University of New Hampshire
University of New Hampshire Scholars' Repository

Master's Theses and Capstones | Student Scholarship

Winter 2016

Interactive Visual Analytics for Large-scale Particle Simulations

Erol Aygar
University of New Hampshire, Durham

Follow this and additional works at: https://scholars.unh.edu/thesis

Recommended Citation
Aygar, Erol, "Interactive Visual Analytics for Large-scale Particle Simulations" (2016). *Master's Theses and Capstones*. 896.
https://scholars.unh.edu/thesis/896

This Thesis is brought to you for free and open access by the Student Scholarship at University of New Hampshire Scholars' Repository. It has been accepted for inclusion in Master's Theses and Capstones by an authorized administrator of University of New Hampshire Scholars' Repository. For more information, please contact nicole.hentz@unh.edu.

INTERACTIVE VISUAL ANALYTICS FOR LARGE-SCALE PARTICLE SIMULATIONS

BY

Erol Aygar

MS, Bosphorus University, 2006

BS, Mimar Sinan Fine Arts University, 1999

THESIS

Submitted to the University of New Hampshire

in Partial Fulfillment of

the Requirements for the Degree of

Master of Science

in

Computer Science

December, 2016

ALL RIGHTS RESERVED

©2016

Erol Aygar

This thesis has been examined and approved in partial fulfillment of the requirements for the degree of Master of Science in Computer Science by:

Thesis Director, Colin Ware
Professor of Computer Science
University of New Hampshire

R. Daniel Bergeron
Professor of Computer Science
University of New Hampshire

Thomas Butkiewicz
Research Assistant Professor of Computer Science
University of New Hampshire

On November 2, 2016

Original approval signatures are on file with the University of New Hampshire Graduate School.

Dedication

This thesis is dedicated to the memory of my grandfather.

Acknowledgments

I hereby acknowledge and thank my mentor, Dr. Colin Ware, who supported this research and made it possible. I would like to thank my committee members: Dr. Bergeron and Dr. Butkiewicz for their interest and expertise. I want to thank to my academic advisors, Dr. Michael Jonas, Dr. Michaela Sabin, and Dr. Philip Hatcher who encouraged me throughout my academic journey and helped achieve my goals.

I would also like to thank my colleagues who provided technical and logistical support throughout this research, including Drew Stevens, Roland Arsenault.

Preface

Initial chapter of this thesis consists of an article that has been submitted for publication [Aygar and Ware, 2016, Ware et al., 2016].

Table of Contents

List of Tables

List of Figures

Abstract

INTERACTIVE VISUAL ANALYTICS FOR LARGE-SCALE PARTICLE SIMULATIONS

by

Erol Aygar

University of New Hampshire, December, 2016

Particle based model simulations are widely used in scientific visualization. In cosmology, particles are used to simulate the evolution of dark matter in the universe. Clusters of particles (that have special statistical properties) are called halos. From a visualization point of view, halos are clusters of particles, each having a position, mass and velocity in three dimensional space, and they can be represented as point clouds that contain various structures of geometric interest such as filaments, membranes, satellite of points, clusters, and cluster of clusters.

The thesis investigates methods for interacting with large scale datasets represented as point clouds. The work mostly aims at the interactive visualization of cosmological simulation based on large particle systems. The study consists of three components: a) two human factors experiments into the perceptual factors that make it possible to see features in point clouds; b) the design and implementation of a user interface making it possible to rapidly navigate through and visualize features in the point cloud, c) software development and integration to support visualization.

Chapter 1

Introduction

Particle based simulations are used in computational models for a number of scientific applications, including simulations of fluids [Müller et al., 2003], to study space weather events [Lapenta, 2012], and to understand the formation of the universe [Habib et al., 2009]. These simulations are powered by high performance computing (HPC) systems. Data generation rates are increasing dramatically with advances in HPC and this is causing problems for visualization applications, and the overall workflow. As a consequence, visual analysis tasks are increasingly performed when data still resides in memory as opposed to after a model run. This method is called *in-situ data analysis* or *in-situ visualization*. These advancements also have impact on the nature of products that offer visual analysis solutions.

In cosmology, particles are used to simulate the evolution of dark matter in the universe [Habib et al., 2009]. Clusters of particles (that have special statistical properties) are called *halos*. From a visualization point of view, halos are clusters of particles, each having a position, mass and velocity in three dimensional space, and they can be represented as point clouds that contain various structures of geometric interests, such as filaments, membranes, satellite of points, clusters, and cluster of clusters. This thesis explores methods for interacting with large scale cosmology datasets represented as point clouds.

The following sections of this chapter begin with a basic description of the role of model simulations in scientific inquiry and then discuss key concepts that provide background information about applications related to the challenges of visualizing large-scale particle simulations. Fi-

nally, the rationale behind this thesis is presented in more detail.

1.1 Model simulation as a scientific inquiry method

Traditional experimental methods rely on data collected from natural events through observations and measurements. This is followed by analysis of the data to construct theoretical explanations and to draw conclusions about the object of the inquiry [Langley and Stanford, 1995, Schickore, 2014]. With the advent of high performance computing (HPC), the process incorporates mathematical models and simulation of the phenomenon in ways that are beyond the reach of experiments. Scientists are now able to incorporate datasets from many different sources, such as data captured by instruments, data collected from sensor networks, and data generated by simulations, to cross-validate their outcomes. This is commonly recognized as data-intensive science or computing [Hey, 2012, Ahrens et al., 2010]. Because of advances in HPC, scientists are able to simulate nature with higher level of detail, precision and accuracy. Consequently, weather forecasts become more accurate, products and services are designed in a more economical way, and we now can study complex physical and engineering systems in a scale and fidelity which otherwise not possible. Thus, model simulations have become an integral part of the scientific inquiry process.

1.2 Workflows

Commonly used visual analysis workflows can be characterized by three main stages: preparing the input (pre-processing), simulation run (execution) and analysis of the data generated by the model (post-processing).

As a part of the post-processing stage, the analysis is performed separately in a computer system dedicated to the task. During a model run, the data is saved in the form of model dumps (checkpoint-restart files). However, the files generated from model dumps are both extremely

large and not ideal for visual analysis. Due to the size of such data sets, tasks like viewing, editing, or presenting the data have become a challenge, as the data is too large to fit into the memory of the visualization computer. In some cases, the data is even too large to move through the network or to store in local storage units. Instead, we need methods for saving model data in ways that are both compact and optimized for visual analysis.

1.3 Visualization Pipeline

One of the most common abstractions used by visual analysis products is what is known as the visualization pipeline. This metaphor provides the key structure in many visualization development systems and incorporates operations such as transforming, filtering, rendering, and saving the data. The visualization pipeline represents the flow of data in which computation is described as a collection of static constructs (data operations) that are connected in a directed graph. Many current projects involve solutions based on pipeline architecture because of the abundance of existing implementations and the flexibility they offer.

Most of the HPC systems are based on massively threaded, heterogeneous, accelerator-based architectures. Traditional pipelines are not compatible with this form of computing. Instead, in-situ visualization is done using customized distributed rendering methods.

There are a number of open source efforts to provide visual analysis. Most of these solutions focus on the traditional post processing approach. Moreland et al provided an extensive survey of visualization pipelines and these efforts [Moreland, 2013].

1.4 Data Size

Large-scale scientific simulations typically output time-varying multivariate volumetric data, and current leading edge computers can perform computation power used in simulations is in petas-

cale range. Consequently, HPC based particle simulations generate petabytes of data that simply cannot be handled effectively by traditional methods. The need for new techniques and tools for storing, transferring, accessing, visualizing, and analyzing large datasets is pressing. For example, a commonly recognized simulation system dedicated to astrophysics and cosmology simulations runs petascale range (10^{15} floating point operations per second) which generates 64 x 10^9 particles with 36 bytes per particle for each time step. The calculation results in a time step of 64 billion particles with 36 bytes per particles, and takes 2.3 TB per time slice. Because of this industry standard scalable storage solution can save only a small fraction of the time slices. Because the data generation rate per time step is greater than this, it is not possible to store the complete set of the simulation data even within a time step. Under normal operating conditions, it takes several minutes to move a time-step worth of data to or from storage unit or through a network. This problem is expected to be much worse in the near future, as the storage read and write bandwidth has not kept pace with computational speed. Solutions to this problem include data reduction through sampling, data compression and in-situ visualization.

1.5 Data Reduction Methods

One way of reducing the burden of storage of HPC is to reduce the size of the data through advanced reduction techniques, such as sampling, compression, and feature extraction. Others involve restructuring the workflow. Following subsections summarize alternatives commonly used to reduce the size of datasets.

1.5.1 Sampling

Both spatial and temporal sampling methods can be applied. In spatial sampling, the spacing between saved data points is increased. In temporal sampling the interval between modal dumps is increased. It is also possible to save data at non-uniform temporal spacing. For example, denser temporal sampling can be done when there is more entropy (the rate of information contained in

a particular set) in the simulation. Another method to reduce the size of data is via statistical sampling of particles or level-of-detail sampling, where the information loss is across spatial dimensions. This approach has the benefit of knowing the approximation error (how accurately sample data represents the original data).

