be the angle of the particle trajectory from the line of symmetry as shown in Figure B.3. The particle's untransformed trajectory is

$$\begin{bmatrix} \tilde{t}_x \\ \tilde{t}_y \\ \tilde{t}_z \end{bmatrix} = \begin{bmatrix} \sin(\varphi)\cos(\xi) \\ \sin(\varphi)\sin(\xi) \\ \cos(\varphi) \end{bmatrix} \tag{B.8}$$

where, by definition, $\varphi = \psi + \Phi$.

The transformation is performed using a series of Givens rotations. If we define the angles θ and ϑ to be related to $\bar{p}$ in spherical coordinates such that

$$p_x = \|\bar{p}\|\sin(\vartheta)\cos(\theta) \tag{B.9}$$

then the transformation of the particle trajectory is performed in the following way:

$$\begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\vartheta) & 0 & \sin(\vartheta) \\ 0 & 1 & 0 \\ -\sin(\vartheta) & 0 & \cos(\vartheta) \end{bmatrix} \begin{bmatrix} \tilde{t}_x \\ \tilde{t}_y \\ \tilde{t}_z \end{bmatrix} \qquad \text{B.10}$$

Since we started out with a trajectory of unit length, normalizing the transformed trajectory is not necessary. The particle velocity at the time of entry for a three-dimensional cone is then defined to be $v_p \bar{t}$.

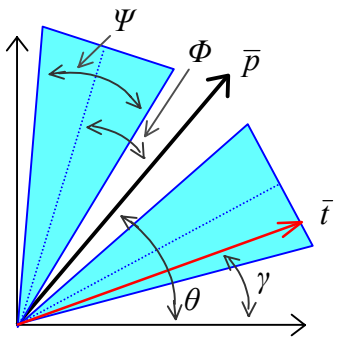

Fig. B.1 Representation of the parameters involved in calculating a particle trajectory for a two-dimensional spray cone.

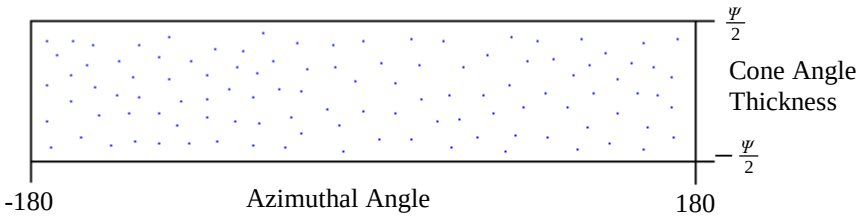

Fig. B.2 Representation of a spray cone cross-sectional area. For purposes of demonstration and clarity, the cross-sectional area has been transformed into its logically equivalent two-dimensional box.

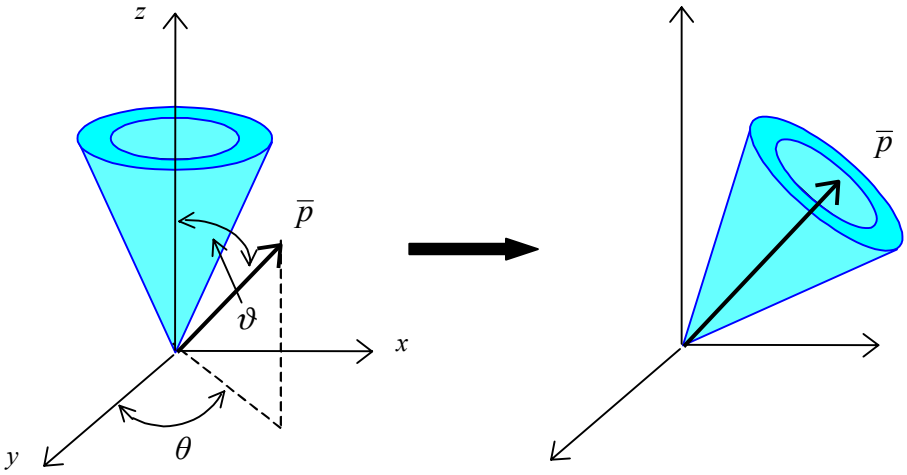

Fig. B.3 Representation of the parameters involved in transforming the particle trajectories for a three-dimensional spray cone.

