ADAPTIVE MESH REFINEMENT AND VISIOMETRICS IN ACCELERATED INHOMOGENEOUS FLOWS

By

SHUANG ZHANG

A thesis submitted to the

Graduate School-New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Master of Science

Graduate Program in Electrical and Computer Engineering

written under the direction of

Professor Manish Parashar

and approved by

_____

_____

_____

New Brunswick, New Jersey

January, 2004

ABSTRACT OF THE THESIS

ADAPTIVE MESH REFINEMENT AND VISIOMETRICS IN ACCELERATED INHOMOGENEOUS FLOWS

BY SHUANG ZHANG

Thesis Director: Professor Manish Parashar

This thesis investigates the issues introduced by the advanced computing technologies applications to our numerical simulation of accelerated inhomogeneous turbulence. The two focuses are adaptive mesh refinement (***AMR***) and data visualization/processing. The insight will lead to direct impact on numerical study of turbulent mixing and mass transport in astrophysics, inertial confinement fusion and internal combustion.

The turbulent environment is initiated by a shock wave hitting a gas inhomogeneity in a supersonic shock tube in both 2D and 3D. Gas bubbles, driving by the vorticity deposited baroclinically on the density interfaces, evolve in time and interact with each other. The fact that the flow field is occupied with isolated structures through out the evolution motivates the introduction of ***AMR*** scheme. In this thesis, we introduced two AMR schemes into our simulation: FLASH and GrACE. And the performance gains are analyzed. However, the complexity introduced by these advanced schemes makes the error analysis extremely difficult ---- the issues of verification and validation of numerical schemes, which haven't been addressed with enough care. In this thesis, we study the convergence of the multi-dimensional AMR application.

On the other hand, it is essential to obtain an access to the physical quantities associated with the bubbles and their evolution during the interaction. This could not be

accomplished by the standard visualization technique. In addition, high-resolution requirements for resolving as many scales as possible in a turbulent study make the data set significantly larger, and more complex if AMR scheme is invoked. Efficient visualization and well-designed data abstraction is necessary to understand the turbulent physics.

In this thesis, we design a pipeline of feature-based analysis to extract regions of interest, then visualize, track, isolate and quantify their evolution. We further extend the feature extraction and tracking scheme in a computational steering environment, to handle large scale AMR dataset. We address quantitatively the spatial and temporal diffusivity of the mixing zone dominated by coherent vortex structures. We study and compare both slow/fast/slow (a helium curtain in air) and fast/slow/fast (a air curtain in helium) cases to illustrate the correlation of mass and momentum diffusivity.

Data projection and space-time analysis are used as meanings of data abstraction to obtain the insight of a complex physical phenomenon. In this thesis, we design a comprehensive visiometrics pipeline, and encapsulate it innovatively into an optimization loop to quantify the error in a feedback manner. We are able to expose the error and correlate these errors to initial physical or numerical parameters during the experimental or numerical investigation. Excellent agreement is obtained between our simulation of a shock bubble interaction and the experiments performed at Los Alamos national lab (***LANL***). Our results outperformed both qualitatively and quantitatively the simulation at LANL. This methodology could be generalized to other disciplines.

ACKNOWLEDGEMENTS

First of all, I would like to thank my thesis advisors, Professor Manish Parashar and Professor Norman J. Zabusky, whose constant support, encouragement, and profound scientific insight make this research possible. The type of trainings I've been exposed, in science as well as in various aspects of life, is something I'll be benefit from through out my life.

I want to express my gratitude to my thesis committee members: Professors A. Zebib and T. Wei for their review of this thesis. I am also indebted to Prof. D. Silver, whose pioneering work and insight advices in scientific visualization are part of the foundation of this dissertation.

I thank Prof. S. Subramaniam and Prof. Pelz, who had been reviewer of my dissertation proposal and gave me a lot of valuable suggestions on the layout of the dissertation. Mr. Gaozhu Peng gives me a lot of constructive suggestions, and I really enjoy many of our valuable discussions.

I want to acknowledge the extensive help I received during the course of this dissertation by the computing stuff at CAIP center, particularly Mr. Bill Kish and James Chun, and at ASCI/FLASH center.

Finally, I want to thank my wife Jian for her love, our parents and sisters for their constant and crucial support, and my son Cheney through whose eye I am re-discovering the world.

DEDICATIONS

To:

Jian

Our parents and sisters

Cheney

Table of contents

Chapter 1 Introduction

1.1 Overview

Parallel computing is now a common approach in CFD of large-scale, complex flow system. Aside from the benefit provided by parallel computing, there are additional features in AIF we can use to invoke further high performance computing (***HPC***) approaches, primarily, adaptive mesh refinement (***AMR***). The advances in computer architecture and algorithms provide the feasibility of these approaches. This thesis will explore the application of the HPC approaches to AIFs flows.

Visiometrics is a powerful methodology in our computational study. It is defined as a data pipeline of visualization, quantification and juxtaposition. Visiometrics can lead us the way to qualitative observations such as phenomena discovery and physical property identification, and to quantitative investigations such as the physical scaling laws and reduced mathematical modeling. HPC approaches bring new challenges to visiometrics because of the increased complexity in data structure. This thesis developed a systematic visiometrics pipeline in the HPC environment, integrated many visiometrics modules developed or enhanced by many researchers in Vizlab including the author. Most noticeably, data projection and space-time analysis allows compressive summary of the simulation data evolving in both space and time; feature analysis allows the access to the localized quantities, VPs in particular, and enables the detail turbulent study as well as reduced modeling.

1.2 Accelerated inhomogeneous flows

A fundamental interaction in compressible hydrodynamics is that between accelerations and density interfaces. This environment is often referred to as the Rayleigh-Taylor (***RT***) or Richtmyer-Meshkov (***RM***) instability environment, or more generally, Accelerated Inhomogeneous Flow (***AIF***) environment [Zabusky, 99].

Our work is motivated by a variety of applications in nature as well as the physical challenges untapped underlying these applications.

1.2.1 Domain of applications

The interactions between the acceleration and density inhomogeneity occur in a myriad of fundamental and applied situations, from supernova explosions in astrophysics to sonic booms in the Earth's atmosphere (geophysics). In supernova explosions, extremely strong shock waves travel through enormous density gradients. The light curve and formation of mushroom structures and subsequent mixing requires an understanding of RM instabilities for an explanation, e.g., supernova (***SN***) 1987A remnant evolution [Arnet et al, 89][Arnet, 00].

Another important application is inertial confinement (laser) fusion (***ICF***), a potential future nuclear powered energy source. Figure 1.1 shows the basic ICF concept [General Atomic, 03]. During the implosion-explosion process, hydrodynamic instability, most noticeably RM instability, of the impulsively accelerated shell containing the deutium-tritium fuel limits the compression of the fuel which is important to achieve the high temperature where the nuclear reaction releases enormous amounts of energy. Thus RM instability represents a significant obstacle to achieving a productive fusion reaction [Lindl, 1995]. The fundamental mechanism of RM instability has also been considered to be of importance to enhancing mixing in supersonic combustion [Yang *et. al.,* 93].

These applications in general contain high energy and high-density fields, and hence are also referred to as high energy density physics. The system is intrinsically complicated, involves chemical reaction, MHD, and many combination of different flow patterns (multi-phase, combustion, hypersonic, etc).

Our focus in this study in on the hydrodynamics part of this high energy density environment, e.g., we assume gamma-law gases all the time and neglect chemical reaction. Even with this simplification, the complexity of the AIF system still left a lot of challenges untapped.

1.2.2 Important physical and numerical challenges

The initial accelerations in AIFs may arise from shock or blast waves (RM environments), bubble (or cloud) collisions impulsive radiation pressure, gravity (RT environments), etc. Most of the community focuses on the instability at interfaces arising from initial acceleration, and the turbulent mixing associated with these accelerations. These topics are also important parts in this thesis, with the assists of vortex paradigm analysis. In addition, this thesis work extends the investigation beyond early time, discovered and quantified the secondary baroclinic process.

1.2.3 Main geometries

Figure 1.2 sketches the major geometries of interests in studying AIF flows driven by RM instability. The first row shows density interfaces of different shape subjected to acceleration, and the second row shows inhomogeneity of different shapes subjected to acceleration. It is obvious that the latter case is of more practical interests.

1.3 Objectives and contributions

1.3.1 High performance computing -- Adaptive Mesh Refinement in parallel environments

The advance of massively parallel computers has enabled one to conduct the investigation of nonlinear phenomena with realistic grid resolutions. On the other hand, the increasing complexity and dimension naturally brought us in front of the problem of computation resources and time. Distributed and adaptive mesh become our very first option to maximize our computation within limited computer resources. All these concepts: parallel, dynamic, distributed, AMR, are part of high performance computing (***HPC***) family.

For the inhomogeneity-dominated flow under investigation, adaptive mesh refinement (***AMR***) is extremely suitable by only refining the region of interests to the higher resolution. This yields more mesh efficiency at the expenses of complex data structure and memory. This thesis explores some numerical aspects associated with AMR scheme, in an application point of view, particularly ParaMesh package and Grid Adaptive Computational Engine (***GrACE***) package.

1.3.2 Visiometrics

Visiometrics, or visualization, juxtaposition and quantification, is a scheme introduced by [Bitz & Zabuksy, 89]. It is first success is on the discovery of solitons with numerical simulation of KdV equation in 1960s. By integrating the modules developed in nearly 15 years since the foundation of Laboratory for Visiometrics & Modeling, 1989, with many new modules implemented during this thesis work, we prototype a comprehensive visiometrics pipeline. Parallel AMR dataset brings more challenges to visiometrics and is also addressed in this thesis.

Table 1.1 summarize the main parallel architectures investigated in this thesis. We give an approximate weight factor to all the machines with regard to the E10K Sparc II CPU.

1.4 Layout of the thesis

Chapter 2 focus on the introduction and some detail implementation of AMR scheme in to AIF environment, and some related high performance computing issues. In Chapter 3 we present the concept and illustrate the key technology underlying visiometrics. Chapter 4 discuss some important discoveries of verification and validation underlying HPC scheme, under our visiometrics mode of working. And Chapter 5 shows the application of visiometrics to uncertainty quantification. In Chapter 6 we conclude.

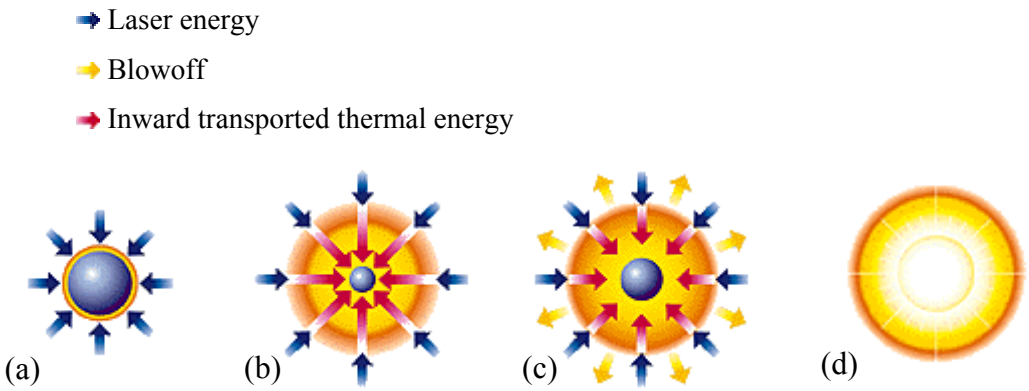


Fig. 1.1 Basic concept of inertial confinement (laser) fusion: a) Atmosphere formation: Laser beams rapidly heat the surface of the fusion target forming a surrounding plasma envelope. b) Compression: Fuel is compressed by the rocket-like blowoff of the hot surface material. c) Ignition: During the final part of the laser pulse, the fuel core reaches 20 times the density of lead and ignites at 100,000,000 degrees Celsius. d) Burn: Thermonuclear burn spreads rapidly through the compressed fuel, yielding many times the input energy [GA Web, 2003].

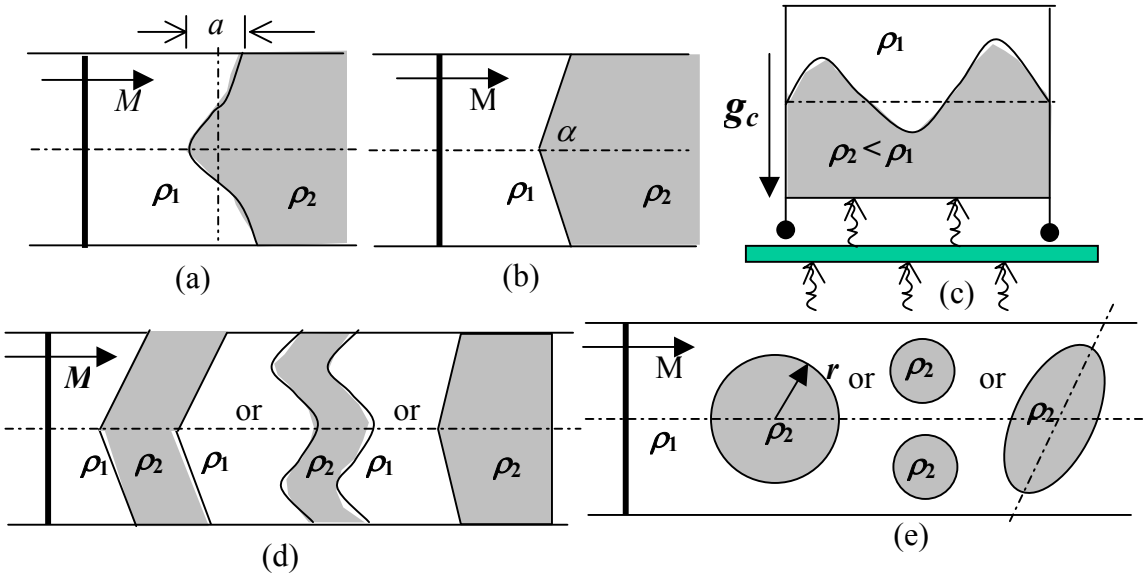


Fig. 1.2 Major geometries of interests in studying AIFs flows. (a). Sinusoidal interface; (b) Inclined interface; (c) Accelerated tank; (d) Curtain (inclined, sinusoidal and chevron); (e) bubble (cylinder/sphere, double cylinder/sphere & ellipse).

Machines (Abbrevation)	versicolor (SGI)	mphase (LAMD)	Teal (SE10K)	Green (SF12K)
OS	Onyx VTX	Linux	Sun OS	Sun OS
CPU type	SGI R10000	AMD MP	Sparc II	Sparc III
CPU numbers	8	44	32	36
CPU speed	194MHz	1.6GHz	400 MHz	900 MHz
Scale factor (w.r.t. E10K) *W*	0.2	1.5	1.0	2.8
Queue System	n/a	SGE	LSF	SGE
RAM	Shared 512MB	Distributed 512MB/per	Shared 16GB	shared 72 GB
Network Bandwidth	100MB	100 MB	12.8Gbytes	12.8 Gbytes

Tab. 1.1 Hardware configurations

Chapter 2 GrACE & FLASH: Adaptive mesh refinement (AMR) in high performance numerical simulation environment

2.1 AIFs and AMR: introduction

One main feature in AIFs is the inhomogeneity throughout the flow field, i.e., stratified flows. In most of the cases, the major physical activities, characterized by evolving and deforming density bubbles and interfaces (e.g., shocks), is highly localized to small fraction of the entire computational domain, while the bulk region of the simulated domain are relatively quiet. Due to the existence of the interface between gases, the localized regions usually require being highly resolved with a large number of numerical meshes. In the traditional uniform mesh simulation, the large amount of meshes in the vast "quiet" regions is a waste of resources. AMR scheme is a natural approach in solving the problem in the most computational time and memory efficient manner.

2.2 AMR: main concept and review

Adaptive mesh refinement is a scheme for finite difference and finite element codes wherein the size and distribution of the computational mesh is changed dynamically so that the solution complies with some specific constraint. Take 2D as an example, Figure 2.1 illustrates AMR scheme and the corresponding data structure [Berger & Oleger, 85]. The basic idea is the dynamic adaptation of the computational mesh to concentrate additional computational effort and resources to only those regions that require them. The method yields highly advantageous cost/accuracy ratio and makes larger scale simulations possible on a given set of resources comparing to static methods.

The superimposed meshes are generated in an autonomous fashion by computing the local error and comparing it with a preset refinement criteria. The error is usually associated with the gradient of certain variables of the flow field. The simulation starts at a certain coarse resolution, which we defined as base mesh, or AMR level 1. According to the refinement criteria, the simulation might put more meshes on the regions with large gradient activity during the evolution, by recursively dividing the mesh size into half. For example, if the level one mesh size is 1.0 cm, level two mesh size will be 0.5 cm, and, level three zone size will be 0.25 cm, and so forth. This could be expressed as:

$$(\Delta x)_n = \frac{\Delta x}{2^{n-1}} \tag{2.1}$$

Where Δx is the base mesh resolution (AMR level 1), *n* is the refine level, and $(\Delta x)_n$ is the mesh size resolution at level *n*. Or the other way around, by pre-defining a de-refinement criteria, the simulation might decide that the finer mesh is not necessary any more and hence remove them.

All AMR decisions are subject to the following two additional constraints: 1) each zone can be refined by a factor of two, i.e., one level, during each time step, and 2) adjacent cells can differ by at most one level of refinement.

Fig. 2.2 shows a typical adaptive mesh hierarchy of a shock cylindrical bubble interaction simulation at *t*=0. The density is displayed together with grid boxes, each corresponds to a 8x8 mesh patch (ParaMesh implementation). The left column displays the visualization for each level and the right shows an integrated image for all levels. There are five AMR levels, with 8x16 base mesh and 128x256 final mesh. Note the localized mesh distribution at level five. The black regions indicate the savings of computation.

2.3 Computational steering

The goal of computational steering is to design a common data structure for execution of interaction, visualization and analysis operations with the simulation. It has a lot of advantages, for examples, runtime control operations can now be directly performed on applications objects such as grids, meshes and trees through their visualization. Parallelization on post processing also becomes straightforward since the data structure is already in a distributed manner [Parashar & Browne, 98]. This is especially important in AMR applications.

In general, the steering environment is object oriented and usually takes modular structure, which allows addition of more HPC or CFD modules easily. In this thesis, two computational steering environments are explored: GrACE-PPM and FLASH.

Fig. 2.3 illustrates the necessity of computational steering with a typical AIF simulation: a shock wave interacting with a gas curtain. As we stated in the previous paragraph, we can see the main physics is highly confined in a small fraction of the simulation domain. Note the vast red region where indicating nothing significant is happening, and hence doesn't need as high resolution as the localized (blue) regions. Dynamically adjusted refinement mesh is a good solution to this type of flow environment.

Let's assume we are running this simulation with two processors. At t=0, we prefer the simulation being partitioned vertically, allowing equivalent distribution of the mesh. At t=23.76, however, the gas curtain evolves into a boundary layer confine to the upper wall, hence a horizontal partitioning is preferable. The preferable partitioning is constantly changing in time. Because static partitioning suffers from imbalanced load, dynamic scheme is required, which is an important feature introduced by computational steering.

2.4 GrACE & GrACE-PPM

Grid Adaptive Computation Engine (***GrACE***, former DAGH, distributed adaptive grid hierarchy), is one of earliest computational steering environment. Like most of the popular AMR package, this C++ parallel hierarchical AMR package implements the original Berger-Oliger scheme [Berger & Oliger, 84], with coarse grained SPMD (single process multiple data) data parallelism model.

Designed as high-level programming abstractions and a general-purpose data management infrastructure, GrACE provides the interface to the application-general features, and leaves application-specific features to be user specified. It has the advantages of scalability, locality, and portability, as well as coarse-grained data parallelism and Fortran compliant data storage.

2.4.1 GrACE performance

As any other the advanced schemes, the computational efficiency and high level steering ability are at the cost of more complex data structure. To validate GrACE environment and evaluate its performance, we introduced a simple convection system:

$$\frac{du}{dt} + \frac{du}{dx} + \frac{du}{dy} = 0 \qquad (2.2)$$

resolved by a Cartesian computational domain. The field is initialized with Gaussian distribution:

$$u_0 = \exp(((x^2 + y^2)^{0.5} - \varepsilon - t_0)/\sigma)^2)$$

And the algorithm is MacCormack (predictor-corrector) method. We fix the number of iterations as 130 and boundaries as outflow.

We study mainly the overhead introduced by the SPMD data parallelism model and the AMR. Two codes solving the convection system are studied: a). sequential uniform grid code, and b). Parallel AMR code interfaced with GrACE. The algorithm of the interface driver is:

Define adaptive grid structure

Define grid functions: containing information for the coordinate

Initialize grid functions: data parallel model, call Fortran CFD initialization

Repeat NumTimeSteps

a. *if* (RefineTime) Refine at Level
b. Evolve at Level: call Fortran CFD kernel
c. *if* (Level+1 exists)

Evolve at Level+1

Update Level from Level+1

End Repeat

Visualization pipeline: 1D xgraph

We also compare the SGI cluster running MPICH and the SUN HPC machine.

We first examine the overhead introduced by the data parallelism in GrACE, by comparing it with sequential code. Note both codes are using a uniform mesh here. Fig. 2.4 shows the run statistics. The overheads introduced by parallelelism are decreasing as the problem size growing. For example, on the SGI64 machine, the uniform grid GrACE run gives (1610-380)/380 = 324 % of overhead for a 64x64 run and (12381-9650)/9650=28.3% of overhead for a 320x320 run. In terms of parallel machine performance, Sun E10k has very good scalability, as shown in Fig. 4.6 with speedup

3.67/4=92% of ideal speedup (comparing to 64.5% speedup on SGI64), although it performs poorly in sequential jobs, an anomaly we don't yet fully understand, but we believe it is associated with the system configuration. It is also shown in Fig. 2.5 that smaller problem has poorer speedups as well as poorer scalability.

We further investigate the overhead introduced by AMR engine in GrACE, by again comparing with the uniform mesh version with the same effective mesh, i.e., the 32x32 AMR run with 3 refinement levels corresponds to 128x128 uniform mesh, according to Eqn. 2.1, shown in Fig. 2.6. Note the AMR run on 1 CPU runs much longer than its uniform mesh correspondence, which is due to:

- The small size of the domain. A significant percentage of the time are wasted on MPI and AMR initialization, communication and synchronization (see the synchronization and recompose time in the table).
- The less scalability of the SGI's, as shown in the previous figures.

2.4.2 GrACE PPM implementation

GrACE provides the opportunity of adding HPC modules and computational steering functionality on top of a existing CFD solver, which is VH1 PPM code in our case. Good understandings on both the GrACE environment and the PPM algorithms are required.

The object-oriented feature in GrACE has the potential of user-friendly interface with any hyperbolic PDE system. The general structure of GrACE-PPM is shown in Fig. 2.7, which has four layers. The lowest layer is the distributed data structures, indexed with space filling curves, as shown in Fig. 2.8. The space filling curve is inexpensive computationally and self-similar (recursive), which makes the original multi-dimensional space being easily encoded. The index corresponding to the right-most figure bottom is

shown at the bottom, with the under-line part corresponding to the refined region in the center.

Next level in the GrACE-PPM structure (Fig. 2.7) is high-level programming abstractions. Three classes are implemented, containing the application fields (Grid Function Abstraction), Grid structure specification (Grid Hierarchy Abstraction) and Grid/Fields correlation (Grid Geometry abstraction).

These abstractions allow the application being interfaced with the lower layer data structure without detail knowledge of the data structure. Between the top layer (PPM application) and the abstractions, there are computational steering modules as the interface, which involves:

1. Multi-grid hierarchy and grid function definition and initialization. A main-shadow structure is used, where the shadow is one level finer mesh.

2. Boundary and initial conditions alignments:

- Physical boundary condition, which is integrated with the hydrodynamic evolution in the original PPM algorithm. This is a special requirement by the inflow/outflow boundary condition – certain number of ghost zones outside the real physical boundary are crucial to absorb the perturbations produced by physical quantities flowing out of domain. In GrACE-PPM, the physical boundary should be categorized into interior mesh (with boundary condition provided by the neighbor grids) and the boundary mesh (with boundary condition provided by the true physical boundary).
- Inter-AMR-level boundary, which is implemented by prolongations (from coarser to finer levels) and restrictions (from finer to coarser levels);
- Inter-CPU boundary;

- Alignment of time evolution in GrACE and time evolution for PPM, which requires the time step not too big, AND not too small.

Fig. 2.9 shows these boundary alignment issues in GrACE-PPM implementation.

3. Perform local error estimation. Aside from using density field as a refinement criteria, vortex paradigm suggests that vorticity field is another important criteria. The error is calculated as local second order gradient (e.g., density, vorticity, or both), clustered in a region pre-set by a cluster threshold, and compared with a preset refinement threshold. If it is greater than the threshold, the refinements to a finer level is performed.