1.5.2 In-situ

One method of data reduction involves processing data at simulation time, which is recognized as in-situ, a Latin phrase that means on site or in position. The method is based on integrating more analysis with simulations during the execution of the simulation, instead of performing those tasks as a post-processing step, thus minimizing, and potentially eliminating, the data transfer need between the stages. A number of in-situ methods have been developed. Some implementations are based on memory sharing, and others share data through high speed networking. A basic in situ method is to compress the data before writing it out to storage instead of saving the full double precision of the computation. One in-situ approach uses parameterized or automated decision making that applies statistical sampling techniques to generate a representative subset of the full simulation data during run time of the simulation. Another method is to do automatic feature extraction in-situ and to save properties of these features. Several toolkits and libraries currently support in-situ visualization.

1.5.3 An in-situ solution based on images (Cinema)

A radically different solution proposed by Ahrens et al. [Ahrens et al., 2014] is called Cinema, an in-situ visualization and analysis framework based on computer generated imagery. The data reduction is attained through sacrificing some viewing parameters for visual analysis, and also through image based compression algorithms that are applied to the dataset.

In Cinema, simulation data are converted into image formats (e.g., png) according to predefined camera positions and angles in line with the simulation run. The set of images and some descrip-

tive information about them (e.g., camera positions, time steps, details about the visualization operators, and statistics about the data) are stored in a separate dedicated image based database. In order to reduce the data even further, these images are also transformed into (in-situ or after a model run) compressed image formats (e.g., jpg, mpeg) by using industry standard image compression algorithms. The imagery databases support basic meta-data and content queries to enable interactive exploration of the data set.

The data reduction through this solution is noteworthy. In traditional in-situ methods, to perform visual analysis tasks, the data files are moved through the network to a dedicated visualization computer after a model run. In Cinema, the entities transferred within the network are the image representation of the simulation data, which are generally smaller than the original model dumps. A typical simulation run (with multiple time steps) produces a data of 10^{15} in size. After saving the data into an imagery database, the size of the imagery data drops to about 10^{6} in size. Thus, the solution offers significant data reduction (order of 10^{9} in size), while preserving important elements of the simulations, and offering interactive exploration of the data during post-processing.

Most of the in-situ approaches, including Cinema, operate on a predefined set of analyses. In order to meet scientist needs, these solutions should preserve important elements of the simulations, significantly reduce the data needed while preserving these elements, and offer flexibility for post-processing analysis. However, the Cinema approach is also inflexible since only the views that have been previously chosen will be available for visual analysis. In general, any automatic selection of data at runtime reduces the type of questions that can be asked about the data during post processing analysis.

1.5.4 Visualization and Perception

From a visualization point of view, the halos simulation generates datasets consisting of clusters of particles, each having a position, mass and velocity in three dimensional space, and these particles can be represented as point clouds. Depth perception refers to the cognitive skill to de-

termine size and distance of objects and their spatial relationships. It is known that structural perception is more accurate when depth cues are available [Norman et al., 1995, Ware, 2012, Sollenberger and Milgram, 1993, Ware et al., 1996, Ware and Mitchell, 2005]. Visual space provides various sources of information about depth called depth cues. The perception of structures in three-dimensional point cloud data requires the use of appropriate visual spatial cues. For example, users can detect paths on a graph more accurately, when motion is provided as a depth cue. Furthermore, when stereo viewing is used in addition to motion cues, this further increases the size of the graph that can be perceived. It is likely that the most important depth cues for visualizing three dimensional point clouds are also kinetic depth and stereoscopic depth. However, the optimal combination of depth cue information is not known for point cloud data.

1.6 Problem Statement

The following subsections highlight the research problems that are addressed throughout this thesis.

Problem 1: Perceptual issues

The perception of structures in three-dimensional point cloud data requires the use of appropriate visual spatial cues. However, the optimal combination of depth cue information is not known.

Problem 2: Data size and data reduction methods

Common approaches to deal with large scale particle simulations use sampling-based data reduction methods, which result in data loss. A promising method (Cinema) is based on generating images from the simulation data by using predefined viewing conditions and let users to visualize the whole simulation but with limited viewing parameters. The specific question addressed in this thesis is how to effectively use Cinema to support the visualization of 3D structures in point

cloud data using visual spatial cues.

Problem 3: Interacting with point clouds

A number of techniques have been developed to interact with three-dimensional point clouds One class of methods involves driving a ray into the scene from the user's cursor until it intersects with the surface of the object of interest. This method has two problems when it comes to point cloud data. The first is that point clusters do not have a single surface, a ray might hit a point in front of the cluster or pass right through it and hit a point beyond it. The second problem is that sequentially searching millions of particles for one that intercepts a ray is inefficient and slow. Other methods that offer area or particle selection interactively constraining the area of interest are also available. These solutions require reorganization of the data and restructuring or indexing methods.

1.7 Approach

The approach is twofold: First, perceptual studies were carried out to determining how in-situ images can support point cloud visualization. Second, an interactive interface was built to explore point cloud data. The following subsections described two major goals of this thesis.

Goal 1: Enhance the imaged based in-situ solution

The first major goal of this thesis is to enhance the imaged based in-situ solution to support the visualization of point clouds. The solution to this problem is the computation of visualization products such as images in-situ along with the model computation. But the main perceptual cue for 3D point data is motion parallax and if images are the basis for motion parallax information many images will be needed.

Fewer images are more desirable from a storage perspective, and it is important to know how

few images can be used in order to still get a benefit from kinetic depth. Also, in order to support stereoscopic viewing, it is necessary to save a pair of images rather than a single image. In order to support kinetic depth perception, it is necessary to save a whole series of images based on a set of viewpoints orbiting a feature of interest, and for reasons of compactness. To investigate the cost benefit trade-offs, we designed and conducted the human factors experiments described in Chapter 4.

Goal 2: Interactive exploration of point clouds

The second major goal was to develop methods for rapidly exploring and visualizing a point cloud data set. This thesis proposes methods for interacting with large scale datasets represented as point clouds. The size of the original data generated by the simulation runs are too large to visualize and in some cases it is even not feasible to move through network or store the data into storage devices. In order to accomplish visualization and enable the interaction with point clouds, this thesis offers visualization methods for efficiently dealing with point cloud data sets consisting of more than 10^{12} point samples (16.7 x 10^6 particles, 32 bytes for each particle, 250 time steps).

1.8 Structure of the Work

We briefly introduce the application domain in Chapter 2. In Chapter 3, a literature review is presented with the focus on perceptual issues relating to the visual cues most important for the perception of features in point cloud data. In Chapter 4, two experiments concerning the optimal parameter for the visualization of point cloud data are presented. In Chapter 5, the software application that was developed for this thesis is explained. Finally, in Chapter 6 the contributions and major findings are summarized. Also some suggestions are given regarding the potential integration of the software with ParaView.

Chapter 2

Application Domain: Dark Matter in the Universe

2.1 Cosmology

The work in this thesis was motivated by the specific need for better ways of visualizing and interacting with the data from cosmological models. Understanding the formation of the universe and its expansion history is one of the prime interests of computational cosmology [Pope et al., 2010]. It is well established that most of the universe (perhaps %95 of it) consist of the material in some as yet not discovered form [Griest, 2002]. Although these materials are not observable using conventional instruments (e.g., ground and space telescopes), the existence of such material (or energy) are inferred from a number of indirect observations involving the cosmic microwave background, galactic dynamics, and gravitational lensing [Jarosik et al., 2011, Rubin et al., 1985, Markevitch et al., 2002].

Halos are theoretical constructs designed to make the universe behave as it should. Theoreticians suggests that halos are made up of Weakly Interacting Massive Particles (WIMPS) which only interact with visible matter by means of gravitational forces. However, observations suggest that universe does not follow the gravitational laws. For example, the angular rotation of galaxies does not vary as it should with distance from the galaxy center. One theory suggests that there exist unseen material or energy that we cannot observe. Dark matter and dark energy is the theory known to explain this phenomenon.

To test and refine this theory, the structure of halos is estimated by simulation models using large particle systems, such as The Roadrunner Universe MC^3.

In the Road Runner Universe, visible matter co-evolves, and is shaped by, dark matter. The resulting structure that emerges in the visible matter component is then compared to the observed structure of the universe. Current halo simulations consist of billions of particles and they begin with these particles forming shortly after the beginning of the universe, following which the particle systems are allowed to evolve through time to the present date. The particles form clusters and sub-clusters and these are the halos.

2.2 The Roadrunner Universe MC^3

Roadrunner (MC^3) is a large N-body cosmology simulation of dark matter physics [Habib et al., 2009]. It was the first computation to break the Peta-scale barrier (in 2008) using a computer system able to churn-out more than a thousand trillion calculations each second [Pope et al., 2010, White et al., 2009]. Figure 2.1 shows the view of the super computing servers.

