Particle3, The Development of a Particulate Visualization Tool

by

Derek D. Plansky

B.A.Sc. University of Toronto (1993)

Submitted in Partial Fulfillment of the
Requirements for the degree of

Master of Science
in
Aeronautics and Astronautics
at the
Massachusetts Institute of Technology

May 1995

© 1995, Massachusetts Institute of Technology

Signature of Author _____
Department of Aeronautics and Astronautics
May 12, 1995

Certified by _____
Professor Daniel Hastings
Thesis supervisor, Department of Aeronautics and Astronautics

Certified by _____
Robert Haimes
Thesis supervisor, Department of Aeronautics and Astronautics

Accepted by _____
Professor Harold Y. Wachman
Chairman, Department Graduate Committee

MASSACHUSETTS INSTITUTE
OF TECHNOLOGY

JUL 07 1995

LIBRARIES

Aero

Particle3, The Development of a Particulate Visualization Tool

by

Derek D. Plansky

Submitted to the Department of Aeronautics and Astronautics
on May 12, 1995 in partial fulfillment of the requirements for the
degree of Master of Science in Aeronautics and Astronautics

Abstract

Particle based simulations are becoming more common, owing to the maturing of fields such as plasma physics and rarefied gas dynamics and the availability of cheap computing power. Computational power is also making the size of the particle simulations grow. Current simulations now use orders of magnitude more particles than they used just a few years ago. Up until now there had not been any dedicated visualization tools with which to efficiently examine the results of these simulations. This thesis presents the development of Particle3, a particle based scientific visualization package which fills this role. Particle3 is a collection of tools which were developed in order to provide an environment to examine and interrogate particle simulations in three dimensions. In order to simplify the development, Particle3 was based on the paradigm of an existing visualization package named VISUAL3. VISUAL3 is introduced and described. The evolution and development of the Particle3 architecture and tools are described. Finally Particle3 is applied to two simulations in the field of space plasma physics, to demonstrate how it can be used to examine particulate data. The first study examines the effects of probe bias on the Charge Hazards and Wake Shield (CHAWS) experiment and the second case shows the back flow contamination from an ion engine on the ARGOS spacecraft.

Thesis supervisor: Daniel E. Hastings
Professor of Aeronautics and Astronautics

Thesis supervisor: Robert Haimes
Engineer in Aeronautics and Astronautics

ACKNOWLEDGMENTS

I owe a heartfelt thank you to many people for contributing to my time at M.I.T.. The following is a partial list of those who's contributions were particularly important.

First, I want to thank my advisors who made this research possible, Bob Haimes--for his patience and enthusiasm, and Professor Dan Hastings--for all the insights he shared about the space industry.

Robie Samanta-Roy and Gabriel Font-Rodriguez warrant special thanks for their individual contributions of test data, computational expertise and all round helpfulness.

Big thanks go to Graeme, Ray, Carmen, Jim, Karen, Guy, Felusho, Mike, Mike, Dave, Brian, and everyone else in the Space Power and Propulsion Lab, both for the working discussions and good goof off company. In particular, Graeme Shaw for the intellectual stimulation and good friendship that helped me through. Some of the things that I will remember about the Lab: egg Latin, brain teasers, baby pictures, cold fusion and the differences between American and Canadian seatbelt politics.

Also a round of thanks should go to the people who made time outside the lab far more interesting, in particular Carlin, Karl, Rob, Dana, Kirsten and Charlotte.

Finally a heartfelt thanks to Paige for all the proof reading, and all the special things she did to make my life both easier and better.

This research was funded by the SANDIA National Laboratories under contract number AI-6484

Table of Contents

List of Figures

List of Tables

Nomenclature

ε	Permittivity [V/C]
ρ	Charge density [C/m^3]
ϕ	Potential [V]
Δt_m	Simulation time step [s]
$\bar{E}$	Electric Field [V/m]
Z	Ionization number
e	Charge of an electron [1.6e-19 C]
n_e	Electron number density [1/m^3]
n_i	Ion number density [1/m^3]
$\bar{r}_0^1$	Position of 0 with respect to frame 1
R_{12}	Rotation matrix from frame 2 to frame 1

CEM	Computational Electromagnetics
CEX	Charge Exchange
CFD	Computational Fluid Dynamics
CHAWS	Charging Hazards And Wake Shield
DSMC	Direct Simulation Monte Carlo
EP	Electric Propulsion
LEO	Low Earth Orbit
MD	Molecular Dynamics
PIC	Particle-In-Cell
PM	Particle Mesh
PP	Particle Particle
PPPM or P^{3}M	Particle Particle-Particle Mesh
WSF	Wake Shield Facility

CHAPTER 1

Introduction

Rapid advances in computational power, both at the workstation level and at the massively parallel level, have allowed such fields as plasma physics and non equilibrium chemistry to perform increasingly large particle based simulations. With the growing size and complexity of these and other particle based simulations, the need for scientific visualization is increasing. Although there have been recent advances in scientific visualization methods and tools, none have focused on particle based simulations.

The lack of visualization capability has traditionally been dealt with in one of two ways. The first method was to convert the particulate data into a form recognized by the more traditional scientific visualization packages, and the second, was to modify or customize an existing graphics package to handle the particulate data. These solutions may have been adequate in the past but are now no longer sufficient.

The new package Particle3 was developed to meet the growing need for a particle based visualization tool. It provides an interactive graphics environment specifically designed for the visualization of multidimensional particle based data.

1.1 Particle Modeling Methods

Particle based simulations are used to model phenomena in which large numbers of discrete lumps or quanta of matter interact. The models accomplish this by creating a system of particles and then tracking these particles, under the influences of internal and external forces, through space and time. Forces are generally classified into two categories, long range or field induced forces and short range or collisional forces. Here collision is assumed to mean any general short range interaction which alters the motion of the involved particles. Although some of the computational techniques used in particle simulations are also used to model continuous phenomena, the fact that these simulations track large quantities of particles, each with their own attributes, distinguishes them from models of purely

continuous phenomena.

Traditional continuum modeling techniques can be broadly classified as either finite element or finite difference techniques, and involve grids or meshes to store the representation of one or several continuous fields. Continuous fields must be discretized in order for a digital computer to be able to store and manipulate them. These field techniques are used in Computational Fluid Dynamics (CFD), Computational Electromagnetics (CEM), structural stress and strain studies or anywhere a continuous field is modeled.

Particle simulations are used when the phenomena to be simulated is inherently not continuous. This traditionally occurs at very small length scales such as modeling a gas at a molecular level, however, particle simulations can be used at widely different time and length scales as summarized in Table 1-1 [12].

Table 1-1 Examples of physical systems represented by particle models

Example	Computer particles	L (meters)	T (seconds)
Covalent liquids	Atoms or molecules	10^{-9} to 10^{-8}	10^{-12}
Galaxy clusters	Galaxies	10^{23}	10^{17}
Collisionless plasmas	"superparticle" of approx. 10^{8} electrons or ions	10^{-4} to 1	10^{-9} to 10^{-12}
Galaxy spiral structures	"superparticle" of approx. 10^{6} stars	10^{21}	10^{13}
Inviscid fluid (vortex modeling)	vortex element	10^{-3} to 10^{6}	10^{-3} to 10^{5}
Semiconductor devices	"superparticle" of approx. 10^{4} electrons	10^{-6}	10^{-9}

There are common features which apply to all classes of particle simulations. Each particle has its own position coordinates and velocity components plus other attributes, all of which are parameters which affect and/or may be affected by the simulation. Particle solutions are inherently time dependent, they track the evolution of the system through the duration of the simulation. A third feature is that particle simulations separate particle motion from particle interaction. With the small exception of some

Molecular Dynamics (MD) codes, which alter the time step to follow individual collisions, most particle simulations use, at a minimum, locally constant time integration intervals. Since these simulations decouple motion from interaction, most of the variation in simulation techniques occurs in how particles interact. Based on this, particle simulations can be classified into two categories, deterministic and non-deterministic. Non-deterministic or stochastic simulations use particle interaction rules which are determined from probabilities which may depend on various parameters of the simulation. Discrete Simulation Monte Carlo (DSMC) is an example of a stochastic simulation technique. Deterministic simulations model interactions, such as collisions, with explicit functions that define the post collision state of the colliding particles as a function of the pre collision state. This means that under identical initial conditions, any number of collisions will yield identical results. A deterministic simulation will reproduce identical results, regardless of the number of times run. Although the deterministic techniques are generally regarded as more accurate, they depend strongly on the accuracy of the interaction models used. A Particle In Cell (PIC) code is an example of a deterministic simulation. In addition there are hybrid techniques which may use deterministic interaction techniques for long range interactions and stochastic models for close range collisions, for example, but these are rare at this point in time.

How the two types of simulations differ is best illustrated through example. To this end the flow of both a DSMC and a PIC simulation will be described.

DSMC, is a non-deterministic particle based simulation technique which is used to model flows where collisional (short range) interactions dominate. It is used to simulate rarefied gas flows and non-equilibrium chemical reactions such as spacecraft re-entry and rocket plumes. It has even recently been extended to model continuum fluid flow, although at very small length scales, which is an area that has traditionally been the realm of the continuous Navier-Stokes differential equations [6], a continuum fluid model. DSMC simplifies the collision modeling, as compared to deterministic methods, by using statistical approximations for the probability that a particle will interact with one of the surrounding particles. The decoupling of the particle motion from the particle interactions is readily apparent in the flow of the steps in a typical DSMC simulation as described below.

1. Initialize the simulation. This includes loading all the particles, setting all their initial parameters, and generating a mesh for the collision modeling.

2. Move the particles. Positions are updated according to their velocity and the simulation time step Δt_m.

3. Boundary accounting. Particle collisions with the domain boundaries and/or particle fluxes at the domain boundaries are dealt with appropriately.

4. Collision partner selection. Likely collision candidates, within individual grid cells, are identified.

5. Collision processing. Using probabilities, which are functions of such parameters as relative velocities and particle collision cross section, the subset of particles that undergo collision are determined.

6. Data accounting. The velocities, energies and other appropriate parameters of these colliding particles are updated.

7. Repeat steps 2-6 until simulation is complete.

Also, at a user defined regular time interval, usually on the order of 100 Δt_m, the flow field is sampled in order to get time history data about the simulation. The sampling may be direct, through particulate data, or indirect, by calculating one or several mesh based field parameters that are determined by interpolating the particle attribute data onto the grid. The above described flow represents the simulation of a non reacting flow. If chemical reactions were to be included in the model, steps 4,5 and 6 would include 'reaction' everywhere the word 'collision' was mentioned.

The PIC technique is a deterministic simulation method that is traditionally used to simulate plasmas that are non collisional. The PIC technique is also referred to as a Particle Mesh (PM) technique [12] because it uses a continuous simulation type mesh. The mesh is used to determine and store the potential and force fields, which are used to calculate the forces acting on the particles and are, in turn, determined from the particles position and other attributes. This technique is preferred to the Particle-Particle (PP) technique [12], which tracks all interactive forces individually, because it is less computationally expensive. The disadvantage of PM

simulations is that they are less accurate over short ranges because the process of calculating the continuous potential field loses short range resolution. The limit of resolution of the particle positions is no better than the size of the smallest grid cell. This is not a problem in plasma simulations if the grid cells are on the order of a Debye length in size. The Debye length represents the minimum length over which a plasma can exhibit collective behavior, and this collective behavior is usually what the simulation is attempting to observe. A standard PIC code simulation used for plasma simulation would run through the following steps:

1. Initialize the simulation. This includes loading all the particles, setting all their initial parameters, and generating a mesh for the field data modeling.

2. Potential Solver. The particles are allocated to the grid in order to approximate the charge density distribution. A field solver is then used to determine the electrostatic potential from the Poisson equation:

$$\nabla^2\phi = \frac{\rho}{\varepsilon_0} = \frac{e}{\varepsilon_0}(Zn_i - n_e) \tag{1.1}$$

where ϕ is the potential, ρ is the charge density, ε is the permittivity, Z is the ionization number and n represents the number density of either the ions or the electrons.

3. Electric Field Solver. The electric field is determined from the gradient of the potential $\vec{E} = -\vec{\nabla}\phi$.

4. Particle update. The particle velocities are updated from the forces applied by the electric field. The field values at the particle locations are interpolated from the nodes of the mesh. The particle positions are then determined from these velocities and the simulation time step.

5. Boundary accounting. Particle collisions with the domain boundaries and/or particle fluxes at the domain boundaries are dealt with appropriately.

6. Steps 2-5 are repeated as often as necessary.

As with the DSMC simulation, there will also be periodic samplings of the data in order to see the progress of the simulation. The above described process is used to model simple plasmas with no magnetic field and no collisions. These effects could both be included, at the expense of increased complexity in the simulation.

In order to simulate deterministic phenomena in which there is no analogy to the Debye length, one must resort to a PP method which tracks all particle interactions regardless of inter particle range. If the number of particles is large a third type of simulation can be used, which is a hybrid of the PP and the PM technique. It is called Particle Particle-Particle Mesh, PPPM or P^3M technique. In simplest terms it uses PM technique to model long range forces (those farther than a grid cell away) and then uses PP techniques to simulate the particle interactions within a cell [12].

State of the art simulations, using DSMC and PIC methods, currently use between 10^6 and 10^8 particles. In the future this is expected to become even larger. Analysis of data quantities of this order will be very time consuming and inefficient if a new approach is not found soon.

1.2 Scientific Visualization

Scientific Visualization is concerned with the processing and display of large amounts of data in a way that allows presentation and analysis in an efficient and effective manner. It is a field which combines the disciplines of computer graphics, image processing, data management and perceptual psychology [19]. Scientific visualization can be interpreted as applying a sequence of transformations which convert a data set into a displayable image. There are typically three transformations in this process, the first is data manipulation, the second is visualization mapping and the third is rendering. Data manipulation consists of interpolation and operations that convert the data into a form that is suitable for the mapping phase. In the visualization mapping phase the data is converted into visualization parameters such as color, texture, and shape. The final phase involves taking the abstract visualization created in phase two and rendering it on the screen. This is where transparency and overlap issues are resolved. In general the first two phases differ the most between different applications and the third phase only differs for different hardware platforms.

Although little effort was spent on scientific visualization in the past, recent advances in computer and graphics technology combined with rapidly growing multidimensional data sets have brought together the means and the need to advance this field. Although computer rendered and printed graphics have existed for decades, true interactive visualization tools have

appeared only recently. One of the first recognized visualization programs was Plot3D, which was developed at NASA Ames [13]. Plot3D was a package designed to visualize 3D data from CFD and to allow the user to examine different aspects of their data. To date most of the development in scientific visualization has occurred in the area of flow visualization. This was in part due to strong interest from the CFD community, some computer company collaboration and the historic tendency for fluid dynamics to involve visualization techniques. Flow visualization has a certain paradigm which makes it a distinct subset of scientific visualization. The fundamental assumption is that a package will be used to visualize data with two and three spatial dimensions and that the data will be associated with a mesh or grid. The data can be either scalar or vector valued. Since the data is based on physical phenomena some of the tools appear to be computerized versions of earlier techniques used in experimental flow visualization such as tufts, Schlieren photography and smoke filaments [21]. These assumptions, which are apparent in many flow visualization tools such as FAST, Ensight, VISUAL3 and Fieldview, make these packages applicable to a broader range of uses [11,22]. Generally flow visualization tools can be used to visualize data from other mesh based simulations such as structural and electromagnetic calculations.

Beyond flow visualization and other application specific visualization tools, several packages have been designed to be general, all purpose visualization tools. Two examples are Ensight and IDL. These packages allow users to create customized visualization environments to meet their own data visualization needs. There are advantages and disadvantages to using these sorts of packages. Broad general packages tend to be able to do many things, but they tend to be complex and have a steep learning curve. In general, the broader the application, the greater the time it takes to become proficient in its use.

In addition to visualization packages, there are also some packages which provide both graphical and analytical tools which can be linked together to create customized data analysis tools. Although the definition of scientific visualization includes analysis, visualization packages tend to use qualitative rather than quantitative analysis tools, and these tools are generally optimized for speed of visualization rather than depth of numerical analysis. Packages which focus primarily on the quantitative aspects, like IDL,

are not as well suited to render large amounts of multidimensional data in its original form.

Particle based simulations are part of the broader set of increasingly multidimensional data. A literature search revealed that as yet there are no tools dedicated to the visualization of particulate data, although examples of particle based simulation and visualization were found [14,15]. Also, both Birdsall and Langdon [2] and Hockney and Eastwood [12] include sections on the use of data visualization as both an analysis and a diagnostic tool. From these visualization sections it is apparent that there is a large amount of data to be analyzed and a mass of printouts would be required to examine everything to the detail specified therein. The described approaches give increased credence to the perceived need for a proper scientific visualization application for particulates. Trends seen in computer power, and simulation size and comments from numerous visualization based papers [1,3,4,19] also confirm the stated need for a more advanced and effective particle visualization method. Ultimately, a visualization system should be able to answer the following questions for the user: where are the features of interest in the data, and what in the system creates these features? Particle3 was designed to assist the user in answering these two questions.

1.3 Scope

This thesis will summarize the development of a scientific visualization package for particulate data, called Particle3. The thesis will describe the flow visualization package VISUAL3 and why it was chosen to be the background for Particle3. The next section will cover the functional requirements and how they are split amongst the architecture and the different tools. The implementation of the architecture and the tools will be discussed, followed by a demonstration of Particle3 as applied to two different space plasma simulations. Finally future additions to Particle3 will be discussed, followed by the conclusion.

CHAPTER 2

VISUAL3

Particle3 is designed to be an expansion module for an existing CFD visualization package called VISUAL3. The benefit to Particle3, in using VISUAL3 as the baseline, was that efforts were focused on the development of new and innovative tools. VISUAL3 is a collection of high level graphics routines that create a visualization environment for discretized vector and scalar data fields which can be either steady or unsteady in time. In addition to providing a collection of standard tools, VISUAL3 provides many access points to allow a user to customize their visualization or create their own tools. Since it is the basis for Particle3, VISUAL3's capabilities and structure will be described in more detail.

2.1 Implementation

VISUAL3 was designed to be a fast, interactive, scientific visualization package for the visualization of three dimensional unsteady or steady vector and scalar field data. Although it was designed primarily for flow visualization, its architecture was kept as general as possible in order to be useful for data from a non CFD source. In keeping with the three dimensional nature of the data, VISUAL3 includes a number of tools which allow the user to efficiently scan through large amounts of data in order to focus in on interesting features in the data set being visualized. When these tools were being developed, they were designed to conform with the two principle ideals of VISUAL3. These were that the visualization shall be interactive, thus minimizing the users idle time during a visualization session, and that the visual representations shall be as true to the data as possible, this is especially important when VISUAL3 is used as a simulation debugging tool.

VISUAL3 is implemented as a scientific visualization environment with a library of functional tools. In order to use VISUAL3 a user creates a code in "C" or FORTRAN which calls the VISUAL3 initialization routine.

VISUAL3 then takes control and creates the graphics environment, which consist of several different windows. The grid coordinates, scalar fields, vector fields, and other data are loaded in through user defined routines called from within the programming structure set up by VISUAL3. VISUAL3 maintains its generality by specifying only the interface to these routines and allowing the user to write the routine through which VISUAL3 can access their data. The VISUAL3 environment is interactive, it allows the user to examine different aspects of his data while the program is running. The user can control the state of the visualization environment with a mouse, the keyboard, or a dialbox (an input device with 8 or 9 dials on it). These inputs are called X-events since the VISUAL3 environment uses X-windows, a UNIX window managing system, in order to manipulate the different windows on the screen. The X-event inputs are processed and appropriate changes to internal state are made and the new window views are rendered, if necessary. This process is shown in Figure 2.1. The loop continues until the user causes the program to stop, which returns control to the original code which initialized VISUAL3. [11].

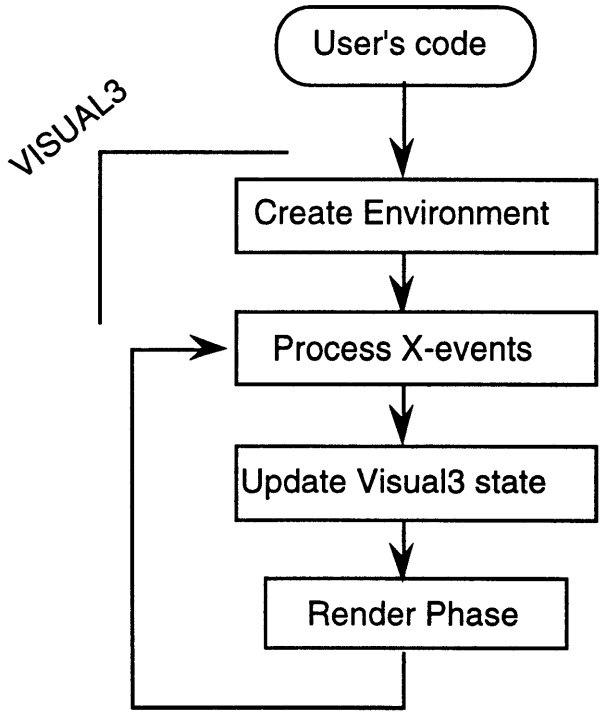


Figure 2.1. VISUAL Process Flow Diagram.

2.2 Environment

The VISUAL3 environment consists of 6 windows, as shown in Figure 2.2, each window with a distinct purpose. There are three viewing windows, these three windows are called the 3D, 2D and 1D window respectively and are named after the type of data they can contain. The 3D window shows the simulation domain in fully three dimensions using shading, perspective and other techniques to achieve that end. The 2D window is used for mappings and surface variations from the 3D window and the 1D window is used for single parameter plots such as the variation along an edge or a streamline. The other three windows are the key window, the text window, and the dialbox window.