4. System dependency, programming language dependency, compiler dependency, initial geometry dependency.

After setting up GrACE environment and the supporting software, most commonly, parameter parser, MPI and HDF input/output, we use the following interface driver algorithm:

```
Initialize Grid Hierarchy
Initialize Grid Function
Initialize MPI
Initialize physical domain over all AMR levels, with proper ghost zones
Recursive_evolve(Level) {
   if (level==0) #Iterations = 1;
   else #Iterations = RefinementFactor;
   Loop over #Iterations {
      if(RegridTime(level)) {
          Evaluate Local Truncation Error(Level);
          Cluster Errors and Regrid current Level;
       }
      CFD Kernel (PPM: sweeping Reimann Solver);
      Update boundary;
      If (Level + 1exists ) call recursive_evolve(Level+1);
```

```
    }
    Increment Timestep on current Level;
    If(Level+1) exists) restrict solution from Level +1 to Level
  }
  IO/Interactive Viz
```

2.4.3 GrACE-PPM: verification

We use a few AIF geometries introduced in Fig. 1.2 to test our GrACE-PPM implementation, comparing with the uniform mesh PPM code VH1.

Fig. 2.10 shows the verification comparison. Note 2D images are good for qualitative comparison but not enough for quantitative comparison. Hence we use projection concepts in our visiometrics environment (which will be discussed in depth in section 4.3) and show comparison of the horizontal slice. All the features during the shock traversing through the gas curtain are captured in the GrACE-PPM implementation, even at late time. Note the figures for the two runs are not to scale.

Fig. 2.11 shows a 2D test run on shock-inclined interface interaction. Here we examine the computational steering issues. Visualization of various domains and boundaries: real physical boundaries; boundaries between refinement levels; and boundaries between different processors. This is for an AMR, distributed, dynamic data set. We see the initial condition of a shock inclined interface interaction, with 8 CPU, 3 AMR levels, and 128x32 base mesh. The mesh re-gridding is according to local density gradient. Note features:

- *AMR*: different color meshes represent different AMR levels (1st level has the same color as the bounding box; 2nd black; 3rd yellow);
- Distributed: different color of bounding boxes represent different processors;

- Dynamic data structure: note the light blue color, balance of data locality and load balance;

Fig. 2.12 and 2.13 shows test runs on shock curtain interaction, in 2D and 3D, respectively. Fig. 2.14 gives a statistics of the GrACE-PPM's parallel performance in 3D.

2.5 FLASH

FLASH is another computational steering environment. It is a modular, adaptive-mesh, parallel simulation code capable of handling general compressible flow problems, mainly in astrophysical environment. FLASH is designed to allow users to configure initial condition and boundary conditions, change CFD kernels, and add new physics modules, all with minimal effort. It uses the PARAMESH library to manage a block-structured adaptive grid. FLASH also uses MPI library to achieve portability and scalability on a variety of different parallel computers. In addition, it has many properties, e.g., MHD, physical viscosity and multi-species diffusion, and is supported by Department of Energy (DoE) ASCI program.

Fig. 2.15 [FLASH, 03] shows the modular FLASH structure. And Fig. 2.16 shows a typical mesh evolution of FLASH juxtaposed with the density field visualization. The simulation is a shock-cylinder interaction configuration. Note in ParaMesh AMR scheme, the mesh box sizes are uniform, an important difference from GrACE AMR scheme and will be addressed later.

We are the first group ported FLASH to SUN HPC system. Fig. 2.17 shows a hardware performance study on three newest architectures, which gives a benchmark of the different architecture and the level of optimization (whether the hardware achieved the expected performance). The machine abbreviations are defined in Tab. 1.1. The specification of the run is:

- FLASH2.1, viscous, shock-cylinder, *M*=2.0, base resolution 8x16, 3 AMR, 2496 time subintervals

For a reasonable comparison, 8 processors are used, which is not overloading any of these systems.

It is obvious in Fig. 2.17a that SF12K is the fastest (about twice the E10K). However, if we consider weighting factors, we found that SF12K is not optimized to the its ideal performance (which is suppose to be 2.8 times faster than E10K).

We summarize itemized timing (% of total) in Fig. 2.17b. Note The main items of the timing statistics are: Initialization, I/O and Evolution. The rest are itemized evolution term. It is a very interesting result. Item 2 (i/o), 6 (guard cell update), 9 (data base updating), 14 (refinement update), and 15 (tree structure update), are all communication (MPI) related, and have significant occupation of the total computing time on Linux AMD clusters, which reflect the bandwidth bottleneck on the privately networked clusters versus HPC platform.

Fig. 2.18 shows performance on SGI machine. The simulation setups are:

- FLASH2.1, 2D sod shock tube problem 6 AMR levels,. 8x8 base level, tmax=0.01, tplot=0.005, time subintervals: 122

And the items are defined the same as in Fig. 2.17b. Note:

1. The three main terms doesn't sum to 100% because the coupling in timing;

2. The IO scheme influence its percentage;

3. From Fig. 2.18b, we can see the percentage of time in evolution reduced in parallel runs, but I/O increased. The balanced result gives the nearly linear scale in Fig. 2.18a.

4. In general, FLASH gives very good scaling in terms of time distribution among tasks and parallel.

5. Great AMR behavior is taken advantage of: on 2 CPU, the run takes 318.269. However, if the domain is doubled, it only takes 323.841s. The rectangular domain is always necessary for AIFs due to the acceleration in one direction, and stretching bubble/interface. AMR is great for this type of problems.

2.6 Summary

Here we summarize the main technical efforts on HPC in this thesis:

1. GrACE-PPM: Interface AMR scheme with an existing 3D Eulerian hydro code [Zhang *et. al.*, 01];

2. Maintain and upgrade the FLASH code on Sun system;

3. Validate the AMR hydro code: GrACE-PPM and FLASH.

4. Performance study of AMR and HPC.

5. Visiometrics of HPC data set.

In Tab. 2.1 we summarize the code used/implemented/updated in this thesis and their main features. VH1 and FLASH are used as the main working codes in this thesis.

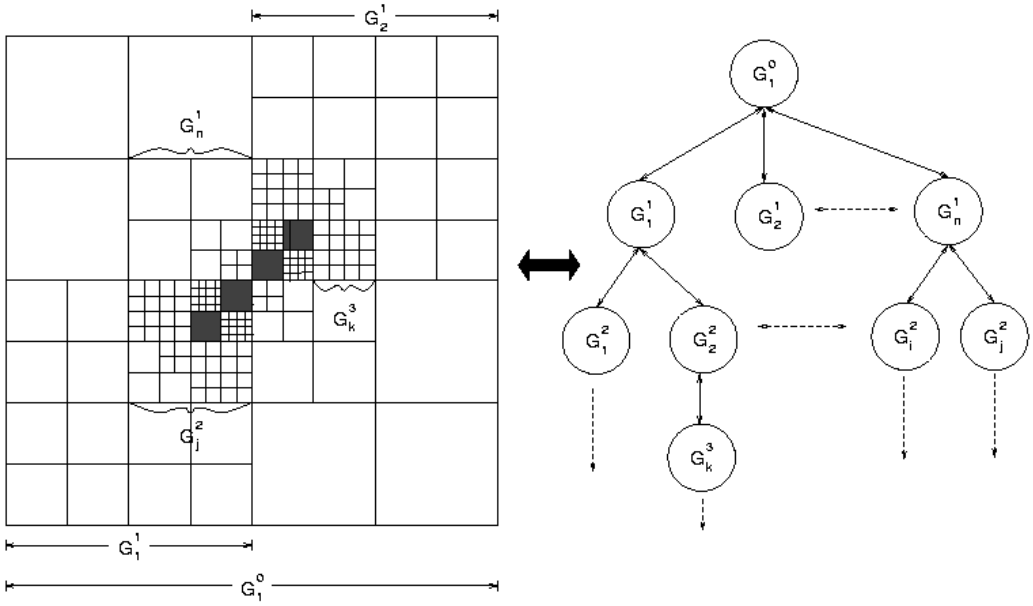


Fig 2.1 AMR Concept: Adaptive Grid Hirarchy (2D)

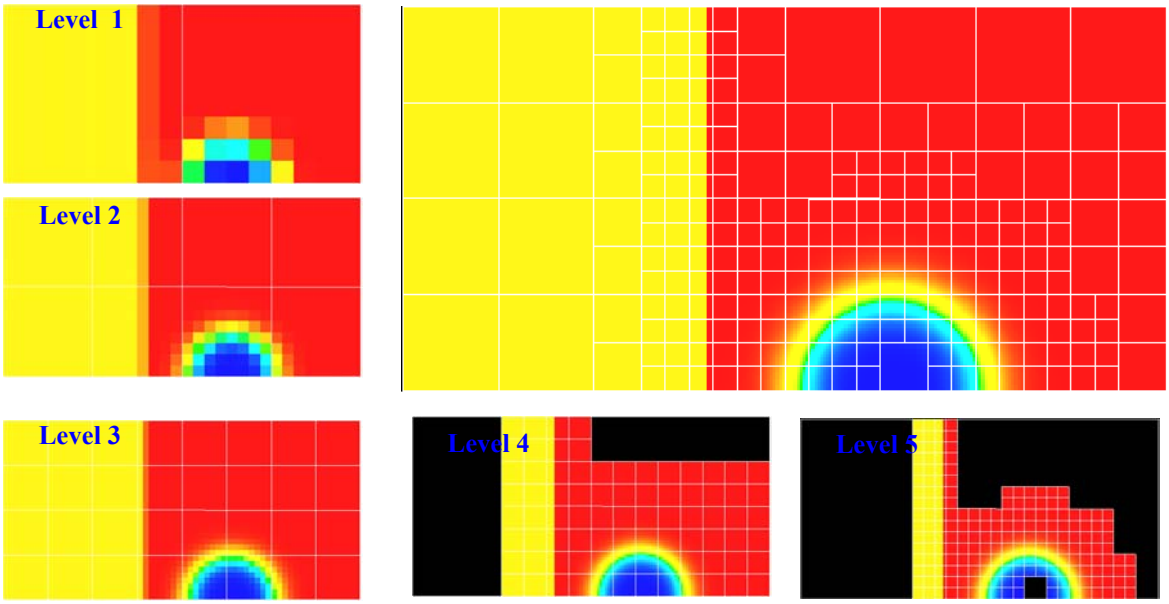


Fig. 2.2 A typical adaptive mesh hierarchy: Shock Gas Bubble Interaction, 5 AMR levels, Level 1: 8x16; Level 5: 128x256

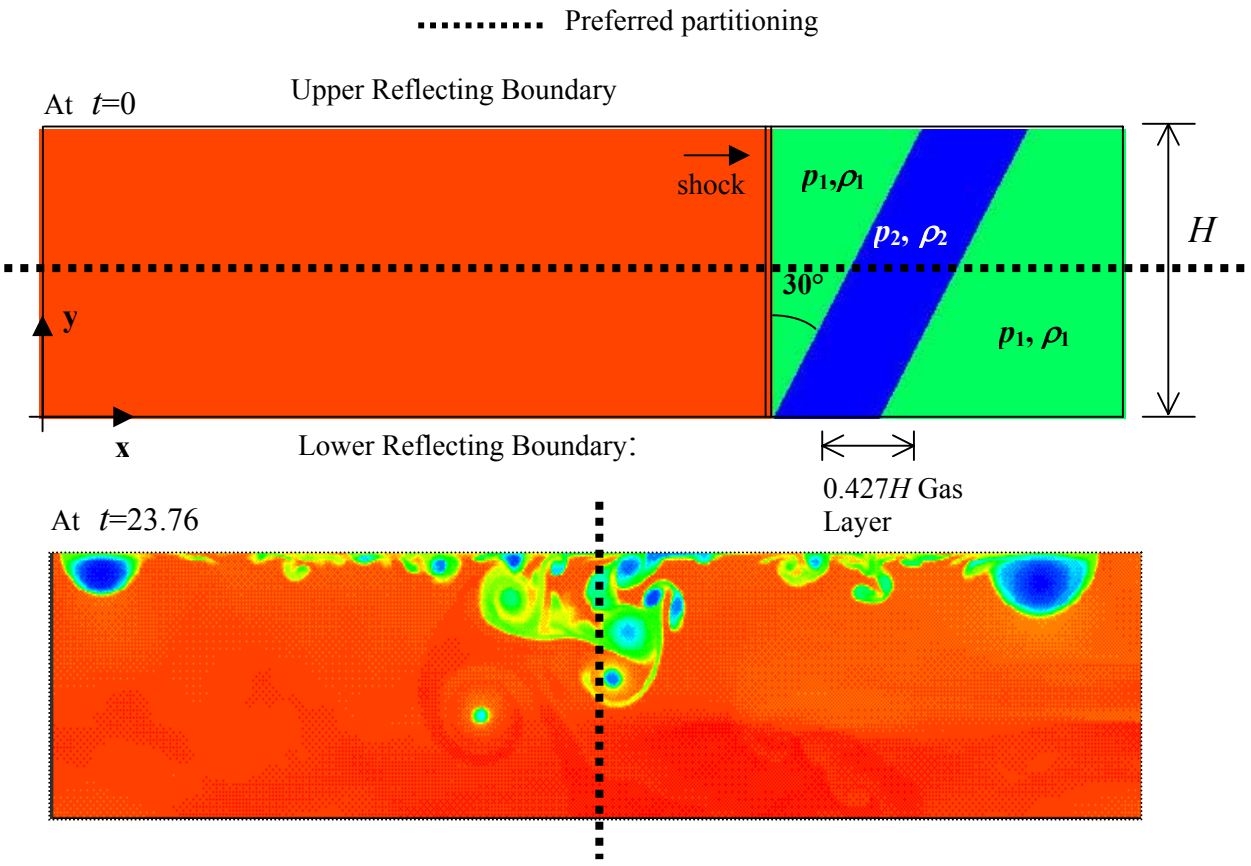
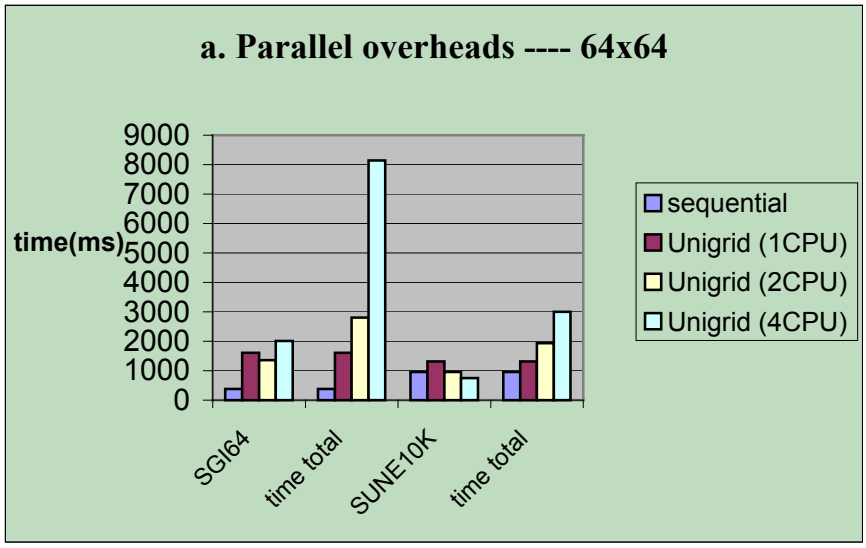


Fig. 2.3 AIFs and AMR: static and dynamic partitioning

Resolution : 64x64

Iteration: 130	SGI64	time total	SUNE10K	time total
sequential	380	380	960	960
Unigrid (1CPU)	1610	1610	1320	1320
Unigrid (2CPU)	1364	2800	959	1931
Unigrid (4CPU)	2007	8140	747	3001



Resolution : 320*320

Iteration: 130	SGI64	time total	SUNE10K	time total
sequential	9650	9650	26300	26300
Unigrid (1CPU)	12381	12381	16143	16143
Unigrid (2CPU)	6754	13552	8194	16420
Unigrid (4CPU)	4860	19582	4404	17644

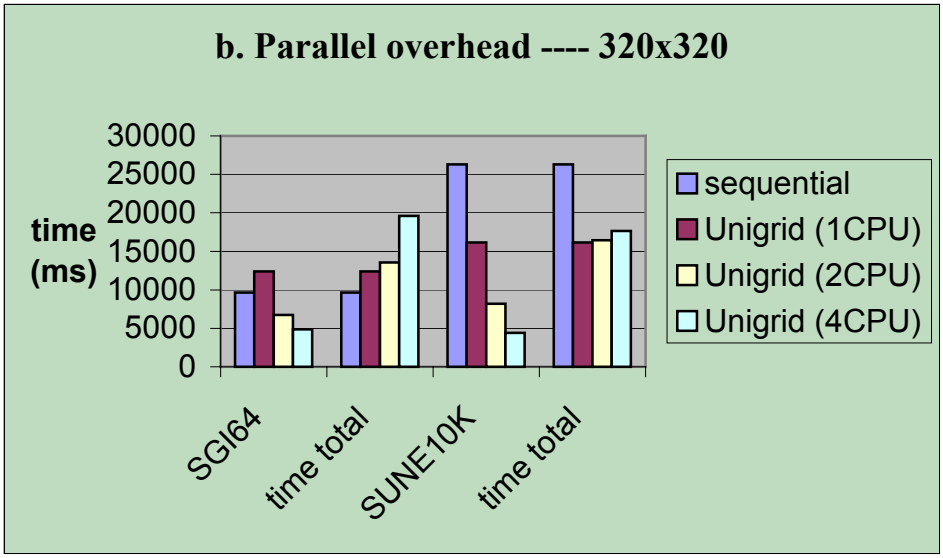


Fig. 2.4 Run statistics examining parallel overhead in GrACE.

speedups	64*64(SGI)	64*64(SUN)	ideal	320*320(SGI)	320*320(SUN)
2CPU	1.18	1.38	2	1.83	1.97
4CPU	0.8	1.78	4	2.58	3.67

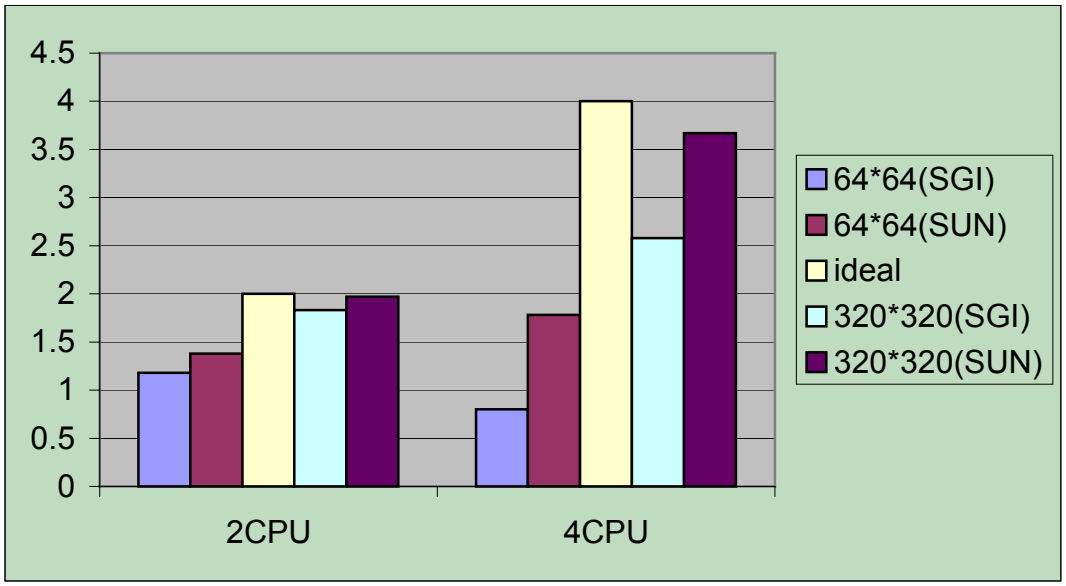


Fig. 2.5 Speedups on uniform grids.

AMR(3 levels): 32*32	p1 time	p2 time	p3 time	p4 time	total time	syn time	recompose time
sequential (32*4 * 32*4)	5.95				5.95		
1CPU	55.974	0	0	0	55.974	24.156	9.719
2CPU	34.767	34.817	0	0	69.584	15.124	9.126
4CPU	33.116	33.147	33.116	33.118	132.497	16.027	9.995

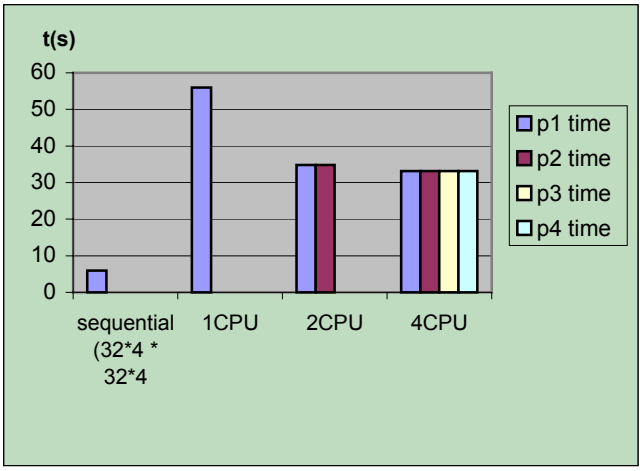


Fig. 2.6 AMR engine performance

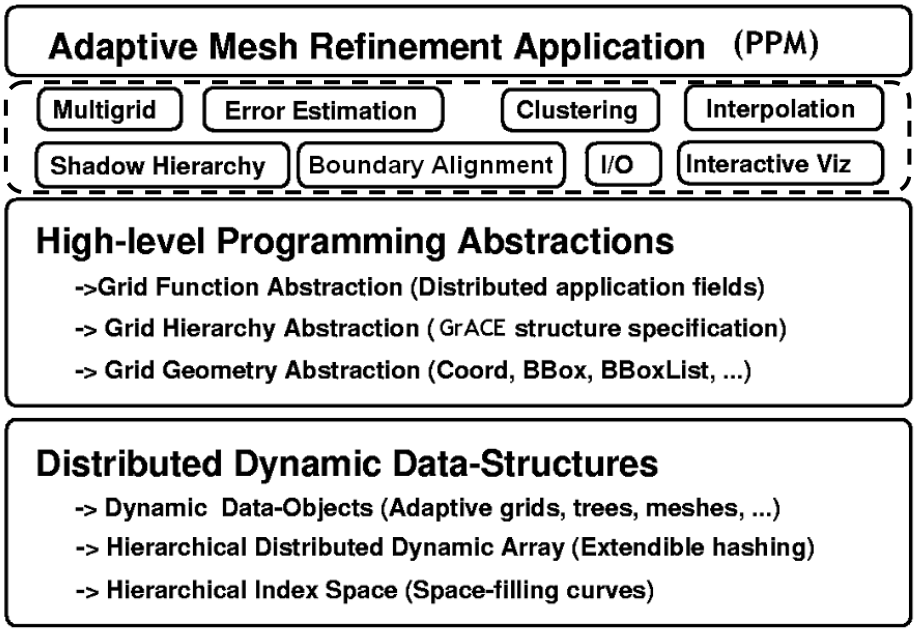


Fig. 2.7 General structure of GrACE-PPM

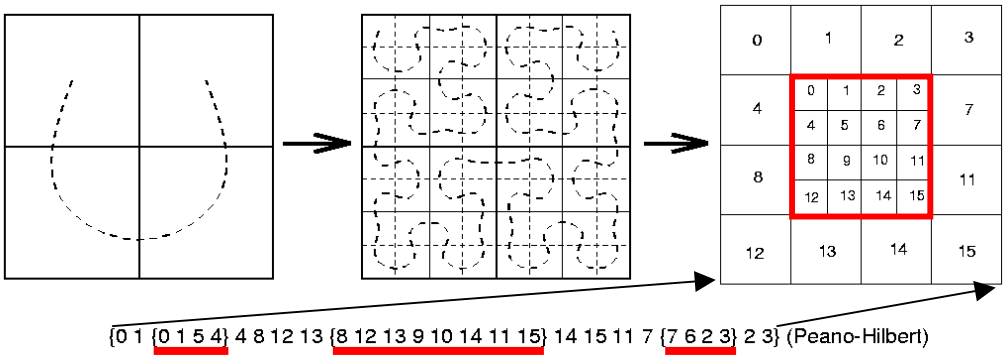


Fig. 2.8 Hierarchical space-filling mappings (Peano-Hilbert)