The platform has been used for studying the structure and the formation of the Universe and its expansion history. The simulation associated with large volume cosmological surveys and observations, such as data collected from satellite missions, large sky area and volume galaxy surveys as well as time varying scans of the sky from space and ground telescopes, to produce cross-validated astrophysical baselines of the Universe [Bhattacharya et al., 2013].

2.3 Dataset

An MC^3 time step has 4000^3 (64 billion) particles with 36 bytes per particles takes 2.3 TB per time slice [Woodring et al., 2011]. As the universe is made up of around 10^{70} particles, each simulated particle represents about 10^{63} actual particles, which is the size of galaxy cluster [Haroz

Figure 2.1: Roadrunner, the first petascale platform in the world, a computer system able to churn-out more than a thousand trillion calculations each second. [LANL, 2016]

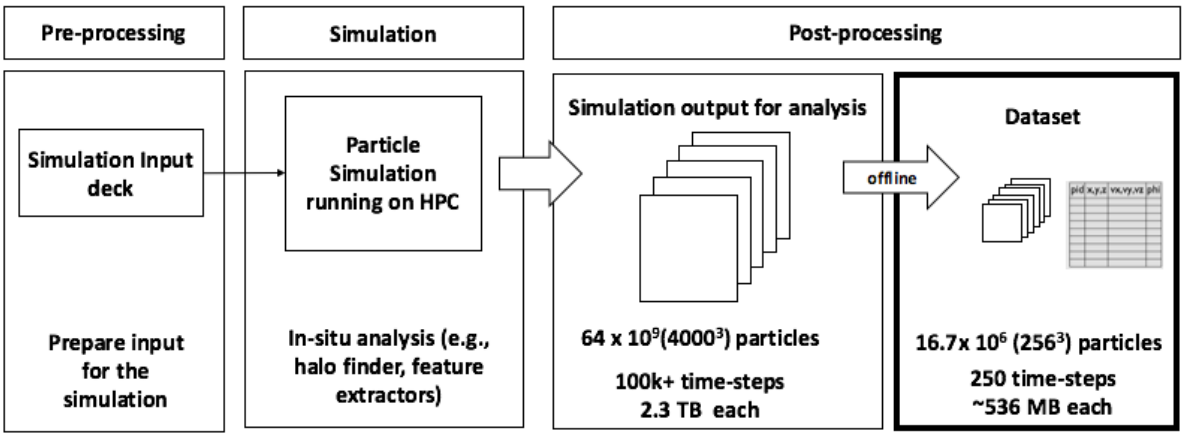


Figure 2.2: End-to-end pipeline from the generation of the data to post-processing stage.

et al., 2008, Haroz and Heitmann, 2008].

An overview of the workflow characteristic of high performance computer simulation is given in Figure 2.2. This shows three distinct stages: preprocessing (preparing the input), simulation (execution) and post-processing (analyzing and visualizing the simulation results).

The size of the original data generated by the simulation runs is too large to visualize and in some cases it is not even feasible to move or store the data locally: A typical run generates from hundreds to millions of time steps, and consequently full simulation output easily occupy petabytes of storage space. Most plausible approaches used to address visual analysis challenges around

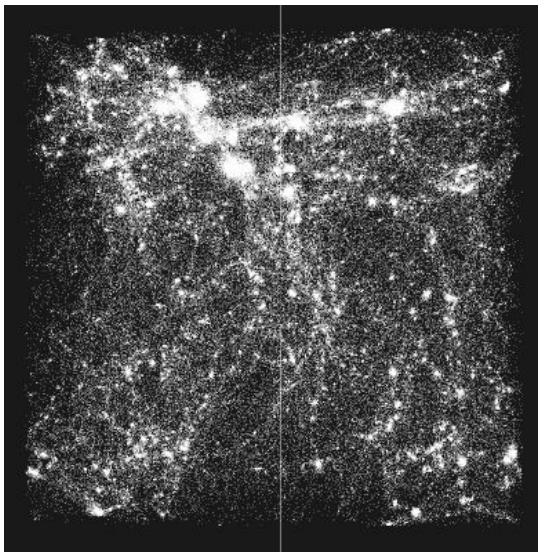


Figure 2.3: Halos data: a single time step with 1.67M particles.

large scale simulation outputs are based on advanced data reduction techniques (e.g., stratified statistical sampling [Woodring et al., 2011], compression [Rogers and Springmeyer, 2011], feature extraction [Ma et al., 2007]), and re-evaluating the processes and methods that are used in visualizing such simulation outputs [Moreland, 2013].

For this thesis the target data set, consists of 250 time steps stored in individual files, is a significantly smaller subset of the original simulation output. Each file includes $\sim$16.7 million (256^3) particles with 32 bytes per particle (position, velocity and mass values), and is $\sim$537 MB in size. The particles are also tagged with unique identification numbers. Figure 2.3 shows an example of a subset of halos simulation data at a single time-step.

Chapter 3

Perceptual Consideration

In this chapter, we review factors relating to the perception of structures in three-dimensional point clouds.

Three dimensional space is interpreted by the brain based on information from the images in the two eyes. The two dimensions orthogonal to the line of sight seem relatively unproblematic since the layout of objects in the visual field corresponds to the layout of the images of those objects as projected onto the retina (albeit reversed left to right and inverted). Information about the depth dimension is more complex, being provided by a set of *depth cues* which are combined by the brain in various ways [Palmer, 1999]. The relative values of different depth cues depends on the nature of the data being displayed and the task [Bradshaw et al., 2000, Ware, 2012].

The so called pictorial depth cues are those that can be represented using a static perspective picture. They include occlusion, linear perspective, shape from shading, and texture gradients, as well as cast shadows. None of these are likely to be very useful in the case of point clouds. For example there is little or no perspective information when the data consists of small points distributed through space. Also, occlusion, a powerful depth cue, whereby objects that block the view of other objects are perceived as closer, is not effective when the objects are small points.

One method for viewing point cloud data is to convert the point set to a 3D field using interpolation or energy functions centered on each particle [Shirley and Tuchman, 1990, Teunissen and Ebert, 2014]. This field can then be visualized using isosurface visualization or volume rendering. In this case, the 3D shape cues of shading and ambient occlusion could be applied [Sweet

and Ware, 2004, Tarini et al., 2006]. Although turning point clouds into surfaces can sometimes be a useful technique, it suffers from the disadvantage that many structures will be hidden due to occlusion.

In the present study, we chose to study the problem of visualizing point clouds as points without the additional step of constructing a field. Another method is to replace each point with a sphere. This can provide depth cues from occlusion and shadows, but is prohibitively expensive on large scale data. In the absence of useful pictorial cues, the most powerful depth cues are structure-from-motion and stereoscopic depth. Although these have not been studied for the visualization of point cloud data, there have been a number of studies of the importance of these cues for path tracing tasks [Sollenberger and Milgram, 1993, Ware and Franck, 1996, Van Beurden et al., 2010, Naepflin and Menozzi, 2001].

3.1 Structure from Motion

Structure-from-motion is a generic term encompassing both kinetic depth and motion parallax. When a three-dimensional form (such as a bent wire) is continuously rotated around an axis and its image is projected on to a flat plane, the observer vividly perceives the 3D shape [Wallach et al., 1953] even though at any instance the only available information is two dimensional. In the computation of 3D shape from motion the brain must assume that object layout is rigid, because if objects were to deform during rotating the motion due to deformation and the motion due to rotation would be impossible to separate. 3D perception from object rotation is called the kinetic depth effect. Motion parallax is the term used to describe the shear effect that occurs when an observer moves laterally with respect to the viewpoint. For example, when an observer looks sideways out of a moving vehicle, closer objects move through the field of view faster than objects at a distance. However, in some cases motion parallax and kinetic depth are essentially the same. For example, if an observer moves around an object (orbiting) while fixating that object, the resulting visual changes are the same as if the observer is fixed and the object rotates. Yet in

the former case this is called motion parallax in the latter it is called kinetic depth.

Both motion parallax and kinetic depth can be obtained through active motion of an observer. It occurs when someone moves their head laterally with respect to an object (motion parallax), or when they rotate a hand-held object (kinetic depth). It can also be obtained by means of passive motion, for example when an object is set rotating in a computer graphics system. Ware and Franck [Ware and Franck, 1996] found no significance between these conditions in terms of errors.

But when stereo was used, the best condition overall was hand coupled object motion. Hubona et al. found no difference between active and passive conditions but for a very different task, that of 3D object matching.

3.2 Stereoscopic Viewing

The stereoscopic depth cue comes from the way the brain interprets differences (called disparities) between the images in the two eyes [Howard and Rogers, 1995]. For example, in the situation illustrated in Figure 3.1, where a vertical bar is fixated, an object closer to the right has less angular separation in the right eye than in the left eye. This difference is the disparity. When disparities are too large, the brain is unable to fuse the two images and observers may perceive duplicated features, or one of the features may be suppressed. The area within which fusion can occur is called *Panum's fusional area*.