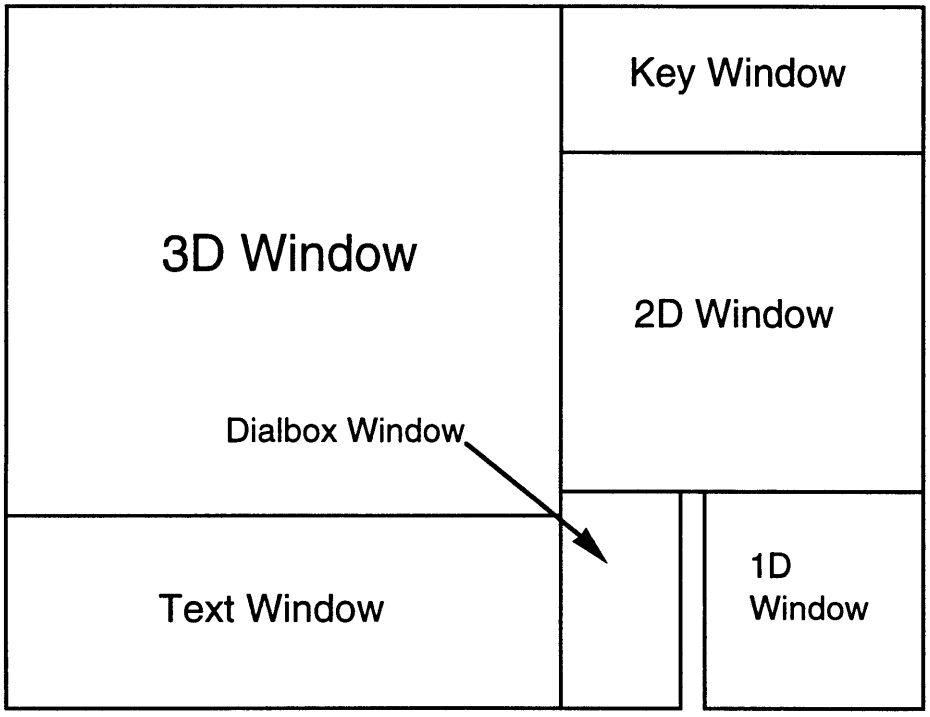


Figure 2.2. VISUAL3 Environment Screen.

The key window shows the color scale used in the 3D and 2D windows and current values of status parameters such as the current scalar field, the color scale limits and, in the case of an unsteady visualization, the current time step. The text window is a window for displaying text messages and receiving text inputs. The dialbox window has two purposes. The first is to represent a virtual dialbox. A dialbox is an input device which uses dials to allow the user to interactively fine control a number of parameters by

adjusting one of several dials. Through the dials the user can change the visualization state parameters such as the view angle, the coloring scheme or the magnification of objects. The second purpose of this window is to control the surface database where the user can control the state of the static and domain surfaces. VISUAL3's use of surfaces is described below in § 2.3

2.3 Tools

VISUAL3 has a large set of tools that are best broken down according to the type of data that they manipulate. The two types of data that VISUAL3 can visualize are scalar field data and vector field data. Before the tools can be described the user must understand how VISUAL3 uses surfaces.

One of the limitations of visualizing three dimensional data is that the entire volume of data can not be viewed at once. Any technique that renders part of the data will obscure the image behind it. Behind and in front are all relative to the current viewing point, the location relative to the simulation domain where the visualization snapshot appears to be from. Computer graphics can currently only show the outer surface of a three dimensional block of data, which implies that three dimensional field data is best examined on two dimensional surfaces. Surfaces are integral to most VISUAL3 tools. Surfaces simplify data examination, and can be used as an aid to define individual points in a two dimensional projection of a three dimensional space. There are several classifications of surfaces in VISUAL3. The first category are domain surfaces, which together define the boundary of the data domain. Domain surfaces are specified when VISUAL3 is initialized. The second category are dynamic surfaces. Dynamic surfaces are those which are activated and controlled by the user, during the visualization session. The dynamic surfaces may move and change, hence their dynamic nature. The final category of surfaces are static surfaces which can not be moved or changed once they are set but are not located at a domain boundary. Static surfaces may be loaded in like domain surfaces, during the initialization phase of VISUAL3, or they may be generated when a dynamic surface is saved into memory. Static and domain surfaces may then be controlled from the surface database in the dialbox window. When a VISUAL3 session is ended the saved surfaces are lost.

2.3.1 Scalar Visualization tools

The following are the VISUAL3 tools which may be used to examine scalar field data. The associated figures are from a data set representing one channel in a turbine.

- Surface rendering.
  Gouraud shaded (smooth color shading) surface contours of the current scalar field can be rendered on any or all of the domain or static surfaces. The correlation between scalar value and color can be determined by looking at the color scale in the key window. Figure 2.3 shows the pressure variation on selected surfaces using this technique.

- Cutting plane
  This tool allows the user to position a square plane at any orientation inside the domain and then show the Gouraud shaded scalar values on the surface of that plane, and in the 2D window. The plane can be rotated about its normal axis and translated without having to re-initialize the plane tool, allowing the user to rapidly sweep through the domain to view the variation of the active scalar field over the path of the plane. Figure 2.4 shows a cutting plane slicing across the channel.

- Iso-surface
  An iso-surface is a surface on which a parameter is constant. In VISUAL3 the iso-surface tool will render the three dimensional iso-surface, where the current scalar field values are constant. VISUAL3 also allows the user to scan through values of the constant in order to see how the scalar field varies in space. Figure 2.5 represents four saved iso-surfaces near the channel exit.

- Thresholding
  The thresholding tool limits the range of values shown on a surface. If a scalar field quantity is outside the regime specified by the user it is simply not rendered on the thresholded surface. This can be used to reduce the clutter on a screen. Figure 2.6 demonstrates the effect of thresholding on domain surfaces.

- Probes

  There are a variety of probes that allow the user to examine the domain data in greater detail. The probes are usually represented in the 1D window.
  - line probe- the user specifies two endpoints in the 2D window and the variation of the scalar field along that line is shown in the 1D window.
  - edge probe- the user can see the variation of a scalar field along the edge of a surface.
  - point probe- the users points the mouse pointer to a specific point in the 2D window and the point's position, scalar and vector field value are written in the text window.
  - Streamline probe- plots the variation of the scalar field along the user selected streamline.

2.3.2 Vector Visualization tools

The following is a summary of the tools which are based on vector field data.

- Streamlines

  A streamline is a line that is everywhere tangent to the local vector field. Streamlines may be launched upstream or downstream (or both) from the cutting plane and are plotted in the 3D window. In addition, variations exist which provide additional information.
  - stream ribbons- show the local twist
  - stream tubes- show local cross flow divergence
  - bubble- the path that bubbles would take in an unsteady flow field

  Figures 2.7 and 2.8 show a number of streamlines and stream ribbons respectively.

- Tufts and Arrows

  Tufts and Arrows are tools that show the local vector field direction next to the cutting plane surface. Arrows originate at the local grid's intersection points with the plane, while tufts are evenly spaced over the plane's surface. Figures 2.9 and 2.10 show tufts and arrows originating from two different cutting planes. In addition to the difference in spacing, tufts have little x's at their base.

- Vector clouds
  An arrow is drawn at every node where vector data exists. These arrows show relative size and direction of the entire flow field. A subset of the arrows can be seen by activating the thresholding.

By using the feature identification tools provided and by using the hierarchy of rendering windows to close in on regions of interest, VISUAL3 enables users to efficiently interpret their data and uncover the underlying behaviors. With this basic understanding of VISUAL3 it became apparent that VISUAL3 was a good choice on which to base the Particle3 application. In addition to the simplification of the development process gained by basing Particle3 on VISUAL3, some of the latter's tools will complement the former, since particle simulations often involve both continuous and particulate components.

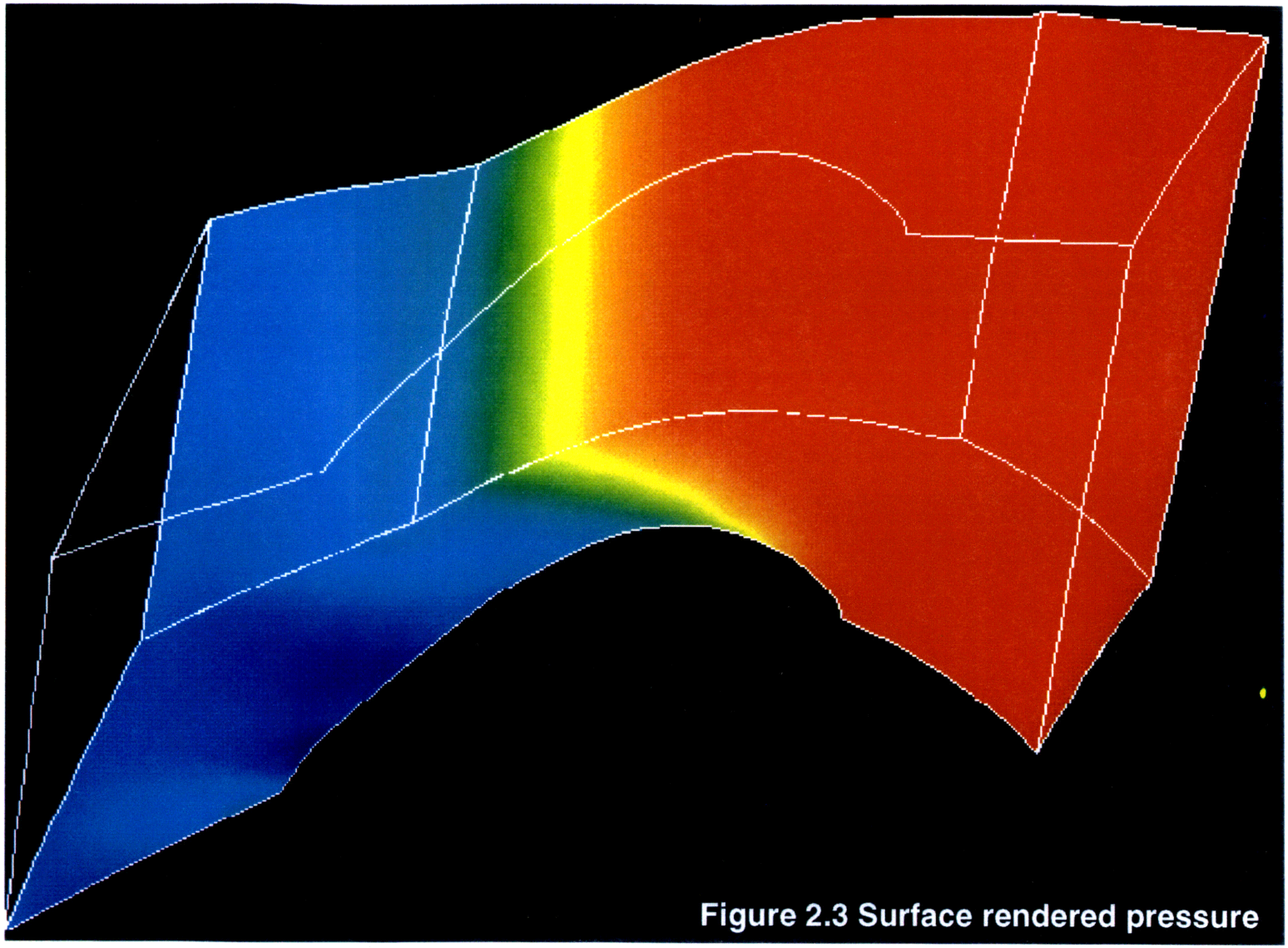


Figure 2.3 Surface rendered pressure

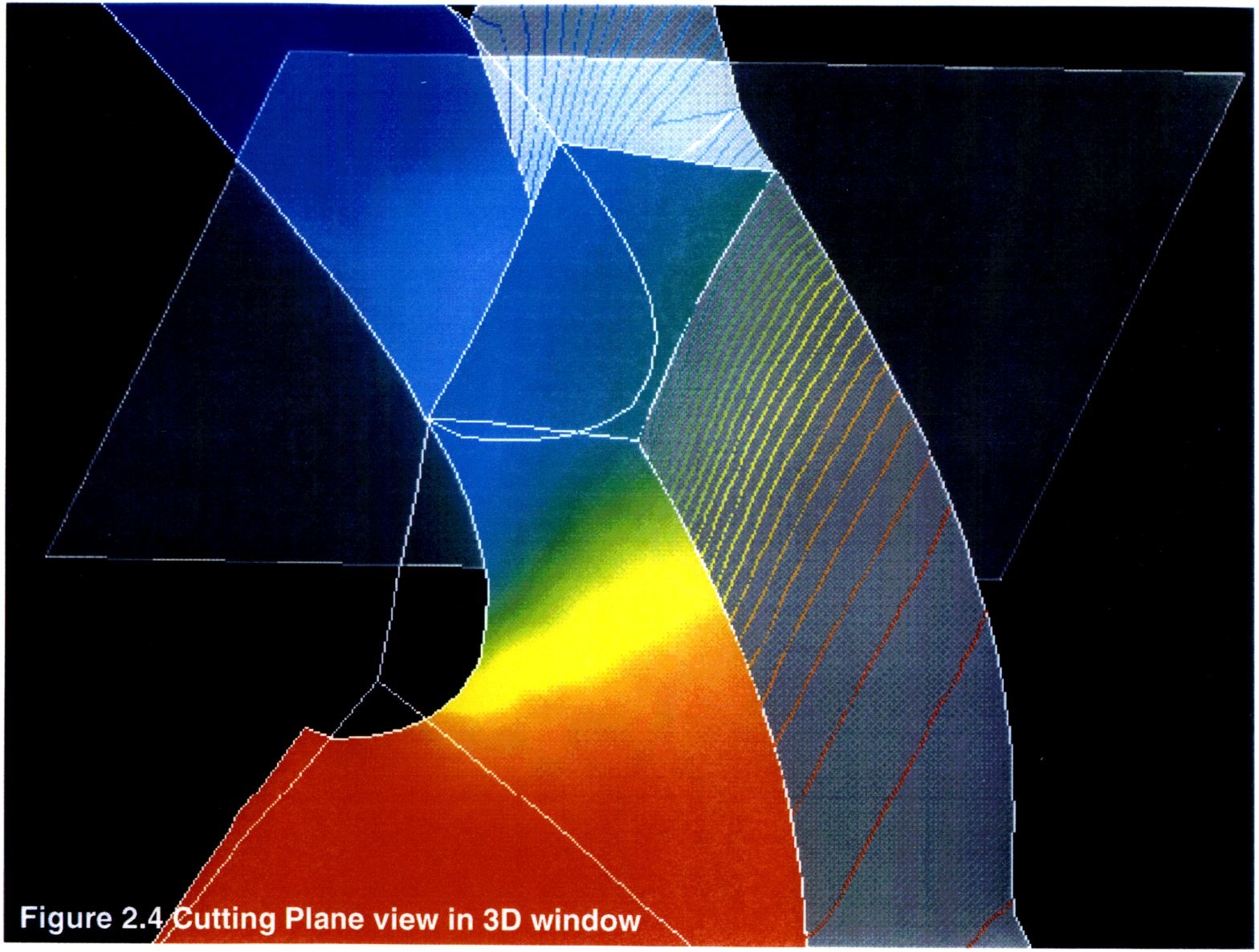


Figure 2.4 Cutting Plane view in 3D window

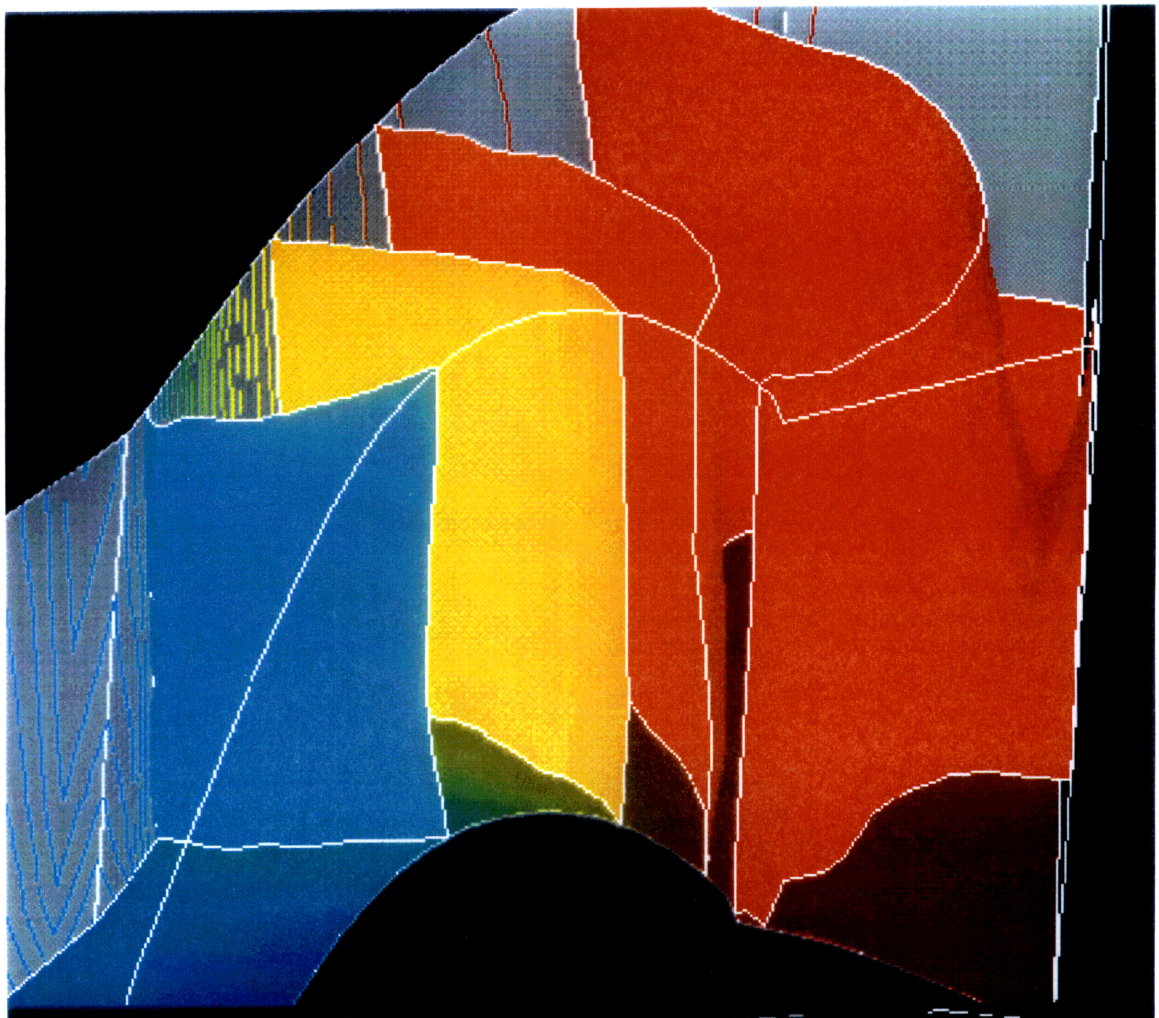


Figure 2.5 Iso–surfaces.

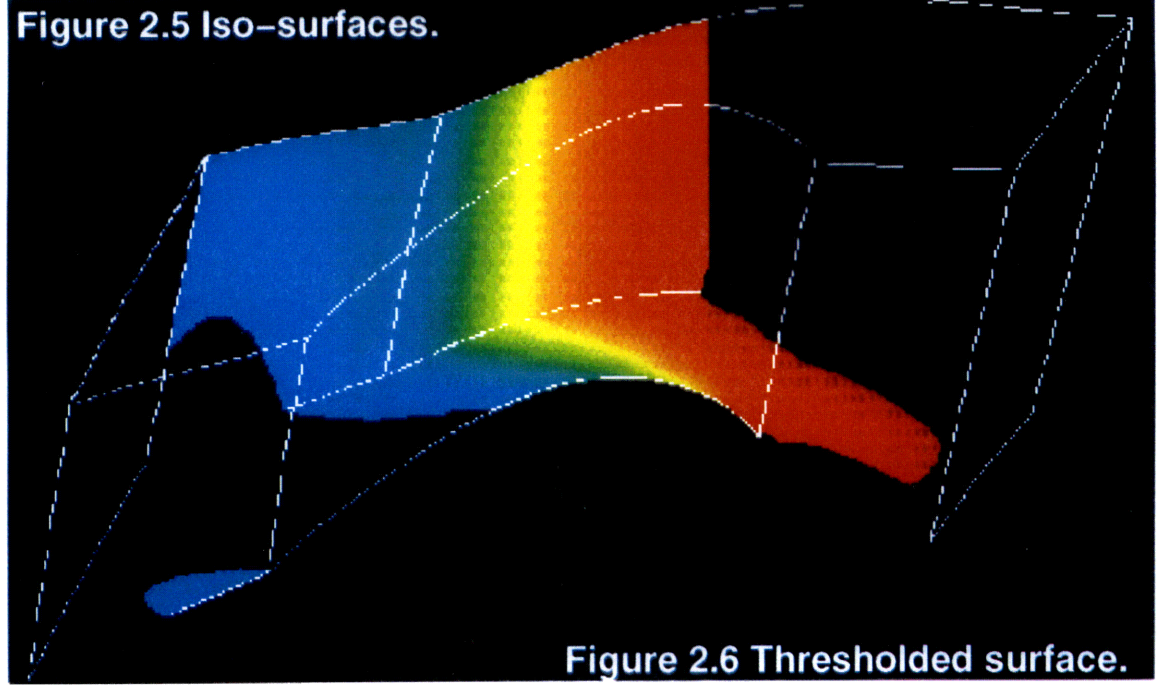


Figure 2.6 Thresholded surface.

Figure 2.7 Streamlines.

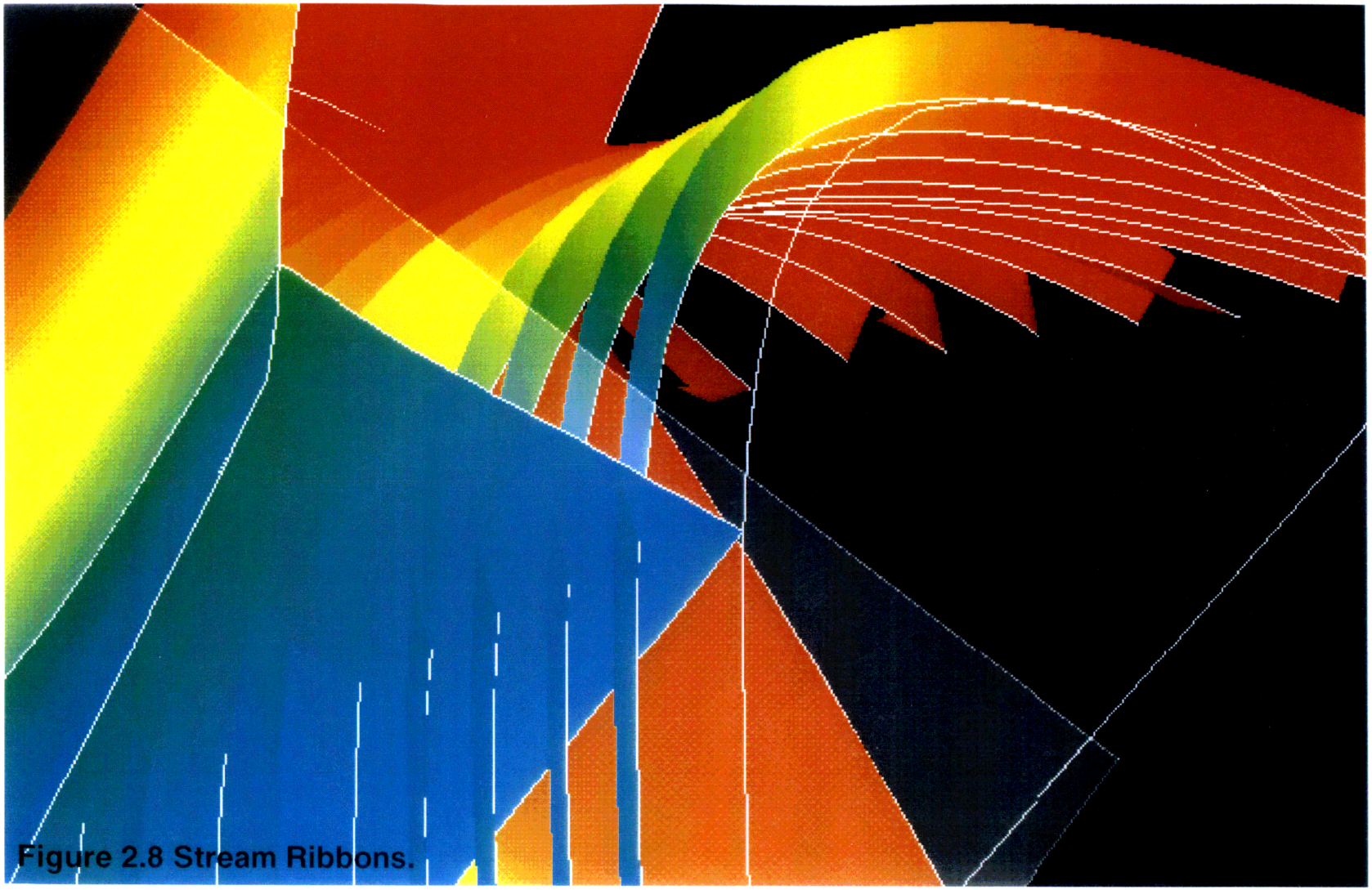


Figure 2.8 Stream Ribbons.