Appendix C

Aerodynamic Drag Implementation

The spray calculations for the aerodynamic drag on the particles are interwoven with the predictor-corrector formulation of the fluid calculations. To accomplish this, the spray calculations are also formulated in a predictor-corrector fashion. The differential form of the Equations 4.1 - 4.3 for the particles translates into the following discrete, finite-difference, predictor-corrector formulation.

$$x^{*}_{p,i} = x^{n}_{p,i} + \Delta\tau_p \left(\frac{v^{*}_{p,i} + v^{n}_{p,i}}{2} \right) \tag{C.1}$$

$$x^{n+1}_{p,i} = x^{n}_{p,i} + \Delta\tau_p \left(\frac{v^{n+1}_{p,i} + v^{n}_{p,i}}{2} \right) \tag{C.2}$$

$$v^{*}_{p,i} = v^{n}_{p,i} + \Delta\tau_p \left(\frac{F^{n}_{fp,i}}{m_p} \right) \tag{C.3}$$

$$v^{n+1}_{p,i} = v^{n}_{p,i} + \Delta\tau_p \left(\frac{F^{*}_{fp,i} + F^{n}_{fp,i}}{2m_p} \right) \tag{C.4}$$

$$F^{n}_{fp,i} = \frac{1}{2} \rho^{n}_{f} C^{n}_{D} A^{n}_{p} \left| I\left(\bar{u}^{n}_{f}, \bar{x}^{n}_{p}\right) - \bar{v}^{n}_{p} \right| \left(I\left(u^{n}_{f,i}, \bar{x}^{n}_{p}\right) - v^{n}_{p,i} \right) \tag{C.5}$$

$$F^{*}_{fp,i} = \frac{1}{2} \rho^{*}_{f} C^{*}_{D} A^{*}_{p} \left| I\left(\bar{u}^{*}_{f}, \bar{x}^{*}_{p}\right) - \bar{v}^{*}_{p} \right| \left(I\left(u^{*}_{f,i}, \bar{x}^{*}_{p}\right) - v^{*}_{p,i} \right) \tag{C.6}$$

where $\Delta\tau_p$ denotes the time during the current timestep, Δt, that the particle is in the domain. It should be noted that $\Delta\tau = \Delta t$ unless the particle has entered the domain through the nozzle during the current timestep. The superscripts indicate the timestep of the variable:

n indicates an old timestep value, * denotes a temporary value (predicted value), and *n+1* indicates the new timestep value. The addition of a second subscript, *i*, indicates the component of a vector. The symbol $I(w,\bar{x})$ represents the interpolation operator of the scalar field *w* at location $\bar{x}$. As mentioned in Section 3.7.1, bilinear interpolation is used for all interpolation operations.

The discretization of the force acting on the fluid due to the particles includes an approximation for the delta function, as described in Sections 3.7.2 and 6.4. In the following expressions, this approximation is represented by a general basis function, $\chi^{\forall}(\bar{x})$, centered at location $\bar{x}$. Thus, the distribution of the momentum source due to aerodynamic drag on the spray is given by

$$F^{n}_{pf,i,IJK} = \sum_{all\ particles\ p} \frac{\Delta\tau_p}{\Delta t}\left(\frac{-F^{n}_{fp,i}\chi^{\forall}\left(\bar{x}^{n}_{p}\right)}{\forall_{IJK}}\right) \tag{C.7}$$

$$F^{*}_{pf,i,IJK} = \frac{\sum_{all\ particles\ p} \frac{\Delta\tau_p}{\Delta t}\left(\frac{-F^{*}_{fp,i}\chi^{\forall}\left(\bar{x}^{*}_{p}\right)}{\forall_{IJK}}\right) + \sum_{all\ particles\ p} \frac{\Delta\tau_p}{\Delta t}\left(\frac{-F^{n}_{fp,i}\chi^{\forall}\left(\bar{x}^{n}_{p}\right)}{\forall_{IJK}}\right)}{2} \tag{C.8}$$

where $F_{pf,i,IJK}$ denotes the i^{th} component of the force on the fluid due to the particles for grid cell *IJK*, and $\forall_{IJK}$ denotes the volume of grid cell *IJK*.