Various computer graphics tools and techniques are currently available to realize stereoscopic viewing conditions. One method is the use of different polarization on different rows of pixels. If the left and right eye images are interleaved correspondingly stereoscopic depth is obtained when appropriately polarized glasses are worn [Lazzaro et al., 1998]. To generate these images, polarizing filters that have different angles of polarization are applied to encode pairs of stereoscopic images. A polarized stereoscopic system uses special glasses to provide three-dimensional im-

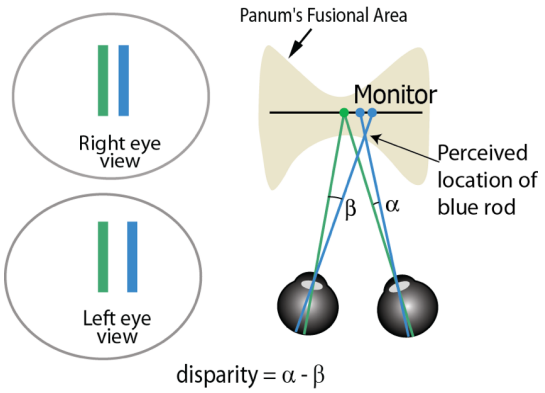


Figure 3.1: Stereoscopic depth is perceived from disparities between features imaged on the retinas in the two eye. The diagram shows the situation when two vertical bars are viewed on a stereo display monitor.

ages to each eye of the users by restricting the light that reaches each eye [Wikipedia, 2016, Light, 1989, Sheiman, 1988]. If the left and right eye images are shown an alternating lines of the monitor, and polarized differently, stereoscopic depth is obtained when appropriately polarized glasses are worn. Another method is to use shutter glasses synchronized with the frame buffer refresh cycle which allows for different images to be shown to the left and right eyes in alternation. Figure 3.1 shows the situation when two vertical bars are viewed on a stereo display monitor, and the stereoscopic depth is perceived from disparities between the features imaged on the retinas in the two eye.

3.3 Multiple Cues

A number of studies have investigated the relative benefits of motion and stereoscopic viewing. Most of these involved path tracing tasks. In some studies the stimuli consisted of wiggly lines connecting points in 3D space and the task was to find the correct terminus for a line with a highlighted start point [Naepflin and Menozzi, 2001, Van Beurden et al., 2010]. In other studies the stimuli consisted of tree or graph structures where the task was to determine whether a short path connected a pair of highlighted nodes [Sollenberger and Milgram, 1993, Ware et al., 1993, Ware

and Franck, 1996, Ware and Mitchell, 2005, Hassaine et al., 2010].

In six of these seven studies, motion was found to be a more effective cue than stereoscopic depth in terms of accuracy, when either one or the other cue was provided [Naepflin and Menozzi, 2001, Van Beurden et al., 2010, Sollenberger and Milgram, 1993, Ware et al., 1993, Ware and Franck, 1996, Ware and Mitchell, 2005]. In most of them the combination of motion and stereo depth was found to be the most effective of all. The exception is a study by Hassaine et al. that found a benefit for stereo viewing but not for motion [Hassaine et al., 2010]. They used a task identical to Ware and Frank but varied the viewpoint density [Ware and Franck, 1996]. Viewpoint density is the angular distance with which new views are provided during lateral head movements. However, a number of studies found that responses in stereoscopic viewing were faster than when structure from motion was the only cue (e.g. [Sollenberger and Milgram, 1993,Naepflin and Menozzi, 2001, Ware and Franck, 1996]. Table 3.1 provides a summary of studies concerning the effect of motion, stereo and their combination.

The path tracing task for 3D node-link diagrams seems similar to the problem of perception of structures in point clouds, because in both cases pictorial depth cues are unlikely to produce much benefit. For tasks other than path tracing, the relative values of different cues are likely to be different [Norman et al., 1995, Wanger et al., 1992, Hubona et al., 1997]. For example, visually guided reaching for objects may be better served by stereoscopic depth than motion parallax [Arsenault and Ware, 2004]. Indeed, one of the main purposes of stereoscopic depth perception may be to support eye hand coordination. There are also cases where stereo and motion may not lead to improved task performance. For example Barfield and Hendrix [Barfield et al., 1997] found no benefit to either stereo or motion cues when the task was matching objects seen in a VR headset to objects drawn on paper. However, they did report that stereo and motion gave them an enhanced sense of *presence*. IJsselsteijn et al., found that motion contributed more than stereo viewing to a sense of presence [IJsselsteijn et al., 2001].

The experiments reported here were in part motivated by the needs of in-situ visualization in

high performance computing (HPC) [Ahrens et al., 2014]. Briefly: because the compute power of cutting edge HPC systems is outpacing the speed at which extreme datasets can be written to permanent storage, it is becoming increasingly difficult to save sufficient data to support post-simulation analytic visualization [Ahern et al., 2011]. The solution to this problem is the computation of visualization products, such as images, in-situ along with the model computation. These visualizations can be much more compact than the complete checkpoint-restart frames (a complete model state dump) that have been until recently the basis for most visualization. But if images are the basis for motion parallax information, then many images will be needed. From a storage perspective, fewer images are more desirable, so it is important to know how few images can be used in order to still get a benefit from kinetic depth.

Article	Techniques	Remarks
[Naepflin and Menozzi, 2001]	Motion: head coupled displa, Stereo: polarizing filters	Motion performed better than stereo in terms of error rat, Motion has less response time, Combination resulted lowest error rate, When stereo is in use, significantly lower time to complete, Stereo provides faster times to completion then motio, Remark: motion needs time consuming movements and performed if necessary
[Sollenberger and Milgram, 1993]	Motion: multiple forms, auto rotation, coupled, Stereo: field-sequential 30Hz each eye	Stereo is better than non-stereo, Rotational motion is superior to discrete mutiple angle, Motion performed better than stere, Combination of motion and stereo performed bes, Remark: When simple rotation, motion is no better than stere, Remark: no contriution of subject-controlled rotation
[Van Beurden et al., 2010]	Motion: multiple forms of motion, compared with/o head coupling, Stereo: glasses free 3D display	Stereo without motion did no increase the accuracy, but positive effect on completion time Remark: No effect of stereo was found on accuracy, contradicts with earlier studies (Solenber 1993 and Ware 1993) suggesting that stereo improved the accuracy and decreased the completion time
[Hassaine et al., 2010]	Stereo: multiview display Motion: head-based motion parallax	Stereo improved task accuracy and reduced latency more than motion. Motion has no evidence of dependence on motion (viewpoint density) Motion had no effect on accuracy when stereopsis was available, head motion parallax did not have any effect on task performance. benefit of motion parallax and stereopsis depends greatly on the task Motion: not have any additive effect with stereo on task performace Explanation: less occlusion, if other cues available (e.g., perpective and shading correspondence is less difficult and motion is less effective)
[Ware and Franck, 1996]	Motion: multiple forms, w/o head coupling	Motion is more important than stereo Head-coupling perfromed better than rocking back and fourth about a vertical axis Remarks: Agreed with Solenberg and Milgram 1993
[Ware et al., 1993]	Motion: multiple forms, w/o head coupling Stereo: frame sequential stereoscopic display, shutter glasses 60Hz each eye	Motion is more helpful than stereo, type of motion is not important Combination of stereo and motion performed best Stereo contributed surprisingly high Remarks: Super-additivity of depth cues observed
[Ware and Mitchell, 2005]	Motion: smooth motion, no coupling Stereo : HD display, mirror based Wheatstone	Head-coupling better perfomed than others such as back and fourth about a vertical axis Motion resulted the lowest error rate Combination is the second Remarks: Agreed with Solenberg and Milgram 1993 , Ware And Franck 1996,

Table 3.1: Summary of studies concerning the effect of motion, stereo and their combination. Motion has been found to be a more effective cue than stereoscopic depth in terms of accuracy, when either one or the other is available.

Chapter 4

Human Factors Experiments

In order to support stereoscopic viewing it is necessary to save pairs of images. But in order to support kinetic depth perception it is necessary to save entire sequences of images from a set of viewpoints orbiting a feature of interest, and for reasons of compactness. To investigate the cost benefit tradeoffs, the first experiment was designed to look at the issue of the number of unique frames needed to support kinetic depth perception and also the relative benefits of kinetic depth and stereoscopic depth. The second experiment was designed to look at the amplitude of oscillation for the kinetic depth effect.

Using structure-from-motion cues in scientific visualization applications can lead to a problem of viewpoint preservation. People often wish to contemplate a structure from a particular viewpoint, but rotation causes a continuously change in the viewpoint. A partial solution is to generate kinetic depth by oscillatory rotation about a fixed axis. If the amplitude of oscillation is relatively small then the viewpoint can be approximately preserved. We generate oscillation using the function shown in Figure 4.1.

$$Angle of Rotation = \alpha * \sin(t * 2\pi/\lambda)$$

where α is the amplitude of oscillation and λ is the period of oscillation.