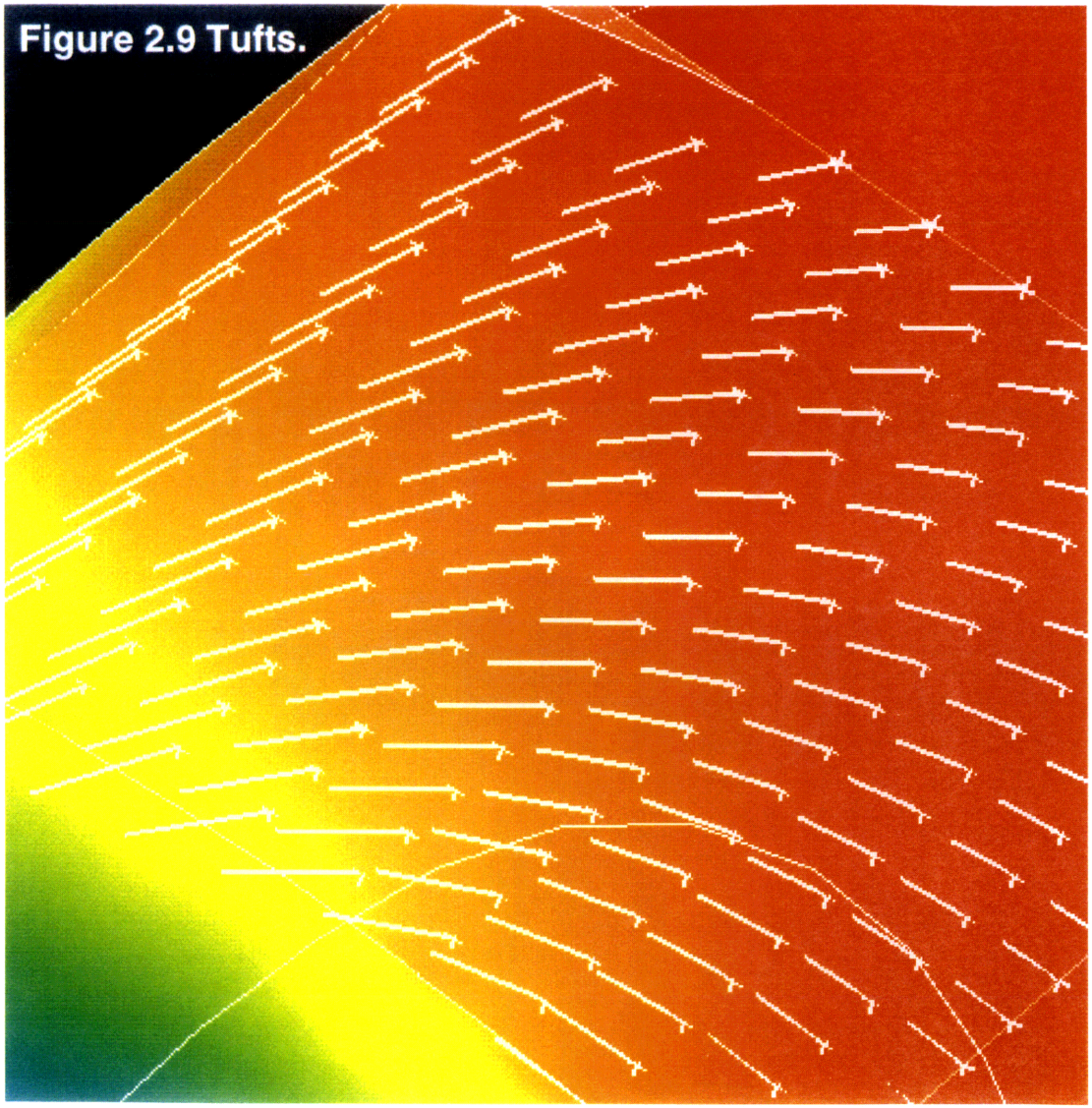


Figure 2.9 Tufts.

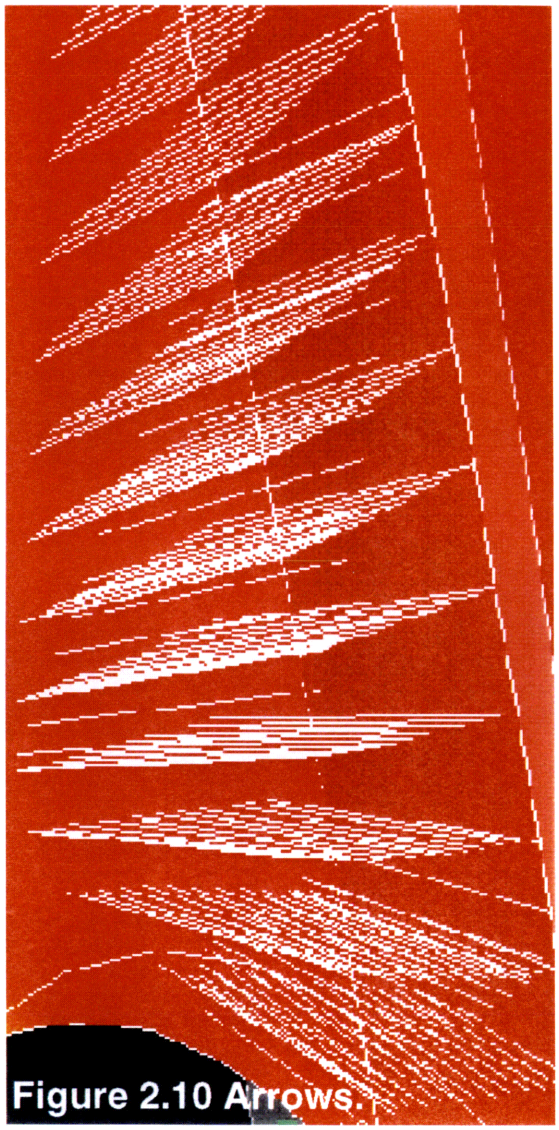


Figure 2.10 Arrows.

CHAPTER 3

Requirements

Particle3 was designed to be a particle data visualization tool. In order to simplify the design process a set of requirements were determined. These requirements were either specified at the project inception, or were derived from research into the sources of data and the criteria suggested by potential users. Once the requirements had been defined the architecture and the visualization tools were specified. The requirements for the data architecture were chosen in order to maintain the visualization architecture for the tools. The functional requirements of the tools were derived from the top level requirements. The set of functional requirements are then shown to be met. The only stated requirements given at the beginning of this project were that Particle3 was to be compatible with VISUAL3 and that it was to help visualize particulate data. The VISUAL3 requirement, though pre-specified, is easily justified. By basing Particle3 on an existing visualization package many benefits are gained. The program will be readily transportable to any hardware platform that VISUAL3 can operate on. Also the three dimensional rendering infrastructure is already in place allowing development time to be focused on new and innovative tools rather than on basic rendering software.

The top level requirements came from two sources, from research into the sources and character of particulate data, and from interviews with the prospective users of Particle3, the people who generate particle based data. The first step in the development of Particle3 was to determine how particulate data was generated, and what form it takes. Particle simulations are used in any field of science when discrete "lumps" of matter interact and move about. They occur frequently in the fields of rarefied gas dynamics, plasma physics, astrophysical motion, and non-equilibrium chemistry where individual lumps of reagent are tracked through chemical processes. Particle based data can also be generated in fluid dynamics (vortex element codes) or in instances where the length scales of matter are small enough that continuum based assumptions break down, in applications such as tracking toner inside a photocopier [14] or tracking photons. It is generally assumed that the particles are based on real material "pieces" of matter but the tools created for Particle3 can still be used to visualize more abstract particles,

although such uses may require more interpretation on the user's behalf. The search for different particulate data sources also demonstrated some of the current standard particle visualization techniques.

Another good source of potential requirements came from several interviews with people involved in particle simulations. These interviews resulted in a relatively clear idea of what potential users of Particle3 would like to see in a visualization package. These comments, combined with the research mentioned above, became the basis of the top level requirements. These requirements are summarized in Table 3-1 below.

TABLE 3-1 Top Level Requirements

Number	Requirement
1	Ease of use
2	Interactivity
3	Phase space representations
4	Distribution functions
5	Data screening/selectivity
6	other new visualization techniques

During development, Particle3 was formally divided into two subsections. The architecture subsection and the tool subsection. The architecture contains the infrastructure which allows all the tools to operate smoothly and efficiently. The tools are the sub-applications which operate within the architecture and perform specific high level visualization functions. The functional requirements were divided between the architecture and the tool subsystems.

Since Particle3 was designed to visualize particles, the form the data representations take are mostly particulate or discrete in nature. Although there are a large variety of techniques that can convert discrete particle data into continuous data which can then be easily rendered on existing visualization packages, these techniques generally loose information in the data conversion process. Particle3 was designed to visualize the particulate data in as pure a form as possible. It strives to show the data without interpolation, smoothing or other computational modification in an attempt to maintain the accuracy and therefore not loose any potentially important

features of the data.

It was assumed that if the user desired to view their particulate data in both particle and continuous form, the user would create a continuous data field from the particulate data with smoothing scheme with user specific accuracy and render the continuous field on VISUAL3. In addition some minor functional overhead associated with the tools was included into the architecture because the requirements did not fall neatly into the tool requirements. This functionality had to do with control and storage of certain parameters which affected the data visualization

3.1 Architecture

The two requirements of ease of use and interactivity are largely determined by the architecture of the application. Particle3 was required to follow the paradigm set by VISUAL3, in order to simplify both the development and the later use by particle visualizers. The VISUAL3 paradigm included the memory allocation structure and the input processing from X-events. Ultimately Particle3 was required to provide an architecture which would store and manipulate the user's data as efficiently as possible. In order to create an optimal infrastructure the data organization was carefully laid out and studied. The following is a summary of what particulate data is comprised of and how it is organized within Particle3.

Particle based data, or particulate data, is quite different in structure from the more common continuous field data. The data is based on individual particles rather than spread around on grid nodes or cells. For reference the set of all particulate data shall be called the particle space. Each element in the particle space represents one particle, and has associated with it three Cartesian position components (x, y and z), three Cartesian velocity components (v_x, v_y and v_z) and possibly other attributes. The six dimensional space spanned by the position and velocity components is referred to as the phase space. The remaining data belongs to the particle parameter space and can be any particle based parameter that the potential Particle3 user deems of interest, such as particle energy, species or mass. Particle3 stores the phase space data of the current simulation. It also stores one parameter value per particle. The architecture allows the user to replace the parameter with another parameter during the visualization. The

parameter stored by the architecture is designated the active parameter and is used by the tools for a variety of different purposes. The current parameter can represent anything the users wishes, it may even be used to store an element of the phase space. The purpose of this will become clear in the next section.

3.2 Tools

Table 2 summarizes the final tool functionality of Particle3. Their function is described in corresponding sections below. The implementation of these tools are described in the next chapter and their actual use is described in [16].

Table 3-2 Particle3 Basic Tools

	Desired tools
1	2 Dimensional Phase Plots
2	3 Dimensional Phase Plots
3	Distinguish Particles by parameter values (coloring)
4	Particle Distribution Plots
5	Isolate particles spatially (projection plane)
6	Isolate particles parametrically (filtering)

3.2.1 Two Dimensional Phase Plots

A standard method used to examine the behavior of a complex system is to observe that system in phase space. By examining the system in phase space important dynamic characteristics are readily observable. A system of particles is no exception to this rule. Two dimensional phase plots for a system of particles take the form of a scatter plot. The screen position of each dot represents the value of the two selected phase space coordinates of the particle and the dot color is used to show the variation of the active parameter. Each axis can represent any one of the six different phase variables, as selected by the user. Figure 3.1 shows the appearance of a generic phase plot.

3.2.2 Three Dimensional Phase plots

Extending phase plots into three dimensions is a logical extension of the two dimensional phase plot tool. By plotting the particles in the 3D window of VISUAL3 the user can control all of the view rotate, zoom and translate parameters available in this window. Figure 3.2 shows a three dimensional particle phase plot.

The three dimensional phase plot in Particle3 is more appropriately called a positional plot, since it only shows the positions of the particles in phase space. Although the six different components of the phase space are effectively interchangeable there are limitations imposed by VISUAL3 to objects rendered in the 3D window. This is discussed more in § 4.2.2.

Due to the large number of particles used in simulations to ensure quality and resolution, the positional plot tool is most effective when it is used in conjunction with the selective rendering of user defined subsets. These subsets are selected by applying filters (refer § 3.2.4).

3.2.3 Coloring

The human eye is best at detecting changes in, and making comparisons through, spatial relations. Position and geometry are strong communicators of information. Using this technique by itself limits computer visualization to two degrees of freedom, perhaps three if perspective is used, with which to communicate information to the user. Fortunately the human eye can see and distinguish color. Although not as accurate as position, color is also a powerful communicator and can be used to convey another degree of freedom of information. All tools in the 3D and 2D window use color to represent the value of each particle's active parameter. On screen particles are drawn with the color derived from a linear color mapping of their active parameter.

Although this coloring is not a tool per se, it does have functionality and requirements. The selection of the active parameter is the responsibility of the architecture, but the mapping of a range of current parameter values to a color scale is given the status, and the corresponding responsibility, of a tool.

3.2.4 Projection Plane

The Projection plane tool allows the user to position and orient a square plane in space, then to project all the particles within a certain distance of the plane, to the plane's surface and then render these particles in the 2D window. The particles are projected down to the plane along lines normal to the surface of the plane. This, in effect, creates a rectangular box in space, with all the particles inside that box projected onto one side of that box. The projection plane tool is designed to work in conjunction with the cutting plane of VISUAL3. The user would position the cutting plane tool in the rendered domain and activate it, just as in VISUAL3. The user then activates the projection plane tool and Particle3 queries the user for a thickness. The thickness specifies the final dimension of the box, and a gray outline is drawn around the volume in the 3D window as a visual reference. Figure 3.3 shows the outline around the cutting plane, signifying that the projection plane tool is active.

Originally the projection plane's capture volume was defined as a certain thickness to one side of the cutting plane, later it was redefined as a certain distance on either side of the cutting plane. In the final version it was decided to make the volume defined on only one side of the cutting plane, in order to allow the user greater precision in placing the volume. If the user wishes to look at the other side of the plane, a negative height is entered when Particle3 queries for the volume thickness.

The main difference between the cutting plane and the projection plane arises from how the domain is sampled. The cutting plane is positioned, then VISUAL3 determines the field values at the intersection of the domain and the cutting plane and maps the data onto the 2D window. With particulate data at distinct locations, the chances of a particle exactly intersecting the plane are small even though many particles may be in the region near the plane. This is resolved by extending the intersection criteria to have a tolerance, effectively giving the plane a finite thickness and turning it into a capture volume. This tolerance can be changed by the user, which allows capture volumes of any size to be defined.

There is an additional advantage to basing the projection plane tool on VISUAL3's cutting plane. The projection plane can be swept through the visualization domain just like the cutting plane which permits large sections of the domain to be interrogated very rapidly.

3.2.5 Distribution Plot

The concept of a distribution function is used frequently when trying to describe a collection of particles. Distribution functions represent the relative likelihood that a particle has a particular value of position, velocity, energy or some other attribute. Although Particle3 has tried to use the raw particulate data as much as possible, some method of quantifying the entire set of particles in a single plot was needed. A histogram type distribution plot was well suited to achieve this end. Particle3's distribution tool shows a plot of number density distribution versus the active particle parameter. The plot is shown in the 1D window. The Particle3 distribution plot tool allows the user to quickly view the distribution of the entire set of particles, or the current particle sub set, if the filter tool is in use. Figure 3.4 shows a typical distribution plot from the 1D window

3.2.6 Filters

VISUAL3 allows users to threshold surfaces and other graphics objects in order to isolate areas of interest. Particle3 can perform the same function by filtering, which allows the user to select a specific particle subset based on a particular attribute. When a filter is applied, only the selected subset of particles appear on the screen. Although previously capable of filtering based on one parameter, the current version of the tool allows the user to select particles based on up to 10 independent bounding criteria. These criteria, or layers, can be applied and removed individually, or, all at once. In addition, the filtering may be applied in the 2D and 3D windows of Particle3 independently. The current version of the filter tool has two components. The first component organizes and maintains the filter structure and the second component handles the logistics of actually applying the filters to data in a user specified window.

A filter is constructed by applying layers, and each layer defines a subset of particles based on one particle parameter. A particle which is outside the boundaries that define the subset is 'tagged'. When the filter is activated, the particles which are tagged are ignored. The filter applies to the phase plots, the projection plane and the distribution plot. The filter also works with other filters in that the criteria applied have a cumulative affect. In order for a particle to be rendered it has to pass all of the filters. By judicious choice of parameters and layers almost any subset of particles can be selected. An example of filtering in the 3D window can be seen in Figure 3.6, where the low energy particles from Figure 3.2 are selected.

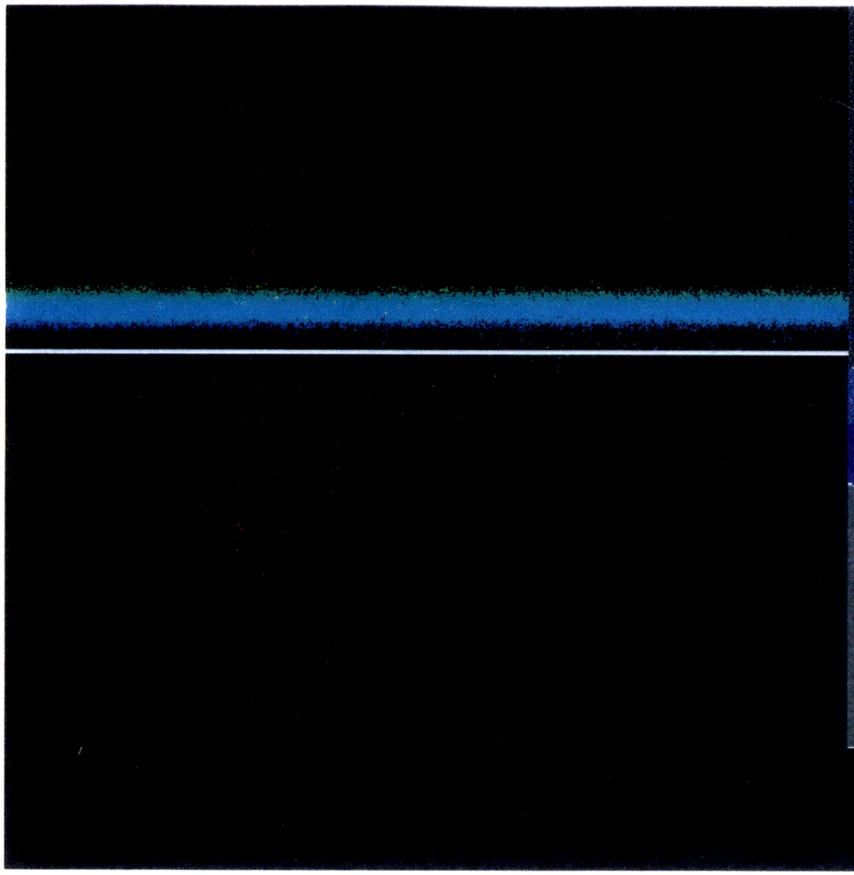


Figure 3.1 Two Dimensional Phase plot.

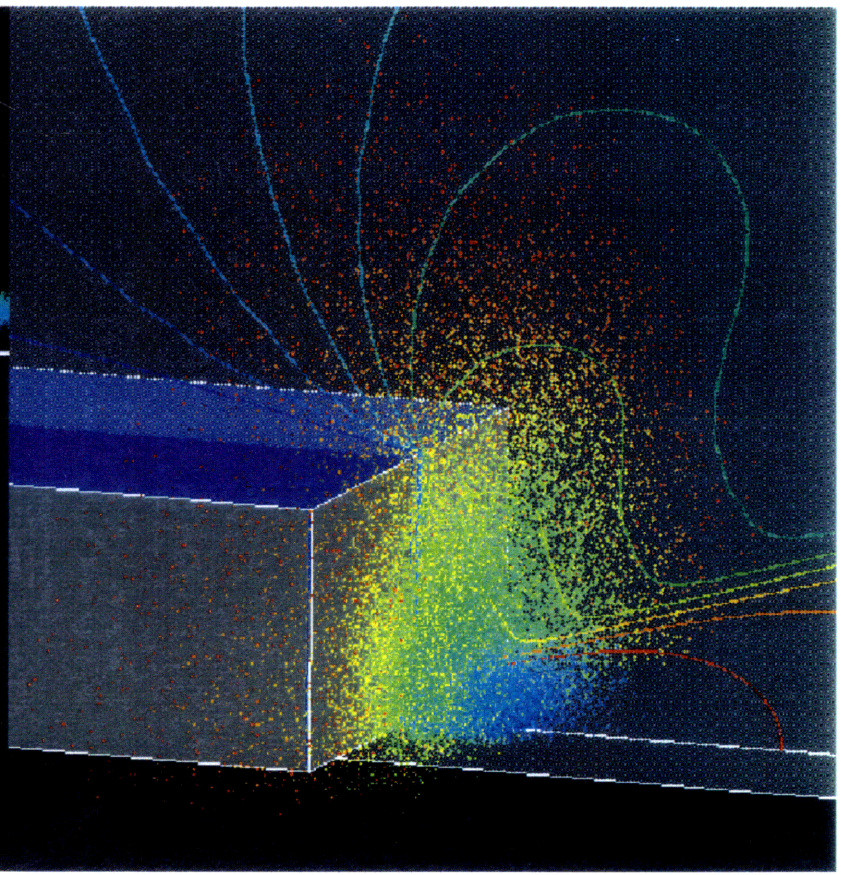


Figure 3.2 Three Dimensional Phase plot.