Appendix D

Turbulence Implementation Details

The turbulence submodel implemented into the AMR spray code is based on the same underlying principles used by the KIVA implementation, as described in Section 4.2, but differs in the details of the implementation. The presence of multiple time scales associated with refined grid levels within the AMR framework requires that the treatment of turbulence be scalable. For the case where the turbulent time scale is less than the current timestep, KIVA's practice of perturbing the particle position and velocity to simulate the passage of multiple eddies during a timestep (see O'Rourke, 1989) raises the question of whether the effect of turbulence on a coarse grid and on a fine grid during the same coarse timestep would be suitably correlated. The AMR spray code thus samples multiple fluctuating velocities and their associated turbulent timescales as needed. In this way, there is some continuity between the effect of the turbulence on a coarse level and turbulence on a finer level

D.1 Single Grid

For a given particle, *p*, pairs of turbulence parameters $\left(\bar{u}_p', t_p^{turb}\right)$ are sampled until the sum of the turbulent timescales, plus the remainder of the last timescale from the previous timestep, add up to at least the current timestep.

$$\left(\delta t_p^{turb}\right)_o + \sum_{k=1}^{m-1}\left(t_p^{turb}\right)_k + \left(\delta t_p^{turb}\right)_m = \Delta t \qquad \text{(D.1)}$$

where $\left(\delta t_p^{turb}\right)_o$ is the portion of the timescale sampled during the last timestep that lies within the current timestep, and $\left(\delta t_p^{turb}\right)_m$ is the portion of the last sampled timescale that lies within the current timestep.

From these pairs of turbulence parameters, an effective fluctuating velocity is calculated for the current timestep.

$$\bar{u}'_{eff,p} = \frac{\left(\delta t_p^{turb}\right)_o \left(\bar{u}'_p\right)_o + \sum_{k=1}^{m-1} \left(t_p^{turb}\right)_k \left(\bar{u}'_p\right)_k + \left(\delta t_p^{turb}\right)_m \left(\bar{u}'_p\right)_m}{\Delta t} \tag{D.2}$$

This effective fluctuating velocity is used everywhere in the calculations where the relative velocity between the particle and the fluid is used. For example, in the aerodynamic drag calculations described in Appendix C, the force on a particle is given by

$$F_{fp,i}^n = \frac{1}{2}\rho_f^n C_D^n A_p^n \left| I\left(\bar{u}_f^n, \bar{x}_p^n\right) + \bar{u}'_{eff,p} - \bar{v}_p^n \right| \left(I\left(u_{f,i}^n, \bar{x}_p^n\right) + u'_{eff,p,i} - v_{p,i}^n \right) \tag{D.3}$$

$$F_{fp,i}^* = \frac{1}{2}\rho_f^* C_D^* A_p^* \left| I\left(\bar{u}_f^*, \bar{x}_p^*\right) + \bar{u}'_{eff,p} - \bar{v}_p^* \right| \left(I\left(u_{f,i}^*, \bar{x}_p^*\right) + u'_{eff,p,i} - v_{p,i}^* \right) \tag{D.4}$$

D.2 Adaptive Grids

The basic approach for finding an effective fluctuating velocity to use within a timestep is essentially unchanged. However, with the presence of finer grids and thus finer timesteps, the issue of when to resample the pairs of turbulent parameters arises. The most direct and simple approach would be to sample the turbulent parameter pairs at the beginning of the coarse timestep and use them for all grid levels. However, this does not take advantage of new and more accurate turbulence field information as it is calculated on the finer grids.

In this implementation, we use the order in which finer timesteps are taken within a coarse timestep to determine when new turbulence parameters need to be sampled. The coarse level samples enough turbulence parameter pairs to cover the coarse timestep, as is done for single grid calculations. The effective fluctuating velocity is calculated and the level 0 advance is performed. For each successive advance, on some level L, the turbulence parameters are handled as follows: If the current level L timestep is the first one after a level L-1 advance, then the effective velocity is calculated from an appropriate subset of the turbulence parameters used for level L-1. The first fine timestep always uses the same fluid information as the previous level's timestep, so there is no sense in resampling the turbulence parameters. For subsequent level L timesteps, the remaining turbulence parameters are ignored and new turbulence parameters are resampled using current level L values for the k and ε fields.

D.3 Discussion

Technically, using each fluctuating velocity separately for the time indicated by the associated timescale would be more accurate within the calculations. In this case, supposing that there were m pairs of turbulence parameters valid for a given timestep, each spray model equation using the fluid's fluctuating velocities would need to be performed m times each timestep and the resulting source terms and change in spray properties summed together. Also, a decision would need to be made regarding how to determine the fluid properties at the sub-timesteps. This type of procedure becomes very expensive when the turbulence timescales are small.

The effects of using an effective turbulent fluctuating velocity instead of considering each fluctuating velocity separately were determined through a couple simple test cases. For the case where each fluctuating velocity was considered independently, the fluid properties were held constant through the timestep, but the new spray properties for each subtimestep were considered.