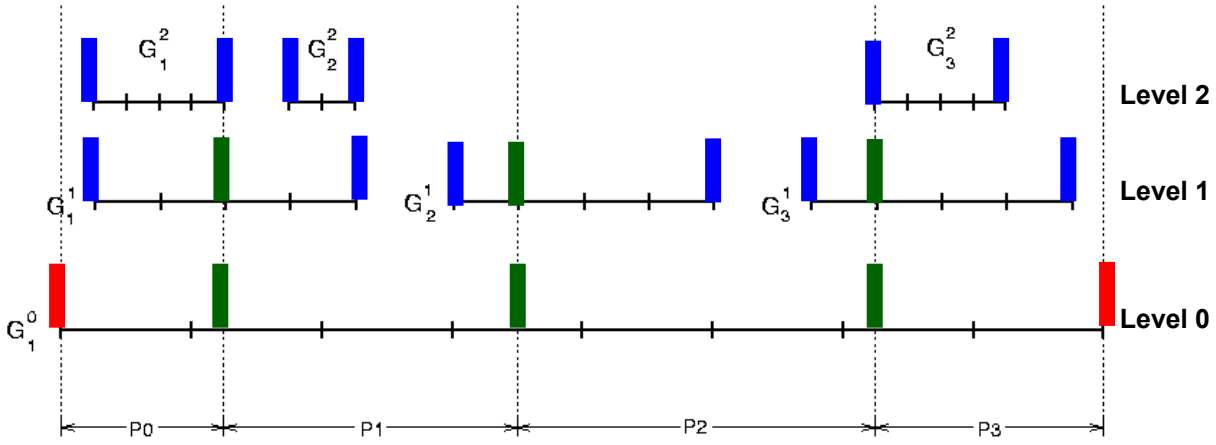
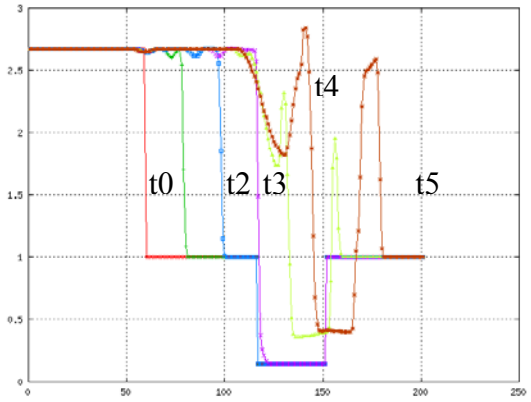
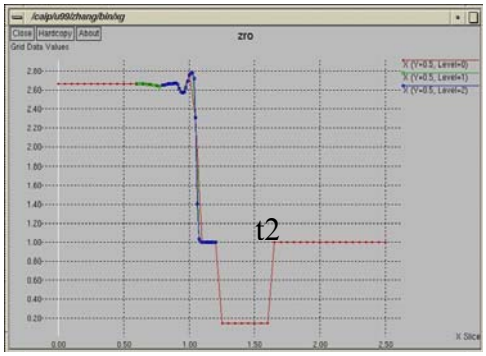


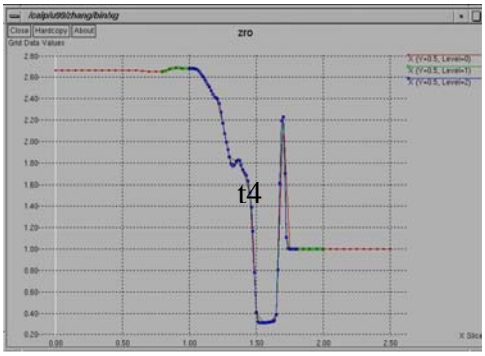
Fig. 2.9 Boundary alignments in GrACE-PPM implementation



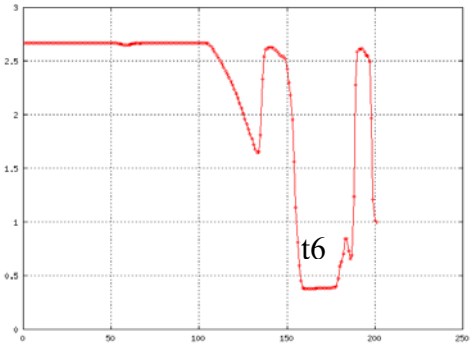
(a). Sequential uniform mesh, resolution 200x80, t0-t5



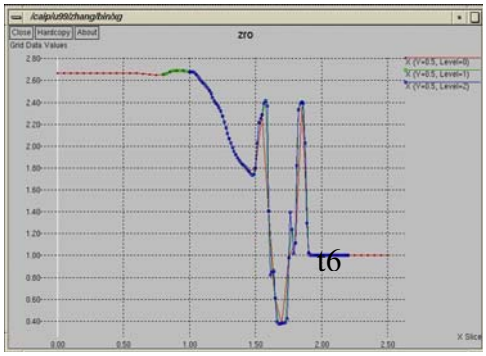
(b). GrACE-PPM, 2 AMR levels, 2 CPU, 50x20



(c). GrACE-PPM, AMR, 2 levels, 2 CPU, 50x20



(d). VH1, Sequential, resolution 200x80



(e). GrACE-PPM, AMR, 2 levels, 2 CPU, 50x20

Fig. 2.10 GrACE-PPM verification – 2D shock curtain interaction, center slice.

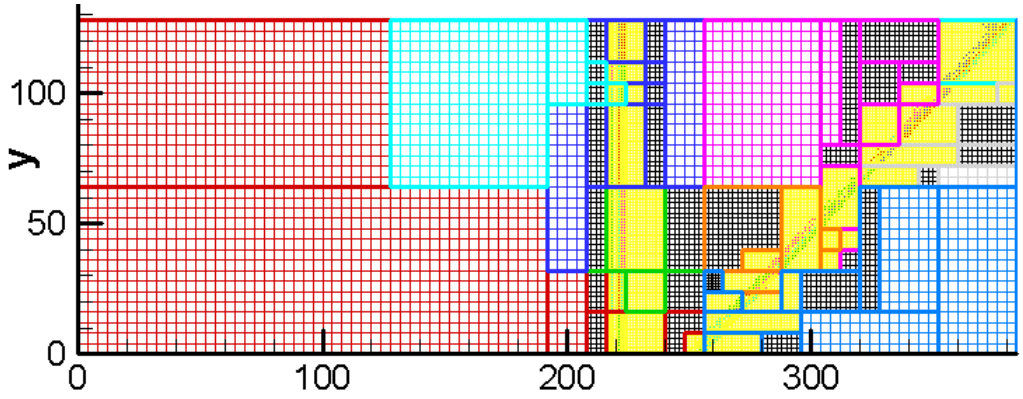
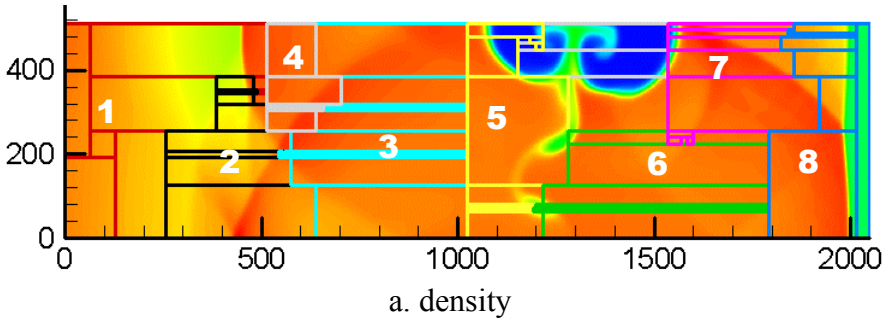
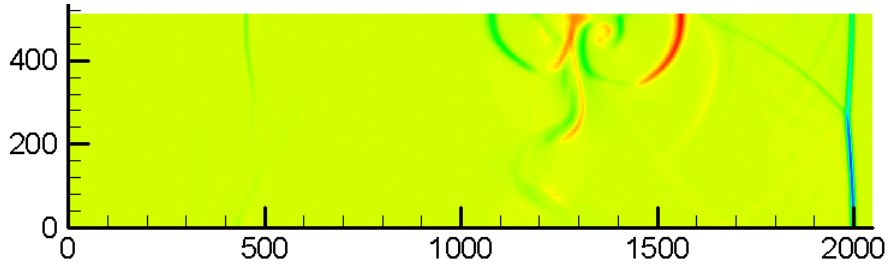


Fig. 2.11 Visualization of various domains and boundaries: real physical; refinement levels; and different processors.



a. density



b. vorticity

Fig. 2.12 GrACE-PPM test: shock curtain interaction at intermediate time, resolution 128x512, with 3 AMR levels, 8 CPUs. Labels in density image are processor numbers.

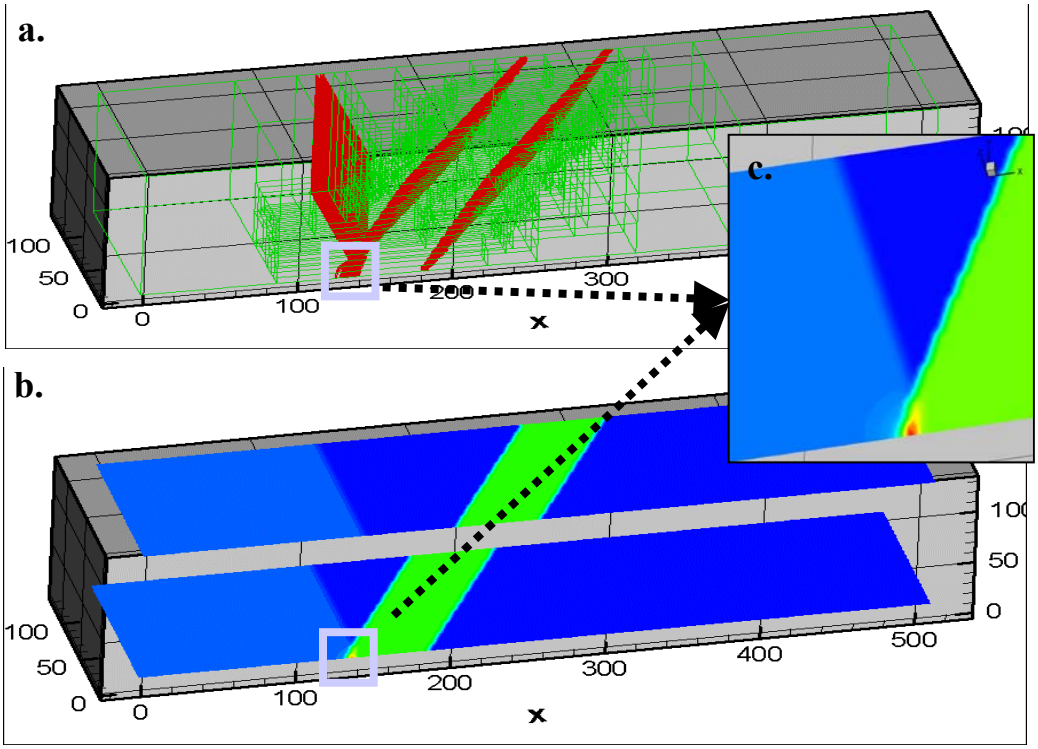


Fig. 2.13 3D GrACE-PPM test: resolution: 128x32x32, 2 AMR Levels, 8 CPUs. (a). density isosurface, with AMR meshes; (b). two slices at j = 10 and 110, respectively; (c). A zoom of slice j=10. Note the transmission and reflection wave front and post shock compression of the curtain mass.

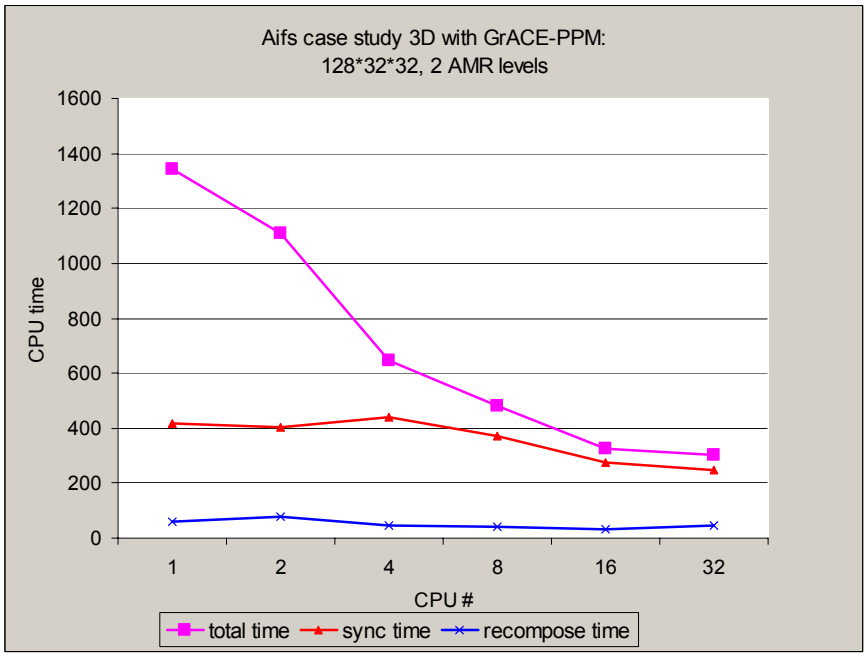


Fig. 2.14 parallel performance of 3D GrACE-PPM simulation

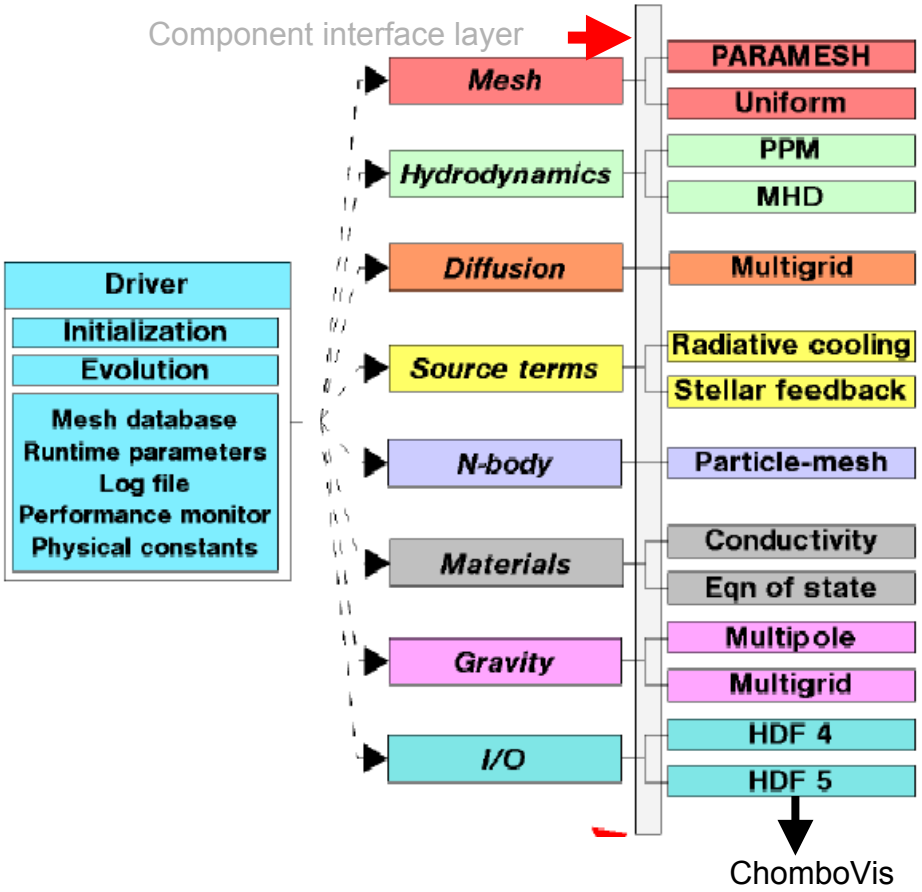


Figure 2.15 Modular hierarchy of FLASH: dynamic degree of complexity.

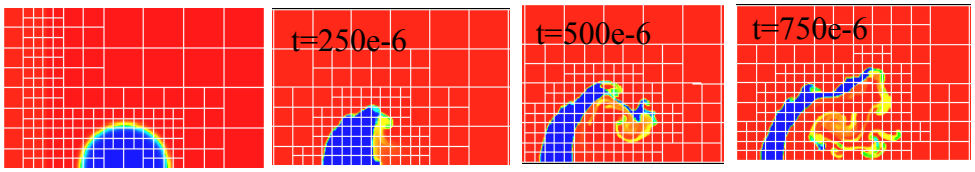
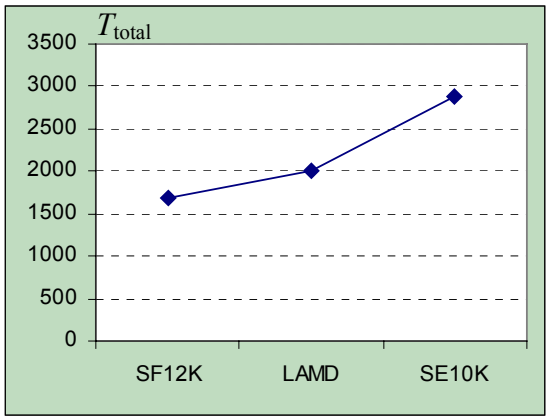
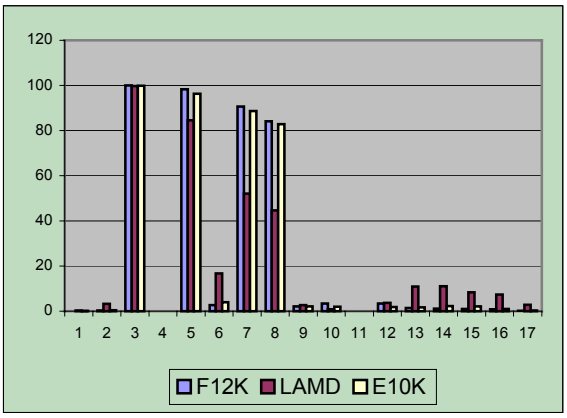


Figure 2.16 A typical mesh evolution in FLASH: 3 AMR levels.



a. 8 CPU Performance on 3 machines



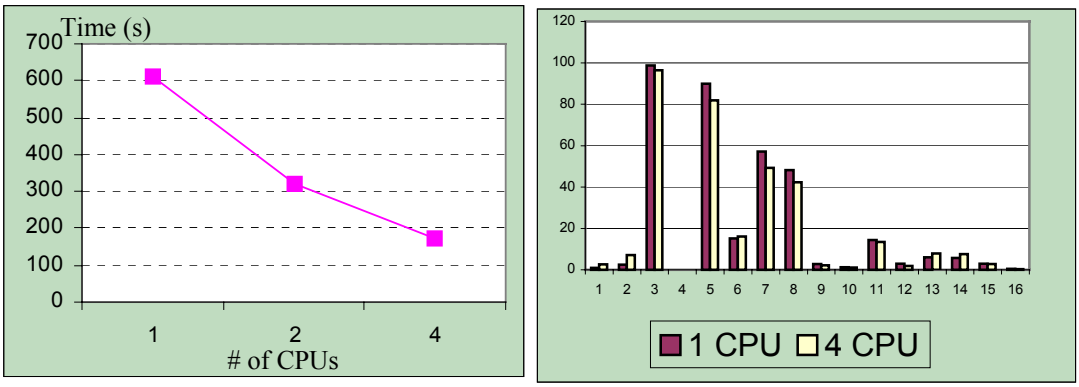
b. Itemized performance

Itemize:

1	initialization	11	diffusion
2	i/o	12	flux conservation
3	evolution	13	eos
4	source terms	14	update refinement
5	hydro	15	tree
6	guard cell	16	guard cell (tree)
7	hydro sweep	17	eos (tree)
8	hydro_1d		
9	dbase		
10	update_soln		

c. Itemized timing

Fig. 2.17 Performance scaling through different hardware: E10K, F12K and AMDMP.



a. Parallel performance

b. Itemized timing (%)

Fig. 2.18 Performance of a typical FLASH run on SGI. a. parallel performance; b. timing.

Codes	CFD Kernel	AMR	Parallel	Accuracy	Function in this thesis
VH1	PPMLR	No	Yes	O(3rd) in S and O(2nd) in T	Main code + Validation testbed
FLASH	PPMDE	Yes	Yes	O(3rd) in S and O(2nd) in T	Main code
GrACE-PPM	PPMLR	Yes	Yes	O(3rd) in S and O(2nd) in T	Validation testbed
GrACE-RM3D	Godnov	Yes	Yes	O(2nd) in S and O(2nd) in T	Validation testbed
WENO	WENO	No	Yes	O(7th) in S and O(3rd) in T	Validation testbed

Tab. 2.1 Summary of codes.

Chapter 3 Visiometrics

3.1 Introduction

Modern computer simulations usually produce large datasets. It is crucial to interpret these datasets through visualization and analysis to obtain physical insight. Visiometrics is our special methodology to accomplish this task.

By visiometrics, we mean the comprehensive data pipeline of visualization, juxtaposition and quantification with modern computers architectures and algorithms. It is a cogent way to explore the dataset for physical and mathematical understanding of phenomena.

The discovery of "Soliton"[Zabusky & Kruskal, 65] via numerical study of Korteweg-de Vries (***KdV***) equation is the first success of visiometrics, pure academically at the time. "Soliton" is now playing a crucial role in optical telecommunication industry. The dispersion managed soliton technology, once converted to industrial productivity, is impacting our life with 10 Gbit/s optical telecommunications systems, with researches on the future evolution towards multi wavelength 40 Gbit/s transmission systems.

The term "Visiometrics" is first made official by the foundation of "Laboratory for Visiometrics and Modeling" (***Vizlab***) at Rutgers and a keynote paper [Bitz & Zabusky, 90]. Vizlab is an interdisciplinary research lab among aerospace engineering, computer engineering, computer science, physics and applied mathematics. Many new physical discoveries were made in the lab under visiometrics mode of working. To name a few in recent years: secondary circulation enhancement [Zabusky & Zhang, 02], baroclinally forced inhomogeneous turbulence [Zhang *et. al.*, 02], and uncertainty quantification [Zhang *et.al.*, 03]. Some of the above examples will be discussed later in this thesis. At

the same time, many new algorithms and software packages were designed and implemented, e.g., Data Visualization and Diagnostics (***DAVID***) and Visiometrics 1.0.

More interestingly, the visiometrics concept, mainly popularized in academia, is catching up in industry by the idea "data mining". It becomes a common belief among spacious researches, both in industry and academia, that a smart way of exploring and presenting the data will be one of the most important research focuses in this information explosive world.

3.2 Scientific visualization in fluid flow

Visualization is one of the most effective tools for flow analysis in both experiments and simulations. It is actually the oldest candidate for visualization ---- the transparent nature of fluid flow and its complexity require smartly designed schematic to reveal the fluid physics without much interference. In terms of computer simulation, visualization could help in revealing scientific insight, as well as in assigning artistic flavors of science in terms of, for example, manipulating colors [Zabusky, 00].

How to visualize the data and which quantities to visualize may determine whether new physics can be discovered, or how much understandings could be obtained from the data. Transformation functions or filters can be applied to flow field variables to generate visual images [Samteney & Zabusky, 00]. Differentiation and threshold are two of the most straightforward and fundamental ones. Take the example of density field. First order differentiation of density filters out the smooth density background and is one of the common methods to visualize the high-gradient density region like contact discontinuity. This method has analogue from both experiment (called schlieren) and image processing (called edge detector). The density gradient field can be further used as the input to more advanced visualization techniques such as skeleton extraction, e.g.,

we used the method in [Chang, 02] to extract skeleton of density gradient magnitude field for our Richtmyer-Meshkov interface simulation [Zabusky & Zhang, 02].

The second order differentiation or Laplacian of density is another technique to visualize the high-gradient regions. We call it numerical shadowgraph in analogy with experimental shadowgraph technique. It is a high-pass filter in imaging processing. Density gradient and density Laplacian visualizations for the interface are smeared over a pretty large space, and consequently are hard for use in further quantitative analysis. One solution to extract the desired curve is the zero crossing of the Laplacian density field [Peng, 03].

For our shock pattern analysis, we use the divergence of velocity field. This will capture only the highly compressed region. Instead, the Laplacian of density can be used to visualize both the shock wave and interface regions at the same time.

[Post *et. al.*, 03] reviewed scientific visualization advances in the past decades, with emphasis on feature-based visualization, a subject in our later discussions. These pre-processing schemes directly associated with visualization are the first step of our visiometrics methodology, and its importance is widely accepted.

3.3 Importance of quantification

Quantification is a step further to the physical understanding and modeling. It is crucial to get physics and mathematics out of the system. There are two categories of quantifications: global and local.

Global quantification measures the total quantities, for examples, circulation, integrated enstrophy, distribution or histogram, power density spectra, moments of statistical quantities (e.g. skewness, flatness et al) and so on. To avoid numerical noise that always associated with the numerical implementation of the boundary condition,

particularly inflow and outflow, we introduce the diagnostic box surround the region of interest to perform global quantification accurately.

Local quantification characterizes the space-time topologies of coherent structures in an appropriate moving frame, including: evolution of critical scales; space-time diagram of integrated vorticity and density; slopes, normals and curvatures of interfaces; and other localized quantities through feature extraction and tracking.

For AIFs and other multi-fluid flows, it is important to extract the localized features for local quantification. These localized features could be coherent vortex structures or the interface curve. Extraction is usually a stand-along post-processing process, but could also be online with the simulations.

We'll discuss the definition of these quantifications at the place we use them, and summarize the details of the procedure on how to get these quantifications in the next section.

3.4 Visiometrics: the comprehensive pipeline

The visiometrics pipeline mainly consists of four stages:

OBTAIN & ASSIMILATE DATA: From numerical simulations, laboratory/filed experiments, and/or observations (e.g., space telescope), with appropriate preprocessing (reduction, filter, statistical modeling and juxtaposition).

VISUALIZE DATA: Use interactive environments and animation, with proper choice of visual/graphics effects, e.g., colormaps.