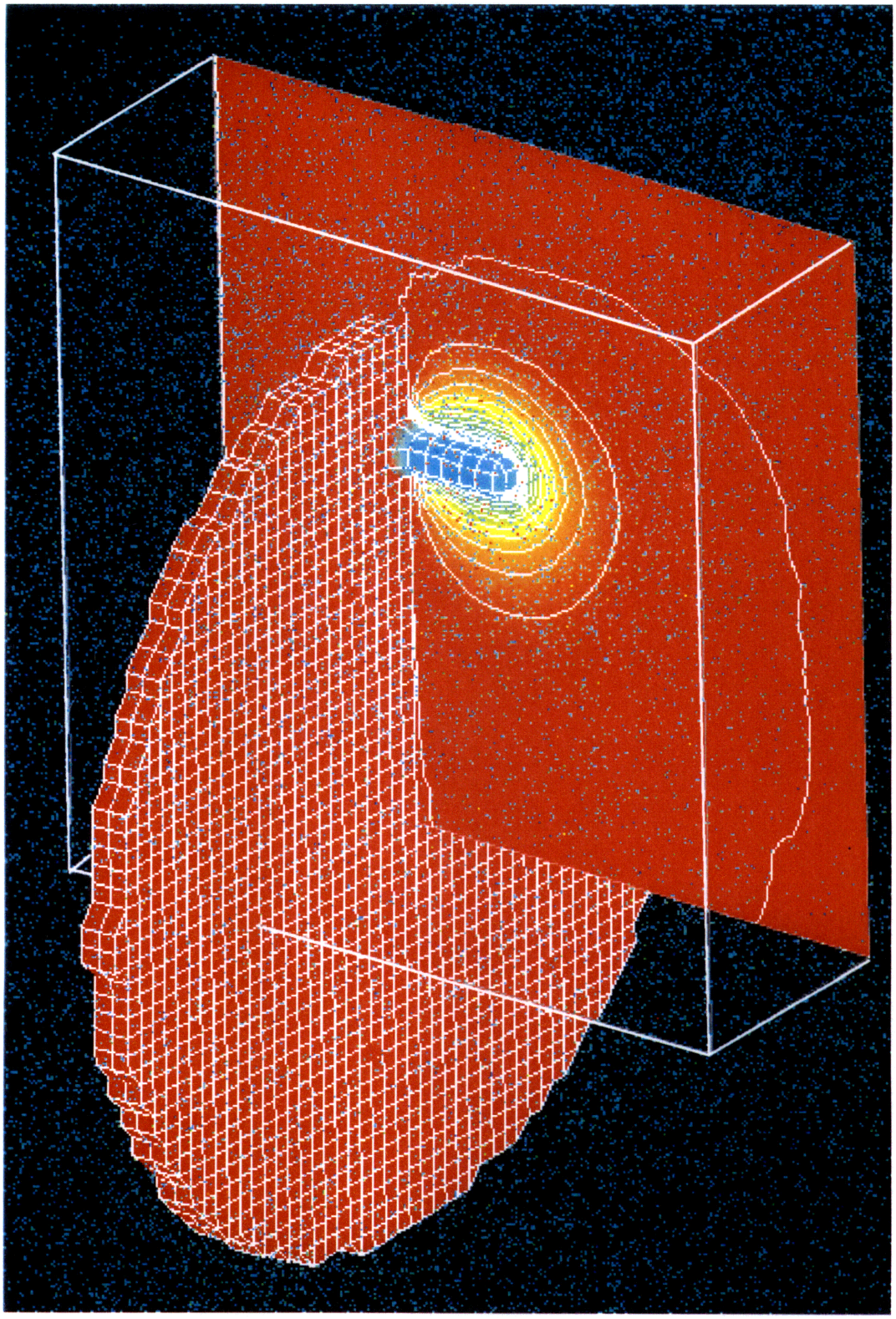


Figure 3.3 Projection Plane Volume in 3D window.

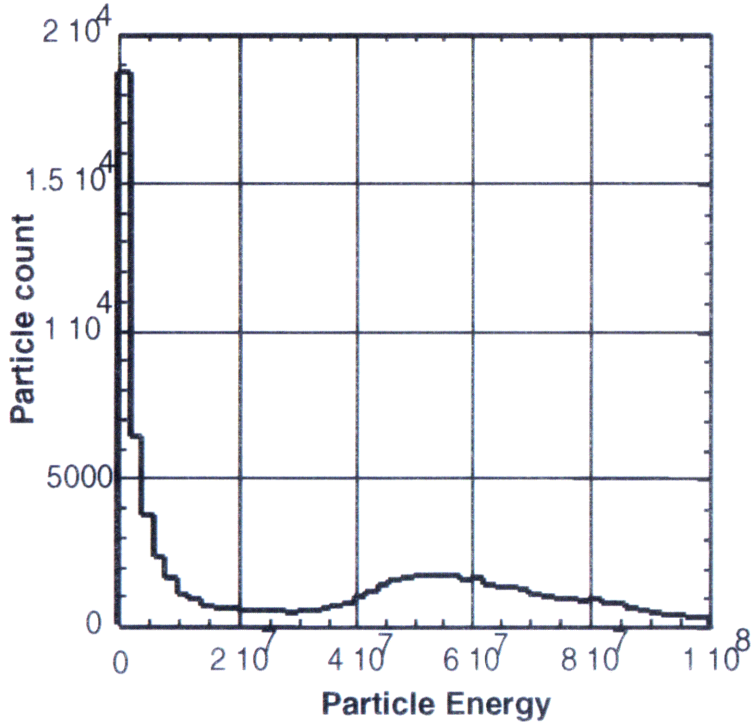


Figure 3.4 Distribution Plot.

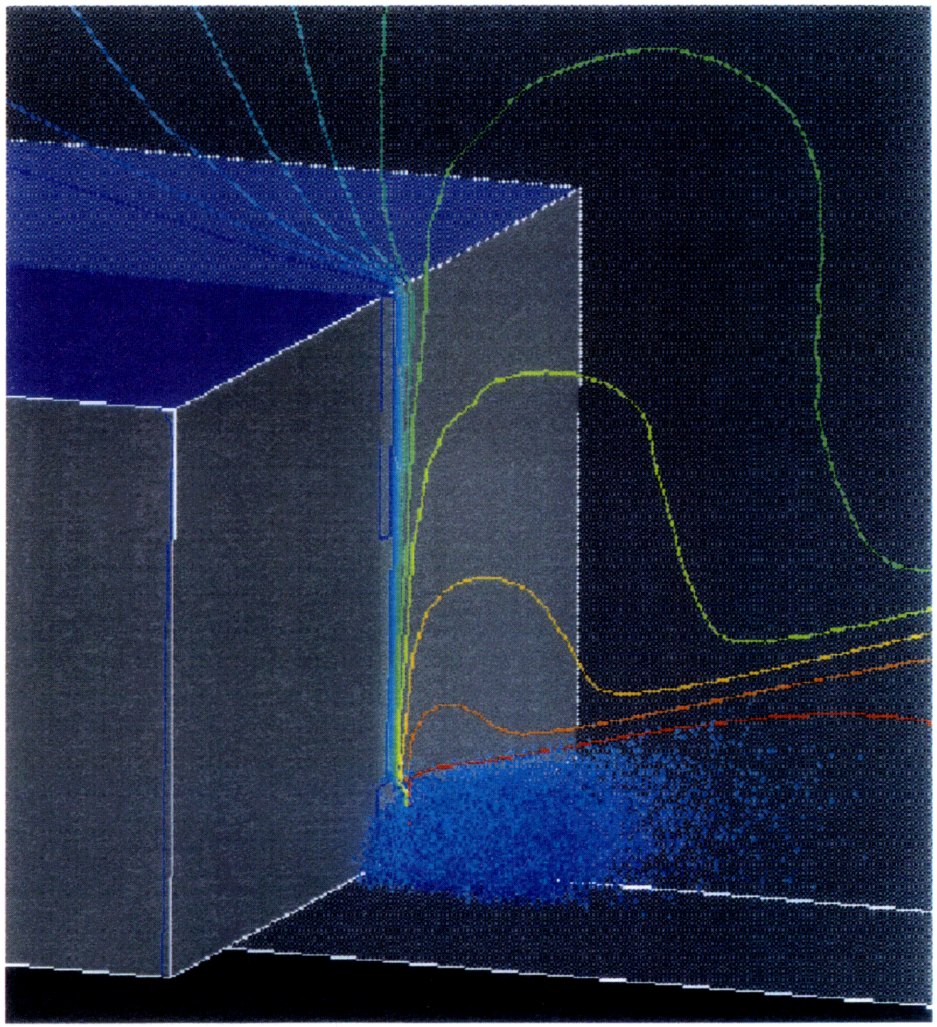


Figure 3.5 Filtered Three Dimensional Phase plot.

CHAPTER 4

Implementation

This chapter describes how the outlined requirements and tools are implemented in the Particle3 software. The descriptions are maintained at a conceptual level, without going into excessive detail. The goal of this section is to give the reader enough information to understand how the individual tools, and Particle3 as a whole, operate.

4.1 Architecture

Previously there were two spaces defined which were subsets of the particle space, the set of all particle data. These two spaces were the phase space and the parameter space. In Particle3, the parameter space data is stored and handled separately from, even though it can have elements which belong to, the phase space. The phase space data is used for the phase plots and the projection plane. The parameter space contains data for use in other visualization tools such as particle coloring or providing an axis for the distribution plot.
Another reason the two spaces are kept separate is because the parameter space can store integer or real valued data, while the phase space data is purely real valued. Examples of real number parameters are particle energy, charge or mass and examples of integer number data are particle species or type.

In addition to the two data spaces described above a third needs to be included, although it is transparent to the user. This space is called the threshold or filter space in Particle3. The filter space is required to maintain the filter layer information.

When Particle3 is initialized three blocks of memory are allocated. They are for the phase space, the active parameter space and the filter threshold space. If N is the maximum number of particles that will be used in the simulation the phase space array will have dimension 6 by N and the other two arrays have dimension N. By using integer and real variables with the same number of bytes each (4), it is possible to use the parameter space memory block for either real or integer parameter data. By only keeping the

active parameter resident in memory inside Particle3 the internal data handling is simplified and memory requirements are kept to a minimum.

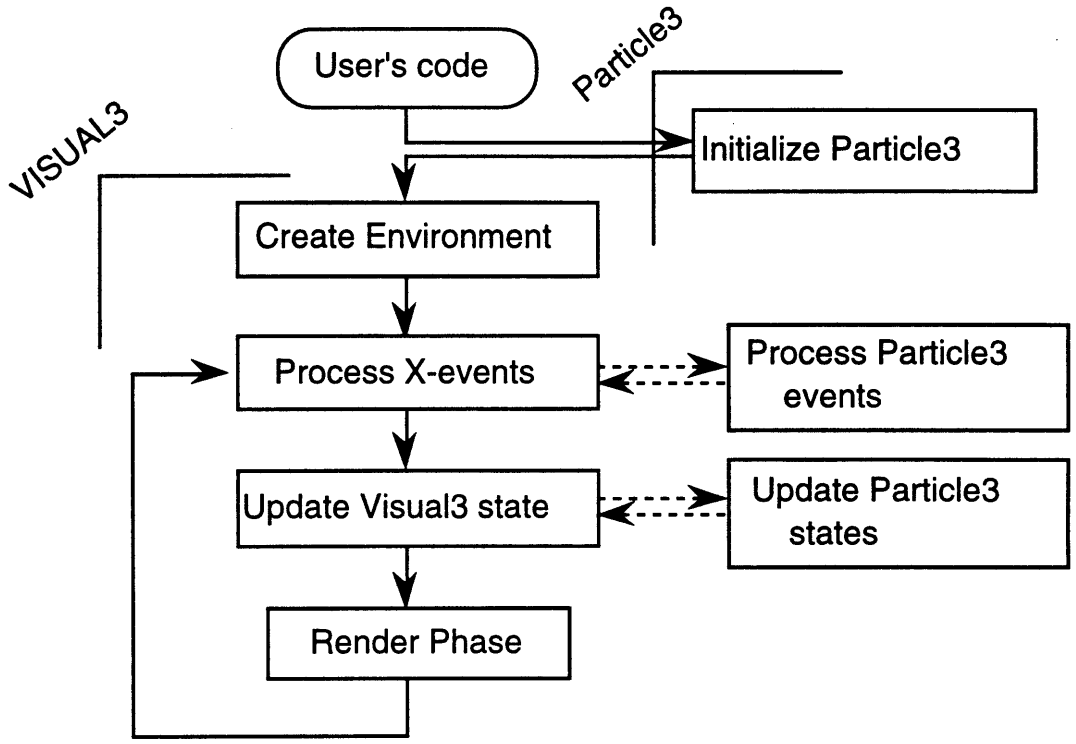


Figure 4.1. Particle3/VISUAL3 Interaction.

Another way of reducing memory, albeit slightly was to use 2 byte integers in the filter space. By using each bit in the 2 byte integer as a binary switch or flag the status of all ten layers can be tracked on a two byte variable. Figure 4.1 shows how Particle3 fits into the operation of the VISUAL3 environment. Before the user's code initializes VISUAL3, Particle3 gets initialized. Then when VISUAL3 processes the X-events, Particle3 also gets to process its events and make the necessary changes to Particle3's control variables.

At Particle3 initialization, one set of arguments that gets passed to the initialization subroutine is the number of different parameters that will be associated with each particle. With each parameter a key code (usually a letter) is defined that will cause that parameter to become active. The coloring map is always based on the active, or current, parameter.

In order to handle the large number of particles that are common in current simulations, only one parameter is actually stored by Particle3 at any one time. When a key corresponding to another parameter's key map is

pressed, a new parameter is activated and the previous one is removed from the parameter storage.

4.2 Tools

4.2.1 Phase plots

This tool produces a scatter plot in the 2D window. The user may select any combination of two of the six phase space variables to be plotted against each other. At startup Particle3 scans the phase space data and determines the minimum and maximum values of each variable. It also does this at each time step in unsteady visualizations. This information is used to normalize the plots to fit exactly into the square 2D window.

Unsteady phase plots will have shifting boundaries and axes. The shifting can be avoided but the alternatives are not considered improvements. The two alternatives are to fix the window normalization at the time step the tool was activated, or to determine the absolute limits for the phase variables over the entire time domain. The former method would allow the particles to drift out of the window, and the latter is inefficient and reduces Particle3's interactivity.

One of the only drawbacks to VISUAL3 is that it is unable to render text in the 2D and 3D window. When VISUAL3 was designed there was very little reason to place text in the 2D and 3D windows since text did not enhance any of the tools which used these windows. Unfortunately Particle3's phase plot tool could be improved through the use of text. Gauging the relative scale and positions of the particles in phase space could be more clear, and this is aggravated because the normalization of the data skews the aspect ratio of the plot. The axes, although specified by the user and drawn on the screen to provide some basic visual cues, have no distinguishing scaling or identifying information on them. An alternative which was investigated but dismissed, was to have some of the basic axis information written in the text window every time the tool was in use. The axis information was not implemented because it cluttered the text window and could not be made to stay in view if tools in other windows were in use. The implemented solution was to draw the visual references mentioned above in the phase plot for the user. By drawing two lines on the plot to represent the x and y axis, the user can identify the relative sign and magnitude of the particulate data. The lines

form a crosshair at the origin of the current phase plot. In the event that either zero axis should be drawn outside the 2D window Particle3 defaults to drawing that axis at the edge of the screen closest to the actual location of the axis. In this way the information provided by the axes is still partially available to the user.

No numerical values of the phase space variables can be rendered in the 2D window, but there is a method to get around this problem. Phase plot features seen in phase space can be correlated with information in the distribution plot to find the associated quantitative values.

4.2.2 Positional plots

Originally it was planned that the 3D phase plot tool would be as general as the 2D version. This would be fine except for a limitation placed on Particle3 by VISUAL3. Conceptually there is no problem with using any three of the six phase space coordinates in the phase plot, however, the infrastructure of VISUAL3, which performs the rotation, translation, shadowing and other data manipulation and rendering actions, is designed to operate in spatial coordinates. This limits the phase plot to three of the six possible degrees of freedom available. The process of creating the capability to manipulate and render six degree of freedom data, boundaries and other surfaces while maintaining compatibility with VISUAL3 would result in a substantially more complicated, visualization package completely distinct from VISUAL3, and was beyond the scope of this research.

It is currently possible to make the three dimensional plots render velocity data in three dimensions, but this requires familiarity with VISUAL3 and Particle3. It is accomplished by changing the indices of the phase space in the user defined data loading routine[16]. This is effectively a "hack" to get the system to plot the velocity components. This technique works. However, if the magnitudes of the velocities are much larger than the magnitudes of the position coordinates, the three dimensional phase plot will be very skewed and likely be difficult to interpret.

4.2.3 Coloring

One of the top level requirements of the Particle3 application was to be able to distinguish one particle from another, based upon some associated field. The simplest method to accomplish this is to color the particles

according to data available from the parameter space. In the phase plot, the position plot and the projection plane plot particles are colored according to a color map based on the current parameter. An upper and lower limit for the linear color scale mapping is stored for each parameter, by Particle3. The mapping is based on these limits and the color scale that is shown in the key window during the visualization session. Particles whose parameters are above or below the limits are colored as if they were at the upper and lower limit of the color scale. The color scale limits can be changed during the visualization session.

4.2.4 Projection Plane

The projection plane tool was designed to be an analogy to the VISUAL3 cutting plane. The cutting plane is used to determine the intersection of a plane and the visualization domain, and then map this intersection to the 2D window. In Particle3 the function of the projection plane is similar, but the mapping process is not. The process used in Particle3 to calculate the particle locations relative to the projection plane can be simplified to a translation and a rotation of reference frames.

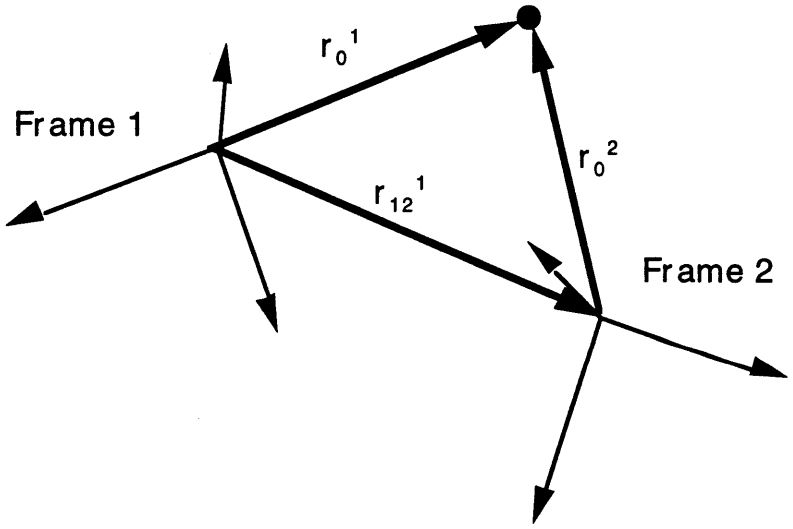


Figure 4.2. Particle Position in Two Reference Frames.

Once the cutting plane is positioned and the projection plane tool is activated, Particle3 queries the user for the cutoff height above the plane. Particle3 then collects the position coordinates of the corners of VISUAL3's cutting plane. This information is used to define the axes of a new reference frame with the origin of the frame at one corner of the cutting plane, the x' and y' (transformed) coordinates lying in the plane, and the z' coordinate

normal to it.

Coordinates in one frame can be transformed to another frame through translation and a rotation. Based on this and Figure 4.2 the position vector of any particle can be written as:

$$\begin{aligned} \vec{r}_0^{\,1} &= R_{12} r_0^2 + r_{12}^1 \\ \therefore \vec{r}_0^{\,2} &= R_{21}\left(r_0^1 - r_{12}^1\right) \end{aligned} \tag{4.1}$$

Where r_0^1 represents the position vector in frame 1 and R_{12} is the rotation matrix from frame 2 to frame 1.

Transforming the particle positions from the visualization frame to the plane's frame simplifies the subsequent calculations and comparisons. To determine which particles are rendered in the 2D window the particle positions are compared to the volume defined by the plane and the user defined height above the plane. If a particle is located within that volume it is rendered in the 2D window. The x' and y' coordinate of that particle are used to plot the particles in the 2D window.

When the height above the plane, or the volume thickness, is specified by the user it should be in the same units which the particle positions were defined in. By using the coordinates specified in each data set, the units are left under the user's control and generality is maintained.

4.2.5 Distribution plot

The distribution plot is a histogram of the number of particles spread out over an evenly segmented range of parameters. When activated, the tool takes the current parameter and it's current range of values, as specified by the coloring limits, and divides the range into a number of evenly spaced segments, called bins. It scans the set of particles, determines the number of particles in each bin and then plots the results as a histogram in the 1D window. The distribution plot tool is an explicitly quantitative tool because it can show numerical values along the axes of the plot. The 1D window in VISUAL3 was designed to present a large variety of single dimensional parameter plots, called probes. Information in one dimensional plots can be much more effectively shown through numbers than through color variations. Also the ability to write titles on the plots help the user know which probe they are currently using. This distinguishes the 1D window from the 3D and 2D windows since the latter can not show alphanumeric text. This is the only directly quantitative tool in Particle3. All other plots show a color interpretation of the parameter values rather than their actual values.

4.2.6 Filters

The Particle3 filter operates by layering several 'acceptance criteria' in form of bounds on specific parameters. In order for a particle to pass through the filter it must pass through each individual layer. At any one time a filter may be comprised of up to 10 different layers.

There are two types of filter layers which can be applied, they are parametric layers and volumetric layers. The former screens particles based on whether or not the particle's parameter is within a certain range. The range is specified by the user when the layer is applied. The parameter is automatically selected to be the active parameter when the layer is applied. The latter selects particles if they are within a user specified volume. This volume is determined by the location and size of the projection plane volume at the time the layer is applied.