Thus, for a case where only aerodynamic drag and turbulence are factors, the resulting force on the particle when fluctuating velocities are considered separately is given by

$$\overline{F}_{fp}(t) = \sum_i F_{fp}\left(t, \overline{u}'_p(\delta t_i)\right)\frac{\delta t_i}{\Delta t} \tag{D.5}$$

such that

$$\sum_i \delta t_i = \Delta t \tag{D.6}$$

and δt_i is either t_p^{turb} or the portion of t_p^{turb} within the current timestep, Δt. Alternatively, when an effective fluctuating velocity is used, the resultant force is given by

$$\overline{F}_{fp}(t) = \overline{F}_{fp}\left(t, \sum_i \overline{u}'_p(\delta t_i)\frac{\delta t_i}{\Delta t}\right) \tag{D.7}$$

The difference between these two approaches was determined to fall within the random deviation found when using different seeds for the random number generator and using effective fluctuating velocities only.

Appendix E

Evaporation Implementation Details

As with the aerodynamic drag submodel, the evaporation submodel calculations are interwoven with the predictor-corrector formulation of the fluid calculations. The equations below are written and described using a general form, without specifying the subcycle of the calculation or whether it is for the predictor or corrector part of the calculation. The first subcycle of each particle's evaporation calculations use the appropriate original or predicted values for the particle and fluid. Following subcycles use the updated particle and fluid properties from the previous subcycle. Most of the equations used in the evaporation model and listed in this appendix may be found in the KIVA II manual (Amsden et al., 1989).

The number of subcycles needed to perform the evaporation calculations is defined such that the heat transfer to a computational particle does not exceed some fraction (in this case, half) of the available energy for transfer during a single subtimestep. (Amsden et al., 1989) Thus, the number of subcycles, N_{ev}, is the smallest positive integer such that

$$N_{ev} \geq \frac{V_{sh}\,\mu_f(T_f)\,4\pi\, r_p N_d}{\rho_f(T_f)\forall_{IJK}\Delta t} \tag{E.1}$$

where

$$V_{sh} = 2 + 0.6Re^{1/2}Sc_d^{1/3}$$

$\forall_{IJK}$ is the cell volume, Re is given by Equations 4.5 or 4.8, and Sc_d is given by Equation 4.16. Once the number of subcycles has been determined, the evaporation timestep, δt_{ev}, can be calculated using the expression

$$\delta t_{ev} = \frac{\Delta t}{N_{ev}} \tag{E.3}$$

The temperature calculation uses the expression

$$\rho_d \frac{2}{3}\left(r_p^v\right)^2 c_{liq}\left(T_p^v\right)\left[\frac{T_p^{v+1} - T_p^v}{\delta t_{ev}}\right] = K\left(T_f^v\right)\left[T_f^v - T_d^{v+1}\right]V_{Nu}^v \frac{\ln\left(1+YR_{fv}^{v+1}\right)}{YR_{fv}^{v+1}} \tag{E.4}$$

$$- L_{vap}\left(T_p^v\right)\left[\rho D_f\left(T_f^v\right)\right]V_{Sh}^v \ln\left(1+YR_{fv}^{v+1}\right)$$

where the superscript indicates which subcycle the value is obtained from, $K\left(T_f^v\right)$ is the coefficient of heat conductivity of the fluid, YR_{fv}^{v+1} is given by Equation 4.13 using the fuel vapor temperature equal to subcycle ($v+1$)'s particle temperature, $L_{vap}\left(T_p^v\right)$ is the liquid fuel's latent heat of vaporization, V_{Sh}^v is given by Equation E.2, and

$$V_{Nu}^v = 2 + 0.6 Re^{1/2} Pr_d^{1/3} \tag{E.5}$$

where Pr_d is given by Equation 4.19.

The new droplet temperature, T_p^{v+1}, is solved implicitly using the Secant method. The initial guesses for the new temperature are the current drop temperature, $z_1 = T_p^v$, and a temperature that is 1% less, $z_o = 0.99 T_p^v$. At each step of the secant method, the residual of the new temperature is determined, i.e. $res_n = T_p^{v+1} - T_p^v = z_n - T_p^v$, using the temperature equation (E.4). A new guess for the temperature is then determined by the expression

$$z_{n+1} = z_n - \left(res_n\right)\left[\frac{z_n - z_{n-1}}{\left(res_n\right) - \left(res_{n-1}\right)}\right] \tag{E.6}$$

given $res_o = 0$.