QUANTIFY DATA:

- Project fields (time evolving scalar, vector and tensor in nD) to lower dimensions (3D, 2D or 1D) by integrating with respect to an appropriate kernel;
- Transform field by taking slices, space-time diagrams and volumes, or with wavelets, etc;
- Identify, extract, juxtapose and track coherent objects and particles (features and trends);

MATHEMATIZE: Obtain reduced models or statistical properties; Find scaling laws, simpler equations of motion and/or simple formulas.

In the following we define a few key technologies underlying these modules. Note our definitions of these technologies are really dynamic: the technique is developing and expected to be enriched.

3.4.1 Data Juxtaposition

Data juxtaposition ---- *Combine and compare data to build correlation, from*

1. Different source (experiments, simulations, field observations);
2. Different quantities (independent or derived) and/or at different time;
3. Different preprocessing procedure; etc…

In complex physical processes, different factors usually interact, correlate and compete with each other. We can obtain different visualization and quantification results from different perspectives. Juxtaposition is a synthetic comparison for insightful analysis by putting side by side different but potentially correlated results (visualized and quantified) of different or similar functions. This is especially true in juxtaposing primary quantity

(e.g., density) with carefully structured quantities (e.g., vorticity). We can also validate the simulation by juxtaposing the experimental results with simulations.

Juxtaposition allows one looking at different aspect of a complex system and derives innovative insights such as correlations and causes.

3.4.2 Data projection

Data projection ---- *Project data with regard to appropriate kernel and reduce the dimension*. The most popular projection methods are:

1. Slice the data (0^{th} order approach);
2. Integrated with regard to a kernel to project the data to lower dimensions (higher order approach);
3. Statistical distributions and correlations; etc…

Project one quantity to the axis reduces the dimension (2D->1D) and makes it possible to juxtapose values of a 2D quantity from different time to capture its time evolution. Fig. 3.1 illustrates an example of a 1^{st} order projection.

Note projection reduces the dimension of the data and make it feasible to juxtapose at this more abstracted level.

3.4.3 Space-time analysis

Space-time analysis ---- *Assimilate the evolution of the data in one parameter space (e.g. temporal) with the other (e.g., spatial).*

- In 1D, it gives a 2D space-time diagram with no data reduction;
- In 2D, it gives a 3D space-time volume with no data reduction or 2D S-T diagram with 1 projection;

- In *n*D, it gives a 3D space-time volume with *n*-2 projection/slice or 2D S-T diagram with *n*-1 projections.

The discovery of soliton is a good example to illustrate the main concept of space-time diagram, as in Fig. 3.2. More complex space-time analysis used later-on in the thesis leads to many new discoveries.

The basic idea in space-time analysis is dimensional transformation, in our case, transformation of spatial dimensions (upon projection) to temporal dimension.

KdV equation is considered the champion of model equations of nonlinear waves. In Fig. 3.2, we can see the PDE system and numerical configurations are on the top. The right column is a sequence of graphs showing the solutions at different times, a set of curves that Korteweg and de Vries were looking at in the 19th century, and hereafter by all the researchers for nearly seven decades.

To the left, a space-time diagram of the same solution is shown, by packaging the curves from all time steps along time axis, and using a appropriate colormap to represent the value of the function (velocity in this case). The eye-capturing large streaks in the diagram are solitons, with some of their mathematical features such as strength, phase shift after intersection, etc, being geometrically visualized.

Through data summarization with space-time diagram, the spatial and temporal evolution is captured together. Important features and their characteristics are easily identified.

Note the extension of the orthogonal Cartesian space to parameter space will be of crucial importance in the extension of using visiometrics to non-scientific data, e.g., business data.

3.4.4 Feature analysis

Features, or patterns, are of crucial importance in any scientific disciplines. This is especially true in AIFs flows, where the gas inhomogeneity is always rolled up by a strong vortex projectile and forms a mushroom dipolar jet as shown in Figure 1.2. The features of interests in fluid dynamics are:

- Vortices
- Shock waves
- Material interfaces (contact discontinuities)
- Separation and attachment lines
- Recirculation zones
- Boundary layers

Etc…

They can be primarily categorized into two groups: volumetric features (e.g., vortices) and interface/line features (e.g., shocks, CDs). Note transformation of the feature from one category to another is always possible, e.g., 3D vortex tube to vortex core line.

Possible events of these time evolving features are summarized in Fig. 3.3. [Zabusky, 99] has a more detail summary of interactions of fluid dynamical features. Clearly, it is very important to ask what happened in the evolution of the features qualitatively (event query) and quantitatively. This could be only answered by tracking algorithms, i.e., correspondence problem. Two main approaches to solve the correspondence problem are: 1. attribute correspondence [Samteney *et. al.*, 92]; 2. full volume correspondence [Silver & Wang, 97]. The former is efficient because it uses the feature attributes as reduced model of the feature and calculates the correspondence of these attributes.

However, it has difficulty in handling features with high complexity, e.g., highly curved features. We use full volume feature correspondence in this thesis.

Fig. 3.4 shows a feature analysis pipeline. The basic steps are: **1**. Obtain data from experiments, simulations or observations, with appropriate preprocessing (e.g., transformation and juxtaposition); 2. Visualize (in traditional way) the simulation data, quantify fields globally and identify main feature of interests (vortices and mass bubbles in this case); **3**. Extract these features and quantify them to get abstract description; **4**. Track the time evolution of these features; **5**. Isolate individual features (interactively) to obtain evolution quantification.

Details on feature extraction and tracking algorithms could be found in [Silver & Wang, 97]. And Fig. 3.5 shows a diagram of using the feature-tracking package.

[Chen *et. al.*, 02] [Chen *et. al.*, 03] extended this package to parallel environment and addressed the correspondance problem of adaptive mesh data.

Aside from extraction and tracking of the volumetric features, the material interfaces between different fluids, referred to as CD, and shock waves, are also of practical interests for extraction and tracking.

[Samtaney & Zabusky, 99] examined the accuracy of zero crossing of the Laplacian density in quantifying the shock and contact discontinuity locations. [Peng, 03] extended this method to initially diffuse interface and developed a simplified interface extraction algorithm using the zero crossing of Laplacian of density in high-density gradient field. Other interfacial-tracking scheme includes media-axis work by [Chang, 02] and those in level set method literature.

3.5 Unique features of visiometrics

Visiometrics has the following features different from the usual visualization:

1.Data abstraction

• Visiometrics increases the information content by abstract representation of the data (with features) and projections/transformations/juxtapositions;

• Visiometrics could reduce the data by the order of 10^3, with high scalability in parallel environment; this data reduction allows real-time interaction between the viz and CFD simulation;

• Visiometrics reduces visual clustering for easier viz;

2.Visiometrics leads to effective physics representation by allowing physicists to relate visual observation to conceptual framework;

3.Feature analysis allows access to the localized features and makes the quantification and reduced mathematical modeling feasible;

3.6 Environments: DAVID & Visiometrics 1.0

The first implementation of visiometrics toolbox is by [Bitz & Zabusky, 90] and [Feher & Zabusky, 95] created and augmented the *DAVID* environment [DAVID, 96] for facile interactive visualization and quantification of simulation data-- single frames of a variable in 2D or a slice from 3D. They introduced the idea of a “diagnostic box” and were able to extract and project coherent structures on which quantifications were done.

A new visiometrics toolbox developed in this thesis dealing with the HPC dataset is visiometrics1.0. It has the following emphasis:

1. It has enhanced features in visualizing, quantifying, scaling and reduced modeling the properties of evolving coherent structures of density, density gradients, velocity, vorticity (or domain circulations and enstrophy), velocity-divergence and baroclinic generation, particularly emphasizing on obtaining

insight to late time turbulent decay and mixing with feature based analysis: feature (volumetric and interfacial) extraction and tracking.

2. It has the ability of handling large, time-varying, parallel, adaptive mesh datasets.

3. It is extended to 3D visiometrics under vortex paradigm framework, to address 3D effect and obtain quantitative insight of 3D physics in AIF.

3.7 Visiometrics for AMR data set – ChomboVis environment.

We discussed in section 4.2 that AMR is a valuable scheme for large multi-scale simulations. However, the significant increase of data structure complexity, across many processors with dynamic features (for load balancing), make it very difficult for visiometrics. For example, data "alignment" is extremely important for visualization and made difficult by various boundaries: real physical boundaries; boundaries of refinement level; and boundary between different processors. We use and augment AMR modules from ChomboVis, a Lawance Berkeley National Laboratory environment [ChomboVis, 03], to visualize AMR data from FLASH and GrACE-PPM.

Figure 4.25 use FLASH data structure as an example to show how FLASH data, with AMR structures, has been processed and visualized. The first approach, Fig. 3.6a, is to output our simulation data with regular HDF5 format, and write a routine to convert the regular HDF5 data to ChomboVis HDF5 format. The second approach is to output ChomboVis HDF5 format directly, as shown in Fig. 3.6b.

Both approaches involves ChomboVis HDF5 data structure, which we'll discuss in the following.

3.7.1 ChomboVis HDF5 data structure

ChomboVis is a program for visualization of 2D and 3D AMR data sets. It is layered on top of the Visualization Toolkit (VTK) and provides a graphical and programmable user interface for interacting with the data set. The primary function is imlemented with C++, but the interface driver is written in Python.

Before we discuss the data structure, we first summarize the datastructure. More details could be found at [ChomboVis, 03].

ChomboVis uses the *native* data type internally, which allows HDF5 to make efficient conversions between binary data representations. Every NATIVE data type translates to a real data type for particular computer architecture.

ChomboVis also defines two new composite HDF5 data types for IntVect and Box (which depend on the dimensionality of the Chombo HDF5 file). An typical 3D description is:

```
DATATYPE ``intvect_id" {
  H5T_NATIVE_INT "intvecti";
  H5T_NATIVE_INT "intvectj";
  H5T_NATIVE_INT "intvectk";
       }
DATATYPE ``box_id"{
  H5T_NATIVE_INT "lo_i";
  H5T_NATIVE_INT "lo_j";
  H5T_NATIVE_INT "lo_k";
  H5T_NATIVE_INT "hi_i";
  H5T_NATIVE_INT "hi_j";
  H5T_NATIVE_INT "hi_k";
       }
```

With the above definitions, the file format could be break-down into the following:

```
GROUP "/" {
   ATTRIBUTE "time" [OPTIONAL]{
      DATATYPE { H5T_NATIVE_DOUBLE }
```

```
  DATASPACE { SCALAR }
}
ATTRIBUTE "iteration" [OPTIONAL] {
  DATATYPE { H5T_NATIVE_INT }
  DATASPACE { SCALAR }
}
ATTRIBUTE "max_level" {
  DATATYPE { H5T_NATIVE_INT }
  DATASPACE { SCALAR }
}
ATTRIBUTE "num_levels" {
  DATATYPE { H5T_NATIVE_INT }
  DATASPACE { SCALAR }
}
ATTRIBUTE "num_components" {
  DATATYPE { H5T_NATIVE_INT }
  DATASPACE { SCALAR }
}
[for n=0,num_components
ATTRIBUTE "component_n" {
  DATATYPE {
    { STRSIZE ;
      STRPAD H5T_STR_NULLTERM;
      CSET H5T_CSET_ASCII;
      CTYPE H5T_C_S1;
    }
  }
  DATASPACE { SCALAR }
}]
GROUP "Chombo_global" {
  ATTRIBUTE "testReal" {
    DATATYPE {H5T_NATIVE_DOUBLE }
    DATASPACE { SCALAR }
  }
  ATTRIBUTE "SpaceDim" {
    DATATYPE { H5T_NATIVE_INT }
    DATASPACE { SCALAR }
  }
}
[for n=0,num_levels
GROUP "level_n" {
  ATTRIBUTE "dt" [OPTIONAL] {
    DATATYPE {H5T_NATIVE_DOUBLE }
```

```
    DATASPACE { SCALAR }
  }
  ATTRIBUTE "dx" {
    DATATYPE {H5T_NATIVE_DOUBLE }
    DATASPACE { SCALAR }
  }
  ATTRIBUTE "ref_ratio" {
    DATATYPE { H5T_NATIVE_INT }
    DATASPACE { SCALAR }
  }
  ATTRIBUTE "prob_domain" {
    DATATYPE box_id
    DATASPACE { SCALAR }
  }
  DATASET "boxes" {
    DATATYPE box_id
    DATASPACE { SIMPLE 1D }
  } # see section 'Data Flattening'
  DATASET "data:datatype=0" {
    DATATYPE {H5T_NATIVE_DOUBLE }
    DATASPACE { SIMPLE 1D }
  } # see section 'Data Flattening'
  GROUP "data_attributes" {
    ATTRIBUTE "ghost" {
      DATATYPE intvect_id
      DATASPACE { SCALAR }
    }
    ATTRIBUTE "comps" {
      DATATYPE { H5T_NATIVE_INT }
      DATASPACE { SCALAR }
    }
    ATTRIBUTE "objectType" {
      DATATYPE {
        { STRSIZE ;
          STRPAD H5T_STR_NULLTERM;
          CSET H5T_CSET_ASCII;
          CTYPE H5T_C_S1;
        }
      }
      DATASPACE { SCALAR }
    }
  }
}]
```

}

The floating point data is stored in the single large data array in /level_*/data:datatype=0. The primary data holder used for each grid in an AMR calculation is defined as:

```
struct FArrayBox
{
  Real* dataPtr;
  int   comps;
  Box   region;
}
```

which is accessed in **Fortran-ordering** fashion with component indexing being the outer index, and zero indexing:

```
for(int comp=0; comp < fab.comps; ++comp, index+=fab.region.numPts())
  for(int k=0 ; k < fab.region.size(2); ++k,
                    index+=fab.region.size(0)*fab.region.size(1))
    for(int j=0 ; j < fab.region.size(1); ++j, index+=fab.region.size(0))
      for(int i=0 ; i < fab.region.size(0); ++i, ++index)
        printf(``fab[%i,%i,%i,%i] = %d",
              i,j,k,comp,fab.dataPtr[index]);
```

For every level of AMR data, the number of components is stored once in /level_n/data_attribtues/comps. All of the Real data per level is then stored in one giant 1D array called /level_n/data:datatype=0. Each FArrayBox data is written as a

contiguous section (an HDF5 *hyperslab*) of this large 1D array Along the 1D data array, the order which things vary is i, then j, k, component, and finally FArrayBox index.

3.7.2 ChomboVis requirements

ChomboVis are primarily tested on Linux. In this thesis, we have succeeded in building and running ChomboVis on Sun Sparc architecture.

Before building and running ChomboVis, a number of supporting packages need to be installed first, include:

- HDF5 1.4.2 - used by FLASH and GrACE-PPM to write data and by ChomboVis to read that data.
- Tcl 8.3.3 and Tk 8.3.3
- Python 2.2
- Pmw.0.8.5 - Python megawidgets.
- Mesa 3.4.2 (if you don't have OPENGL).
- VTK 3.2

Note the version number is extremely important. A number of packages that are almost universally installed are also necessary, e.g., GNU make (v3.77 or higher), GNU zip, autoconf, and text tools like grep, sed, and awk. In addition, GNU C++ (gcc), version 2.95 or later, except version 2.96 is required.

3.7.3 Interface function and parameters

Although ChomboVis was designed to use together with Chombo package (A variety of ParaMesh), the only point of contact between the two is that ChomboVis reads the HDF5 file format that Chombo writes. Any package could write out ChomboVis compatible data as far as it follows the data structure. The most important step is

knowing where in the large 1D data array to start a read or a write for a particular FArrayBox. Offset calculation and hyperslab construction is a large part of the code complexity in the ChomboVis HDF5 (combined with the complexity of the flexible HDF5 binary portability). Upon completely reading in all the region information from the "boxes" data set, and reading the number of components, then one can calculate where to start any particular read/write operation. This is also an essential element of how to perform I/O in parallel, with every processor individually figuring out where in the giant array to write its particular data sections.

As pointed out in the previous sections, an important difference between ParaMesh based AMR package (e.g., FLASH and Chombo) and GrACE based package (e.g., GrACE-PPM) is that the former has uniform size for all blocks. This makes it significantly easier for the offset calculation in visualizing ParaMesh based data than GrACE based data.

For non-uniform grid size, we need to retrieve the box information as part of the Grid Hierarchy as well as part of Grid Function. Then at the time we are allocating memory for the 1D array, we also had the information of the box size and assign the memory accordingly.

In the following, we show a typical interface function:

void write_block_to_chombo_file (int* block_tot, // accumulating: total number of blocks been processed, for file close control

int* no_at_level, // accumulating: number of blocks at current level been processed, for computing offset of HDF5 dataspace

char* fileout2, // filename

int* tot_blocks, // total number of blocks, for file close control together with block_tot

int* numLevels, // total number of levels, for initialize and close level-wise HDF5 memdataspace

int* ndim, // number of dimensions

int* nvar, // number of variables

int* nxb, // size of the box

int* nyb,

int* nzb,

int* lrefine, // current level

int* min_coordinate[ndim], //index coordinates of current box

int* max_coordinate[ndim],

double* unk[ndim[0],ndim[1],ndim[2],nvar], // Data: note the order (i, j, k, nvar)

double* simtime, // simulation time

double* dt, // time step

int* first_call, // boolean controling the file to be open only once for each time step

int* no_at_level_tot[numLevels], // An array contents total number of blocks at each level, for memdataspace allocation

double* dx[numLevels], // An array contents step size at each level

char* compNames // An array contents name of components**);**

Two block indexes is required to build the 1D array, as shown in Fig. 3.7:

1). The total number of blocks for book-keeping: ***block_tot*** (accumulating) and ***tot_blocks*** (=6) in this case.

2). The number of blocks at each level for dataspace memory allocation ***no_at_level_tot.***

The interface algorithm outputting FLASH data (different levels, multiple grid function) at each time step to one file in ChomboVis HDF5 format is summarized as following:

1. Open file if called the first time (controlled by ***first_call***);

2. Define and insert compound data strcture, including box, attibute (***ndim, numLevels, nvar, simtime***);

3. Open and write out CHOMBO HDF5 level-wise: a data space is created for each level, and allocated with the size: ***no_at_level_tot[i]* *nvar * *nxb * *nyb***;

(Step 1, 2, 3 is within *chomboinitializedfile*)

4. Compute level-wise offset of the data in the data space;

5. plug the data into the appropriate data space according to the computed offset;

6. Close the file if all the block numbers have been processed

3.7.4 Examples

Fig. 3.8 shows a typical ChomboVis visualization interface. Most of AMR visualization in this thesis is based on ChomboVis, however, the GUIs are omitted for all other cases.

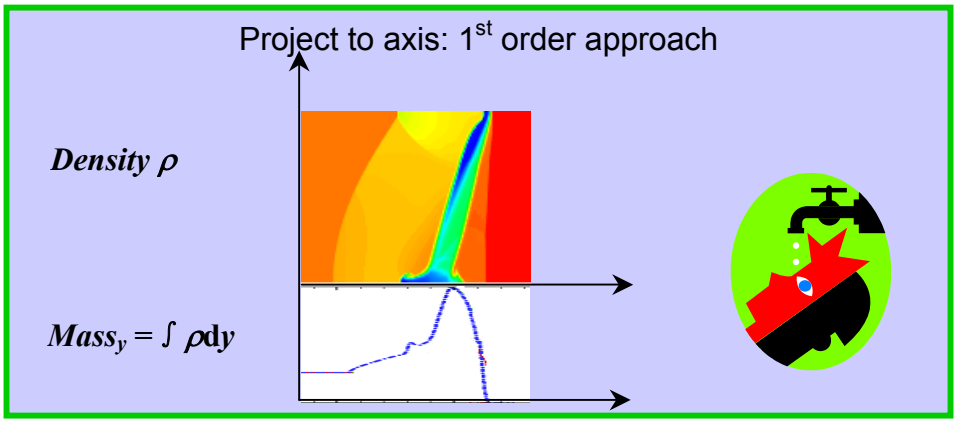


Fig. 3.1 An Illustration of data projection

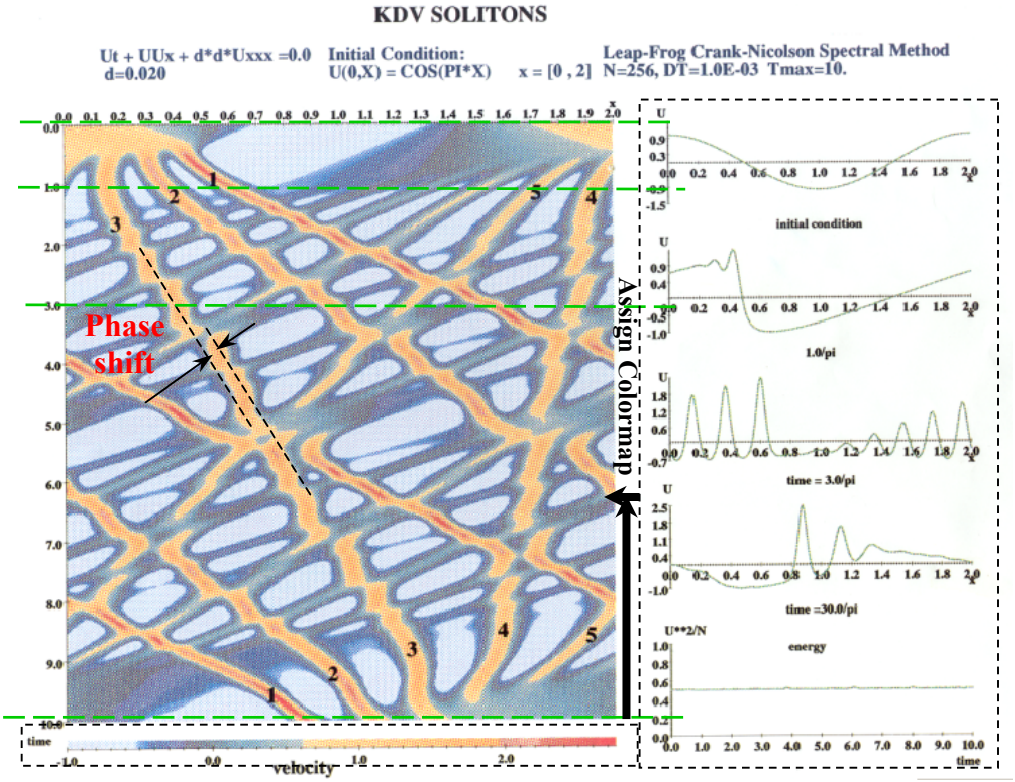


Fig. 3.2 KdV soliton: an example of space time diagram.

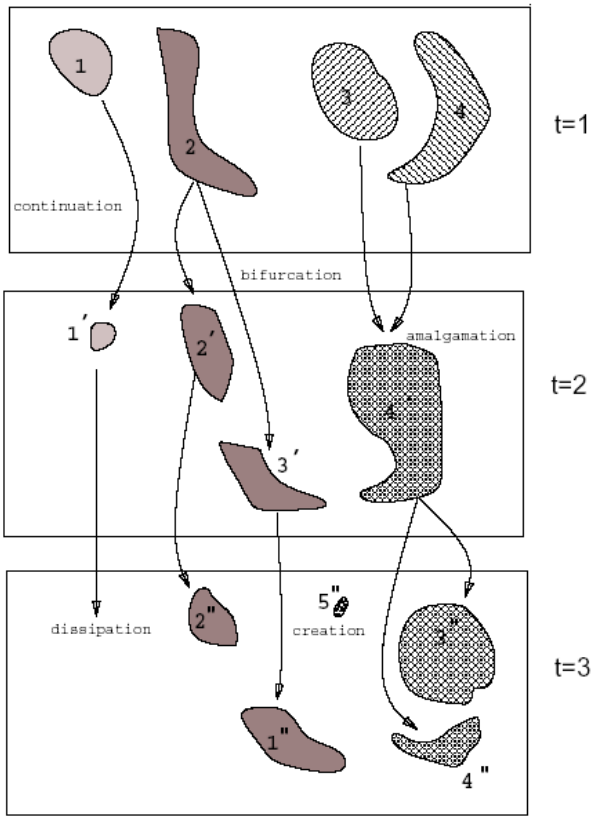


Fig. 3.3 Events in 2D turbulent mixing.