When the user applies a layer, the entire set of particles is sorted and those which fail to meet the criteria, by being outside the bounds of the parameter range or the projection plane selection volume, are tagged. The user can apply the filter either to the particles in the 3D window, or to the particles in the 2D and 1D windows. The particles which were tagged do not appear in the windows where the filters were applied. When the filters are activated, the rendering and counting routines check if each particle has been tagged (failed to meet a filtering criteria), and if the particle has, the routine simply does not draw or count the particle. This simplifies the processing routines since they need only know if the particle has to be included, not at which layer the particle had failed. This 'tagged' information is stored in what is called the filter space. The filter space allows Particle3 to know exactly which layer failed to meet the criteria. This simplifies the tracking of the particles when layers are removed or added. (see § 4.1)

In order to simplify the creation and use of the filter tool, several organizational capabilities have been added to Particle3. In addition to the filter activation/deactivation toggle, the user may call up a list of the current layers in the filter, which includes the parameter names and limits associated with each layer. Volumetric layers appear on the listing simply by number, for example volumetric layer #1, #2 and so forth. The user may clear the entire filter with one key press, or he or she may remove one layer at a time. Layers, of course, may be added.

CHAPTER 5

Applications

Particle simulations are used in the study of non-equilibrium chemistry, rarefied gas dynamics, plasma physics and many other fields. Demonstrations of two space plasma physics simulations are presented to show the uses and capabilities of Particle3. The two different data sets were generated at the Space Power and Propulsion Lab at M.I.T. [5,17]. Previous examples of visualization of plasma particle simulations, such as found in Wang's Thesis [23], generally only present continuous field results. Two dimensional contours and arrow plots are effective and are preferable for presentations of large numbers of standardized plots. The current use of particulate visualization is limited to scatter plots and a mere two dimensions. Examples of current particulate visualization can be seen in Figures 5.2 and 5.3. Figure 5.2 demonstrates two phase plots showing the evolution of a two beam instability, and Figure 5.3 shows a combined visualization screen showing the current state of a plasma at a wall. These two figures show the limitations of the pre-Particle3 visualization tools. Particle3 distinguishes itself when applied to the study of complex three dimensional simulations such as the two cases explored below. It will be demonstrated that Particle3 provides both insight and diagnostic aid for particle based simulations in three dimensions.

5.1 CHAWS

The first simulation was a PIC simulation used to model the Charge Hazards and Wake Shield (CHAWS) experiment onboard the Wake Shield Facility (WSF) satellite [5]. The goal of the simulation was to determine the effects of spacecraft charging in the wake region behind the spacecraft and to try and capture the experimental results in a simulation, where both theory and other simulations had been unable to. Before presenting the visualization results a brief background will be discussed [6].

Although always a concern, the importance of spacecraft charging is growing, as space missions increase in duration and as the frequency of multiple spacecraft encounters increase. One particular problem of spacecraft

charging occurs when two spacecraft of distinctly different sizes interact in Low Earth Orbit (LEO). The problem is caused when the larger craft shields the smaller from the ion bombardment of the orbital plasma. The electron velocities in LEO are much larger than the orbital velocities, and the ion velocities are lower, which indicates that the plasma can be called mesosonic. Under the shielded conditions the amount of charge deposited onto each spacecraft is significantly different, and a large potential difference is created between the two spacecraft. The interspacecraft potential can create an arcing hazard if the two vehicles are brought sufficiently close.

Charging is also important from the spacecraft engineering standpoint. Electric fields caused by the power system or spacecraft experiments can redirect the trajectories of charged particles and cause impacts to unexpected or unwanted locations.

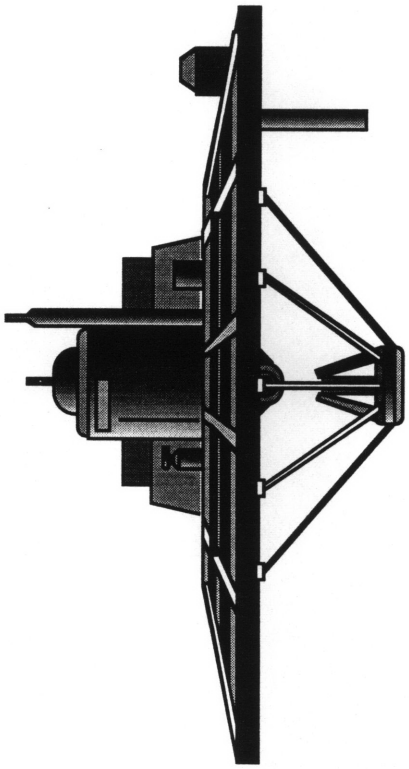


Figure 5.1 Wake Shield Facility.

The simulation was an attempt to model the plasma behaviour around the CHAWS experiment and the WSF spacecraft. The CHAWS experiment consisted of a front (ram) side particle detector and a special Langmuir probe in the wake. The front side particle counter was used as to measure the ambient conditions and provide calibration for the particle

counters mounted on the probe. A Langmuir probe is a sensor which determines the amount of current collected from a plasma when the probe is biased to a specific voltage. Previous theoretical and computational calculations had shown a linear relation between the CHAWS probe potential and the current it collected, but the experiment detected a nonlinear relation between current and probe voltage.

The WSF is a four meter diameter disk with number of experiments on board, including a Molecular Beam Epitaxy experiment and CHAWS. It was carried into orbit on STS-60 on February 4th, 1994. Figure 5.3 shows aside view of the WSF spacecraft. During the CHAWS experiment, the orientation of the satellite was such that the axis of the satellite's disk was in line with the orbital velocity vector and the probe, which was off center, was on the back side of the WSF. Since the environmental plasma in the orbit is approximately stationary with respect to the earth and the satellite is moving in a low earth orbit (LEO) the plasma appears to be moving at nearly 8 km/s towards the spacecraft along the velocity vector. The front, or ram side, of the WSF blocks the ions and causes a wake region to form behind it with a very low ion density. One goal of the simulation was to determine what happened to the ions and the wake structure as the probe was given a large negative voltage. The ions were positively charged and attracted towards the negatively biased probe and some of these ions were counted by the probe particle detectors during the experiment. The simulation had four scalar fields, one vector field and a set of approximately fifty thousand particles. The four scalar fields used were the total ion density, the electric potential, the body field and the strike field. The electric field was the vector field. The body field was used as a simple way to store the location of the WSF spacecraft. The ion density, potential and electric fields were used in the PIC simulation to simulate the ion motion. The electron distribution was assumed to Boltzmann, which means it was always in equilibrium with the potential field, and hence did not have to be stored in an additional field. The body field was used to represent a simplified model of the Wake Shield Facility and was used to determine if particles had struck the body or the probe of the spacecraft. These collisions were stored in the strike field, which demonstrated the variation of the ion flux over the WSF. The simulation was a time dependent three dimensional PIC code which contained the entire wake shield in an equispaced structured grid with 75x75x50 cells. The

simulation whose results follow used a probe bias of -1500 Volts.

Figure 5.4 shows the simplified simulated version of CHAWS in the simulation domain. This figure was generated by saving an iso-surface of the body field of the WSF. The probe is facing away from the ram side of WSF and the two blue planes are the inflow and outflow surfaces of the domain boundary. The grid on WSF displays the grid intersection with the spacecraft body representation. The next two figures, Figures 5.5 and 5.6 show VISUAL3's cutting plane in two different orientations. The surfaces are colored according to the ion density scalar field. In this image dark blue represents low density, green intermediate density and red indicates high density. In Figure 5.5 there is a region of reduced density in the free stream ion flow over the top of the WSF. Here the biased probe has pulled in ions from the free stream and changed the character of the flow. The density around the bottom of the spacecraft appears to be a characteristic expansion of a supersonic flow which would be expected if the probe was not biased. The flow at the bottom of WSF seems to indicate that the potential of the probe is quite localized and probably shielded by the ions it is drawing in towards itself, hence the high density near the probe. Figure 5.6 confirms that the simulation has one axis of symmetry.

One of the most significant impacts of the visualization, was its ability to show the potential field structure around the probe. The visualization demonstrated that until very low (near zero) voltages the potential distribution around the probe was spherical. The validation of the spherical field distribution assumption was important to the to the subsequent analysis of the experimental space flight data by Shaw [20]. The potential structure can be seen in both figures 5.7 and 5.8. Figure 5.7 shows the grid outline of three nested iso-potentials in a probe on view of the WSF. The iso-potentials shown are actually three saved VISUAL3 iso-surfaces of potential. The probe is at a voltage of -1500 Volts, and the iso-surfaces are at potentials of -50, -5 and -3 Volts looking from the probe outward. Controlling these surfaces from the surface data base allows just their grid outline to be shown. Near to the probe the iso-potential is nearly spherical, and even as the potential decreases in magnitude the portion of the iso-surfaces which approach the free stream are nearly spherical, it appears that most of the perturbation from a truly spherical distribution only occurs in the wake region. Figure 5.8 shows the color and contour variation of potential on a cutting plane in the plane of

symmetry. In these two figures the color red represents potentials near zero and blue represents large negative potentials. A contour is a two dimensional analogy of an iso-surface. The contours also confirm that electric potential around the probe is spherical for large potentials, and that the potential field becomes more distorted towards the center of the wake side of the WSF.

The use of two dimensional phase plots provide a simple method to observe the particle behaviour. Figures 5.9 and 5.10 both represent a x vs. y phase plot. This phase plot shows the distribution of particles that would be seen when looking down onto the WSF from above. Figure 5.9 shows the entire set of particles and Figure 5.10 is a filtered subset thereof. In Figure 5.9 the wake is not very distinct because the particles in the free stream around the spacecraft obscure it. In order to make the wake more distinct a volumetric filter layer was applied. The defined volume was a rectangular slab around the probe extending into the free stream at the sides of the WSF but not above and below it. A small subfigure on Figure 5.10 shows the position and size of the selecting volume relative to the spacecraft. The wake region clearly stands out in figure 5.10. Another insightful phase plot in this simulation is the x vs. v_x plot as shown in Figure 5.11. In this plot the particles are colored according to their energy values, red indicates higher energy than blue. The energy is calculated from the three velocity components of each particle. There are two conclusions that are reached, based on this plot. First, the dense fuzzy line of particles implies that the majority of particles are nearly mono-energetic and traveling at a finite positive x velocity, which corresponds to the orbital velocity. Second, high energy particles appear at a specific range of x values. The distribution of high energy particles near and downstream from the probe can be explained as follows. The highly biased probe accelerates the particles as they pass near the probe, but not all particles influenced by the probe are trapped by its potential well, and those that escape, flow downstream.

The distribution plots of the particles are relatively simple. Figure 5.12 shows the energy distribution of the simulated particles and it appears to be a shifted maxwellian, as expected. Figure 5.13 shows the energy distribution of the particles with a normalized energy of less than fifty. This was generated by using the Particle3 filter. Note the difference in particle counts between Figures 5.12 and 5.13.

The positional plot does not really offer much additional information

in this simulation. The large number (approximately 50,000) of particles in the simulation clutter the 3D window, unless they are selected through the projection plane. With the projection plane, note the outline of the volume attached to the plane. Figure 5.15 shows the 2D window view of the projection plane shown in Figure 5.14.

5.2 Plume Simulation

The second simulation to which Particle3 was applied is also a PIC simulation of the plasma environment around a satellite. The purpose behind this simulation was to model the plume effects and contamination caused by an electric thruster on a satellite with highly biased surfaces. The results generated by this model were presented in Samanta-Roy's thesis [18]. Before describing the use of Particle3 to visualize the simulation, the background and motivation for this simulation is briefly detailed.

The potential problems of spacecraft contamination by the effluents of electric propulsion (EP) thrusters have been known for some time. However, ground-based experiments produce estimates of thruster contamination that are questionable due to vacuum chamber facility effects such as residual chamber gases and chamber wall effects. An alternative to ground experiments is to turn to massively parallel computing technologies in order to simulate the complex interactions. The interest in this problem exists because EP is earnestly being pursued for commercial satellite applications. The induced environment in the vicinity of an EP propelled spacecraft consists of neutral gases, plasmas, and fields produced by the complex interactions between the ambient environment, the thruster effluents and the spacecraft itself.

There are a variety of ways for a contamination plume to degrade the performance of a satellite. The thruster plume may cause electromagnetic interference due to the enhanced plasma density in the spacecraft vicinity. Sputtered metals from the EP thruster can stick to and therefore degrade sensitive surfaces, or a low energy plasma may be generated by charge-exchange processes which may cause a power drain on solar cell arrays. For other sources of degradation or a more detailed description of these, refer to Samanta-Roy [18].

The simulation uses a hybrid plasma PIC model (particle ions -fluid

electrons) and includes thruster beam ions, charge-exchange (CEX) ions and non-ionized neutrals in the simulation. The CEX collisions between beam ions and the neutrals are simulated on a production rate basis near the exit of the EP thruster. The EP thruster is attached to a spacecraft with solar cells. The production of slow CEX ions at the thruster exit are moved outwards due to the radial fields in the thruster plume. These ions contribute to a "bubble" in the electric potential which grows until it reaches a steady state. The induced electric field causes the ions to move outwards from the "bubble" including a component back towards the spacecraft which gives rise to the term backflow.

The simulation models the backflow over half of the USAF Advanced Research Global Observation Satellite (ARGOS). ARGOS was modeled with a rectangular half body with a planar solar array at one end. By capitalizing on one plane of symmetry in the satellite the computational effort required was reduced. The resulting computational domain was 3.2m by 4.5m by 3.0m in the x, y and z directions. The satellite dimensions were 1.5m by 0.5m by 1.0m with 4.1m solar arrays at one end of the spacecraft and a xenon ion thruster at the other. The simulation domain was broken into a rectangular grid with varying spacing with 139x241x281 nodes along the x, y and z axes, for a total of over 9.4 million nodes. There were 35 million particles in the simulation. In order to run the simulation the computations were performed on two massively parallel computers, a CRAY T3D and on an Intel Touchstone Delta, both from the Concurrent Supercomputing Consortium in Pasadena, California. The non particle field data generated consisted of ion densities, electric potentials and electric fields. Although the simulation was run with this level of resolution, due to memory constraints on the platforms on which Particle3 was running the number of nodes and particles had to be reduced in order to make the data set more manageable. The visualization data set consisted of 784,000 grid points and 200,000 particles.

The first figure, Figure 5.16, shows three parallel surfaces intersecting the spacecraft half body. The planes were generated by scanning the cutting plane through the domain and saving the surface when it was at a desired location. The planes are colored by the CEX ion density field. The dark blue represents low densities and red high. The body of the spacecraft provides partial shielding of the solar cells from the backflow plume which can be seen in the expanding low density region behind the thruster.

Figure 5.17 again contains saved surfaces, but in this figure the surfaces

are iso-surfaces, or more precisely, iso-potentials. The iso-potentials show the expansion of the potential field in three stages. First the positive ion thruster beam is shown in light green. The beam is relatively unperturbed here because of the high velocity ions that are being emitted to provide thrust. The second surface, which was rendered translucent in order to show the previous surface, begins to show the "bubble" in the field structure induced by the CEX ions leaving the thruster plume. The third surface shows the potential field as it folds over the spacecraft body, in particular the influence of the spacecraft asymmetries at the corners are noticeable on this iso-potential. For reference the solar array and thruster exit are indicated in this figure.

The next two plots show the particles in the plume. Figures 5.18 and 5.19 both show a view of the thruster exit, near where the CEX ions are produced. The flow of particles up and around the thruster exit is noticeable in Figure 5.18. The particles are colored based on their kinetic energy, which shows low energy particles in blue and high energy particles in red. As the particles get drawn into the backflow the electric field accelerates them, which increases their energy. Figure 5.19 shows just the low energy particles which were selected by applying a parametric filter based on energy. The particles seen represent the CEX ions shortly after their production, before they drift out of the ion beam and contribute to the backflow.

Figures 5.20 and 5.21 are images generated by using the VISUAL3 cutting plane positioned to show a cross section of the entire domain. The rectangle in the lower left is the spacecraft body, and the solar cells are along the left edge of both plots. In Figure 5.20, the potential contours are rendered. These show the smooth variation of potential, and in particular show the "bubble" in the field caused by the positive CEX ions. Figure 5.21 shows the CEX ion density, which is highest, as expected, near the exit of the thruster. This plot is similar in structure to the previous plot, but the backflow is more distinct. When viewed from the side, the contours have an expansion wave quality over the spacecraft top corner nearest the thruster.

Figure 5.22 is an energy distribution plot generated by Particle3. There appear to be two distinct energy distributions combined in this simulation. The first hump, with the large peak represents the CEX particles that have been newly generated in the ion beam. This subset of particles was the shown in Figure 5.19. The second, more diffuse peak, represents the ions that have

been drawn into the backflow.

Figure 5.23 shows a projection plane view of the particles generated from the same cutting plane that was used in Figures 5.17 and 5.18. This plot shows the energy variation in the particles very clearly. The newly created low energy ions at the core gradually get drawn into the backflow where they are accelerated outwards by the backflow field structure.

Figure 5.24 shows a Z versus Y phase plot from an earlier and smaller simulation run. This plot demonstrates how Particle3 was instrumental in finding a problem in the simulation that might otherwise have gone undetected. The plot should have an even azimuthal distribution but there are three regions with substantially fewer particles emanating from the thruster axis. These gaps were caused by a problem in the segment of code which loaded the particles into the domain.

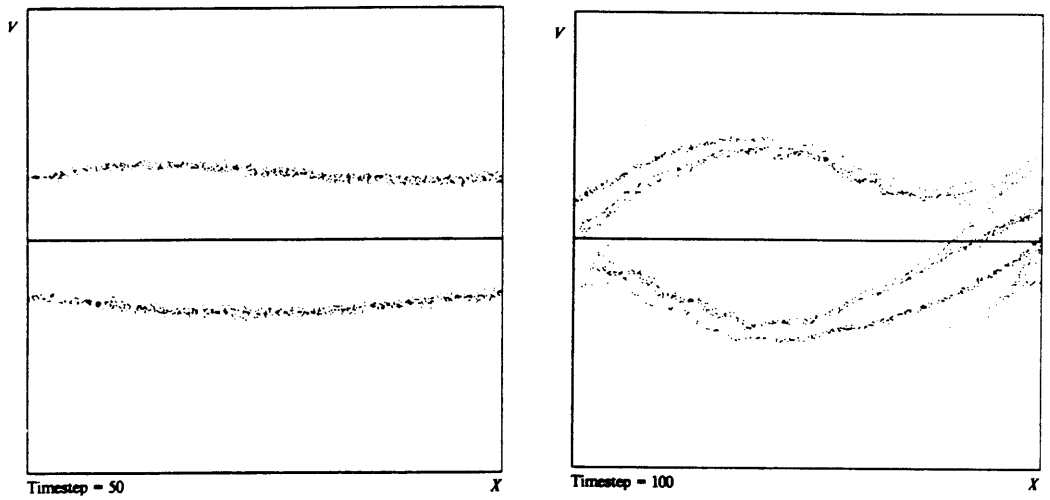


Figure 5.2 Phase space plots of the nonlinear evolution of a counter streaming beam instability. Taken from Hockney and Eastwood [12]. Reprinted with permission from IOP Publishing Ltd.

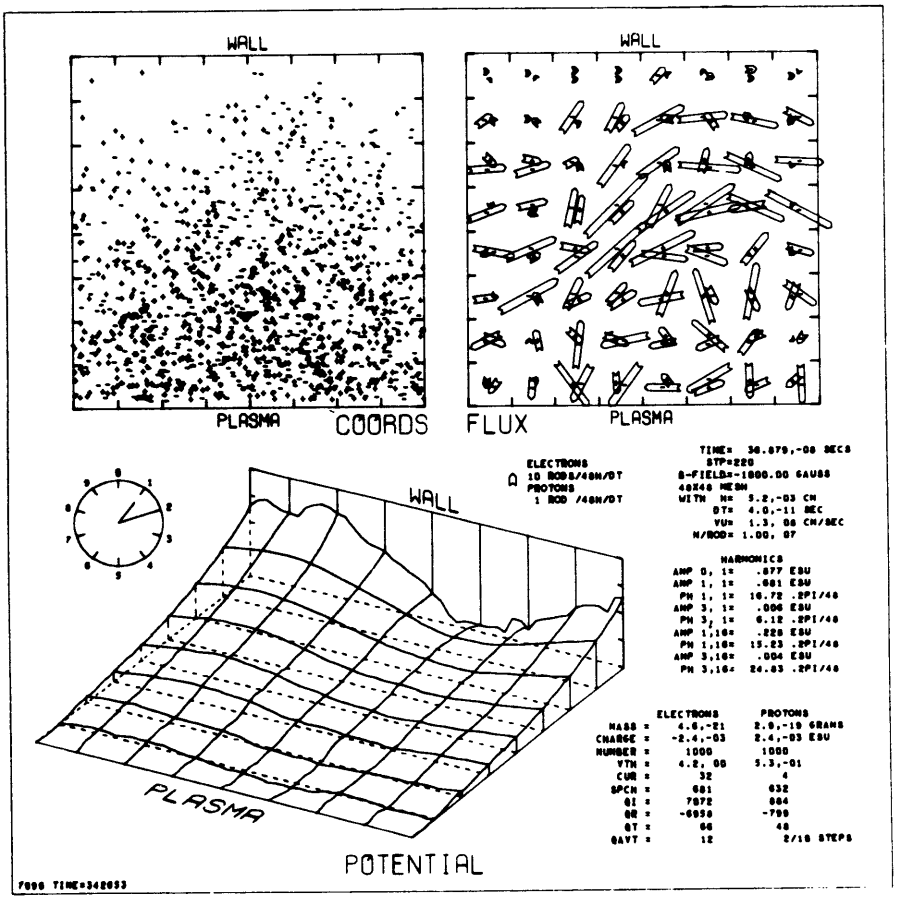


Figure 5.3 Integrated particle simulation plots from Hockney and Eastwood [12]. Reprinted with permission from IOP Publishing Ltd.