Once the temperature iteration converges, the new drop radius is calculated using the expression

$$\left(r_p^{v+1}\right)^2 = \max\begin{cases} 0.0 \\ \left(r_p^{v}\right)^2 - \delta\, t_{ev}\, \dfrac{\rho\, D_f\left(T_f^{v}\right)}{\rho_d} V_{Sh}^{v}\, \dfrac{\ln\left(1+YR_{fv}^{v}\right)+\ln\left(1+YR_{fv}^{v+1}\right)}{2} \end{cases} \qquad \text{(E.7)}$$

insuring that the radius can never become unphysically negative.

Changes in the fluid properties can now be calculated from the new particle temperature and droplet radius. These changes are added to the local fluid properties used to perform the evaporation, but also added to fluid source terms that are applied to the fluid later. The changes in fluid properties are applied as follows.

For the fuel vapor mass and the total fluid mass,

$$m_f^{v+1}\left(\bar{x}_p\right) = m_f^{v}\left(\bar{x}_p\right) - \frac{4}{3}\pi\,\rho_d\left[\left(r_p^{v+1}\right)^3 - \left(r_p^{v}\right)^3\right]N_d \qquad \text{(E.8)}$$

$$\dot{\mathrm{M}}_p = \dot{\mathrm{M}}_p - \frac{4}{3}\pi\,\rho_d\left[\left(r_p^{v+1}\right)^3 - \left(r_p^{v}\right)^3\right]N_d \qquad \text{(E.9)}$$

where $m_f^{v}\left(\bar{x}_p\right)$ is the mass of the fluid in the cell surrounding the particle, and $\dot{\mathrm{M}}_p$ is the mass source term to be distributed at the location of the particle.

For the fluid enthalpy,

$$(mh)_f^{v+1}\left(\bar{x}_p\right) = (mh)_f^{v}\left(\bar{x}_p\right) - \frac{4}{3}\pi\,\rho_d\left[\left(r_p^{v+1}\right)^3 h_p\left(T_p^{v+1}\right) - \left(r_p^{v}\right)^3 h_p\left(T_p^{v}\right)\right]N_d \qquad \text{(E.10)}$$

$$\dot{\mathrm{H}}_p = \dot{\mathrm{H}}_p - \frac{4}{3}\pi\,\rho_d\left[\left(r_p^{v+1}\right)^3 h_p\left(T_p^{v+1}\right) - \left(r_p^{v}\right)^3 h_p\left(T_p^{v}\right)\right]N_d \qquad \text{(E.11)}$$

where h_p is the enthalpy of the particle, and $(mh)_f^{v}$ is the mass times enthalpy of the fluid in the cell surrounding the particle.

For the fluid temperature, the definition of the relationship between temperature and enthalpy is used.

$$\dot{T}_p = \frac{\left[{(mh)_f^{v+1}} \Big/ {m_f^{v+1}} \right] - h_f}{c_p} \tag{E.12}$$

$$T_f^{v+1} = T_f^v + \frac{\left[{(mh)_f^{v+1}} \Big/ {m_f^{v+1}} \right] - h_f}{c_p} \tag{E.13}$$

where h_f is the enthalpy of the fluid, and c_p is the fluid's specific heat, both determined from the fluid properties at the beginning of the timestep.

Once evaporation for all of the particles is complete, the predicted (or corrected) values for the particle mass, radius and temperature are set and the fluid source terms are passed on. For each particle, the predicted values for the particle mass, radius, and temperature are defined as

$$m_p^* = m_p^V \tag{E.14}$$

$$r_p^* = r_p^V \tag{E.15}$$

$$T_p^* = T_p^V \tag{E.16}$$

and the corrected values as

$$m_p^{n+1} = {1}/{2}\left(m_p^n + m_p^V\right) \tag{E.17}$$

$$r_p^{n+1} = \sqrt[3]{\frac{3m_p^{n+1}}{4\pi\,\rho_d N_d}} \tag{E.18}$$

$$T_p^{n+1} = {1}/{2}\left(T_p^n + T_p^V\right) \tag{E.19}$$

where the n superscript indicates "old" values, the $n+1$ indicates the "new" values, and the V superscript indicates the value of the property after the final subcycle of the evaporation calculation. The corrected values for mass and temperature are obtained by averaging the predicted and corrected change of these values.

Appendix F

Breakup Models

Both breakup models use a collection of placeholders to keep track of particle information over several timesteps (also see Appendix A). These placeholders include:

Reitz Wave Model

shedmass - A parameter that denotes the amount of mass that has been shed from the particle since the last breakup.