Figure 4.1: The function used for generating oscillatory rotation

Our hypotheses are shown in Table 4.1. We also measured response times, but did not generate formal hypotheses for this variable.

H1: Kinetic depth will improve target detection in point cloud data.
H2: Stereo viewing will improve target detection in point cloud data.
H3: Kinetic depth will be more effective than stereo viewing in improving target detection in point cloud data when motion is smooth.
H4: The value of kinetic depth will be reduced for lower update rates.
H5: The value of kinetic depth will be reduced for smaller amplitudes of motion.

Table 4.1: Hypotheses

4.1 Experiment 1

The objective of the first experiment was to discover the relative benefits of stereo and motion viewing conditions in the detection of three-dimensional patterns in point clouds. The approach taken was to use artificial point clouds resembling the halos computed by domain scientists. Using artificial rather than real data makes it possible to generate large number of trials containing unique patterns. Artificial targets were placed within the point clouds, and study participants were trained to detect them.

Figure 4.2 shows a sample view of the artificial *universe* consisting of 1.63M particles.

The whole scene was either stationary or in oscillatory motion. The motion was generated by rotating around a central vertical axis at the center of the point cloud using the equation in Figure 4.1. The period of oscillation was 2 seconds. and the amplitude of oscillation was 15.0 degrees.

4.1.1 Conditions

Six distinct redraw rates were compared (0, 3, 6, 15, 30, and 60 updates per second). There were also two viewing conditions: stereo display and no-stereo display. A redraw rate of zero results in no motion so this is provides a condition with no kinetic depth.

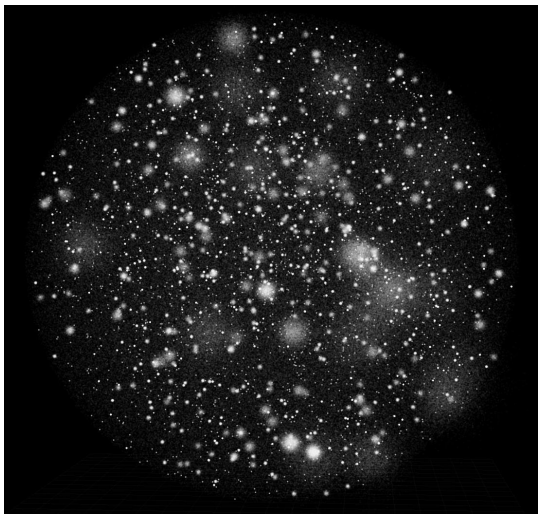


Figure 4.2: Artificially generated background consist of 1.63M particles in 3D space

4.1.2 Test Patterns

Our test data was artificially generated to have some of the qualities of dark matter simulations. To provide a background *universe* for the target features five distinct sizes of point clusters were generated and randomly positioned in the sphere as described in Table 4.2.

Each cluster contained the number of points specified normally distributed with the specified standard deviation. The background had radius of 16.0 cm. All points were rendered as pixels with a transparency of 0.2.

On a given trial there was either no target, or one of three types of target pattern was generated and randomly placed within the background field within a radius of 0.75 of the overall radius.

The three target patterns were as follows:

1. **Filament:** A set of clusters on a natural cubic spline. The spline path was drawn through five control points which were randomly chosen from a spherical space having a radius 3.8 cm. Seven clusters (2 large, 2 medium and 3 small) were uniformly distributed along the spline path.

2. **Satellite:** Cluster with multiple satellite of clusters. A cluster of points was generated using 7500 points normally distributed with a standard deviation of 0.28 cm. Ten satellite clusters (4 large, 3 medium and 3 small) were randomly positioned around the center cluster at a distance of 1.2 cm

3. **Enhanced filament:** Enhanced pattern based on double filaments. Two filaments were constructed. A cluster of 4500 particles (sd=0.32 cm) was positioned at their point of intersection.

Examples of the three target pattern types are given in Figure 4.3.

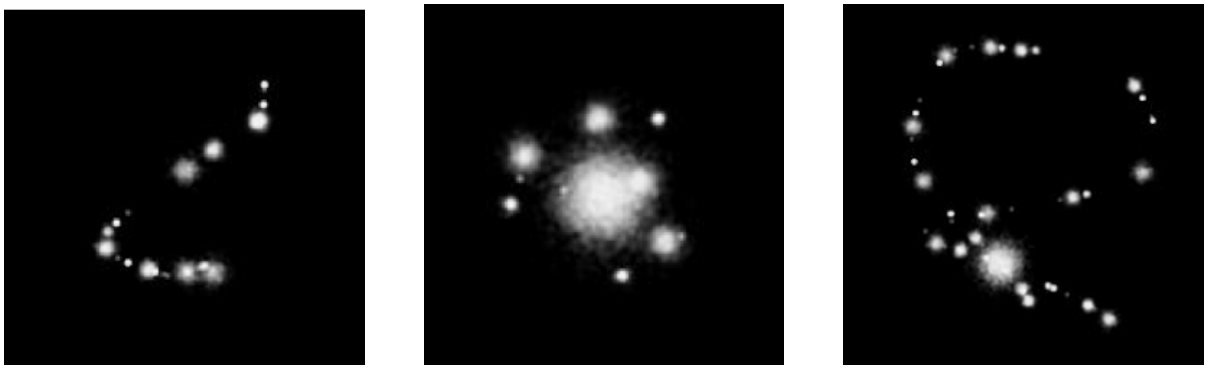
Background	Number of clusters	Number of particles in each cluster	Number of particles	Minimum sd (cm)	Maximum sd (cm)
X-Large clusters	25	12500	312500	0.14	0.84
Large clusters	70	2250	15700	0.032	0.14
Medium clusters	700	700	490000	0.038	0.067
Small clusters	2000	60	120000	0.0094	0.038
Tiny clusters	350000	1	350000	1 pixel	1 pixel
Total	352795	-	1430000	-	-

Table 4.2: Parameters used to generate background data

The code that we used to generate the dataset can be reached through the code repository link provided on Appendix C.

4.1.3 Task

The participants' task was to respond *YES* or *NO* depending on whether they detected a target in the background pattern. The display was presented for two seconds, and then the screen was cleared. They responded using one of two specially marked keys on the keyboard. [Yes] (Overlaid on the V-key) if the pattern was present and [No] (overlaid on the N-Key) if it was not. Participants were instructed to leave their left index finger on the [No] key and the right index finger on the [Yes] key.



(a) Filament (b) Satellite (c) Enhanced Filament

Figure 4.3: Sample structures randomly created on three-dimensional space.

4.1.4 Participants

There were 14 participants (12 females and 2 males) on the first experiment. They were paid for participating.

4.1.5 Procedure

There were 36 main conditions (6 redraw rates x 3 test patterns x stereo vs no stereo). Trials were given in blocks of 60 for each test pattern. Each block was constructed of 5 trials at each redraw rate with the target present and 5 at each redraw rate with the target absent. All the trials in a block of 10 had the same type of target patterns, although the actual target pattern was generated for each trial based on random parameters. The first three blocks used the three different test patterns given in a random order, and this was repeated with a new random order. There were six trial blocks for the stereo condition and six for the no-stereo condition with stereo and no-stereo conditions being given on different days, randomly assigned as stereo first and no-stereo first. Overall there were 720 trials (10 target-present trials and 10 target-absent trials in each of the 36 main conditions).

Prior to the experiment proper, a training session was given where participants ran through a re-

duced set of conditions. Only three of the redraw rates were used (60, 15, 0) and there were 6 trials per condition. In this phase participants were given directions on what to look for and corrected when they made an error. Prior to each trial block, participants were shown two examples of the target pattern with no simulated halos background.

4.1.6 Display/Hardware

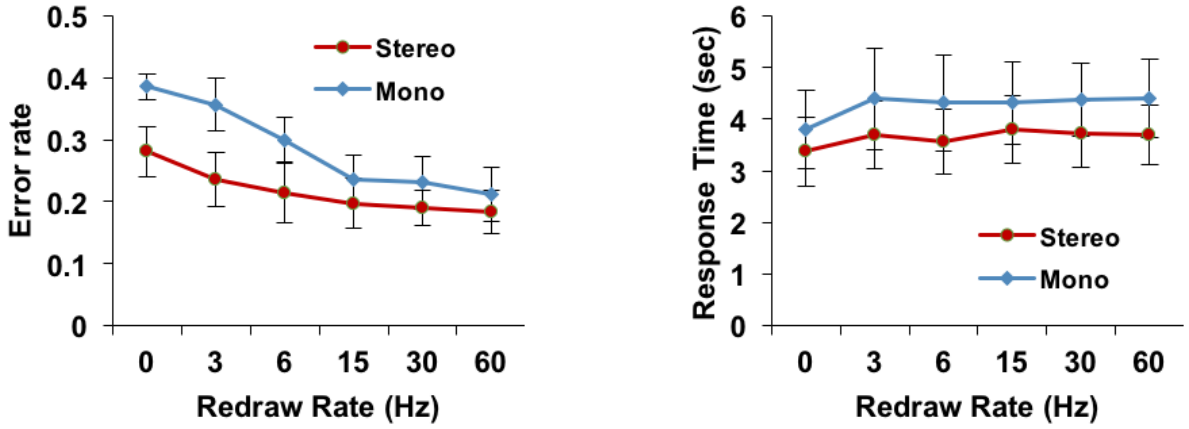
Stereoscopic imagery was presented using a frame sequential display with alternating left and right eye images and synchronized shutter glasses, each eye receiving a 60 Hz refresh rate. The perspective was based on a viewing distance of 75 cm from the screen and the experimenter was instructed to seat participants at this distance. The center of the simulated universe was place a further 20 cm behind the screen.