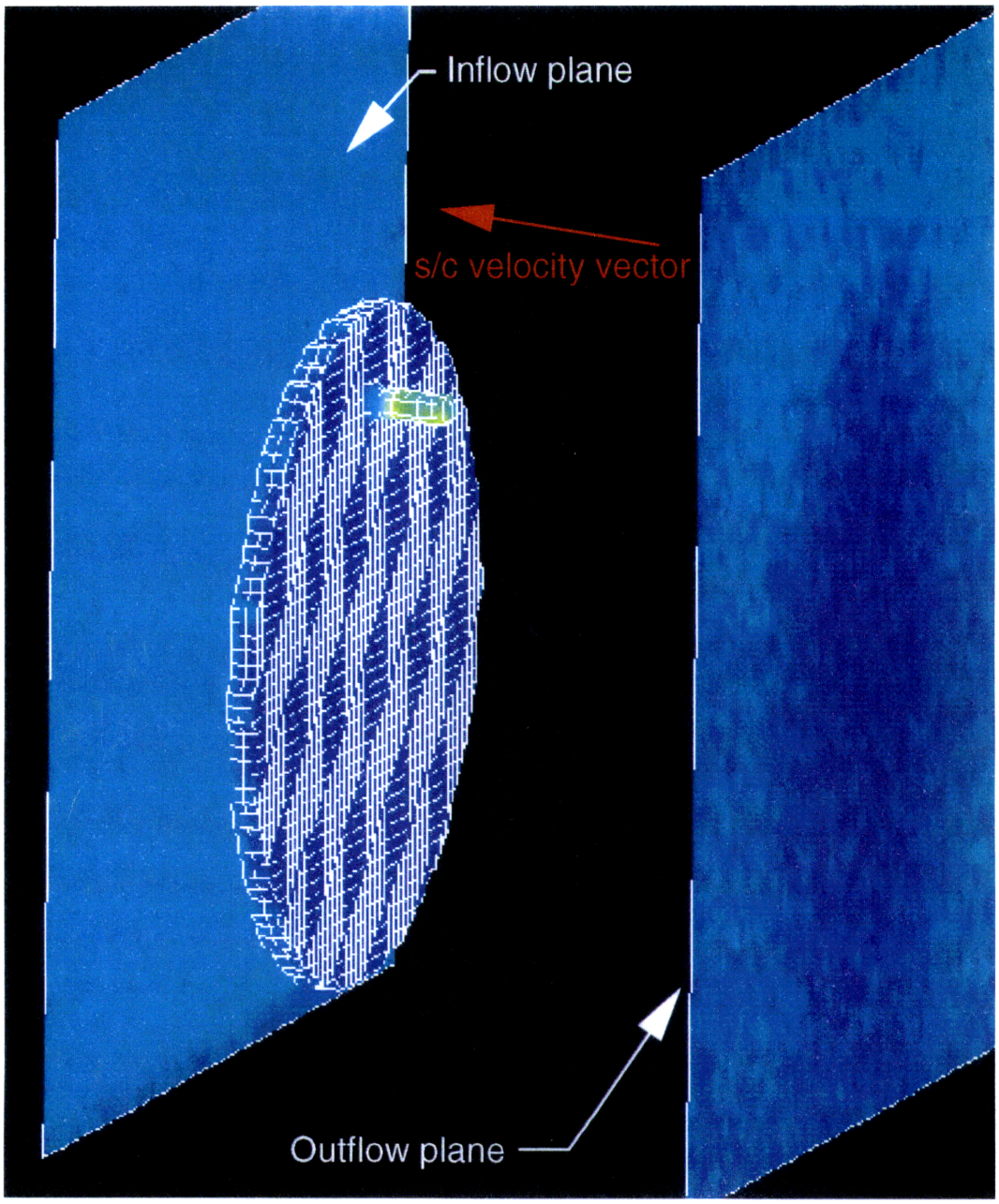
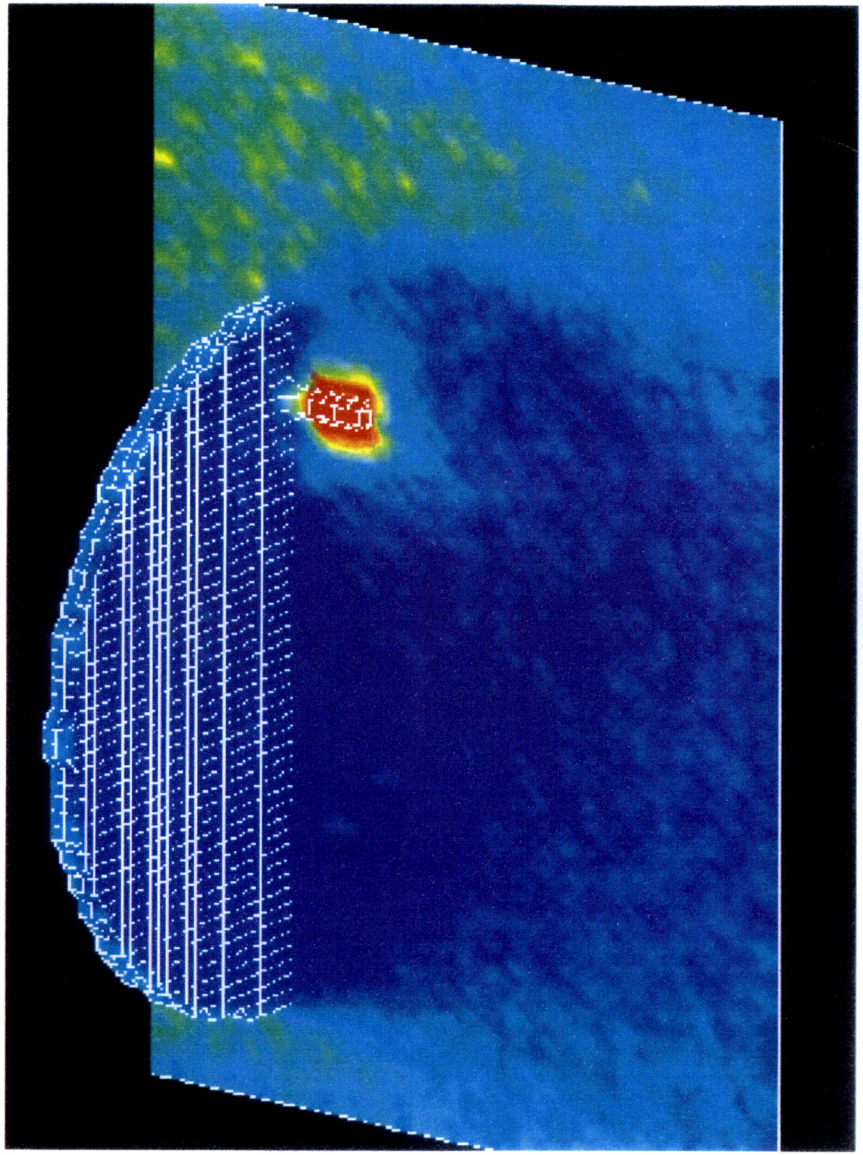


Figure 5.4 CHAWS Domain.



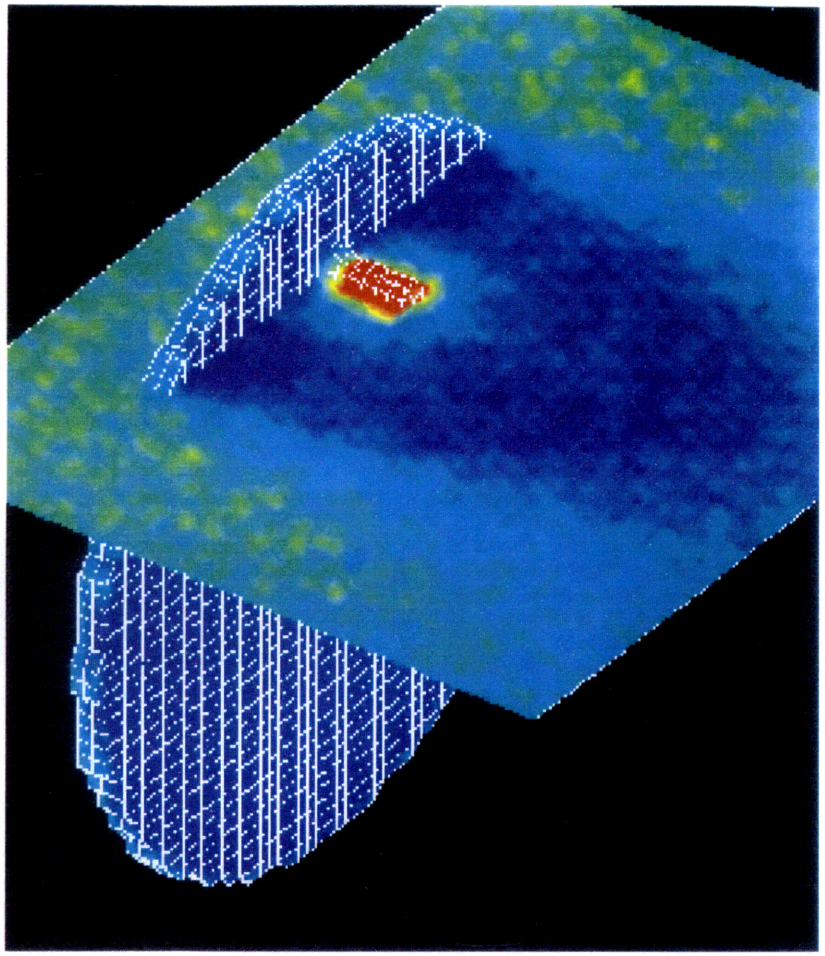


Figure 5.5 Vertical Cutting Plane, Ion Density Field. (left)

Figure 5.6 Horizontal Cutting Plane, Ion Density Field. (above)

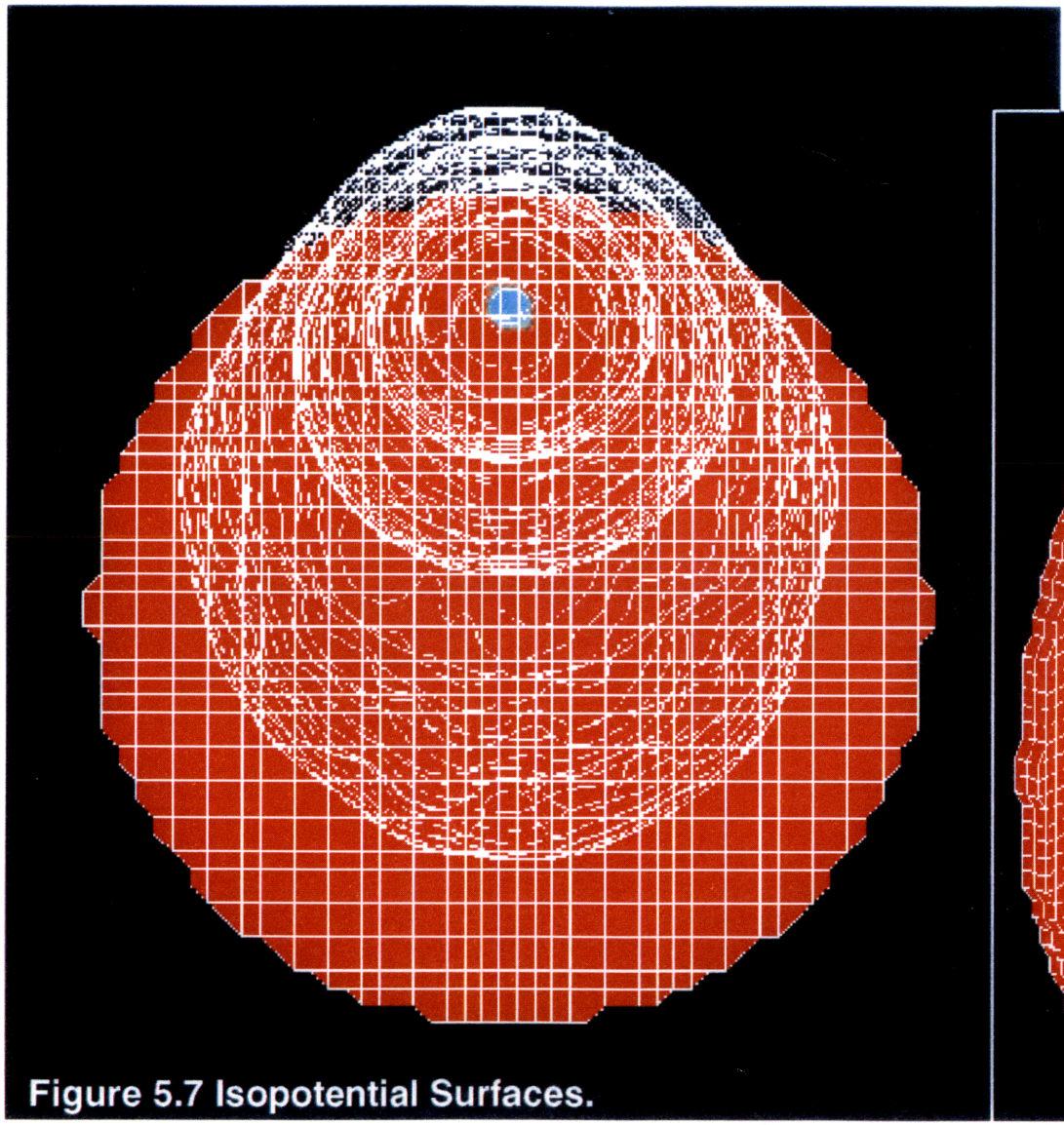
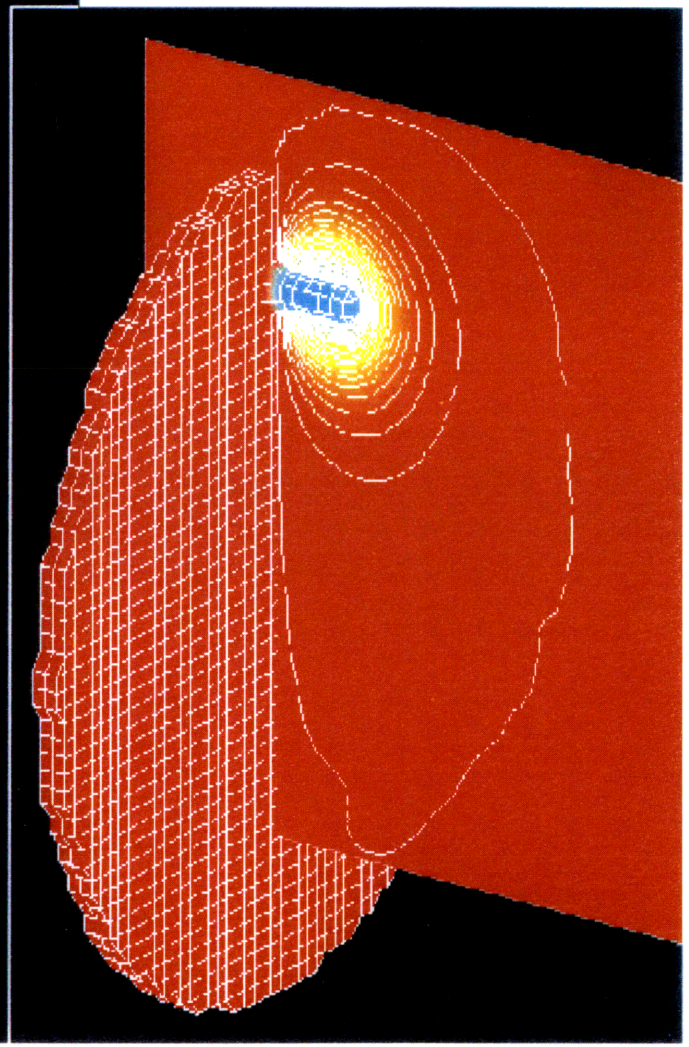


Figure 5.7 Isopotential Surfaces.

Figure 5.8 Planar cut with Potential Contours.



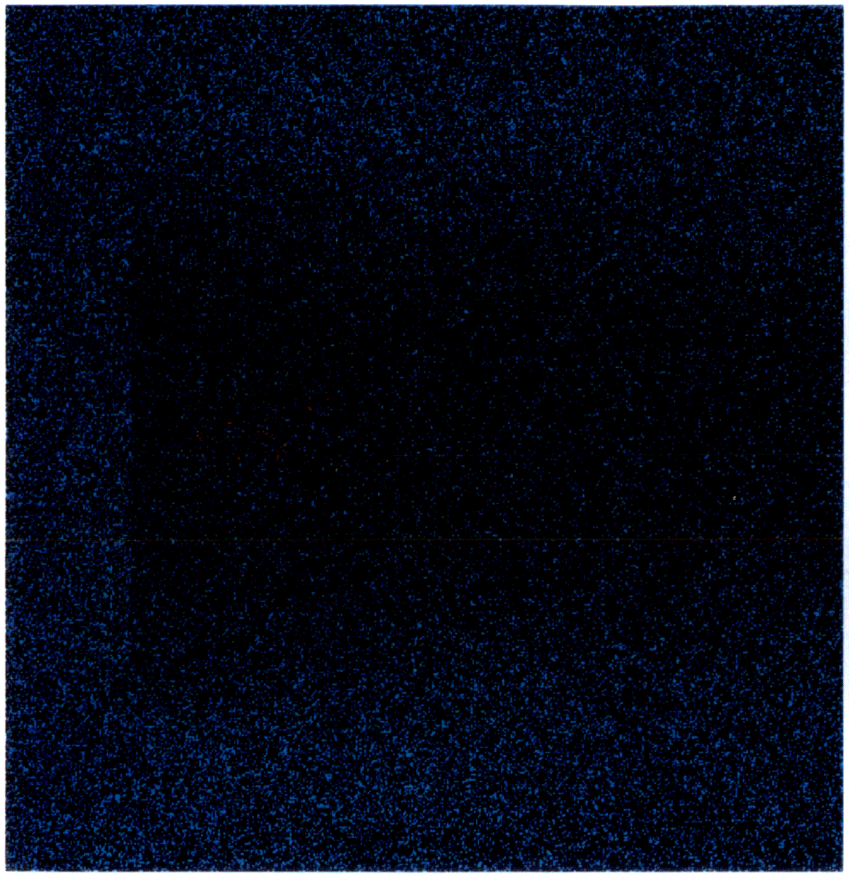


Figure 5.9 X vs.Y Phaseplot.

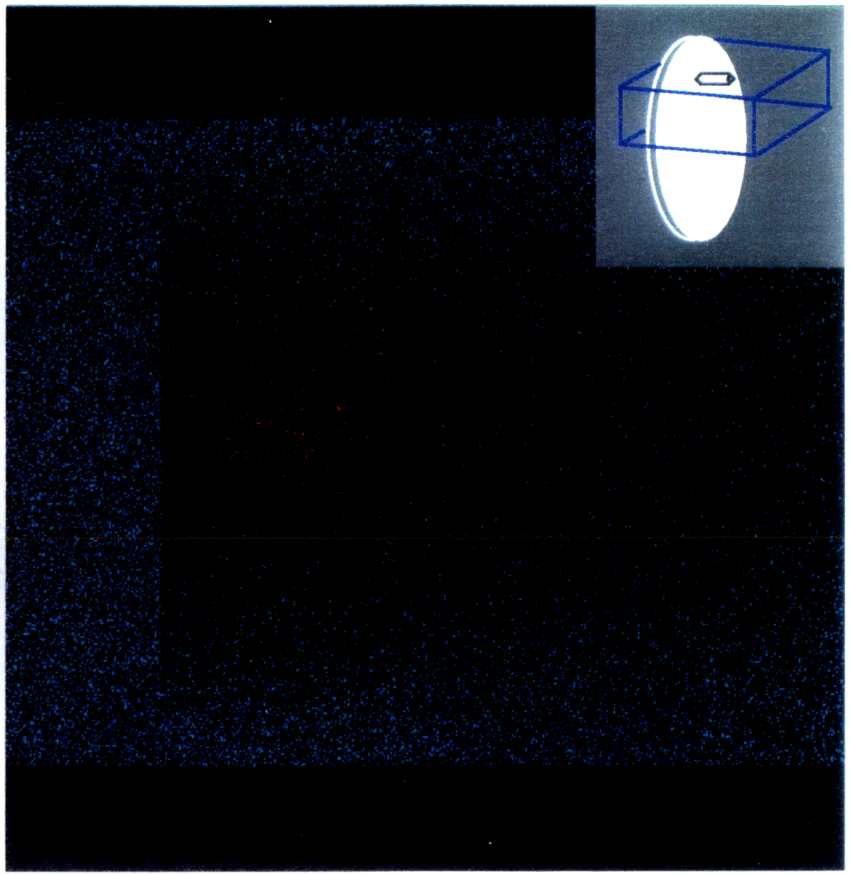


Figure 5.10 X vs.Y Phaseplot w/ Filter Applied.

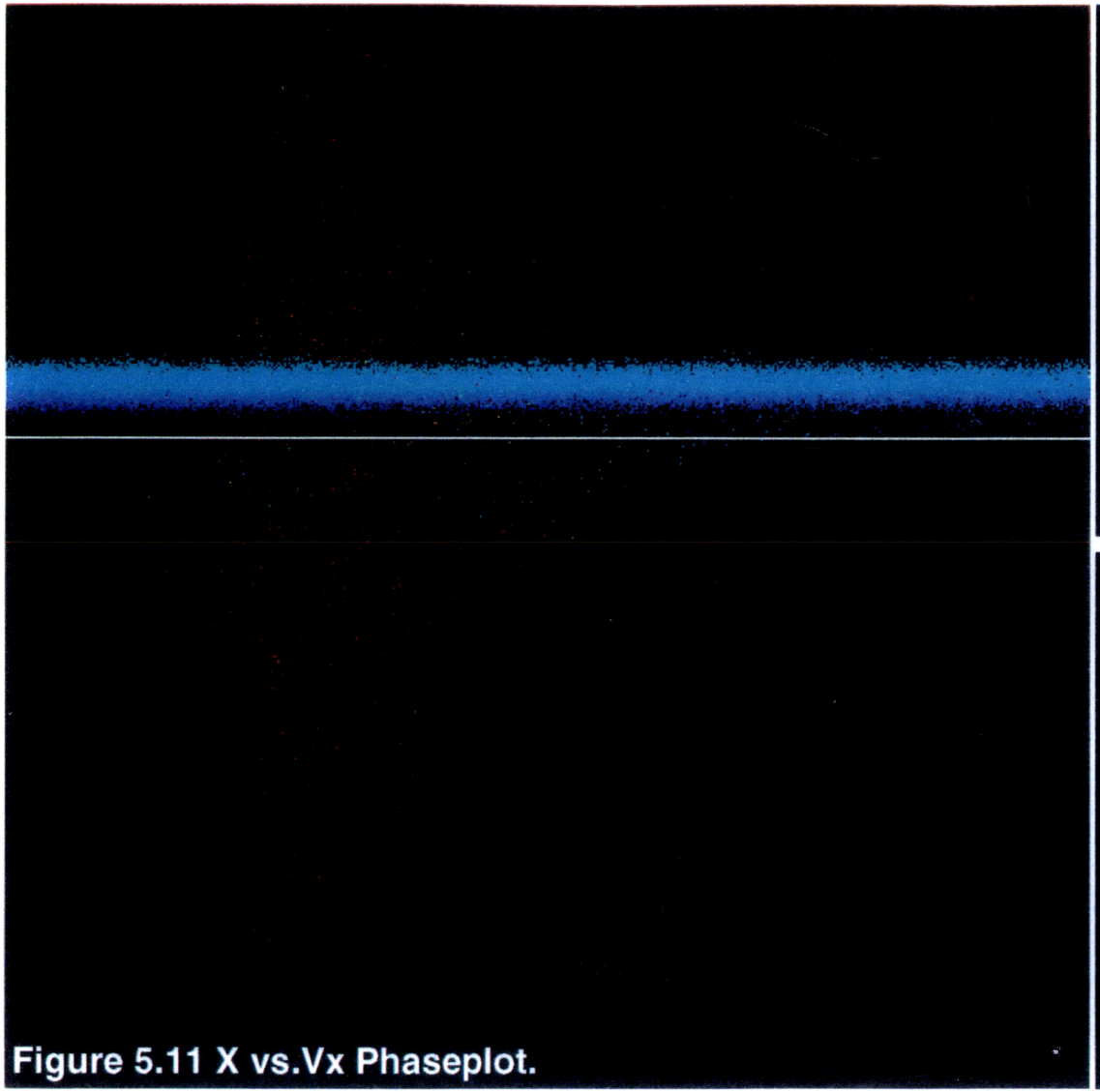


Figure 5.11 X vs.Vx Phaseplot.