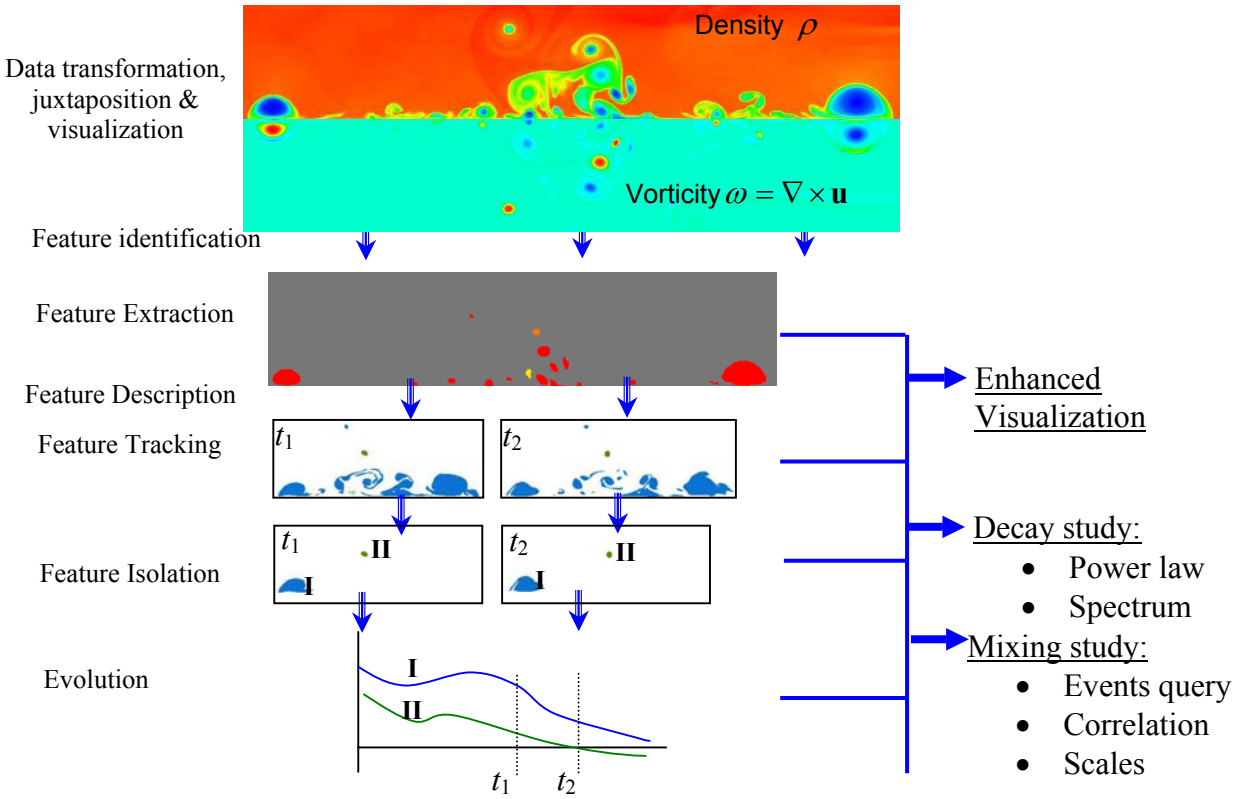


Fig. 3.4 Feature based analysis in AIF study

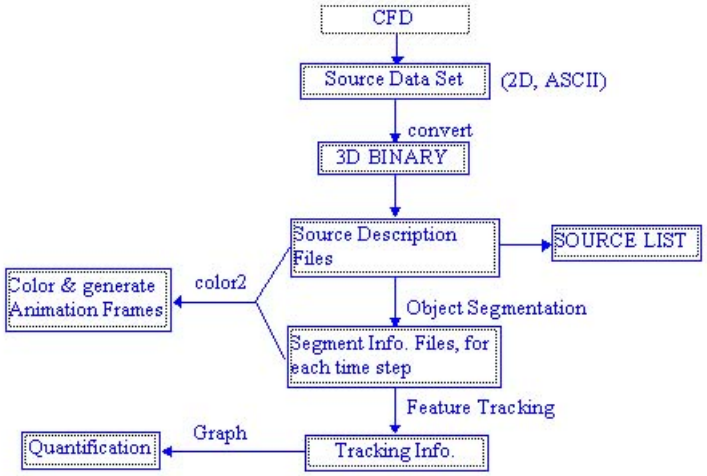
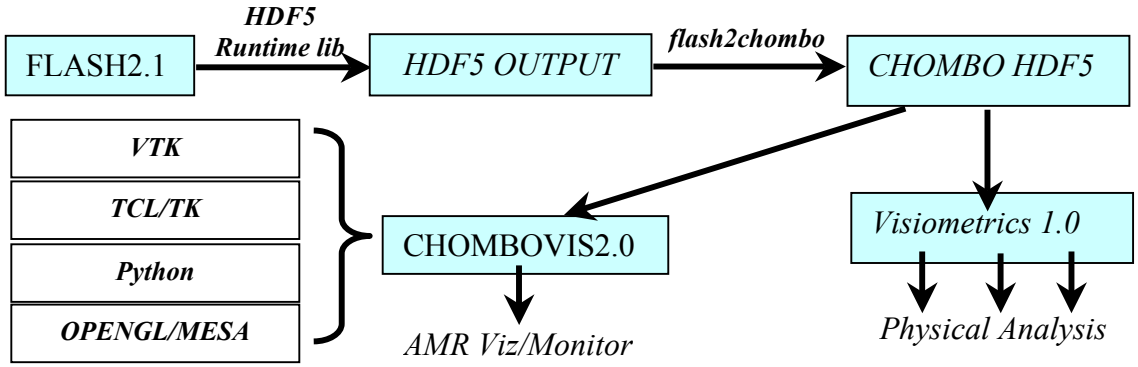
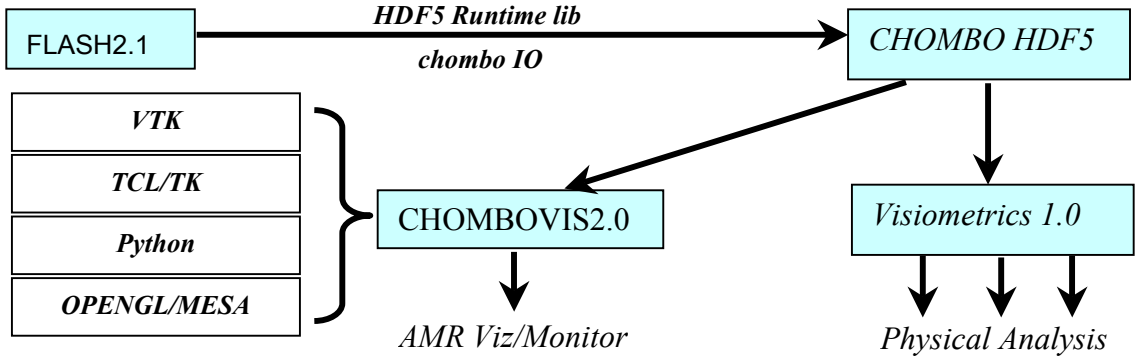


Fig. 3.5 Diagram of the feature tracking package usage.



a. Processing FLASH AMR data: approach 1



b. Processing FLASH AMR data: approach 2

Fig. 3.6 Typical processing diagram for HPC data: FLASH AMR data as an example. Note the *italic* modules are software support implemented or imported or updated by this thesis.

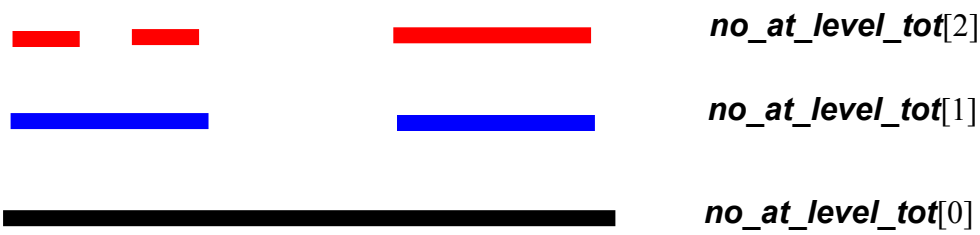


Fig. 3.7 Level-wise block indexes in ChomboVis HDF5 data.

Fig. 3.8 ChomboVis GUI.

Chapter 4 Verification and validation

4.1 Introduction

Verification and validation, referred as **V&V** hereafter, are fundamental steps in developing any new simulation technology. The goal is assessing the credibility of modeling and simulation. Verification means demonstrating that a code or simulation accurately represents the conceptual model – solves the equations right. Validation of a simulation means demonstrating that the simulation appropriately describes nature – solves the right equations. The scope of validation is therefore larger than that of verification, which involves comparison of numerical results with experimental or observational data [Calder *et. al.*, 02].

The issue of verification and validation of a numerical scheme is very important but not yet being addressed with enough emphasis. According to a statistics shown by Steven Orszag at an APS Centennial Meeting, 1999, there are only 6.7% of totally 89 published CFD papers, randomly sampled in Journal of Fluid Dynamics and Physics of Fluids from 1990 to 1999, that addressed carefully the numerical accuracy issues, while 68.5% has little or no tests at all.

This V&V problem requires mathematical and physical insights into the problem to avoid wrong answers. In addition, careful examinations of the output data are crucial when theory is not available or not well-developed, which is the focus of this thesis. Here we use the validation of PPM as an example to show our vortex paradigm mode of V&V.

4.2 Dissipation of PPM

In this section, we examine the dissipation effect of PPM algorithm in different implantations. These dissipations are very important to intermediate to late time dynamics, and could be treated as a sub-grid model of turbulence.

We first study the effect of numerical precision Fig. 4.1. We define circulations as our convergence criteria:

$$\Gamma_+ = \int \omega_+ dxdy\,,\ \Gamma_- = \int \omega_- dxdy\,,\ \Gamma = \int \omega dxdy \qquad (4.1)$$

The simulation we are looking at is shock curtain interaction, with M=2.0, η=7.14, t_{end}=200, resolution 2048x256, on SUN E10K machine. A double precision run uses CPU Time 261,078.56 sec, while a single precision uses CPU time 192,396.91 sec.

Only starting from the end of second time epoch (eII), the influence of numerical error becomes significant, i.e., dissipation starts at an later time in double precision run.

In Fig. 4.2, we show the convergence of circulations for runs with zero-viscosity and with two numerical methods: PPM and Weighted essentially non-oscillatory (***WENO***) method (two runs with WENO at 5th and 7th orders of accuracy respectively). The physical conditions are an M =1.2 shock traversing a SF_6 gas cylinder in a 200x800 uniform mesh numerical shock tube. The circulation differences are unobservably small during shock passage and within 5% for later times. These differences come from the formation of small-scale structures beyond the intermediate time, which are also observed experimentally. All qualitative features in density and vorticity images are common to all the runs. Although PPM is only upto 3rd order accurate, its result is comparable to 5th order WENO run.

4.3 FLASH modules: constant viscosity, gamma blending and AMR numeric

We introduced in this thesis a constant kinematic 3.2x10^{-3} cm^2/sec to momentum evolution equation:

$$\text{Momentum: } \partial(\rho\mathbf{u})/\partial t + \nabla\cdot(\rho\mathbf{u}\mathbf{u}) + \nabla p = \rho\mathbf{g} + \nabla\cdot(\nu\nabla\mathbf{u}) \qquad (4.4)$$

In Fig. 4.3, we show that this viscosity does affect circulation evolution in a shock cylinder interaction simulation, with $\Delta x = 0.01$ cm. Fig. 4.3a shows that a viscous flow with viscosity 3.5x10^{-5} cm^2/sec is essentially equivalent to an inviscid flow at this resolution. Fig. 4.3b shows that the circulation is smaller as the viscosity in this range increases.

Note, the FLASH compressible code omits viscosity in the energy equation and thus provides only heuristic results for small scales at late times. For comparisons with experiments, we do not believe that this omission and the lower viscosity used is an important effect for the large and intermediate structures up to intermediate times. In the future, we hope to investigate the role of physical and numerical diffusivities at late times when the small-scale structures are be in a turbulent state.

We note that the AMR implementation and some FLASH modules which introduces more complex physics into the simulation, e.g., gamma blending of different species, make the simulation subjected to more numerical errors.

In Fig. 4.4, we show the error introduced by AMR comparing with the viscosity. We see that the circulation is actually enhanced by AMR scheme, a phenomenon we will discuss in more detail later.

In the following, we further show the solution convergence and accuracy issues on the gamma-blending and AMR. In FLASH, the equation of state of gamma-law gases evolve

with different specific heat ratio, where for the gas mixture, the specific heat ratio is defined as weighted average adiabatic index [FLASH, 02]:

$$\frac{1}{\gamma - 1} = \sum_i \frac{1}{\gamma_i - 1} \frac{X_i}{A_i} \tag{4.5}$$

Where X_i is the mass fraction of the *i*th element, *A* is atomic mass, *i* is 1 and 2 in our case, for air and SF_6, respectively.

Fig. 4.5 shows a convergence study of gamma blending modules in FLASH. Note for all five runs, the net circulations are the same, which means the basic physics for all the runs are consistent. The difference comes only from secondary structures. The hydro solver of VH1 and FLASH are almost identical, revealed by the solid and dotted lines, except a short region between $0.0007 \leq t \leq 0.0011$, where FLASH circulations are smaller. Dashed line shows the effect of multi-species, which makes second baroclinic increase phase start at a significantly early time. Dark solid lines shows deviation from adaptive mesh. The difference is not negligible and is under investigation. We show in addition the simulation with a small diffusion 1.73E-5 here.

Now we switch to another geometry: curtain to examine the convergence issues at late time while turbulence developed. Fig. 4.5 show four time steps for three shock curtain runs:

Run 1: VH1, uniform mesh 256x2048, single species;

Run 2: FLASH, AMR (32x256, 4 AMR levels), multi-species;

Run 3: FLASH, AMR (8x64, 6 AMR levels), multi-species.

It is important to prevent the CDs from being ill-refined to different level of mesh. Hence, an optimization study on the refinement threshold is performed and 0.2 is used for 4 AMR levels and 0.08 for 6 AMR levels.

Figure 4.6a shows the mesh distribution of run 8 with FLASH. Each box in the Figure corresponds to an 8x8 mesh block. Clearly, 4 level of meshes are observed, and the discontinuities are well refined.

Figure 4.6b captures the curtain when it is just hit by the incident shock and some shock wave interactions. The specific heat ratio is an important factor influencing the time scale, which result in the different position of the reflected shock wave front at the same time for different runs. Transmitted shocks for FLASH runs are not captured because of the colormap setups. The VBL driven curtain rolls up almost the same way for all three runs. However, small differences in secondary structures are presented. This continue to be true to intermediate time at t=2.5, Fig. 4.6c. But at late time, Fig. 4.6d, t=6.9, when turbulent mixing dominants the flow, the simulations agrees only in a very qualitative sense: the AMR runs with different γ's are more turbulent. In addition, numerical effects become significant, which is certainly more severe in AMR runs: even the two AMR runs at this time do not agree well.

We amplify the above statement by showing quantifications of Run 1 and Run 2 in Fig. 4.7. In both Figs. 4.7a and b, we see that at early time, the circulation and the enstrophy agree. However, at t> 2.5 the results diverge. In Fig. 4.7a, the positive circulation seem to be subjected to a much stronger secondary baroclinic enhancement for the FLASH run than the VH1 run. This phenomenon is also seen in the stronger mixing process observed in Fig. 4.6 and the y-integrated vorticity in Fig. 4.7c, where we see locally, there is much more positive circulation in FLASH run (dash-dot curve) than VH1 run (solid curve). However, the negative circulation, obtained mostly from the strong localized vortex projectiles is subjected to much less mixing,

In Fig. 4.7d we explain the above deviation by plotting the density gradient distribution. At t=2.5, we see the gradient distributes almost identically, while at t=6.9, the

distributions diverges. In the high gradient region, $|\nabla\rho|>10^3$, VH1 has a higher gradient value because of its intrinsic ability in maintaining contact discontinuities, that is, it is less dissipative. In the large intermediate gradient range $10<|\nabla\rho|<10^3$, we see, FLASH gives a much wider gradient distribution, which contributes to the stronger circulation enhancement in **eII**. This arises because of the stronger mixing obtained from the different equations of state of the two gases and AMR implementations.

In summary, we find that the combined effect of gamma-blending and AMR preserves very high gradient structures, which greatly change the circulation and enstrophy behaviors at and beyond the intermediate times.

It is also interesting to note the difference of the two PPM implementations: PPMLR and PPMDE. Although according to [Woodward & Collela, 84], they are almost identical, VH1 group pointed out that PPMLR is less dissipative and thus better at maintaining contact discontinuities, which is consistently observed here by the slightly larger circulation.

4.4 AMR error exposure with visiometrics

The efficiency in time and computer resources introduced by AMR is at the sacrifice of certain accuracy, as already shown in the global quantifications in the previous section. The import of AMR data hierarchy requires further numerical procedure, and consequently introduces more numerical error. For example, the time step splitting at finer levels introduces more numerical steps and hence more round off error.

These errors are not being fully aware of by the AMR community and hence careful validation is extremely important. This section summarizes the error exposed during the course of this thesis and discusses the solutions.

4.4.1 AMR error

We further focus on the numerical artifacts introduced by AMR. In Fig. 4.8 we show a comparison of AMR introduced numerical dissipation with viscosity. We see the error introduced by AMR is within 1% of the circulations, and will be overwhelmed by the constant viscosity greater than 3.5×10^{-5}.

4.4.2 Ill-imposed initial AMR mesh: refine criteria and local error clustering

Efficient and accurate schemes for refinement and de-refinement of the variables are a crucial element in any adaptive scheme. The refinement and de-refinement strategies used here take advantage of the 1:2 and 2:1 ratios between parent and child cells, allowing these processes to be carried out rapidly and in a conservative manner.

The usual procedure of refinement and de-refinement is driven by the calculation of a local error comparing with a preset error threshold – referred to as refinement criteria. In FLASH, the following error estimator is used [Lohner, 87]:

$$E_{i_1 i_2 i_3} = \left\{ \frac{\sum_{pq} \left(\frac{\partial^2 \rho}{\partial x_p \partial x_q} \Delta x_p \Delta x_q \right)^2}{\sum_{pq} \left[\left(\left| \frac{\partial \rho}{\partial x_p} \right|_{i_p+1/2} + \left| \frac{\partial \rho}{\partial x_p} \right|_{i_p-1/2} \right) \Delta x_p + \varepsilon \frac{\partial^2 |\rho|}{\partial x_p \partial x_q} \Delta x_p \Delta x_q \right]^2} \right\} \tag{4.2}$$

where the ε term in the denominator acts as a filter, preventing refinement of small ripples. Partial derivatives are evaluated at the center of the $i_1 i_2 i_3$-th zone.

These errors computed at each cell were clustered to compare with the refinement threshold. In AIFs flow, the situation illustrated in Fig. 4.8a is possible, which gives the ill-imposed mesh because the material interface haven't been refined to the same level. For the type of flow that is extremely sensitive to initial condition, the boundary of different refinement level at the interface provides a perturbation which propagates

through the time evolution, and result in wrong secondary instability as in Fig. 4.8b. For comparison, the correct result is shown in Fig. 4.8c with appropriate refine criteria (usually lower). Note the best criteria is always a result of optimization because if the criteria is too low, it will waste a lot of mesh on refining less significant regions, with errors coming from maybe only numerical noise.

4.4.3 Error exposure with visiometrics and anomaly

Most of the AMR convergence study is concentrated on 1D problem with known analytical solution. The addition of AMR levels are usually shown to converge to the analytical solution[FLASH, 02]. However, few studies have been performed further.

We set up a straightforward experiment to investigate the convergence of 2D simulation. The strategy is: we keep the numerical resolution the same, while varying the combination of the base mesh and AMR level, which gives the same numerical resolution. Four runs are performed, as listed in Tab. 4.1. The VH1 run is only shown for reference.

Figure 4.9a shows the density images for run 1 and 3. There are subtle differences in the two runs which are hard to identify through this qualitative comparison. We need to get quantitative with our visiometrics toolbox.

The first step, we check in Fig. 4.9b the global conservation of the run:

$$Mass(t) = \iint \rho(t) dxdy \tag{4.3}$$

We note:

1. The increase of the mass is due to the compressibility of the flow, which is balanced by the inflow/outflow boundary condition and gives mass conservation;

2. Mass evolution in PPMLR run (VH1) is offset a bit from the PPMDE run (FLASH2).

3. Three AMR runs has the error around 0.1% ((0.0001/20) / 0.0045);

4. The number of refinement levels in AMR runs are guaranteed by the refinement criteria to refine to the finest mesh, as shown in the following picture

This is the stage of most AMR validation study.

We further project the mass to the axis as in Fig. 4.9c:

$$Mass_y = \int \rho dy \tag{4.8}$$

We start seeing difference between the first three runs:

1. Each peak corresponds to a roll of the vortex structure, roughly speaking;

2. AMR errors are larger inside the mixing zone;

3. AMR makes the mass less mixed

More AMR, higher value inside the elongated bubble;

More AMR, lower value in the mixing zone;

In Fig. 4.9d we plot the axis slice, which shows consistency with previous conclusion: AMR produces more gradient and hence less mixing.

In Fig. 4.10, we plot the circulation budget of the runs. We note that the net circulation (solid curves) are consistent among all FLASH runs. Hence we conclude that the positive and negative circulation differences are due to the small-scale structures, the same course as in the mass mixing, and also with the same trend. AMR runs are less dissipative.

All the above quantifications expose the anomaly in AMR: that is, AMR runs are less dissipative than the uniform mesh, which is conflict with the fact that AMR actually introduces more error to the simulation and hence has larger numerical dissipation.

We leave the detail investigation of this anomaly to future study.

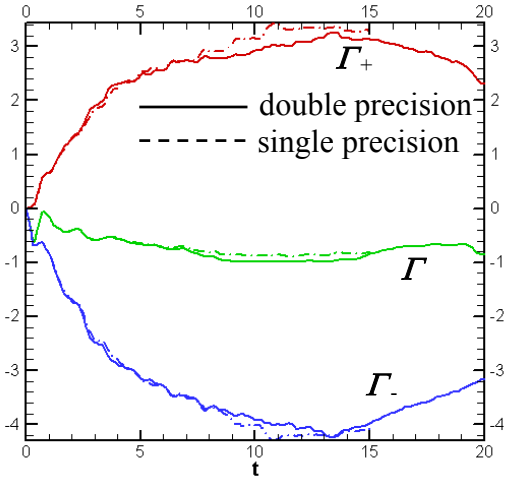


Fig. 4.1 Precision study: effect of precision to intermediate to late time phenomena. Use global circulation as diagnostics.

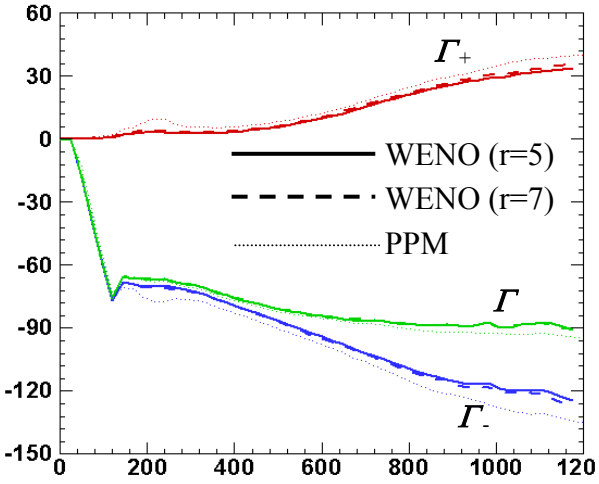


Fig. 4.2 Validation of the numerical schemes: positive, negative and net circulation evolution for an M = 1.2 shock interaction with an SF_6 bubble. The resolution is 200 X 800.

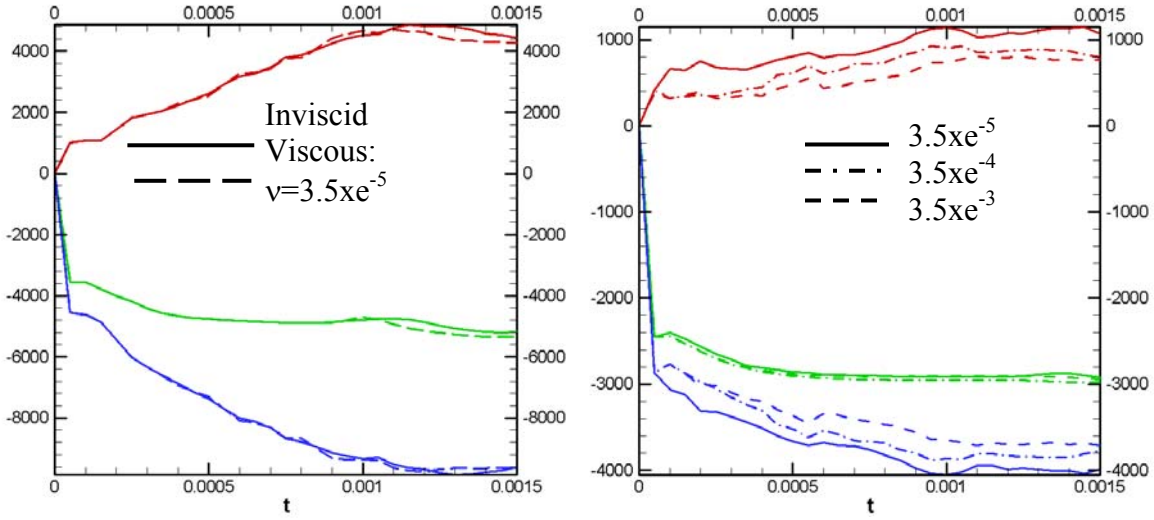


Fig. 4.3 Validation of physical viscosity and numerical viscosity. a. test runs for inviscid flow and viscous flow with kinematic viscosity 3.2xe-5; b. three viscous test runs with different viscosity. Note all the simulations, the resolutions are the same at 128x256.