WgrwYN - A flag that indicates whether the droplet radius of the particle has been allowed to enlarge (ie. assumed to have broken off from the 'liquid core' of the spray). This flag is also set the first time a child particle is created (for both the parent and child particles), or the first time the particle breaks up according to the Rayleigh-Taylor model.

drop_num_orig - A parameter that denotes the original number of droplets in the particle, or the number of droplets in the particle just after the last child particle was created.

Rayleigh-Taylor Model

TRT_break - A parameter that denotes the amount of time that has passed since the last breakup (Rayleigh-Taylor or Reitz wave model).

RTbrkYN - A flag that indicates whether the particle is allowed to breakup due to the Rayleigh-Taylor model. This flag is set when it has

traveled a user-defined distance from the nozzle, or when the particle is a child particle of an earlier Reitz wave model breakup. When breakup occurs according to one of the models, the information placeholders for both models are reset and the breakup process starts over using the new particle properties.

A set of user-defined constants that tune the behavior of the breakup models is also used. The role of these constants is noted in Section 4.5, and can be described as follows:

bw_rad - Also noted as C_{wave_rad}, it is used to scale the radius of the shed droplets for the Reitz wave breakup model.

bw_time - Also noted as C_{wave_time}, it is used to scale the breakup timescale for the Reitz wave model.

bw_vel - Also noted as C_{wave_vel}, it is used to scale the magnitude of the velocity perturbation for new droplets in the Reitz wave model.

crt_dist - The breakup distance for the Rayleigh-Taylor breakup model.

crt_time - Also noted as C_{RT_time}, it is used to scale the breakup timescale for the Rayleigh-Taylor breakup model.

crt_rad - Also noted as C_{RT_rad}, it is used to scale the estimate of the fastest growing wavelength in the Rayleigh-Taylor breakup model.

Appendix G

Collision Models

G.1 Data Structure

The collision submodels are the only physical spray submodels, other than the initialization routines, that are implemented entirely in C++ rather than in Fortran. Utilizing C++ for these routines permitted the use of a container class in the C++ Standard Template Library (STL) that was convenient for organizing the particles by cell and then retrieving the needed particle information at low computational cost. Unlike the other spray submodels, the collision submodels do not require any fluid property information, relinquishing the need to use Fortran.

The container class used to hold and sort the particles by cell was the STL's multimap that holds pairs of data consisting of a "value" (which can be of any type or class) and a "key" (which can be of any type or class that has a defined ordering). As pairs are added to the multimap, they are organized/sorted according to each pair's key. The multimap container class allows multiple pairs to have the same key. The STL also provides iterators that allow access to the multimap pairs in an array-like fashion for any given key. For the case of holding the spray information, the pair consists of a pointer to a Particle and the Particle's associated collision cell indices.

G.2 Symmetry Boundaries

Special handling was implemented to handle symmetry boundaries with the NTC and Proximity collision models. When particles were clustered around a symmetry boundary,

such as when the symmetry boundary runs down the spray's axis of symmetry, the particles were observed to 'pull away' from the symmetry boundary over time. This behavior is familiar to KIVA users where the collision grid corresponds directly to the fluid grid and thus remains stationary. In these cases, particles tend to coalesce away from the collision grid cell boundaries, collecting in the center of the cells. It is hypothesized that a similar phenomenon occurs at the fixed symmetry boundary even though the remainder of the collision grid is perturbed at each timestep for the NTC and Proximity collision models.

To counteract the effect of the symmetry boundary, the particles along the symmetry boundary were duplicated and reflected across the boundary, though not officially added to the spray, to fill the portion of the collision grid cells that lay outside the symmetry boundary. It was hoped that this would effectively remove the influence of the symmetry boundary. One negative side effect is the introduction of a non-conservative factor to the spray. Mass from a duplicated particle outside the domain could coalesce with particle inside the domain, resulting in either more or less total mass in the spray depending on which particle received the mass from the other. At the end of the collision calculation, the duplicated particles are discarded.

The 'pulling away' behavior of the spray was somewhat alleviated by the above measures, but not removed entirely. Further, when the spray is dense along the symmetry boundary, the creation of and collision with the duplicated particles significantly increased the computational cost of the collision models. For these reasons, it was decided not utilize symmetry along the spray axis.