The hardware consisted of a NVIDIA Quadro 4000 graphics card, NVIDIA 3D Vision P854 Wireless Active 3D Glasses and Acer HN274H 1920x1080@120 Hz monitor with a screen 59.3 cm x 33.6 cm.

4.1.7 Results

One of the 14 participants had an exceptionally high error rate, performing at chance on some conditions. That data was removed and the analysis was done based on data from the remaining 13 participants. Figure 4.4 summarize the results from the first experiment.

As Figure 4.4a illustrates, error rate performance for stereo was always better than performance with no-stereo for a given update rate. The error data were subjected to a 2-way repeated measures ANOVA (redraw rate x stereo/no-stereo). Hypotheses H1,H2 and H4 were supported. There were highly significant main effects for both redraw rate [($F[5,60]=26.9$, $p < 0.001$) and stereo/no-stereo ($F[1,12]=27.9$, $p < 0.001$). There was also a significant interaction ($F[5,10]=3.2$, $p < 0.05$ with a Greenhous-Geisser sphericity correction). The interaction suggests that the benefit of



(a) Mean error

(b) Mean response

Figure 4.4: Experiment 1 results. Vertical bars represent 95% confidence intervals.

stereoscopic display decreases as the motion becomes smoother.

The response time data were subjected to a 2-way repeated measures ANOVA (redraw rate x stereo/no-stereo). There were significant main effects for both redraw rate ($F[5,60]=5.93$, $p<0.01$) and stereo/no-stereo ($F[1,12]=5.77$, $p<0.05$) (both with Greenhouse-Geisser sphericity correction). There was no significant interaction. The time to respond was shorter (on average by approximately 630 ms) for stereo viewing. Response times increased with the redraw rate.

4.2 Experiment 2

The goal of the second experiment was to determine how target detection varied with the amplitude of the motion. Because small amplitude motion is desirable, we were interested in finding out how much amplitude could be reduced before performance declined significantly. In most respects the procedure was the same as for the first experiment. The same method was used to generate both the background point field and the target patterns. In all conditions the redraw rate was 30 Hz.

4.2.1 Conditions

Six amplitude (angle of rotation) values were compared (0, 1, 2, 3, 8, and 16 degrees) both with and without stereo viewing. Error rate and time to respond were dependent measures.

Trial blocking was the same as for the first experiment, with the different amplitudes of motion substituted for the different redraw rates. As with the first experiment the experiment was run over two different sessions with stereo for one session and no stereo for the other (order randomly selected).

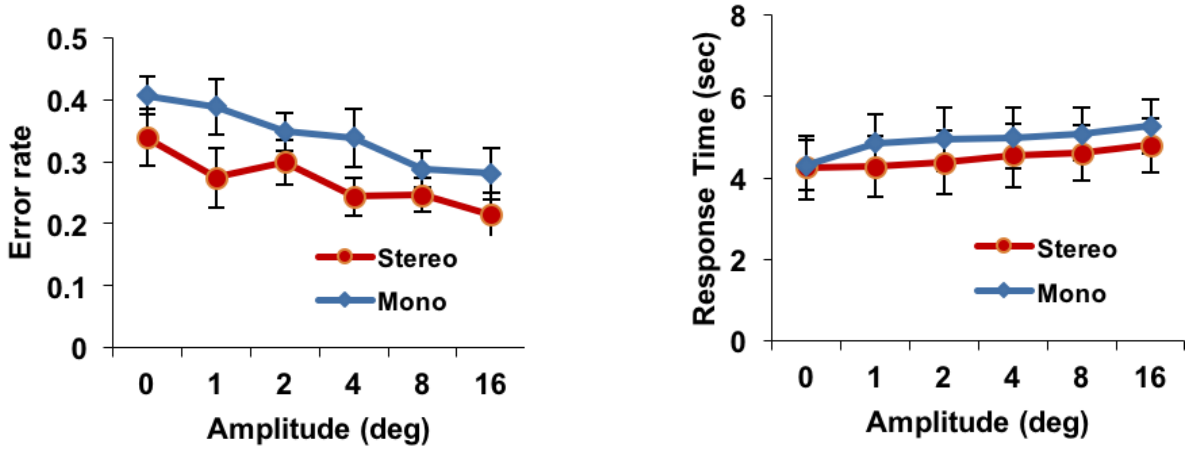
4.2.2 Participants

There were 14 participants (5 females and 9 males) on the second experiment. They were all paid for participating.

4.2.3 Results from Experiment 2

Figure 4.5 summarize the results from the second experiment. As Figure 4.5a illustrates, error rate performance for stereo is always better than performance with no stereo for a given update rate. The error data were subjected to a 2-way repeated measures ANOVA (amplitude x stereo/mono). There were highly significant main effects for both amplitude ($F[5,65]=22.7$, $p < 0.001$) and stereo/mono ($F[1,13]=39.3$; $p < 0.001$). There was no significant interaction.

The response time data were subjected to a 2-way repeated measures ANOVA (redraw rate x stereo/mono). There was significant main effects for amplitude ($F[5,60]=14.6$, but no significant effect for stereo/mono. There was a significant interaction effect ($F[5,65] = 2.19$; $p < 0.05$ with Greenhouse-Geisser correction). Greater amplitudes resulted in slower responses.



(a) Mean error

(b) Mean response

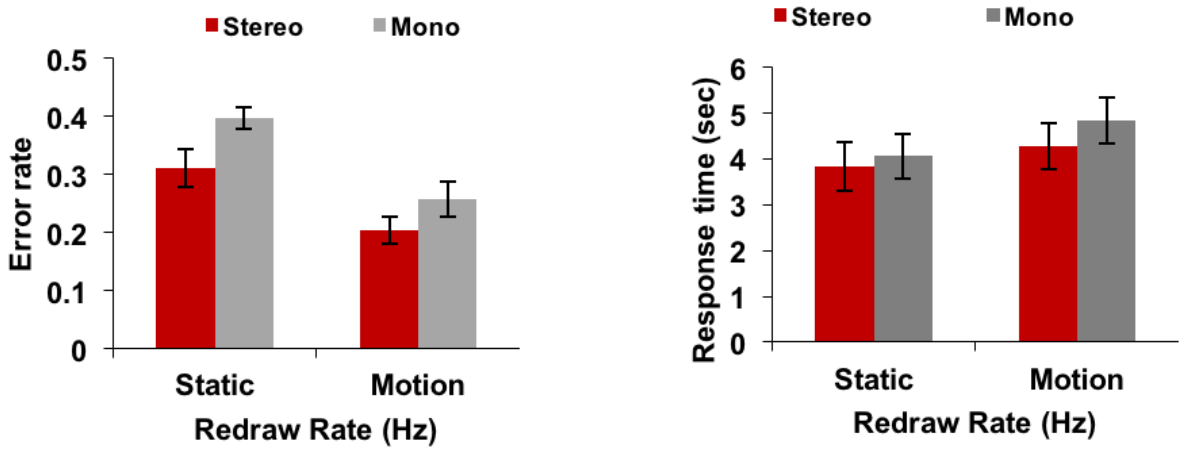
Figure 4.5: Experiment 2 results. Vertical bars represent 95% confidence intervals.

4.3 Meta-Analysis

Four of the conditions from the two experiments were close to identical. Both experiments had identical stereo and no-stereo conditions with no motion. Experiment 1 had conditions with an amplitude of 15 degrees and 30 Hz update rate (stereo and no-stereo). Whereas Experiment 2 had the same conditions with an amplitude of 16 degree. Accordingly we combined the data for these conditions yielding 27 participants. The results are summarized in Figure 4.6a and Figure 4.6b.

We subjected this to a 2-way repeated measure ANOVA for errors. There was a main effect for stereo/mono ($F[1,26] = 142.9$; $p < 0.001$) and a significant main effect for motion ($F1,26] = 37.0$; $p < 0.001$), with no interaction. The overall effect of stereo was a 21% reduction in errors. The overall effect of motion was a 35% reduction in errors. Combining motion and stereo viewing reduced errors by 49%.