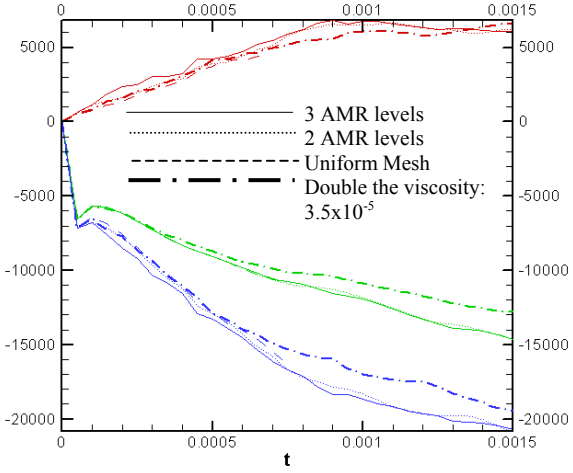


Fig. 4.4 Numerical dissipation introduced by AMR and viscosity

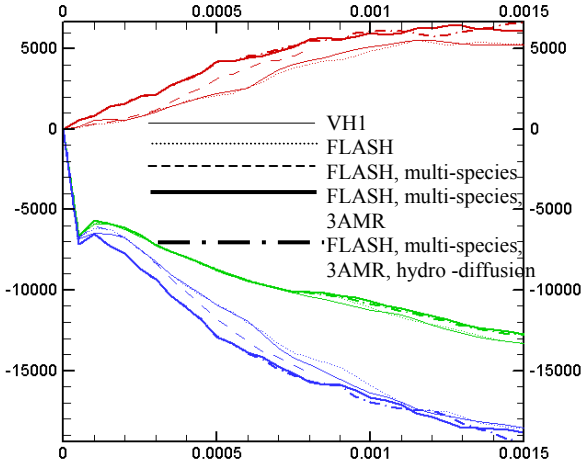
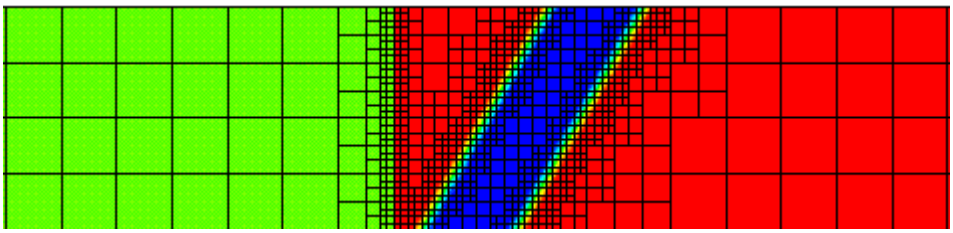
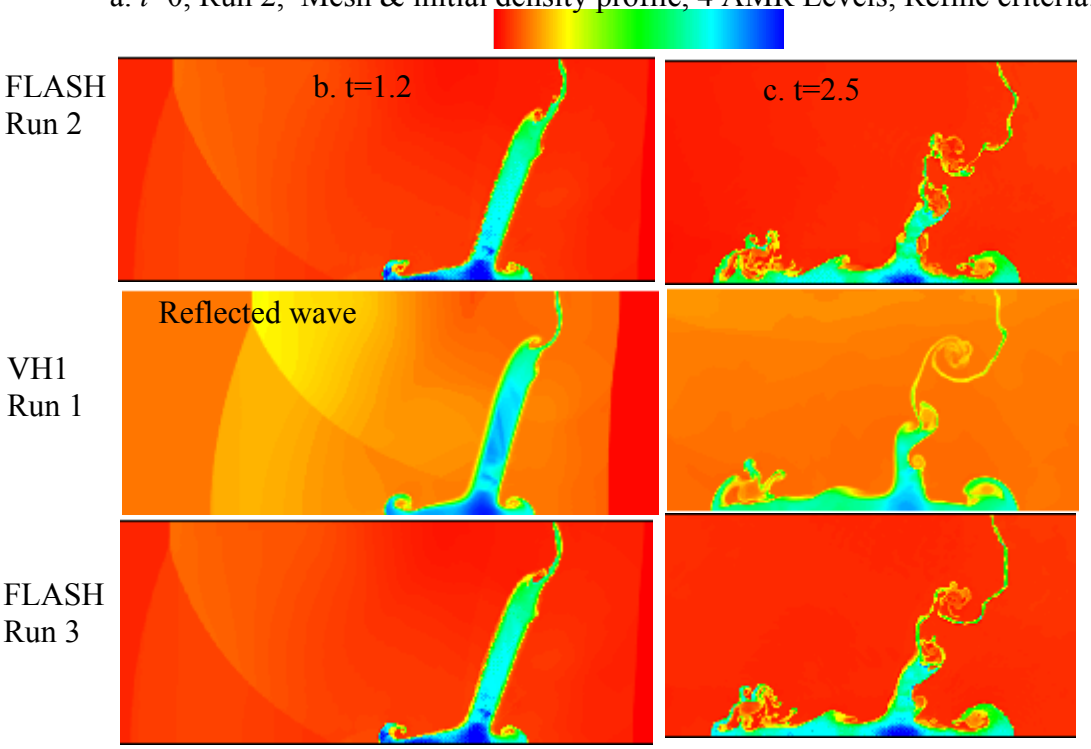


Fig. 4.5 A study of FLASH parameters: gamma blending and viscosity



a. *t*=0, Run 2, Mesh & initial density profile, 4 AMR Levels, Refine criteria: 0.08



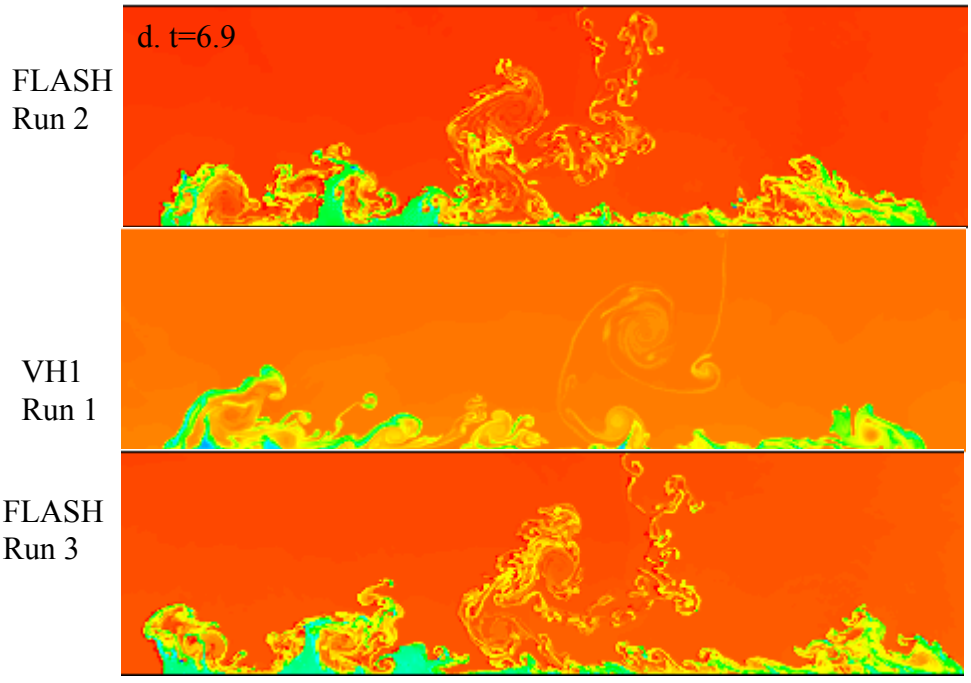


Fig. 4.6 VH1 vs. FLASH: species and AMR modules. Sampled density images for three runs.

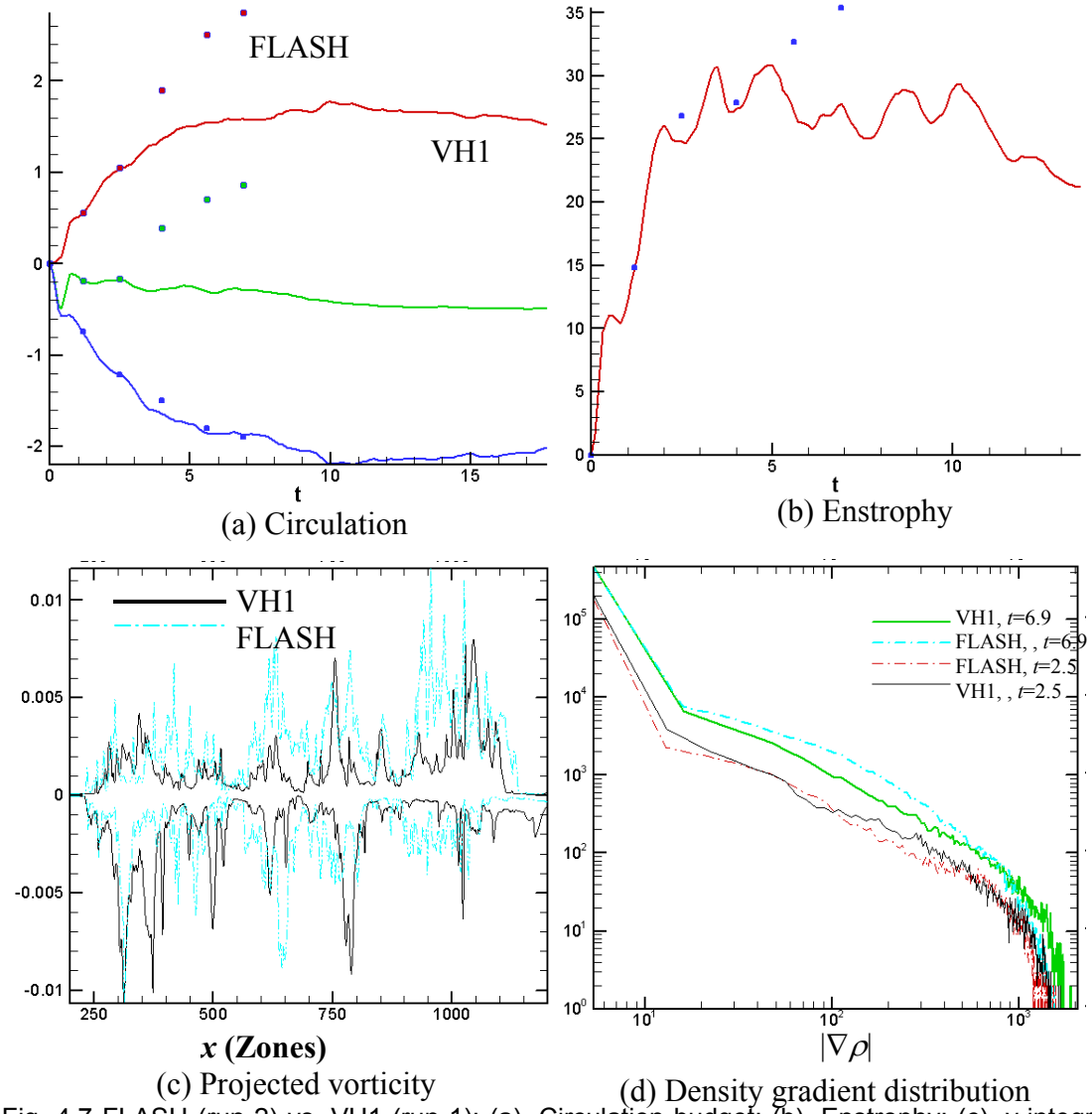
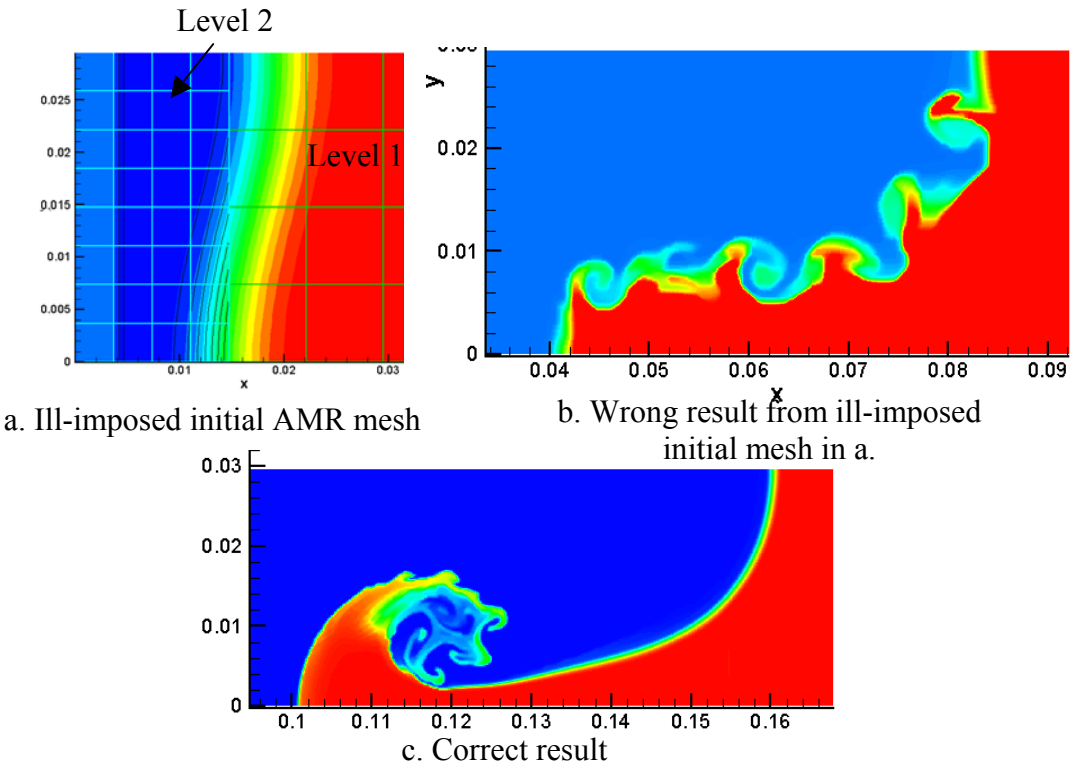


Fig. 4.7 FLASH (run 2) vs. VH1 (run 1): (a). Circulation budget; (b). Enstrophy; (c). y-integrated vorticity at *t*=6.9; (d). gradient distribution at *t*=2.5 and 6.9.



a. Ill-imposed initial AMR mesh

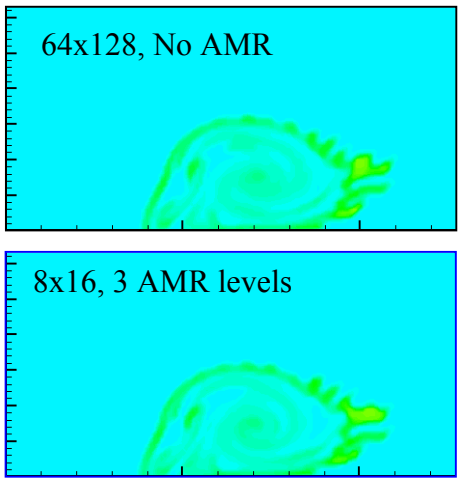
b. Wrong result from ill-imposed initial mesh in a.

c. Correct result

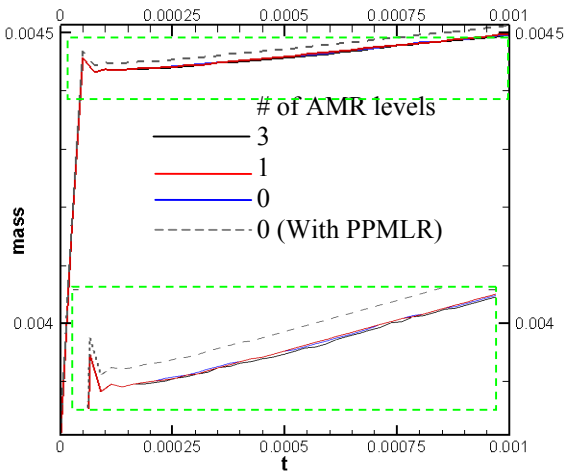
Fig. 4.8 Error produced by ill-imposed initial AMR mesh.

Run	Code	Resolution	Base Mesh	AMR Level
0	FLASH	128x256 Shock Cylinder Sun E10K 64 bit, Double precision	64x128	0
1	FLASH		32x64	1
2	FLASH		8x16	3
3	VH1		64x128	0

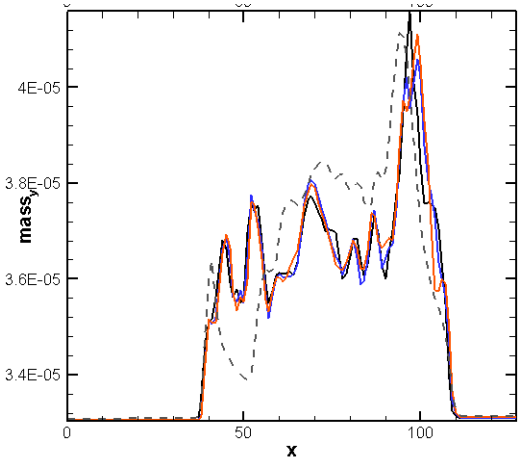
Tab. 4.1 Simulation summary of the FLASH AMR convergence study



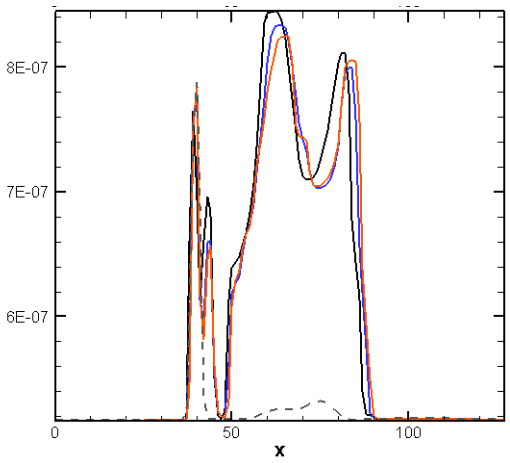
a. Density images for two FLASH runs, with refine threshold 0.2



b. Convergence study: no error (<0.1%) on mass evolution $Mass(t) = \iint \rho(t)dxdy$



c. Error exposure by projections: Axis Integration $Mass_y = \int \rho dy$



d. Error exposure by projections: Axis slice

Fig. 4.9 Error exposure by visiometrics in high performance computing

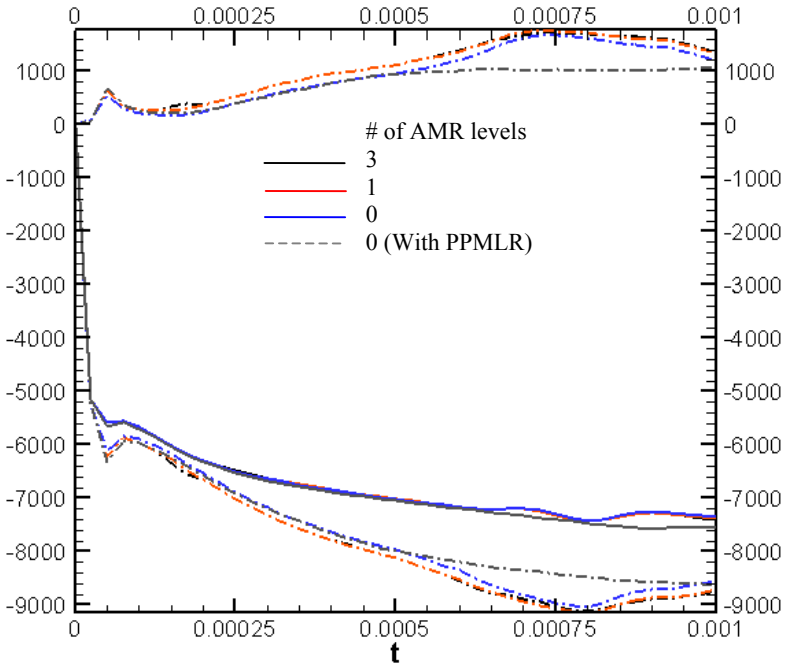


Fig. 4.10 Circulation budget of different AMR runs.

Chapter 5 Experiments/Simulation comparison and uncertainty quantification

5.1 Introduction

Laboratory studies have a lot of uncertainties: repeatability, limitation of measurements, three dimensionality, undesired boundary layers etc. In many cases, a membrane separates the two gases - the energy expended in breaking the membrane has been cited as the reason for the discrepancy between numerical and experimental values. Although advanced experimental technology has provided more accurate and detailed measurements of the RM environment without using membrane, e.g., [Zoldi, 02] and colleagues at Los Alamos National Laboratory investigated an M=1.2 shock cylinder interaction with Particle Image Velocimetry (***PIV***) and Planar Laser Rayleigh Scattering (***PLRS***), still, large discrepancies are observed between the experiments with their simulations.

People in the field realized these issues and started addressing these uncertainties, for example, [Benjamin, 91] modeled the membrane as a thin layer of dense gas and concluded that the effect of the membrane should be negligible. The complexity of this type of uncertainly, in terms of the number of uncertain parameters and the range of the uncertainty for each parameter, demands a systematic way of addressing these uncertainties, featured by a workshop “quantifications of uncertainties in physics simulations” held at LANL, 2002.

Although the presence of strong shocks and density contact discontinuities (***CD***) make it a challenging task to perform trustworthy simulations beyond early times, it is

nevertheless extremely helpful to interpreting experimental data, because of its explicit control on the initial condition. Since the first numerical calculations [Meyer & Blewett, 72] obtained good agreements with Richtmyer's experiments, numerical simulation becomes a very important tool to verify the experimental conditions and analytical results, as well as allows predictions of complex system, thanks to the advances of the modern computers and numerical scheme.

An important conclusion from 2D numerical work is that the late—time behavior of the RM instability is sensitive to the initial condition. [Mikaelian, 88] has reported that this dependence at late times is weak. His computations were supposedly identical to the experiments of [Sturtevant, 87], however, Sturtevant's experiments had ill-defined initial conditions due to the gas separation scheme: a plate was used and withdrawn before the shock reached the interface.

The literature shows many efforts addressing the uncertainties in initial conditions in the experiments with simulation and analytical tools, by usually looking at one or two particular parameters. For example, [Cloutman & Werner, 92] studied the effect of boundary layers. An analytical treatment of the effects of viscosity was presented by [Mikaelian, 92].

5.2 Uncertainties and Dynamic Validation

5.2.1 Optimization prototype

Verification of codes and validation against experiments is a very important issue in numerical simulations [Calder et. al., 02]. In particular, the problem of reduction of order of accuracy of some codes when shocks and high-gradient regions arise in the flows has not been widely appreciated. For example, recent studies observed non-convergence behavior under mesh refinement in Rayleigh-Taylor simulation with FLASH code [Calder

et. al., 02]. In the discussion just below, we assume that the code has sufficient accuracy and resolution to solve the equations accurately and focus on the validation process.

If an experiment exists, the goal of simulations is to capture the large, intermediate and small scale features in space and time. Often it is convenient to introduce symmetries into the simulation, based on remarks of the experimentalist and the requirements of simplicity (e.g., 2D vs 3D). Most often one deals with data and images in the literature and essential information is unavailable. On the other hand, even the experimentalists are not able to explain some phenomena, due to the limitation of experimental technique. Some typical experimental uncertainties are:

- 3D nature of the experiment;
- Undesirable boundary effects (boundary layers, wave reflections, perturbations, etc);
- Causes of the asymmetry in the initial bubble shape due to the setup of the experiment;
- Thickness of ITL at shock arrival;
- Effect of seeding on initial true density and during evolution;
- Correlation of experimental visualization with physical quantities;
- Reproducibility of the initial conditions for multi-experimental study of evolutions.

5.2.2 Simulation parameter space

In Fig. 5.1, we summarize our dynamic initial condition validation process in obtaining the good agreements with both experiments shown in previous sections. The solid arrows indicate the process going forward from experimental data to the simulation setup, and the dashed arrows are feedbacks of information obtained from the visiometrics of simulated data.

The basic ideas are: from the experimental data and documentation, we obtain the experimental configuration, visualization technique and time evolution from which we choose the parameter sets. We limit the experimental uncertainties to a few parameters by simplification and modeling. For example, we concentrate on the ITL thickness, ITL profile, and SF_6 concentration in this paper. These parameters are used as initial condition to invoke the simulation. Evolutionary data is fed into the visiometrics environment, where we define, extract and quantify features to modify control parameters, e.g., the bounding box dimensions (see Figs. 7.4 and 11) and the velocity distribution function in an extracted frame (see Figs. 7.8 and14c). The uncertainty parameters initially fed into the simulation as initial condition are adjusted by producing optimal agreements on these control parameters.

Two optimization cycles are defined in Fig. 5.1: Loop 1, the optimization loop explored in this paper, and Loop 2, optimization and direct feed back from experimental initial condition. In a proper collaboration the simulator would interact with the experimentalist to elucidate possible experimental errors and discrepancies [Thurber & Hanson, 01] [Baltrusaitis *et. al.*, 96].

Note, in Zoldi's attempt on juxtaposition with her own experiment using the Radiation Adaptive Grid Eulerian (***RAGE***) code, the comparison is less comprehensive and not as good without identifying the importance of the information feedback and optimization.

Table 5.1 shows the parameters explored in this study, which is a extended table to Tab. 2.2. The adjustment of numerical parameters is mainly for validation purpose as discussed in the previous section. The physical parameters shown in the table correspond exactly the range of ambiguities in the experiments.