G.3 Algorithm Cost

The collision model implemented in KIVA checks each pair of particles in the domain to see if they occupy the same cell. This algorithm has a cost of $O(N_p^2)$, where N_p is the number of particles in the domain (Schmidt and Rutland, 2000). The KIVA collision model implemented in the AMR spray code first sorts the particles by collision grid cell, with a cost of $O(N_p \log N_p)$ (Robson, 1999), then checks each pair of particles found in each cell. The resulting cost of this algorithm is still $O(N_p^2)$ since all of the particles could be in the same collision grid cell, but could do as low as $O\left(\frac{N_p^2}{N_c}\right)$ where N_c is the number of collision grid cells in the domain. Because the particles are first sorted by collision grid cell, the algorithm cost is bounded below by $O(N_p \log N_p)$.

As noted in Schmidt and Rutland, the NTC scheme has a cost of $O(N_{pc})$ per collision grid cell, where N_{pc} is the number of particles in the cell. The NTC scheme therefore has an overall cost of $O(N_p N_c)$ but could do as well as $O(N_p)$. However, since the particles must first be sorted according to collision grid cell, the NTC scheme cost is bounded below by $O(N_p \log N_p)$.

The Proximity collision scheme uses essentially the same algorithm for choosing potential collision pairs as the KIVA collision model, with the exception that particles may exist in neighboring cells. Since the number of possible neighboring cells is essentially constant, the overall cost of the Proximity collision model is identical to the KIVA scheme.

Appendix H

Species Transport Properties

The AMR code utilizes the Los Alamos database, TRANSPORT (Kee et al., 1986), which works in conjunction with CHEMKIN (Kee et al., 1980), to determine the transport properties (ie. viscosity, heat conduction coefficient, and diffusion coefficient) of the fluid given the local composition and properties of the fluid. While using the TRANSPORT database is sufficient for most species commonly found in the fluid, it is not appropriate for fuel vapor that is added via spray evaporation. TRANSPORT is based on the Lennard-Jones molecular model, which assumes that the molecule of each species is roughly spherical and non-polar. The molecules of the fuels used in this study are long hydrocarbon chains that do not conform to the assumptions used by TRANSPORT. Thus, a method of specifying alternate transport properties of the fuel vapor was needed.

It was decided that utilizing constant, user-defined transport properties for the fuel vapor was preferable to calculating variable properties with TRANSPORT based on estimated parameters resulting in unrealistic transport behavior. To this end, the routines for obtaining the transport properties were modified such that the TRANSPORT database was used to obtain the transport properties for all fluid species except the fuel vapor. The constant fuel vapor properties were then used with the properties of the remainder of the fluid to obtain the composite fluid transport properties.

In later tests, it was determined that the effect of using constant transport properties over using a “best guess” in the TRANSPORT database for the fuel vapor had a minimal

effect on the resultant calculated liquid fuel and fuel vapor penetrations. The differences in these measurable quantities fell within the margin of error due to random number effects.

Appendix I

Divergence Constraints and Variable Ambient Pressure

The pressure field is calculated as part of the projection procedure in which the incompressibility velocity divergence constraint is enforced. Detailed explanation of the divergence and projection calculations will not be provided here. The reader is referred to the relevant AMR literature, such as Bell and Marcus (1992), Pember et al. (1998), Almgren et al. (1998), etc.

In the unaltered form of the LBNL-CCSE AMR code, the calculated pressure field is a perturbation of a constant and uniform field of a user-defined magnitude. This constant ambient pressure is used primarily when determining thermodynamic and transport properties of the species that compose the fluid. Though the composition of the fluid may change due to chemistry, no mass is created or destroyed and thus the ambient pressure remains unchanged. Diesel sprays, in practice, occur within closed domains. By modeling the liquid spray and the surrounding fluid as two separate entities, the transfer of fuel from the liquid state to the vapor state due to evaporation appears as a mass source to the ambient fluid. In the closed domain, this results in an increase of the ambient pressure.

I.1 Single Grid

Incorporating a variable ambient pressure into the AMR code for single grid cases consisted of essentially three different parts: a method of calculating the current ambient pressure, the creation of time variable parameters needed to perform the calculation, and the

alteration of the AMR code to refer to the new variable pressure instead of the constant ambient pressure.