Figure 4.6b shows time to respond results. A two-way repeated measures ANOVA revealed main effects for both stereo/mono ($F[1,26]=142$; $p < 0.001$) and for motion ($F(1,26)=36.9$; $p < 0.001$) with no interaction. Having stereo resulted in faster responses, but having motion resulted in slower responses.



(a) Mean error

(b) Mean response

Figure 4.6: Experiment 1 and 2 results combined. Vertical bars represent 95% confidence intervals.

4.4 Discussion

This study has been the first to look at the relative values of stereo and motion cues for the task of finding features in point cloud data. It suggests that both kinetic depth and motion are useful cues for helping people find features in point clouds, and they should be applied wherever possible. The utility of motion was found to increase with amplitude, but good results were found for amplitudes of 4 degrees and greater.

The results are broadly in agreement with previous studies using path tracing tasks. As discussed in our introduction six of seven prior studies found structure-from-motion to be a more powerful cue than stereoscopic viewing in terms of error rates for the task of tracing 3D paths. Also like some of these prior studies we found that structure-from-motion increased response times.

Motion was not a stronger cue for all conditions, its value was found to vary strongly with both the number of distinct frames per second and the amplitude. Motion was only a more effective cue for amplitudes greater than 4 deg and with update rates of at least 15 Hz. The majority of

prior experiments have only used a single motion condition and a single stereo condition (an exception is [Hassaine et al., 2010]). As a broad comment on these kinds of experiments, general statements, such as motion was a more powerful cue than stereo, must always have the caveat for the particular viewing conditions used. Obviously if motion is very slow, or much too fast its value will be minimal.

There are also different stereo viewing parameters that can be adjusted. For example, virtual eye separation can be adjusted to increase or decrease disparities [Ware et al., 1998]. It might be thought that enhancing disparities is the best approach, but because Panums' fusional area can be small, reducing virtual eye separation may be better if the features of interest have a substantial depth extent.

These kinds of consideration all suggest that for any given visualization design, many factors must be considered. Are the users likely to be willing to invest in stereo viewing hardware? If not, motion is really the only option for enhancing raw point cloud data. Is target detection the paramount consideration? If it is, motion and stereo viewing will be the best choice (assuming stereo is available). Is interaction necessary? If so, it may be necessary to halt motion to facilitate selection, and in this case stereo viewing will facilitate eye hand coordination. Is the goal the presentation of results to a larger audience? In this case, we must consider the fact that stereo viewing is not yet widely available whereas the ability to show short animated movies is becoming ubiquitous so structure from motion is a more useful cue. On the other hand, if virtual reality headsets become widely available, then providing both cues may become commonplace.

There are many aspects of this problem still to be explored. Although we varied amplitude of oscillation and update rate, we always had the same frequency of oscillation. Also there is the issue of active and passive motion for point cloud data. There have only been a few studies of this issue with path tracing tasks and in any case, we cannot assume that the same results would be found for point cloud data with different kinds of tasks.

There is also the nature of the data itself. Point cloud data is being generated in vast quantities by

laser mapping technologies. In some cases there is a need to view and edit the raw data [Wand et al., 2007]. It is likely that the results we obtained for the simulated halos model data would also apply to laser scans, but this is not certain. In the case of laser mapping, the data tend to lie on distinct surfaces unlike the data we simulated and this could lead to different outcomes.

In conclusion, the results from this study suggest that for the task of perceiving features in point clouds structure from motion is a more valuable depth than stereoscopic viewing, although ideally both could be used since the effects were found to be additive. But motion should be reasonably smooth and if motion is oscillatory an amplitude of at least four degrees should be used.

Chapter 5

The Interactive Interface

In this chapter, a tool designed to let users interactively navigate within a point cloud dataset is proposed. The optimum viewing conditions (such as redraw rate, the form of the motion to generate motion parallax, frustum dimensions, and so on) that were studied on the formal experiments discussed in Chapter 4 are taken into account. A commonly used flow visualization technique (streamlets) is implemented as a visualization method in order to represent particle and cluster orientations based on their position and velocities. User requirements elicited from domain experts based on a literature survey are also taken into account in the application design. In the following sections, first, the rationale for developing the interactive application is laid, and a literature review concerning methods for interacting with virtual environments, specifically with point clouds, is presented. Finally, the software application developed for this thesis is described.

5.1 Rationale

One of the major goals of this thesis is to develop methods for rapidly exploring and visualizing a point cloud data set. This section outlines functional requirements derived from user expectations on performing the visual analysis task on a large point cloud dataset. The target audience is the scientists working in the field of computational cosmology.

5.1.1 Functional Requirements

The following list outlines the functional requirements in designing the interactive exploration application:

FR1: Interaction

(a) Ability to navigate the viewpoint easily and intuitively through the point cloud.

(b) Ability to scale the scene up and down about a point of interest.

(c) Ability to select a subset of particles and animate their trajectories through space and time from the beginning of the modeled universe to the present.

FR2: Data management.

In order to provide visualization and interactive exploration of such a large scale dataset, which consists of particle samples distributed into separate time step files, a data management method for efficiently dealing with the dataset is needed. A solution likely to be in a form of indexing and spatial partitioning. The system should support a structure such that, as the system user navigates within the data set, the desired set of particle values should efficiently be pulled from the corresponding data file.

FR3: Visualization parameters

(a) Representing particle velocities

A method is needed to represent both the speed and 3D direction of particles in the point cloud.

(b) Tracing particles over time

Visualizing traces of particles or clusters within a spherical point of interest (represents a focal area of interest as if the user is holding a 3D scope inside the virtual world) is desired. Observations of clustering events of halos, such as their birth, death,

merge, split and continuation, should be observed. To do this, retrieving the traces of particle clusters from the multiple time-steps is also required, as the data is stored in separate files for each time step.

(c) Support for optimum viewing parameters

In Chapter 4, two formal experiments evaluated perceptual requirements for perceiving 3D structures in point clouds. The system should consider the parameters tested in these studies. First, an auto-rotation of the whole scene with a type of oscillatory (or rocking) fashion to enable the contribution of depth cues (kinetic depth, motion parallax) should be available. Second, the system user may choose to utilize stereoscopic viewing features, and the system should support appropriate stereoscopic viewing peripherals, such as shutter glasses, high resolution computer monitors and projectors that support over 120Hz refresh rates.

FR4: Software Considerations

There are a number of open source efforts that aim to provide visual analysis solutions for data intensive analytics needs. O The system architecture designed as part of this thesis should be compliant with the pipeline structure, so that the artifacts of this thesis, the interactive application, could be turned into modules (data reader, renderer), which can be then integrated into one of these libraries as a plug-in. Secondly, the system should utilize GPU programming to parallelize calculations (such as velocity representation calculations) when possible for optimization and improved performance.

5.2 Literature Review

The fundamental forms of interaction in a computer-generated world can be categorized into four basic actions: navigation, selection, manipulation and data input by a system user [Mine, 1995]. These actions can be performed in a variety of modes, such as direct user interactions

(e.g., hand tracking, gesture recognition, pointing, gaze direction) or through using controllers which can be virtual or physical (e.g., buttons, touchpads, pens, sliders, dials, joysticks, wheels). Figure 5.1 presents different methods and devices that are developed to interact with scientific datasets. However, the methods to interact with a virtual environment depend on not only the task at hand but also the structure of the dataset. Therefore, the interaction method should ideally be designed for the specific nature of the dataset and the features of interest within it. The following section presents a literature survey of interaction methods that are related to the problems of viewing point cloud datasets. Also considered are open source software libraries and toolkits that are commonly used in handling large scale point cloud datasets to develop visual analysis applications.

5.2.1 Interacting with Point Clouds

A number of techniques have been developed to interact with 3D objects in virtual worlds. Most of the studies are based on spatial selection and involves driving a ray into the scene towards the object of interest from the user's cursor until it intersects with the surface of the object [Pierce et al., 1997, Hand, 1997, Zeleznik et al., 1997]. In computational geometry, this method is recognized as ray-surface intersection or ray-casting. Grossman et al. provide an extensive survey of 3D selection techniques based on ray casting for volumetric displays [Grossman and Balakrishnan, 2004].

There are also methods developed to interact specifically with point cloud [Wiebel et al., 2012, Remondino, 2003, Scheiblauer, 2014, Bacim et al., 2014] The common idea of these techniques is based on a spatial specification of a sub-space that contains the elements to be further explored. In order to make interaction simpler, these subspaces are defined using primitive geometric forms, such as cones, cubes, circles, lasso, polygons, which usually do not exist within the original data [Liang and Green, 1994, Pfister et al., 2000, Bowman et al., 2001, Steed and Parker, 2004, Yu et al., 2016, Valkov et al., 2011, Shan et al., 2012]. Figure 5.1a shows examples of simple geometric

shapes that are commonly used to define a sub space to interact with point cloud data.