A diffused ITL between the gas bubble and the ambient is obvious in both experiments (Fig. 7.3a for Jacobs and Fig. 7.11a for Zoldi). The medium seeded to the original SF_6

gas (biacetyl in Jacobs' and glycol in Zoldi's), necessary for laser-induced visualization technique, makes the observation ill-informed especially at the interface. A common assumption of the ITL profile is the error function, solution of diffusion equation. However, other profiles such as Gaussian [Zoldi, 02] are also used based on the assumption that the exact profile has smaller effect than the introduction of ITL itself.

The seeding medium, on the other hand, changes the density of the bubble. As a result, the concentration of the initial bubble gas, another quantity that is difficult to measure, is also a critical parameter, since it actually determines the initial deposition of the baroclinic circulation by the incident shock. Variation of the concentration changes the mass of the bubble gas too, although we keep the mass conserved while adjusting the thickness of the ITL. Consequently, it will influence the distribution of the mass, hence the distribution of the density gradient.

Note with initial concentration less than 100%, either because the diffusion time is so long that all original bubble gas is diffused, or because of the seeding, it is very possible that there are actually no pure bubble gas present initially, i.e., the whole bubble should be modeled as an transition layer. Consequently, the exact profile of the ITL has more ambiguity, because no diffusion law holds if one gas is completely mixed.

We choose a constant viscosity 3.2×10^{-3}, an order of magnitude smaller than the real SF_6 viscosity, taking account of the mixing and the assumption of its negligible influence on energy equation. We assume gamma law gas properties for both air and SF_6, and use the specific heat 1.4 for both gases.

Other parameters which are important to the experiments, but we consider them having smaller effects includes: the flow rate of the SF6 (necessary to produce the cylinder) in the 3rd dimension; the initial asymmetry of the gas cylinder; the percentage of the

seeding media in the bubble and their dynamical significance; three dimensional effect; and wall effect.

In the following sections we discuss three of our simulations, with parameters also listed in Tab. 5.1. Simulation cyld2 and cyld3 compare with Jocobs' experiment at M=1.095, and focus on the effect of ITL parameters and SF_6 concentration. cyld4 compare with Zoldi's experiments at M=1.2, with parameters adapted from the previous simulation-experiment comparison.

5.3. Comparison with Jacobs' experiment and dynamic validation of experimental initial condition

5.3.1 Validating experimental initial condition

We extract the colormap used in Jacobs' paper [Jacobs, 93] and reconstruct it for our simulation. We find that the profile is nearly linear and thus accounts for our **cyld2**, as shown at t=0 in Fig. 5.2 We are aware that it is difficult to associate a PLIF image precisely with the density of the carrier gas for a variety of reasons. Cyld3 is therefore based on the validity the diffusion equation and used an error function profile.

In Fig. 5.2, we show the comparison of Cyld2 (column 2) and Cyld3 (column 3), with Jacobs' PLIF images (column 1). The rows correspond to different times. Note in Fig. 5.2, column 1, the original sequential marks of the experimental PLIF images are kept for reference. They are shown at only selected times here. For column 2 and 3, each picture is composed of two images, density (above) and the vorticity (below), except for t=0 ms, when vorticity is zero and we show instead the initial on-axis density profiles ($\rho(x,y=0)$).

For a quantitative comparison, we define three bounding box dimensions: the width, height and neck of the evolving bubble, shown in the embedded image at upper right

corner in Fig. 5.3. For the experiment, we measure these scales directly from the PLIF images. For the simulations, we use an interface extraction and tracking algorithm with density gradient. All the measured values are normalized by the initial diameter of the bubble, 2*R*.

We adjust by trial and error other parameters to get good agreements for both simulations, for examples, radius/thickness of the ITL and SF_6 concentration [Baltrusaitis, *et. al.*, 96]. We believe that an automatic optimization process is necessary to make efficient initial condition validation.

Both simulations quantitatively reproduce the experiments in large and intermediate scales as shown in Figs. 5.2 and 5.3. We now discuss the subtle differences that exist in the two simulations and thus provide a better understanding of the initial parameters.

In Fig. 5.3, we see the solid curves for width and height from cyld2 agree well with the first two points of Jacobs' experiments. However, at later times there is more disagreement, especially the height. With the same colormap, cyld3 does not compare as well at early time but does much better at intermediate to late times, on all three scales. Thus we conclude that cyld3 is overall a better simulation and use it for our further simulation in next section. More details of the differences in evolutionary phenomena are discussed in the following sub-section.

The most important inference of these subtle differences is that none of these two analytical forms of the initial ITL is exact correspondence with the experimental condition.

5.4. Comparison with Zoldi's experiment

5.4.1 Overview

Following the previous analysis and resulting parameters, we further validate our initial condition reconstruction process by comparing our simulation (Cyld4) to recent experiments performed by Zoldi2002 at Los Alamos National Laboratory. With advanced experimental technique, a stronger incident shock wave is studied (M=1.2) and more quantitative information is available. The attempt of numerical validation with Radiation Adaptive Grid Eulerian (***RAGE***) code is made, with difficulties shown, discussed but unfortunately not completed.

With the same dynamic validation process as in the previous section, we obtain excellent agreements with Zoldi's experiments, even without the direct access to the experimental data. As we shown in Tab.1, we use exactly the set of parameters used in Cyld3, except a larger radius of the gas bubble and a stronger shock wave, both correspond to Zoldi's experiment.

Zoldi's RAGE simulation uses the initial condition constructed directly from experimental data, i.e., including the factor of initial asymmetry and CD diffusion. However, the ambiguities are still overwhelming and the comparison is poor. With certain validation effort, better insight is obtained and the comparison is improved. Numerical errors arise from ill-imposed adaptive mesh, e.g., the boundary of the experimental data is refined as a small but sharp discontinuity upon plugging into the ambient gas in the simulation. Details of the numeric such as those associated with adaptive mesh, and the optimization of the numerical parameter set, are beyond the scope of this paper, and we'll discuss in the future.

Aside from more careful numerical and physical setup, our FLASH simulation introduces physical viscosity. The initial symmetry is assumed in our simulation, since the asymmetry is smeared during time evolution in both Zoldi's experiments and RAGE simulation.

By defining the characteristic length scale and distributing effective mesh accordingly, our FLASH simulation, on a SUN Enterprise 100000 machine, achieve great efficiency (a factor of 3) than the RAGE simulation.

5.4.2 Evolution morphologies

Figure 5.4 illustrates the evolution of the major mass and vortex phenomena in our simulation (Cyld4) comparing with the experiments. Fig. 5.4a shows the experimental images (volume fraction of SF_6). Fig. 5.4b shows the simulation result, density above and vorticity below.

Initially, note again the asymmetry in the experimental image. The time used by Zoldi starts after shock passage, which is ambiguous. We find a shift of 75ms is necessary to account for the shock passage time and match the time scales between the simulation and experiment.

The simulations represent well the large and intermediate scale features. Fig. 5.5 shows the measurement of the bounding box dimensions: height, width and neck, as defined for Fig. 5.3. Note the measurement for the neck for the experiment is not available, due to the poor quality of the experimental images. The agreements in width and height are excellent.

At t =125ms in Fig. 5.4b, our simulation captured the upstream indentation (indicated by **I** in the figure), with almost exactly the same curvature as in the experiments, and the axis downstream protuberance (**P**). The vorticity image at the same time reveals the

same VBL configuration as in lower Mach number simulations. At around time *t* = 325-475ms, the secondary instabilities appear on the interfaces of the deformed bubble. Fig. 5.4c left show a zoom at *t* = 325ms, with vorticity above and density below.

The secondary structures correspond well with the experiment. For example, at *t*=800ms, (also shown with a zoom in Fig. 5.4c right panel, vorticity above and density below), we indicate the velocity components (arrows), and three pockets of *entrained air* (marked by **1, 2,** & **3).** Another very important observation is associated with the development of the outer shear layer: ① the location of the appearance of the secondary instability; ② the first merger of nearby localized vortices; ③ the extension of the downstream side.

5.4.3 Velocity field validation

In Fig. 5.6, we look into more detail of the velocity field and compare carefully with the experiment. Fig. 5.6a shows an experimental PIV image, which is the basis of all experimental analysis and quantification. The resolution and the windowing domain are fixed. To match the experimental data, we extract and average

the appropriate portion of the simulation data to compute the velocity vectors which is shown in Fig. 5.6b.

We compare in Fig. 5.6c the velocity magnitude distribution. The range of the velocity magnitude and the value of the peak region agree extremely well. This indicates the validity of both our data processing process and the simulation results for large and intermediate scale features. In fact the detailed shapes of the curves are in good agreement, as seen by shifting the simulation data to higher velocity in the inset figure (See the arrows for close matching of features).

During the comparison, we conclude that the following factors will influence the final velocity distribution result: 1). Size of the windowing domain; 2). Portion of cylinder in the data-cut; 3). Data reduction rate (averaging); 4). Number of bins taking the distribution (larger # of bins yields more shallower profile with lower peak value); 5). Frame velocity.

5.5 Conclusion

With comprehensive visiometrics of intermediate and late time simulation data, it is possible to obtain a parameter space that corresponds to the experiments, i.e., validate the experimental initial condition dynamically. In this thesis, we introduce a prototype of this systematic approach of quantifying the uncertainties in the experiments and integrate the visiometrics pipeline into a feedback loop to obtain excellent simulation/experiments comparison.

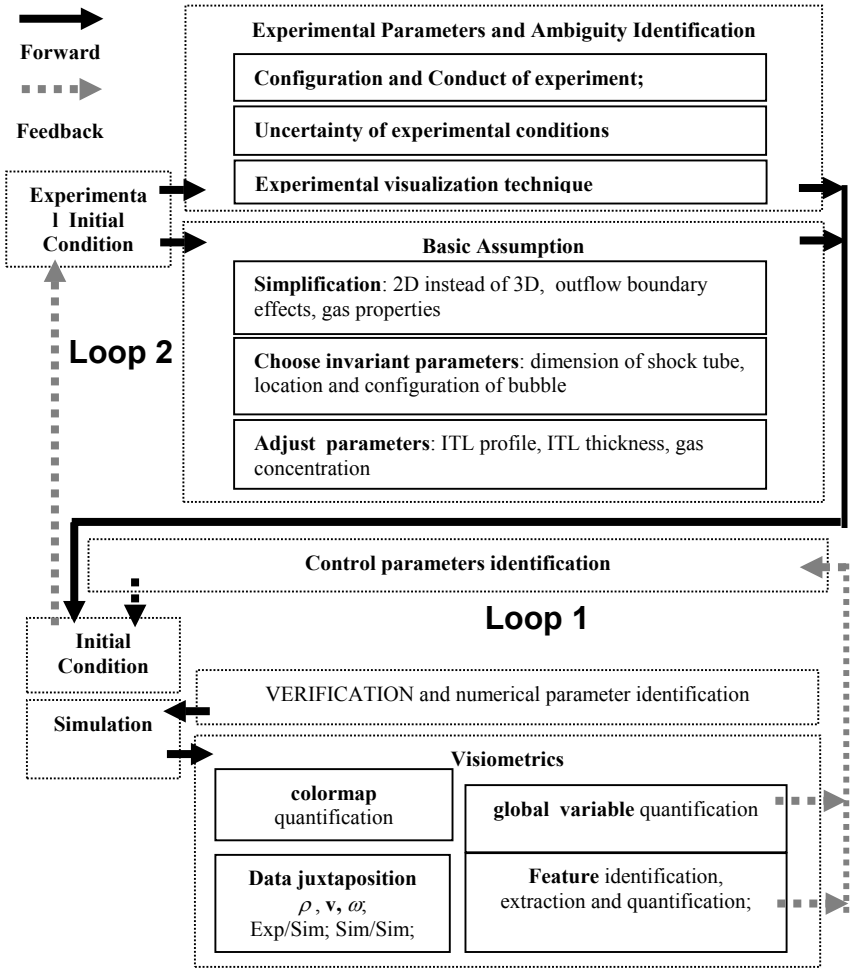


Fig. 5.1 Validating experimental initial condition

Parameters		Studied	cyld2	cyld3	cyld4
Numerical parameters	Mesh Adaptivity	AMR Levels: [0,5]	4	4	4
		Base mesh: [8,200]	16	16	16
	Resolution (*H*x*L*)	64x128; 128x256; 256x512	128x256	128x256	128x256
	Numerical scheme	PPMLR, PPMDE, WENO	PPMDE(FLASH)	PPMDE(FLASH)	PPMDE(FLASH)
Physical Parameters	viscosity	[1.7e-5, 3.5e-2]	3.2e-3	3.2e-3	3.2e-3
	ITL profile	Linear, Gaussian, Error function	smoothed linear	Error Function	Error Function
	ITL thickness (δ/2)	[20%,100%]*R*	100%*R*	100%*R*	100%*R*
	SF6 concentration	[60%, 100%]	100%	60%	60%
	Gamma law	$\gamma_{air}=\gamma_{SF6}=1.4$ $\gamma_{air}=1.4$, $\gamma_{SF6}=1.1$	$\gamma_{air}=\gamma_{SF6}=1.4$	$\gamma_{air}=\gamma_{SF6}=1.4$	$\gamma_{air}=\gamma_{SF6}=1.4$
	Radius (*R*)	[8%,15%]*H*	10%*H*	15%*H*	23.4%*H*
	Shock strength	1.095, 1.2, 1.5	1.095	1.095	1.2
	Y Boundary condition	Reflecting, outflow	outflow	outflow	outflow

Tab. 5.1 Simulation parameter space summary.

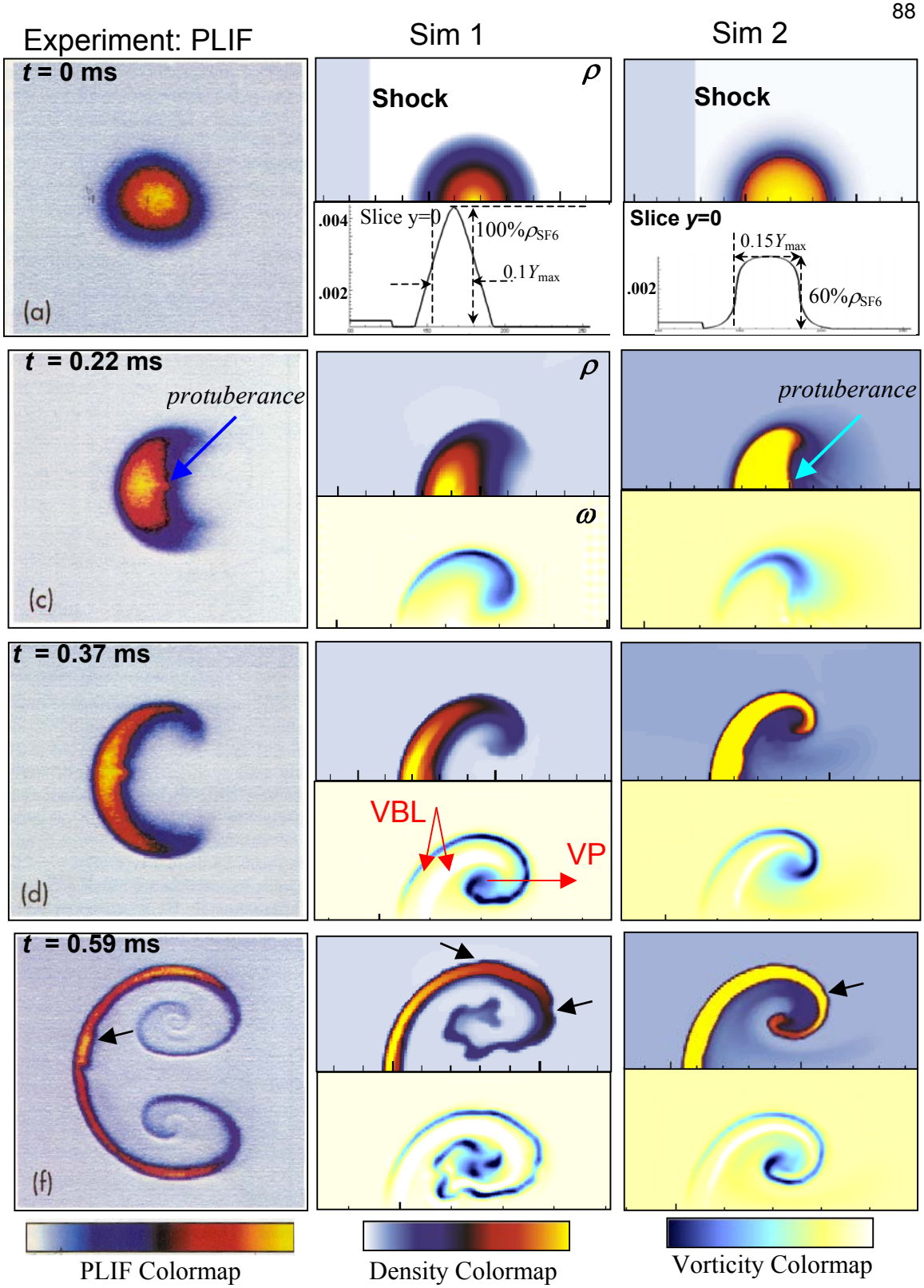

Experiment: PLIF
Sim 1
Sim 2
t = 0 ms
(a)
ρ
Shock
.004
Slice y=0
100%ρSF6
0.1Ymax
.002
Shock
Slice y=0
0.15Ymax
.002
60%ρSF6
t = 0.22 ms
protuberance
(c)
ρ
ω
protuberance
t = 0.37 ms
(d)
VBL
VP
t = 0.59 ms
(f)
PLIF Colormap
Density Colormap
Vorticity Colormap

Fig. 5.2

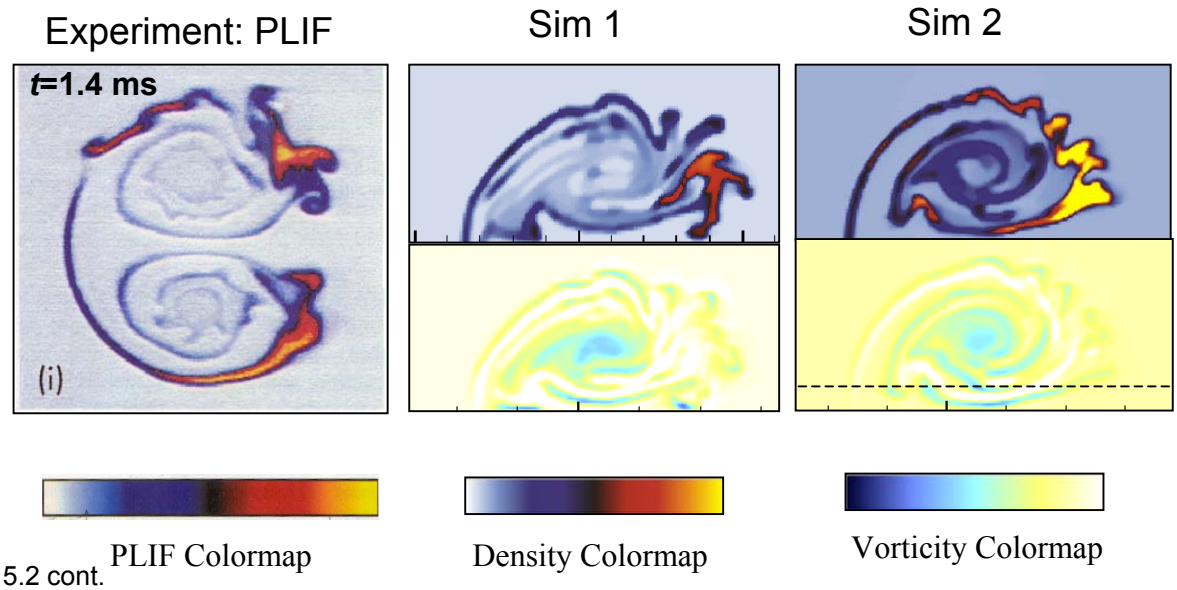


Fig. 5.2 cont.

Compare of experiment (Jacobs, 93), Cyld2 and Cyld3. Column 1: PLIF images from experiments; Column 2: Cyld2 with linear ITL, 0.1*Y*max radius, and 100% SF6 concentration; Column 3: Cyld3 with error function ITL, 0.15Ymax radius, and 60% SF6 concentration. Each row shows a different time, and each picture in column 2 & 3 has two panels: density above and vorticity below, except the first row at *t*=0, with density above and *y*=0 density slice showing initial ITL parameters.

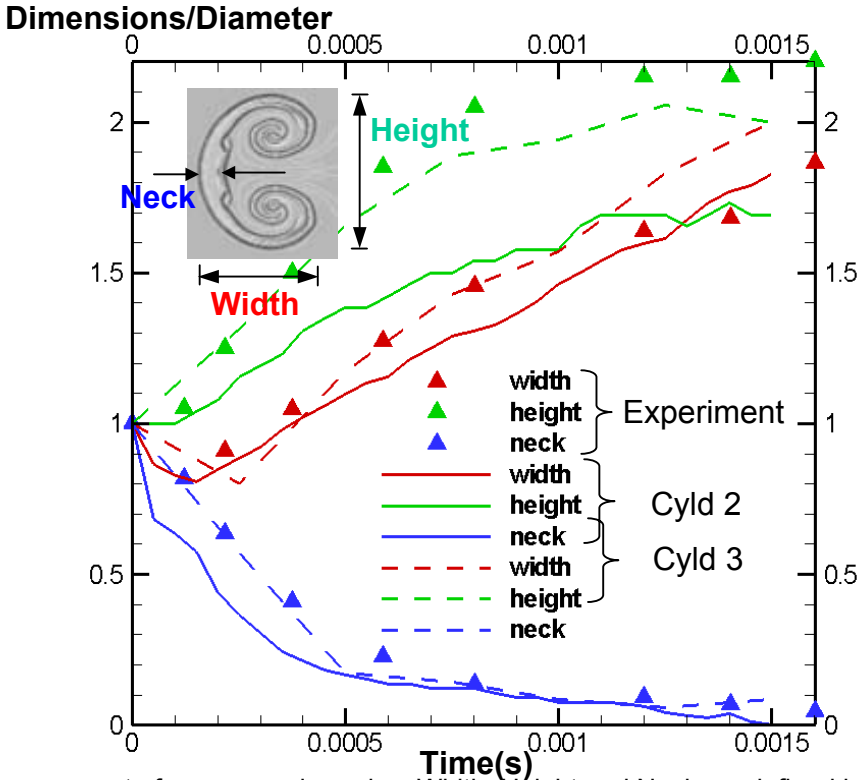
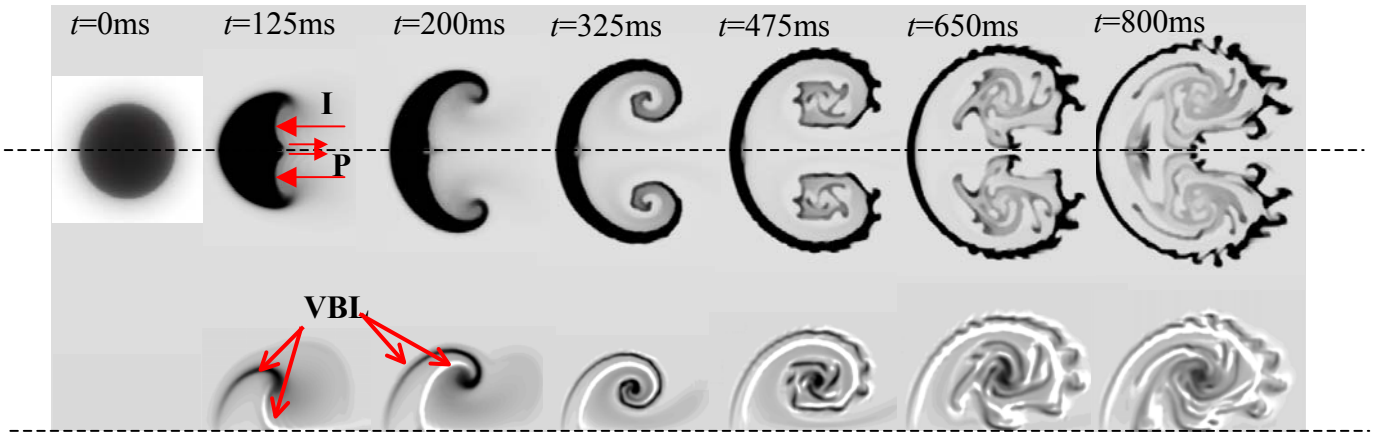
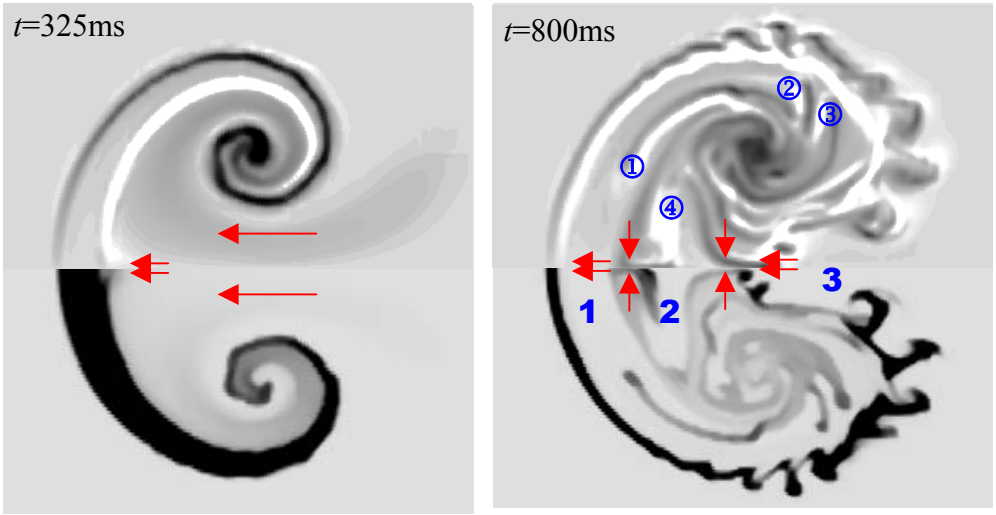


Fig. 5.3 Measurement of macroscopic scales: Width, Height and Neck, as defined in the embedded image at upper right corner, and normalized by the initial diameter of the bubble.