Through consultation with the LBNL-CCSE collaborators, it was determined that the rate of change of the ambient pressure was proportional to the average of the velocity divergence field, ie.

$$\frac{\partial p_{amb}}{\partial t} = \frac{p_{amb}\int_{\forall} \nabla \cdot \bar{u}_f \; d\forall}{\forall} \tag{I.1}$$

As the velocity divergence field is calculated twice for each timestep calculation as part of the predictor-corrector solve, the calculation of the ambient pressure should take into account the most recent divergence information. The ambient pressure calculation, for a single, unadapted grid, was then determined to be of the form

$$p^{*}_{amb} = p^{n}_{amb} e^{\Delta t \tilde{\upsilon}^{n}} \tag{I.2}$$

$$p^{n+1}_{amb} = p^{n}_{amb} e^{\frac{\Delta t}{2}\left(\tilde{\upsilon}^{n}+\tilde{\upsilon}^{n+1}\right)} \tag{I.3}$$

where

$$\tilde{\upsilon} = \frac{\int_{\forall} \nabla \cdot \bar{u}_f \; d\forall}{\forall} \tag{I.4}$$

p^{n}_{amb} is the ambient pressure at the beginning of the timestep, Δt is the local timestep and $\forall$ is the total volume of the domain.

To facilitate the calculation of the ambient pressure at any point during the calculation, several parameters were created to hold the needed information. One parameter held the ‘old’ ambient pressure, p^{n}_{amb}, one provided a placeholder for the spatial average of

the velocity divergence to be placed whenever it was calculated, and two parameters held the spatial average of the velocity divergence to be used in the ambient pressure calculation, $\tilde{v}^n$ and $\tilde{v}^{n+1}$, retrieved from the aforementioned placeholder at the appropriate points during the calculation.

The AMR code alterations were largely incorporated into the fluid-spray interface with a couple of exceptions. The actual calculation of the fluid viscosity, heat conductivity and diffusivity occur in Fortran routines. Since the relevant functions could not be overridden with 'virtual' functions, as is possible with C++, these routines were modified directly.

I.2 Adaptive Grids

The inclusion of adaptation complicates the issue of calculating the ambient pressure. For calculations on each finer level, the ambient pressure must be consistent with that used for the coarsest level calculations. To meet this restriction, it was decided to calculate the change in ambient pressure through time on the coarsest level then use this information to interpolate the appropriate ambient pressure in time for each finer timestep calculation. It should be noted that this approach does not make use of the more accurate information provided by the finer levels as it becomes available.

I.3 Discussion

The issues involved with incorporating a variable ambient pressure into the AMR code had not previously been considered by the LBNL-CCSE code developers. Unlike the addition or modification of a scalar or vector field, for which there are many existing

examples, the transformation of the ambient pressure from a constant parameter to a variable one introduced numerous questions regarding the appropriate handling of the new variable. While the ambient pressure is an integral part of the AMR fluid code, the LBNL-CCSE researchers had no immediate need to alter the constant ambient pressure assumption.

The LBNL-CCSE researchers attempted to help formulate the changes needed to implement a variable ambient pressure through numerous discussions. However, due to limited experience in the intricacies of the AMR code, limited available support from those with more experience, and the inclusion of adaptivity issues that had not been yet been considered in depth, the attempt to implement a variable ambient pressure were unsuccessful. In single-grid, closed-domain, evaporation tests, the ambient pressure itself appeared to be varying in a reasonable fashion, but resulted in convergence problems in other parts of the fluid code. These problems were not resolved. Basic adaptivity handling of the ambient pressure was implemented but never fully tested.

In the end, it was advised that this issue be put aside to be addressed at a later date. To avoid problems associated with mass source terms when using the evaporation submodel, at least one boundary was left open for all evaporation test cases. The presence of an outflow boundary allows the AMR code itself to adjust the flow as needed in response to the added mass. It was determined that an open boundary would have a minimal effect on the development of the spray and the associate fluid flow, as long as the boundary was sufficiently far from the active spray area.

Appendix J

Random Number Generators

The spray model implemented into the AMR code is strongly based on stochastic concepts and makes broad use of random numbers. When this project began, the LBNL-CCSE AMR code had no provisions for generating random numbers. Thus, a variety of random number generators was needed to support the spray part of the code.

Random number generators typically use mathematical expressions for generating numbers that belong to particular random number distributions. These generators are never truly random, and different algorithms have varying levels of success. To obtain good quality random number generators, publicly available routines from well-respected sources were investigated. For this project, a series of random number generators were obtained from the DATAPAC package (Filliben, 1977) of numerical routines.

The routines in DATAPAC are written in Fortran and only use single precision variables. However, the routines provide a full suit of random number generators for a wide variety of probability distributions, and permit the user to control the number generation through the definition of the random seed. It is suggested that these routines be replaced in the future by better quality routines as they become available.

The random probability distributions that are used in the spray code include:

- uniform distribution between 0 and 1
- normal distribution around 0 with a standard deviation of 1
- negative exponential distribution
- χ-squared distribution