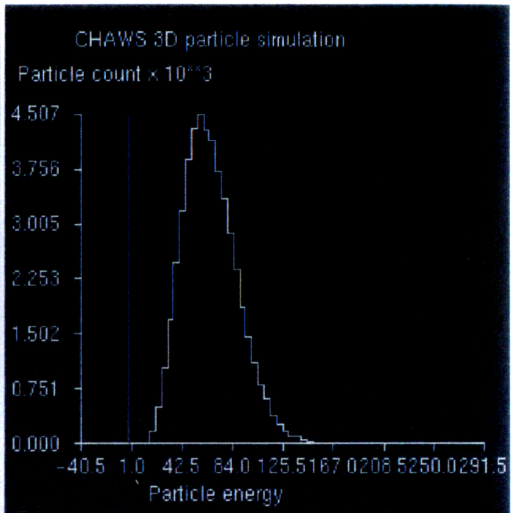


Figure 5.12 Histogram.

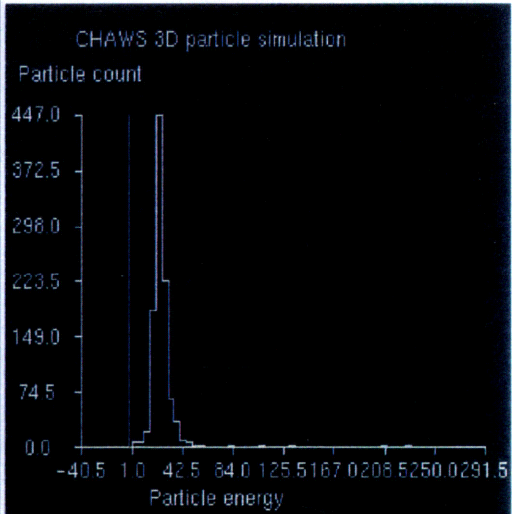


Figure 5.13 Filtered Histogram

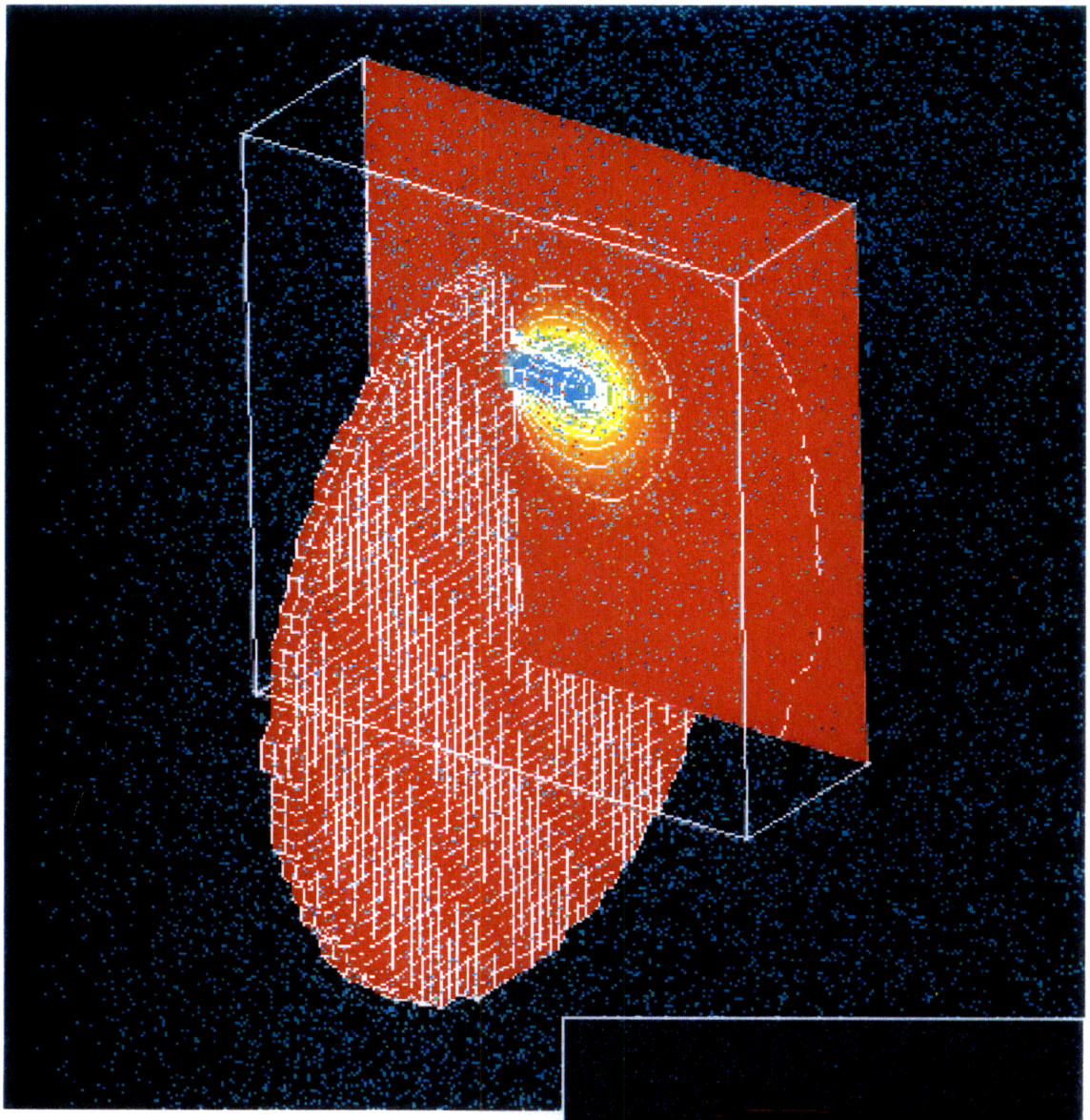


Figure 5.14 Planar Cut with Projection Plane Activated in 3D Window.

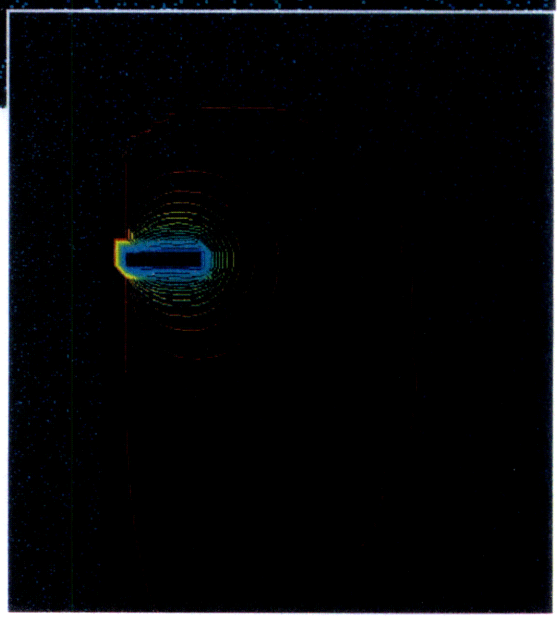
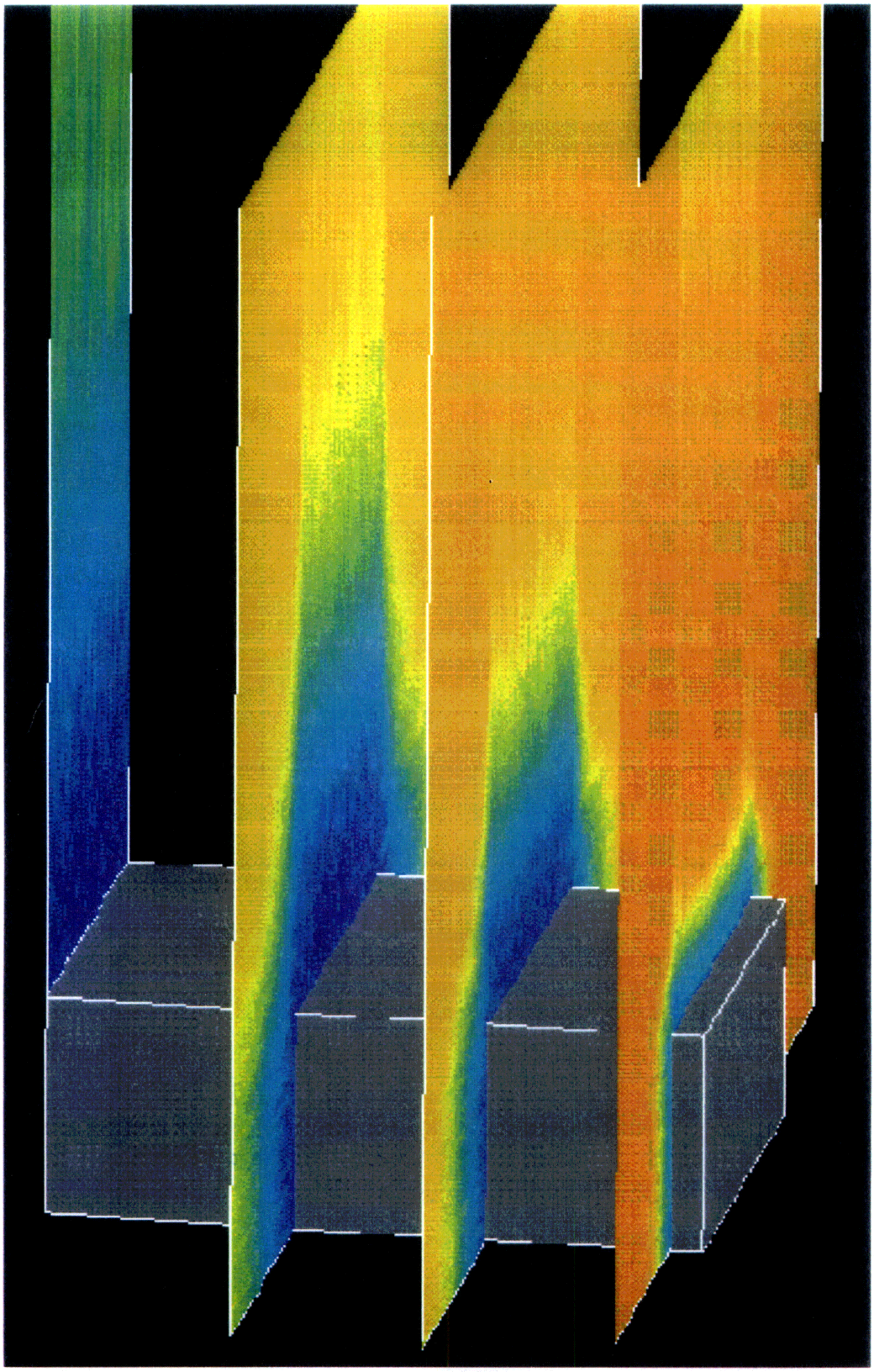


Figure 5.15 Projection Plane View in 2D Window with potential contours.



5.16 Multiple saved cutting planes – CEX density field

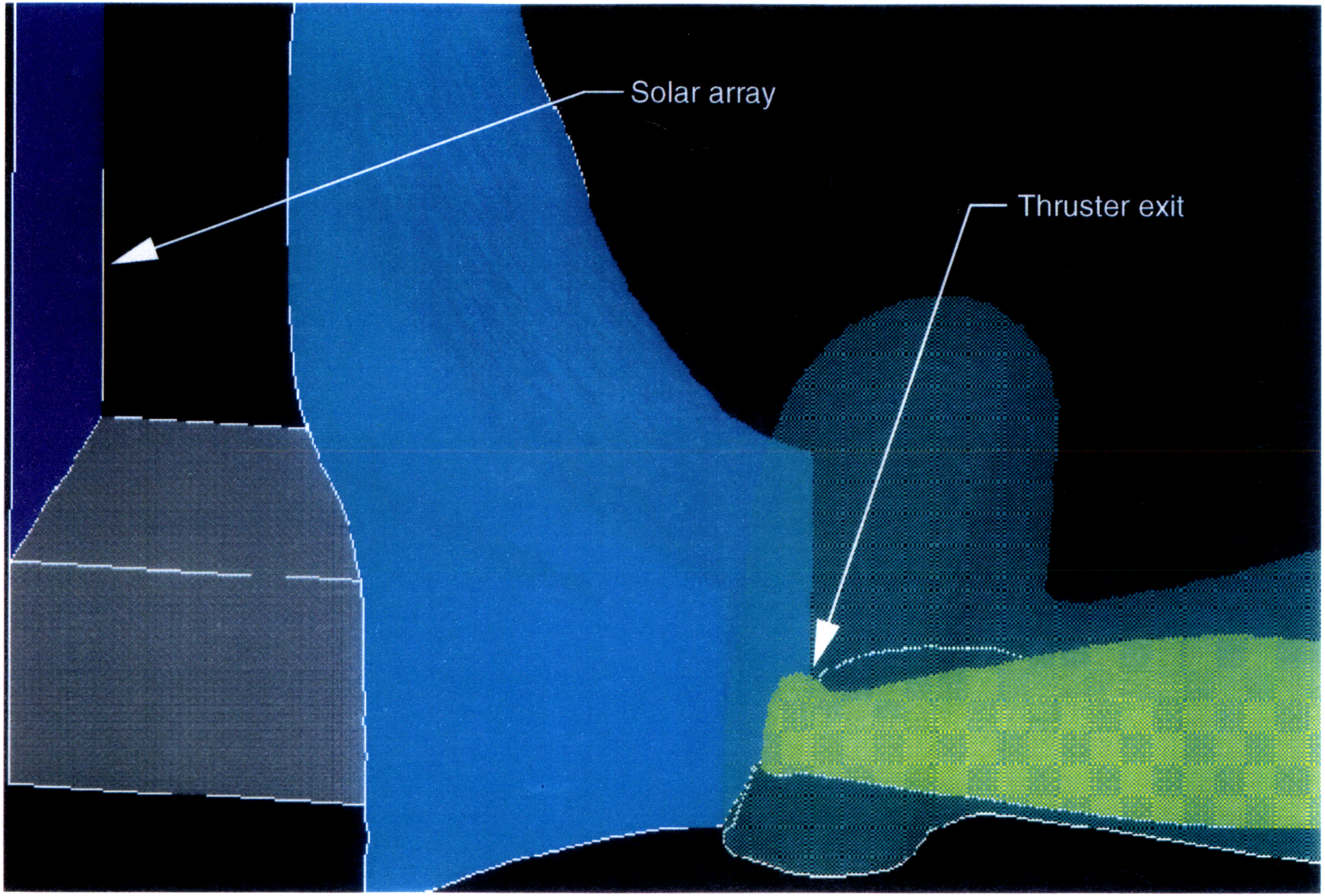


Figure 5.17 Multiple saved Iso–potentials.

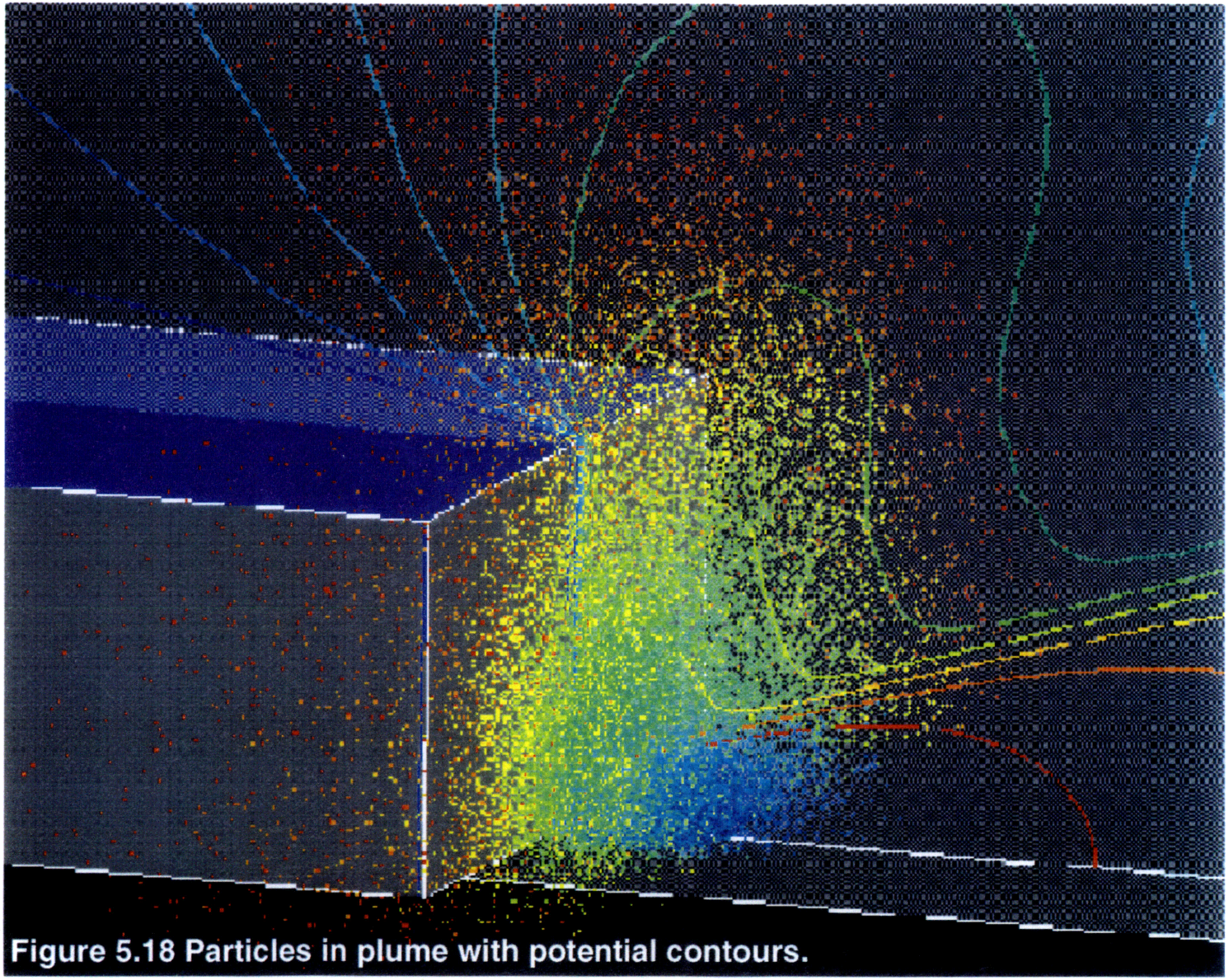


Figure 5.18 Particles in plume with potential contours.

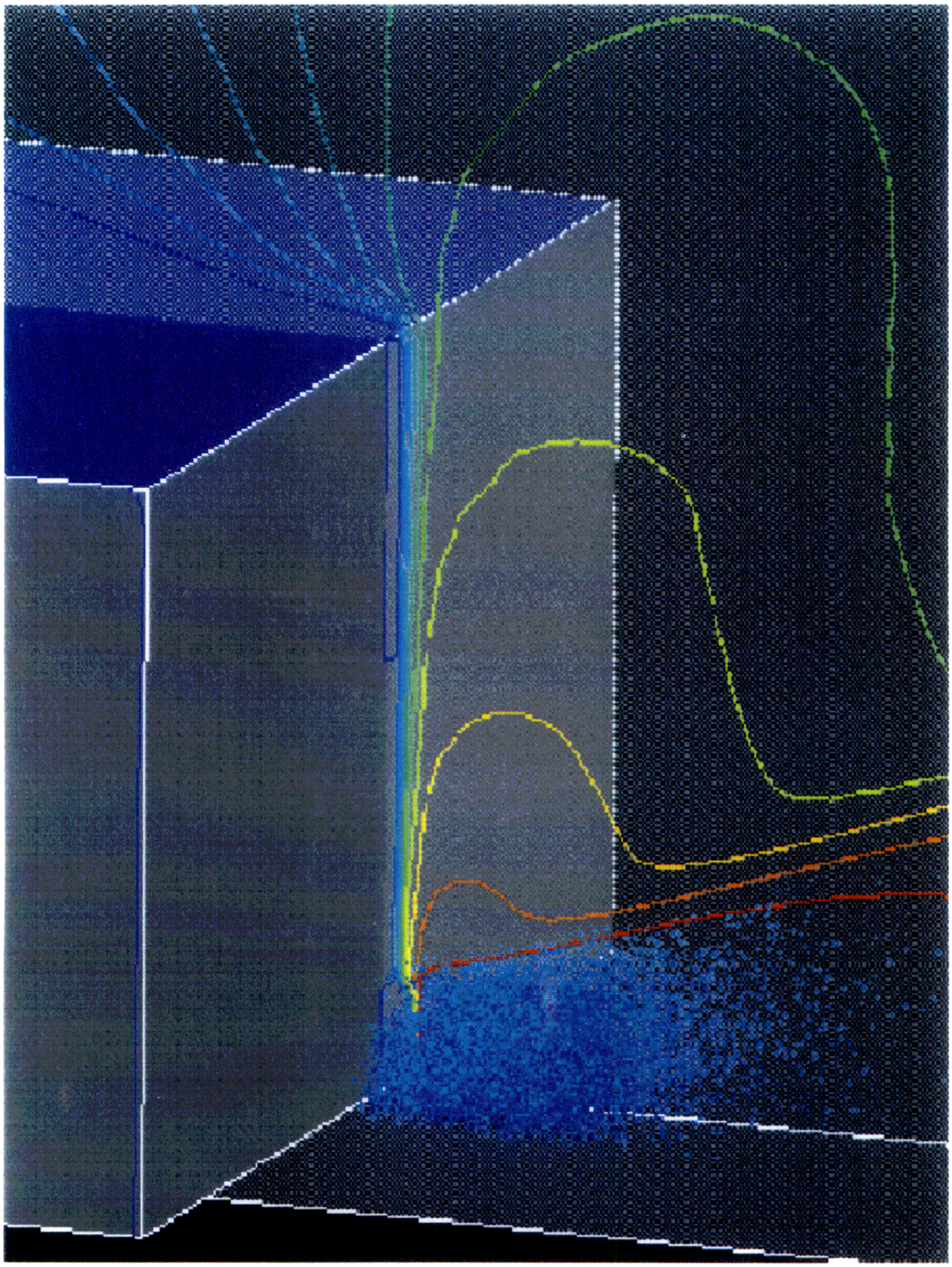


Figure 5.19 Positional Plot with filter activated.