(a). Experimental images



(b). Simulation images: density above and vorticity below



(c). Enlarged simulation image at *t*=325ms and 800ms, vorticity above and density below

Fig. 5.4 Simulation and experiment compare at *M*=1.2.

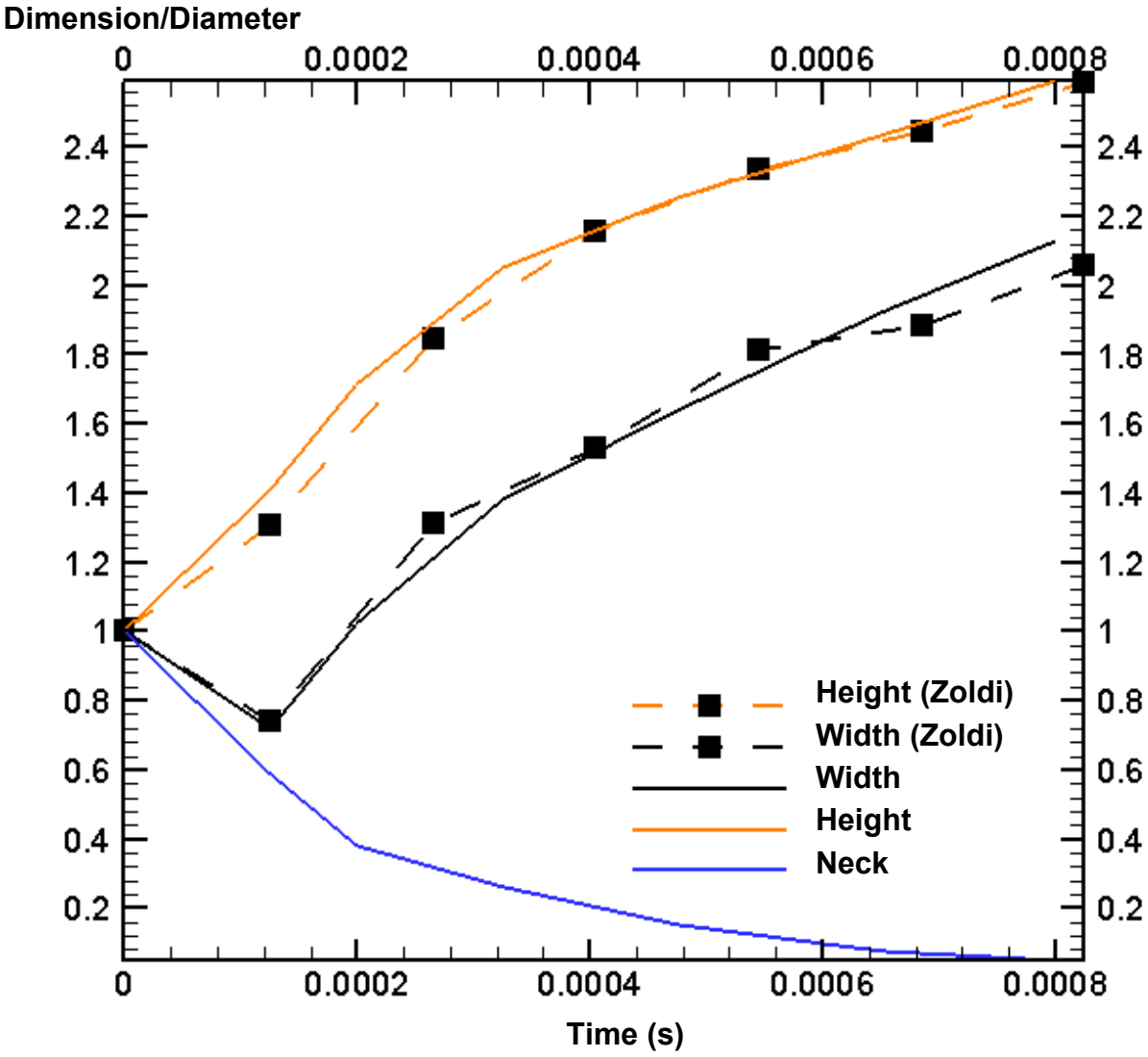
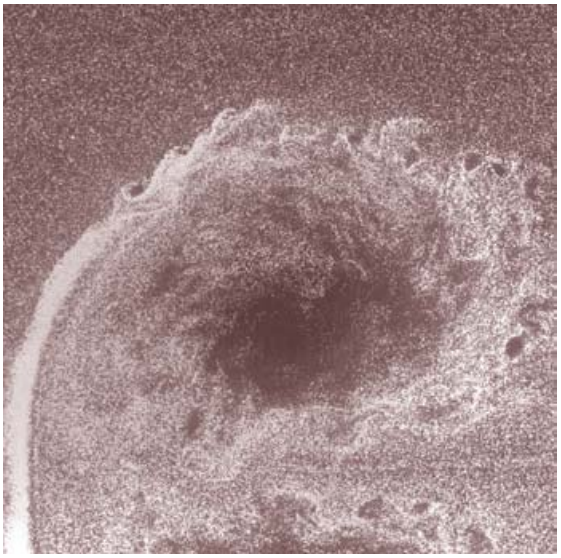
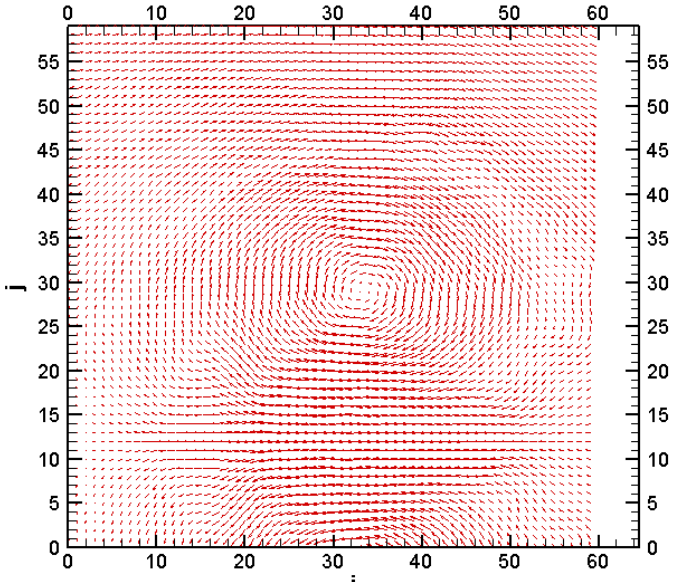


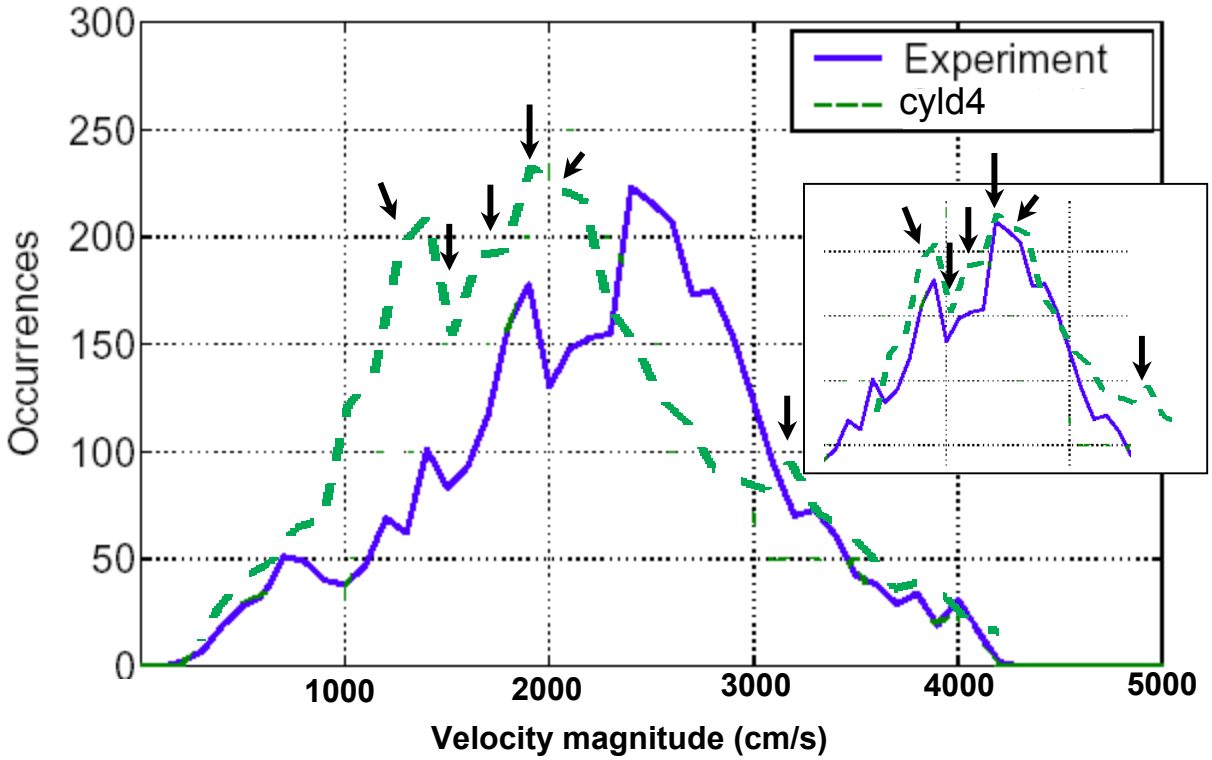
Fig. 5.5 Measurement of macroscopic scales: Width, Height and Neck, as defined in Fig. 5.3, and normalized by the initial diameter of the bubble.



a. PIV image (Experiments)



b. Velocity vector field (Cyld4)



c. Velocity magnitude distribution

Fig. 5.6 velocity magnitude distribution. (a). PIV image from Zoldi's experiments, at the resolution of 11.7μm/pixel, and velocity vectors every 187 μm. (b). Velocity vector field from Cyld4. Note the data is windowed and averaged according to the experimental configuration. (c) Velocity magnitude distribution: experiments (solid) versus simulation (dashed). The embedded figure shows the peak region after shifting a frame velocity difference between the experiment and our simulation.

Chapter 6 Summary and Conclusion

Fig 6.1 shows a comprehensive HPC visiometrics framework developed in this thesis. There are three main parts in the pipeline, the HPC modules (top part), discussed in chapter 3, the visiometrics modules (central part), discussed in chapter 4.

The bottom part of Fig. 4.1 shows the hardware environment supporting our HPC simulation and comprehensive visiometrics. In summary, the major visiometrics work is done on an 8-processor (3 nodes with 2 CPU each and 2 nodes with 1 CPU) DELL precision workstation Win2K cluster. A 16-processor (2 nodes with 4 CPU and 1 nodes with 8 CPU) SGI IRIX64 Cluster and a 44-processor Linux cluster are mainly used as a validation and benchmarking platform. All the major simulations are performed on a SUN Enterprise 10000 HPC machine with 64 processors and 32 GB RAM, and most recently SUN Fire 12K with 32 processors and 144 GB RAM, through Center for Advanced Information Processing (CAIP) at Rutgers University.

The numerical schemes, together with the advanced computational modules, are subject to verification (i.e., check the errors produced by the numerical approximation of equations – whether we solved the equations correctly) and validation (i.e., check the errors produced by the modeling of real world with the equations based on certain assumptions – whether we solved the correct equations). The verification and validation (***V&V***) issues are not addressed with enough care and are discussed in chapter 5, with the assists of comprehensive visiometrics. Excellent comparison results are obtained with multiple experimental date sets.

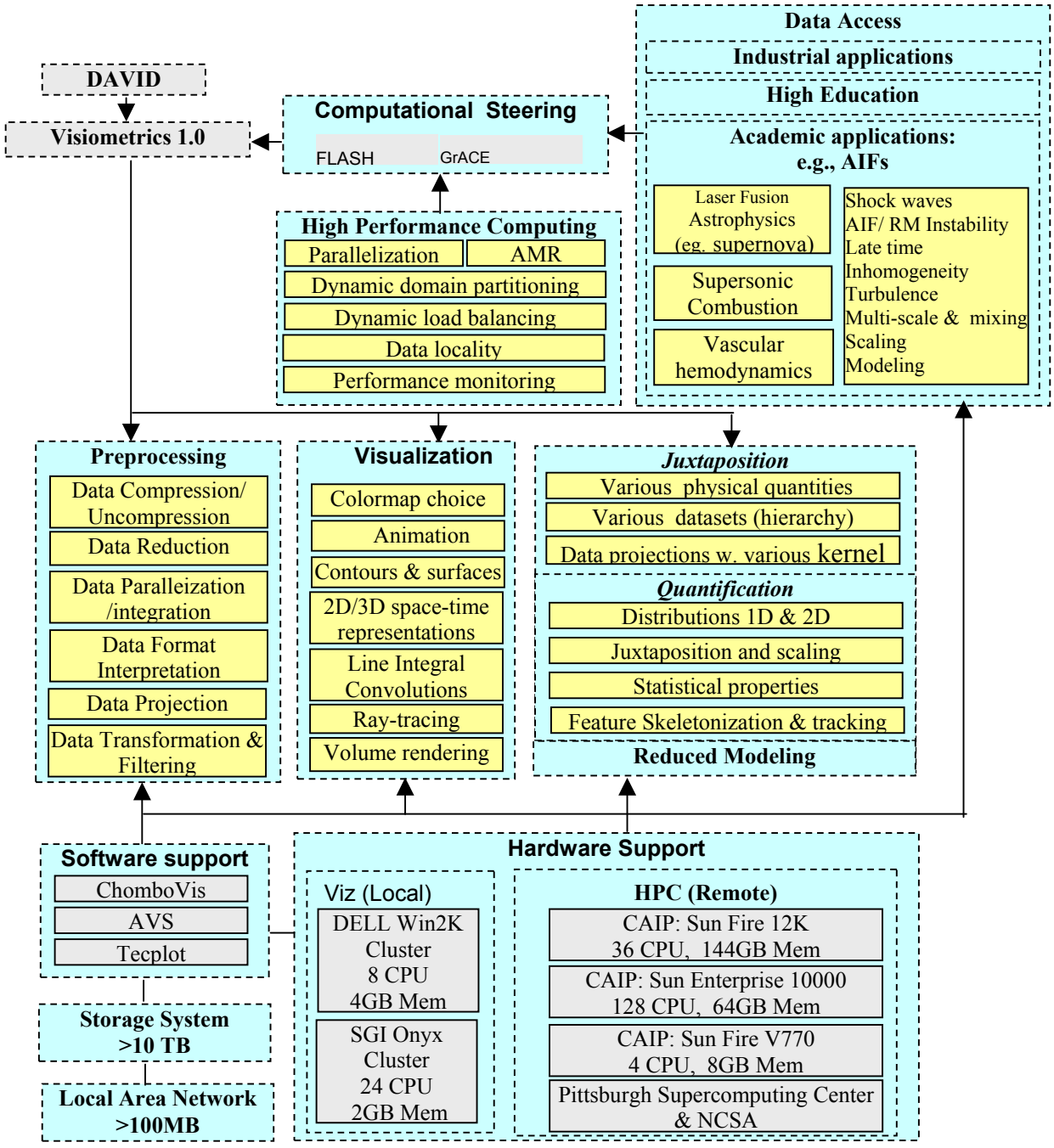


Fig. 6.1 Comprehensive HPC visiometrics pipeline.

References

1. Arnett D. 2000. “The role of mixing in astrophysics”. *Astrophys. J. Suppl.* **127**, 213-217.

2. Arnett W. D., Bahcall J. N., Kirshner R. P. and Woolsley S. E.. 1989. “Supernova 1987A”. *Ann. Rev. Astron. and Astrophys*, 27:629.

3. Baltrusaitis R.M., Gittings M.L., Weaver R.P., Benjamin R.F., and Budzinski J.M. 1996. “Simulation of shock-generated instabilities”. *Phys. Fluids*, 8(9):2471-2483.

4. Benjamin R. 1991. “Shock and reshock of an unstable fluid interface”. In *Proceedings of the 3rd International Workshop on the Physics of Compressible Turbulent Mixing*.

5. Berger M. J. & Oliger J. 1984. “Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations”, *J. of Comp. Phys.*, pp. 484-512.

6. Bitz F.J. & Zabusky N.J., 1990. “DAVID and ‘visiometrics’: Visualizing and quantifying evolving amorphous objects”, *Computers in Physics* Nov/Dec, 603-614. Cover of this issue showing color isosurface of vorticity during 'winding reconnection.

7. Calder A. C., Fryxell B., Plewa T., Rosner R., Dursi L. J., Weirs V. G., Dupont T., Robey H. F., Kane J. O., Remington B. A., Drake R. P., Dimonte G., Zingale M., Timmes F. X., Olson K., Ricker P., MacNiece P. & Tufo H. M. 2002. “On Validating an Astrophysical Simulation Code”. *Astrophys. J. Suppl.* 143, 201.

8. Chang S-K. 2002. “Sketching Skeletons of Elongated Objects Using Scan Line Algorithm”, Ph.D. Dissertation, Department of Computer Science, Rutgers University, NJ, USA.

9. Chen J. & Silver D. 2002. “Distributed feature extraction and tracking”. SPIE, Visualization and Data Analysis Symposium.

10. Chen J., Silver D., & Parashar M. 2003. “Real time feature extraction and tracking in a computational steering environment”. *Proceedings of the High Performance Computing Symposium, HPC2003*, Society for Modeling and Simulation International, San Diego, pp 155-160, March 2003.

11. ChomboVis Website. http://seesar.lbl.gov/anag/chombo/chombovis.html

12. Cloutman L.D. & Werner M.F. 1992. “Numerical Simulations of Richtmyer-Meshkov instabilities”. *Phys. Fluids A*, 4(8):1821.

13. DAVID Website:

http://www.caip.rutgers.edu/vizlab_group_files/RESEARCH/VISIOMETRICS/DAVID/index.html

14. Feher A., and Zabusky N. J. 1996. An interactive imaging environment for scientific visualization and quantification,. *International Journal of Imaging Systems and Technology* 7 121-130.

15. FLASH Group. 2002. “FLASH User’s Guide”. University of Chicago.

16. FLASH Group Website. 2003. http://flash.uchicago.edu/

17. General Atomics Website. 2003. http://web.gat.com/icf/concept/.

18. Jacobs J.W. 1993. "The dynamics of shock accelerated light and heavy gas cylinder", *Phys. Of Fluids A*, **5**, 2293.

19. Lindl, J. 1995. “Development of the indirect-drive approach to inertial confinement fusion and the target physics basis for ignition and gain”. *Phys. Plasmas*, **2**, 3933-4024.

20. Lohner, R. *Comp. Meth. App. Mech. Eng.*, 61, 323, 1987.

21. Meyer K.A. & Blewett P.J. 1972. “Numerical investigation of the stability of a shock-accelerated interface between two fluids. *Phys. Fluids*, 15:753-759.

22. Mikaelian K.O. 1988. Numerical simulations of turbulent mixing in shock-tube experiments. *Lawrence Livermore National Laboratory, Paper*, (UCRL-10098).

23. Mikaelian K.O. 1992. “Effect of viscosity on Rayleigh-Taylor and Richtmyer-Meshkov instabilities”. *Phys. Rev. E*, 47:375.

24. Parashar M., & Browne J. C. 1998. “Integrated Data-Management for Computational Steering,” *31st Annual Hawaii International Conference on System Sciences,* Kohala Coast, Hawaii, CDROM, IEEE Computer Society Press, 10 pages, January 1998.

25. Peng, G. 2003. “Richtmyer-Meshkov instability for a compressible 2D environment: simulations and visometrics for late-interdiate times”. M.S. Thesis, Dept. of Mechanical & Aerospace Engineering, Rutgers University.

26. Post F.H., Vrolijk B., Hauser H., Laramee R.S., Doleisch H. 2002. “Feature Extraction and Visualization of Flow Fields”, in: D. Fellner, R. Scopigno (eds.), *Eurographics State-of-the-Art Reports*, pp. 69-100 (ISSN 1017-4565).

27. Samtaney R., Silver D., Zabusky N.J., & Cao J. 1994. “Visualizing Features and Tracking Their Evolution”. *IEEE Computer*, pp. 20-27.

28. Samtaney R. & Zabusky N. J. High gradient compressible flows: Visualization, feature extraction and quantification, in *Flow Visualization: Techniques and Examples*, Editors T. T. Lim and A. Smith, Imperial College Press, 2000.

29. Silver D. and Wang X. 1997. “Tracking and Visualizing Turbulent 3D Features”. *IEEE Transaction on Visualization and Computer Graphics*, Volume 3, No 2.

30. Sturtevant, B. 1987. In "Shock Tubes and Waves". Edited by H. Gronig (VCH, Berlin), pp 89.

31. Thurber M.C. and Hanson R.K. 2001. "Simultaneous imaging of temperature and mole fraction using acetone planar laser-induced fluorescence". *Experiments in Fluids*, 3093-101.

32. Woodward P. & Colella P. 1984. “The numerical Simulation of Two-Dimensional Fluid Flow with Strong Shocks. *J. Comp. Phys.*, 54:1, 115-173.

33. Yang J., Kubota T. & Zukoski E.E. 1993. “Application of shock induced-mixing to supersonic combustion”. *AIAA J.* **31**, 854-862.

34. Zabusky N.J. 1999. "Vortex paradigm for accelerated inhomogeneous flows: Visiometrics for the Rayleigh-Taylor and Richtmyer-Meshkov environments". *Ann. Review of Fluid Mechanics*, 31:495-535.

35. Zabusky N.J.. 2000. “Scientific Computing Visualization – a new venue in the arts”. In *Science and Art Symposium 2000*, Eds., A. Gyr, P. Koumoutsakos, & U. Burr. Kluwer Academic Publishers.

36. Zabusky N. J. & Kruskal M. D. 1965. “Interaction of ‘solitons’ in a collisionless plasma and the recurrence of initial states”. *Phys. Rev. Lett.* 15, 140-243.

37. Zabusky N.J. & Zhang S. 2002. “Shock planar-curtain interactions in two dimensions: Emergence of vortex double layers, vortex projectiles, and decaying stratified turbulence”. *Phys. Fluids* **14**, 419-422.

38. Zabusky N.J & Zhang S. 2003. “High Performance Computing and Visiometrics in Computational Sciences” *CAIP Annul Research Review 2003*.

39. Zhang Q. & Sohn S. 1997. “Nonlinear theory of unstable fluid mixing driver by shock wave”. *Phys. Fluids* **9**, 1106-1124.

40. Zhang S., Chen J. & Zabusky N.J. 2003. “Turbulent decay and mixing of accelerated inhomogeneous flows via a feature based analysis”. SIAM *Journal on Scientific Computing*, Revised Submitted.

41. Zhang S., Parashar M. & Zabusky N. J. 2001. "GrACE-PPM: A distributed dynamic adaptive mesh CFD Environment for accelerated inhomogeneous compressible flows", *The 54th Annual Meeting of Division of Fluid Dynamics, American Physics Society*, San Diego, CA.

42. Zhang S. & Zabusky N.J. 2003. " Shock–planar curtain interactions: Strong secondary baroclinic deposition and emergence of vortex projectiles (VPs) and decaying inhomogeneous turbulence ". *Laser and Particle Beams*. To appear.

43. Zhang S., Zabusky N.J., Peng G. & Gupta S. 2003. “Shock gaseous cylinder interactions: validation of experimental initial conditions through intermediate to late time visiometrics and vortex paradigm analysis”. *Phys. Fluids*. Revised submitted.

44. Zhang S., Zabusky N. J., & Nishihara K. 2003. "Vortex Structures and Turbulence Emerging in a Supernova 1987A Configuration: Interactions of ‘Complex’ Blast Waves and Cylindrical/Spherical Bubbles ", Laser and Particle Beams. To appear.

45. Zhang S., Zabusky N.J. & Peng G. 2002. “Vortex bilayers, vortex projectiles and decaying inhomogeneous turbulence for shock-planar heavy curtain interactions”. *J. Fluid Mech*. Submitted.

46. Zoldi C.A.. 2002. “A numerical and experimental study of a shock- accelerated heavy gas cylinder”, PhD Thesis, SUNY Stony Brook.