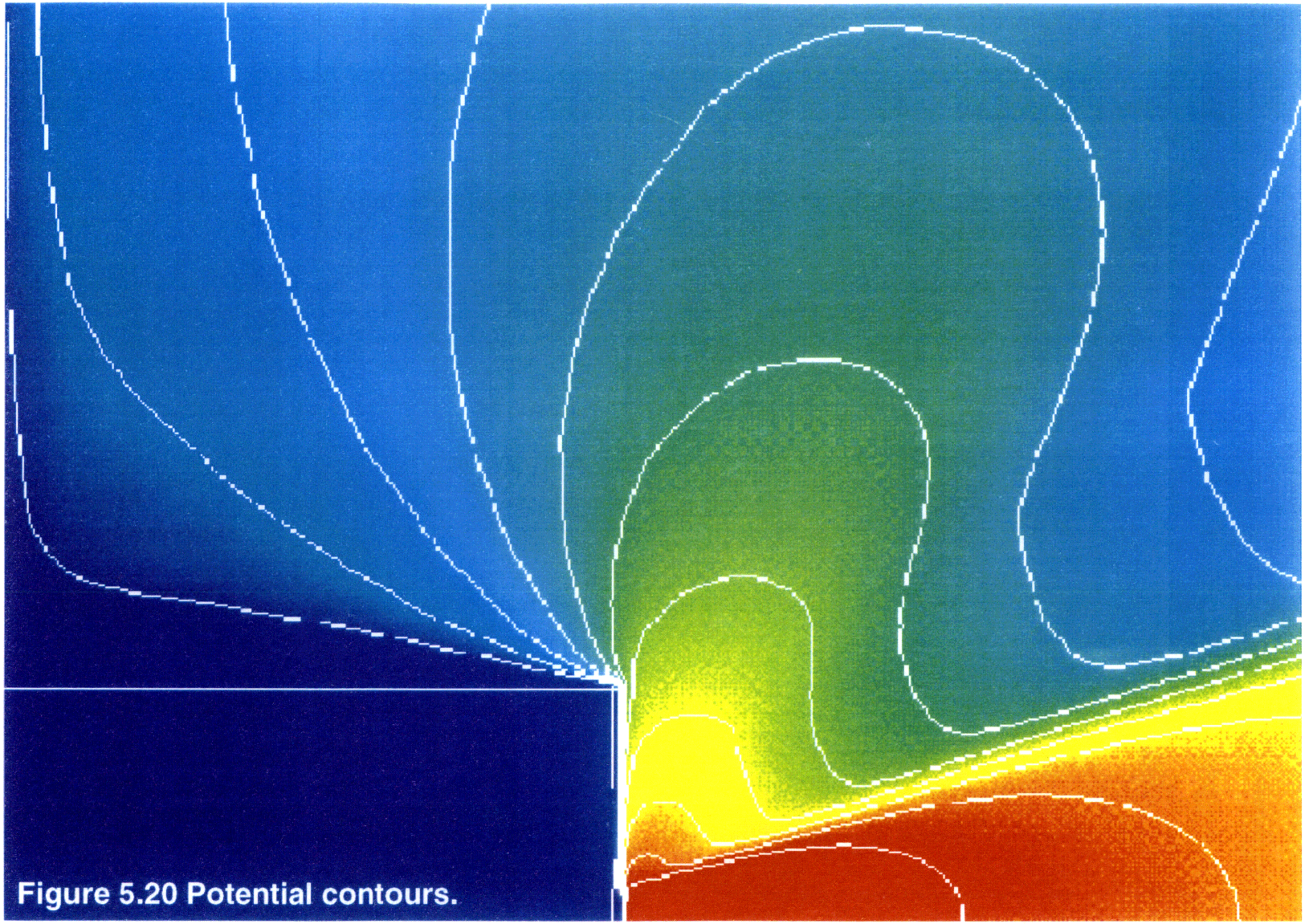


Figure 5.20 Potential contours.

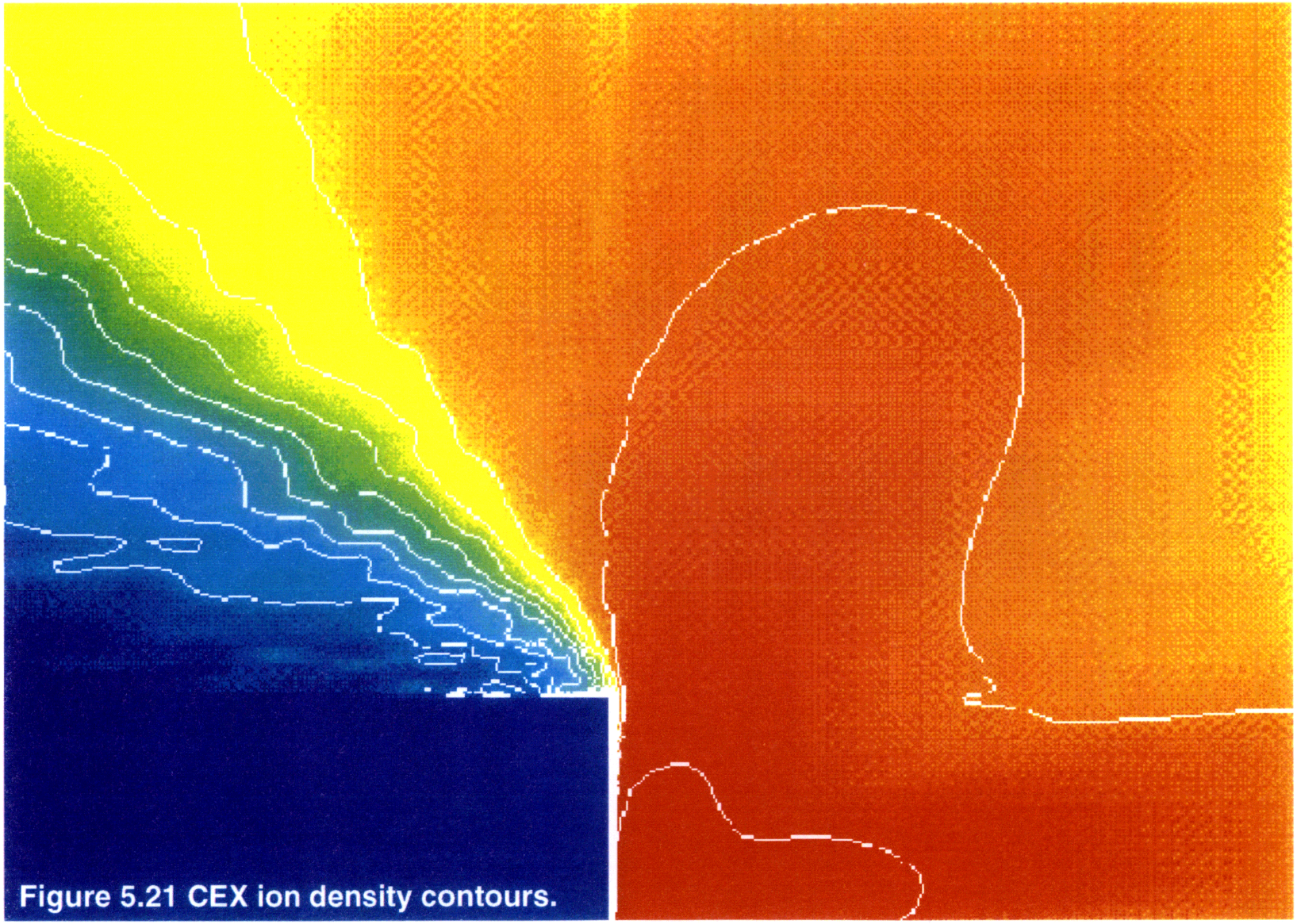


Figure 5.21 CEX ion density contours.

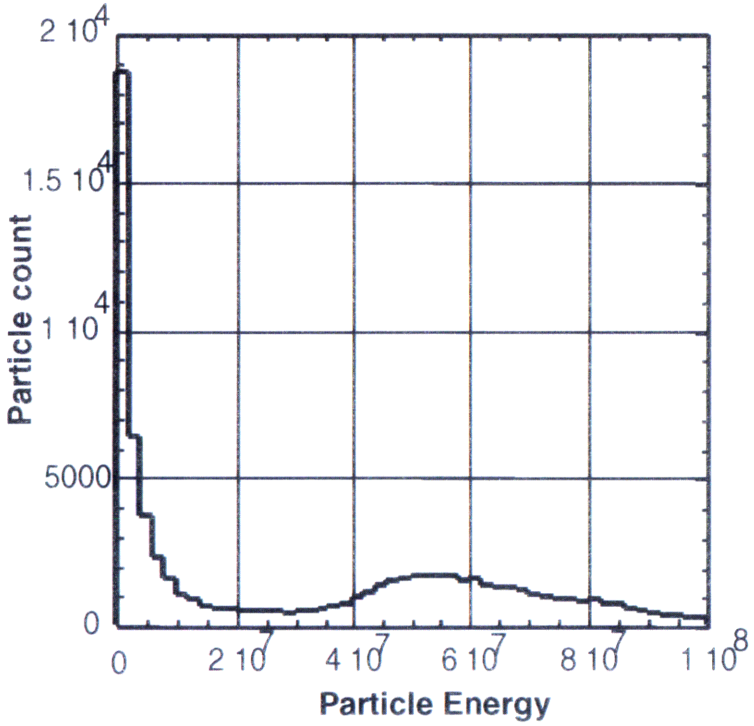


Figure 5.22 Particle Distribution Plot

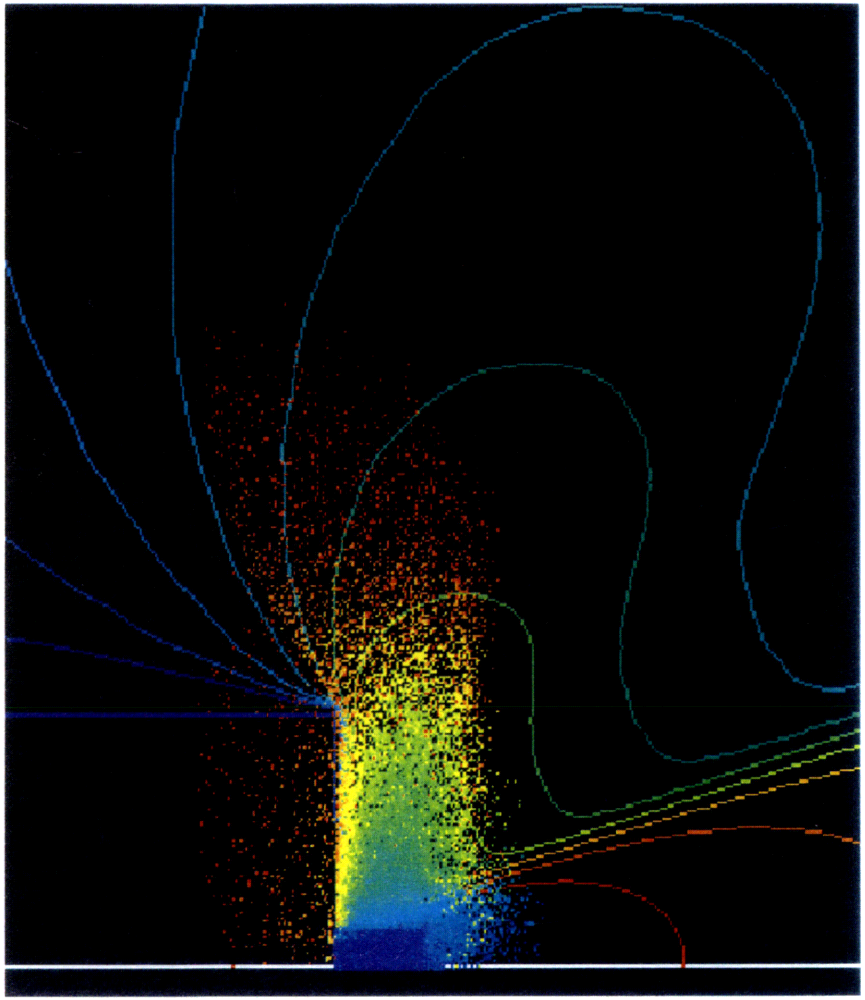


Figure 5.23 Projection Plane View

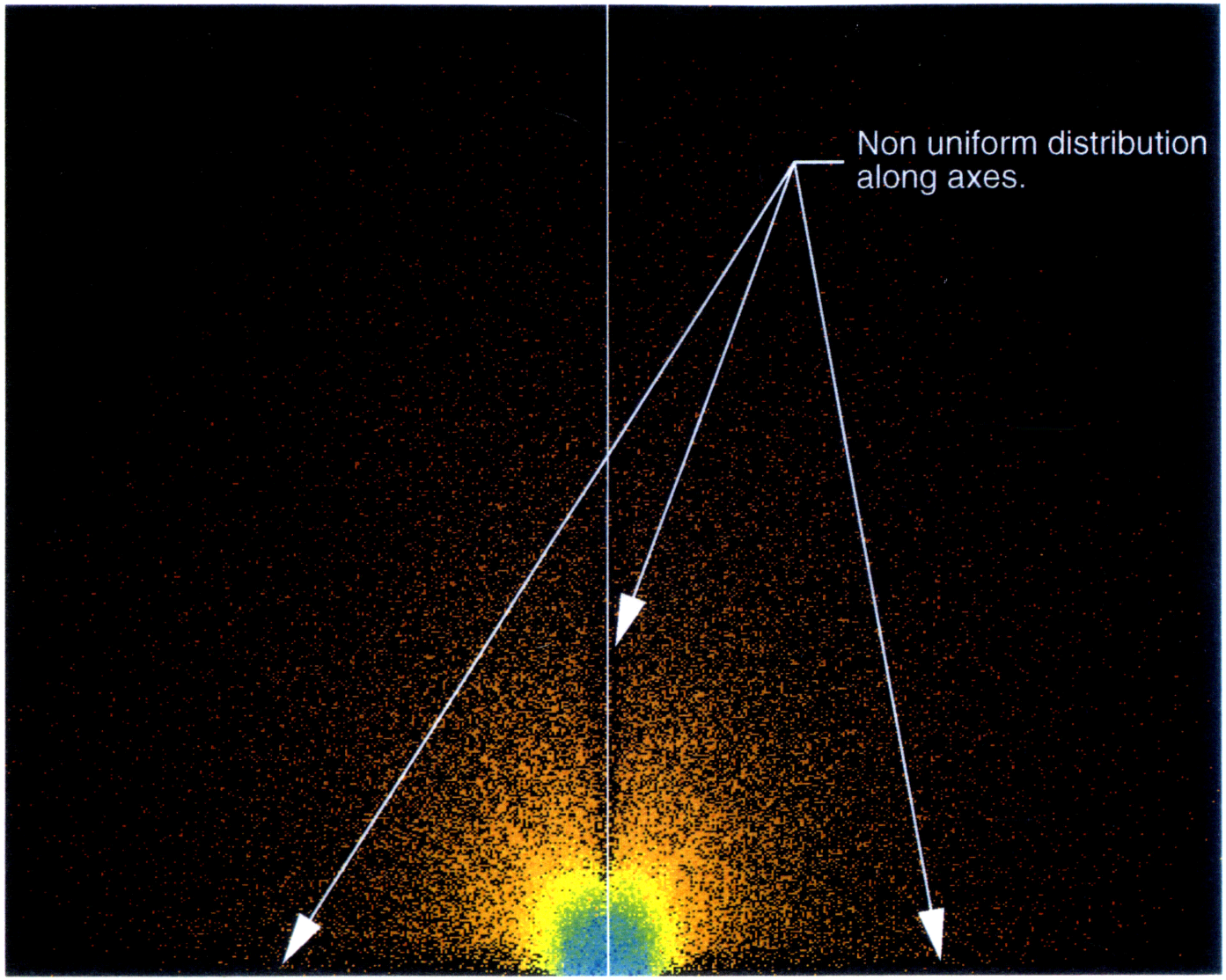


Figure 5.24 Phase plot of old data.

CHAPTER 6

Future Work

Although Particle3 is a complete collection of tools (when used with VISUAL3) there are a number of potential expansions that could be made.

• Programmable Projection Plane. VISUAL3 allows a user to define his or her own cutting surface, with the condition that the general equation of the surface must be written as f(z')= constant. It would be logical to extend Particle3 to be able to use this user defined cutting surface in conjunction with the projection plane tool.

• Particle Trajectory Plotter. This functionality was jury-rigged into a specific simulation in order to get some specific results but was never formally implemented into Particle3. The main difference between a streamline and a particle trajectory is that a particle trajectory requires the particle mass and initial velocity information, in addition to the standard information required to plot a streamline. The ability to show particle traces is expected to be included in a future release of VISUAL3 [10].

• RAVEN implementation. The RAVEN scientific visualization application [3] uses VISUAL3 underneath a streamlined graphical user interface which reduces learning time and generally makes visualization simpler. RAVEN was produced with input from Robert Haimes who created VISUAL3. Since Particle3 was also designed to work with VISUAL3 and could benefit from a menu driven, mouse operated user interface, it was deemed to be a worthwhile future pursuit. A Particle3 window in RAVEN would streamline the entire simulation process and allow much greater control over the sub parameters within the simulation, particularly the filter sections.

CHAPTER 7

Conclusions

Developments in the various research fields that use particulate simulations have advanced further than corresponding developments in visualization software. Particle3 was devised to address and resolve this discrepancy. The program was based on an existing three dimensional CFD visualization program called VISUAL3 in order to take advantage of the rendering "infrastructure" which enhanced the development effort.

The structure, implementation and capabilities of VISUAL3 have been described. VISUAL3 provides a comprehensive set of interactive flow visualization techniques which can be applied to a variety of different simulations. VISUAL3's collection of scalar and vector visualization tools allows data to be quickly and effectively visualized. The operational structure of VISUAL3 was noted and applied to the development of Particle3.

As is integral to any development process, the requirements of Particle3 were defined. These requirements were established in two ways. First, requirements were defined through research into scientific visualization and computational particle simulation. Second, the details were expanded upon, based on interviews with a sampling of potential users.

The functional requirements were divided amongst the proposed tools and the package architecture. Tools were proposed and developed. The tool design was driven by the need to provide informative and insightful techniques for analyzing particulate data. The architecture design was driven by the need to be compatible with VISUAL3. The defined tools were then created and integrated into the final product, Particle3.

Particle3 allows particle data to be handled and observed in its original form. Particles may be viewed in real and phase space. Their distributions may be probed and regions of interest may be focused upon and examined. Unsteady simulations may be viewed as easily as time steady ones, and continuous data may be viewed simultaneously with the particulate data. By creating a coherent collection of tools and an environment in which to use them, Particle3 provides a tool which can improve a user's ability to examine his or her data.

This final version of Particle3 was applied to two different three dimensional Particle-In-Cell simulations and made contributions to each. The visualization of the CHAWS simulation led to the discovery of the potential field structure around the Langmuir probe. This discovery led to an improved numerical model to fit the experimental data generated by the actual CHAWS flight experiment. In the plume contamination simulation Particle3 proved to be a valuable diagnostic tool. The phase plots and positional plots confirmed a problem with a segment of the early simulation code, which was then remedied. In the final versions of the plume evolution the behaviour of the simulation could be clearly seen with both VISUAL3's continuum based tools, and Particle3's particle based tools.

In summary Particle3 was defined, created, and applied successfully. There now exists a comprehensive integrated toolbox of particulate visualization tools for use with VISUAL3, to meet the expected continued growth in demand for particulate visualization.

References

[1] S.F. Beam. "RIAD Visual Imaging Branch Assessment." NASA document N94-27895, December 1993

[2] C.K. Birdsall and A.B. Langdon. *Plasma Physics via Computer Simulation*. IOP Publishing Ltd., 1991.

[3] D.E. Edwards and R. Haimes. "RAVEN: A Rapid Visualization System for Interpretation of Engineering Data." AIAA Paper 94-0204, January 1994.

[4] J.W. Feuquay. "Data visualization techniques for hyper dimensional data." SPIE Vol. 1938, p 435-440

[5] G. Font-Rodrigues. CHAWS PIC code simulation, July 1994

[6] G. Font-Rodrigues, D.E. Hastings, D.L. Cook, "Transient Beam Phenomena due to Spacecraft Charging in the CHAWS experiment.", AIAA Paper 95-0490, January 1995.

[7] D. Gonzales. Personal Communication. April 1995.

[8] R. Haimes. *Advanced Programmer's Guide for VISUAL3* Rev. 2.20, August 8,1994

[9] R. Haimes. Personal Communication. December 1994.

[10] R. Haimes. Personal Communication. November 1994.

[11] R. Haimes. *VISUAL3 User's Guide and Programmer's Manual* Rev 2.15, September 2, 1993

[12] R.W. Hockney and J.W. Eastwood. *Computer Simulations using particles.*, IOP Publishing Ltd., 1989.

[13] T. Lasinski, P. Buning, D. Choi, S. Rogers, G. Bancroft, and F. Merritt. Flow Visualization of CFD using Graphics Workstations. AIAA Paper 87-1180, 1987

[14] M.H. Lean. "Simulation and Visualization of 3D Particle Cloud Electrodynamics." IEEE Transactions on Magnetics, Vol.28, No.2, March 1992.

[15] M.M. Novak. "Visualizing the Dynamics of Particle-Field Interactions." *Fractals in the Fundamental and Applied Sciences* ,p285, 1991.

[16] D. Plansky and R. Haimes. *Particle3 Programmer's Guide.* Rev 1.1, August 10, 1994.

[17] R. Samanta-Roy. EPEX backflow simulation data. November 1993.

[18] R. Samanta-Roy. "Numerical Simulation of Ion Thruster Plume Backflow for Spacecraft Contamination Assessment", Ph.D. Thesis. June 1995.

[19] H. Senay and E. Ignatius. "A Knowledge Based System for Scientific Data Visualization." NASA document N94-23507, 1994.

[20] G.B. Shaw "Analysis of the Ion Current Collection in the Plasma Wake During the Charging Hazards and Wake Studies (CHAWS) Experiment", M.S thesis, to be published.

[21] M. Van Dyke. *An album of Fluid Motion* , Parabolic Press, Stanford, 1982.

[22] P.P. Walatka et al. *FAST User Guide.* NASA Document RND-92-015. November 1992.

[23] J.J. Wang. "Electrodynamic Interaction between Charged Space Systems and the Ionospheric Plasma Environment", Ph.D. Thesis, M.I.T., June 1991.