A group of methods involve free-hand gestures; however, these techniques require dedicated 3D input devices. For example, Bacim et al. used sub spaces controlled through 3D gestures Figure 5.1f. Butkiewicz et al. use a multi-touch based 3D positioning in a form of coupling fingers and the cursor to manipulate the 3D subspace for this purpose [Butkiewicz and Ware, 2011]. Figure 5.1e shows this technique (pantograph) in use. Others used 3D mice, optical tracking devices and so on. For example, Figure 5.1c shows a cubic mouse designed to manipulate a virtual object in 3D space by using a cube-shaped device with three rods coupled with primary axes [Frohlich and Plate, 2000, Bowman et al., 2001]. Another example, Cabral et al positioned a cube with two handed sphere [Cabral et al., 2014].

Another group of techniques use conventional user pointing devices, such as mouse and keyboard, to manipulate subspaces in the virtual environment by using geometric shapes calculated as a function of user inputs and predefined geometric forms. For example, Wiebel's method allows users to select spatial position in volumetric renderings by simply enclosing an area of interest by drawing a free form shape on the 2D projection of the 3D dataset, which is then used to derive a sub-space within the volume [Wiebel et al., 2012]. Another example is Yu et al.'s method presented in Figure 5.1b [Yu et al., 2012]. This method is based on a shape drawn on the 2D screen to define a cone based subspace in a 3D dataset. Then a grid is constructed by estimating densities of the particles inside it, in order to segment the sub-space into further smaller sub spaces. A grid constructed in the box of selected area, and a smaller cubes are used to enclose sub areas to further investigate (Figure 5.1a). An enhanced version of this approach is also available that a solution utilizes scoring functions to interactively constraint the area of interest (or the sub-space) as the user navigates within the dataset [Haan et al., 2005].

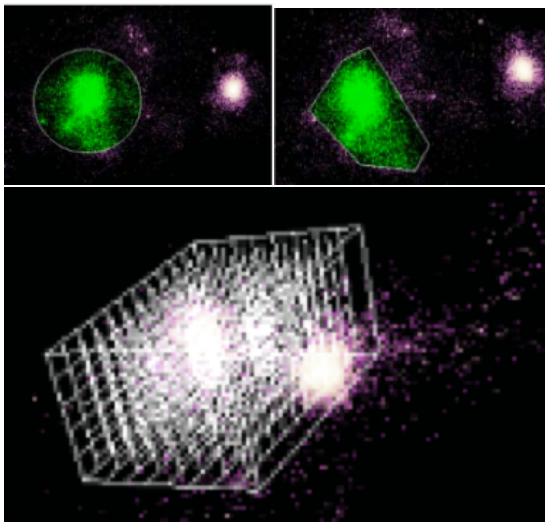
One form of ray-casting technique is based on progressively reducing the set of selectable objects to refine the area of interest to make the selection task with lower precision [Kopper et al., 2011]. However, for point cloud datasets, object selection methods do not apply because groups or clus-

ters of particles are more important for analysis than any single particle. Therefore, instead of gradually simplifying the area of interest, increasing the number of points in a sub-space is more suitable (because such datasets likely have much more particle data available than possiblly be visualized on the screen).

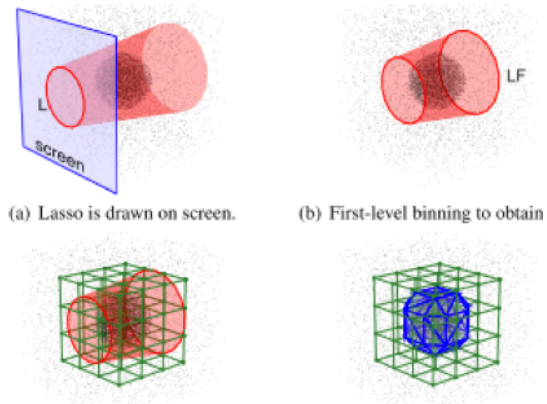
Spatial selection by using ray casting methods have two problems when it comes to point cloud data. The first is that point clusters do not have a single surface, a ray might hit a point in front of the cluster or pass right through it and hit a point beyond it. The second problem is that sequentially searching millions of particles for one that intercepts a ray will relatively be inefficient and slow.

5.2.2 Open Source Toolkits and Libraries

There are a number of open source visualization libraries and toolkits that provide visual analysis solutions for data intensive analytics needs. As stated in the Section 1.3, these tools generally focus on a traditional post processing approach to data visualization for high performance computing. Moreland et al. provide an extensive survey of visualization pipelines and open source visualization libraries. Ma et al. reviewed these solutions from in-situ processing and visualization perspective [Ma et al., 2007]. The following are examples of open source software libraries and frameworks that are commonly used to in data analysis, visualization and data exploration tasks: Application Visualization System (AVS) [Upson et al., 1989], DataVis [Hils, 1991] apE [Dyer, 1990], Vista [Tuchman et al., 1991], VISAGE [Schroeder et al., 1992], VASE [Jablonowski et al., 1993], Iris Explorer [Foulser, 1995], OpenDX [Abram, 1995], CUMULVS [Geist II et al., 1997], SCIRun [Parker and Johnson, 1995, Johnson et al., 1999], VisIT [LLNL, 2005] and the Visualization Toolkit (VTK) [Schroeder et al., 1996]. Visualization applications, such as ParaView [Squillacote et al., , Ahrens et al., 2005], VisTrails [Bavoil et al., 2005], and Mayavi [Ramachandran and Varoquaux, 2011], are also commonly used by the domain experts to perform the analysis tasks via graphical user interfaces.

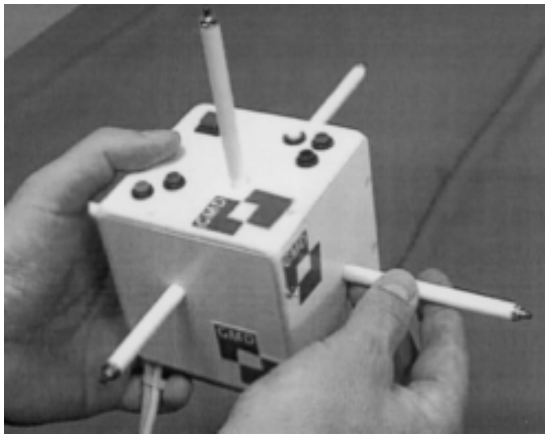


(a) Different forms of primitive shapes are used to define a sub space to interact with point cloud data [Yu et al., 2012]

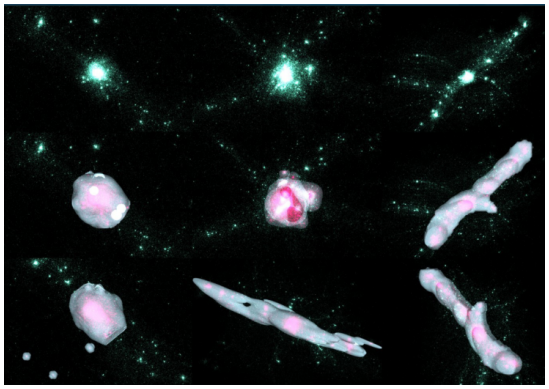


(a) Lasso is drawn on screen. (b) First-level binning to obtain

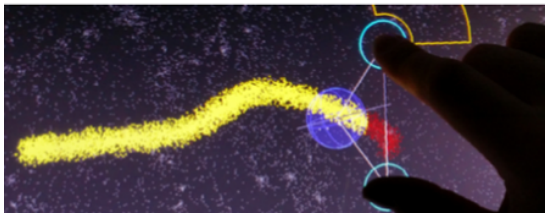
(b) A shape is drawn on 2D screen to generate a cone based subspace in 3D dataset. Estimating the densities of the particles, a grid is constructed to segment the subspace into further smaller sub spaces, called marching cubes (Yu et al)



(c) The cubic mouse designed to interact a virtual object in a 3D data space [Bowman et al., 2001]



(d) A context aware technique selecting different forms of clusters in a point cloud datasets [Yu et al., 2016]



(e) An interaction method using fingers of a single hand on a touch screen, recognized as pantograph (Butkiewicz et al.)

(f) A technique for interacting with point cloud dataset by using gestures and free hand movements to annotate point cloud datasets. The dataset is iteratively refined as user divides the space [Bacim et al., 2013]

Figure 5.1: Different forms of interacting with point clouds. Most of the techniques are based on specification of 3D subspace that contains elements to be further analyzed and progressive refinement strategies

The following are the most related ones for this thesis.

1. Visualization Toolkit (VTK)

   VTK is a software library for 3D computer graphics, image processing, and visualization [Schroeder and Martin, 2005]. It consists of C++ class libraries and supports interface layers including ParaView and ParaView Catalyst.

2. ParaView

   ParaView is an open-source data analysis and visualization application that enables users to build visualizations to analyze their data using qualitative and quantitative techniques, and enables interactive data exploration in 3D [Henderson, 2004]. The application supports large datasets using distributed memory computing resources.

3. ParaView Catalyst

   Catalyst is a software library for in situ analysis and visualization [Ayachit et al., 2015]. The library provides methods to integrate simulation and post-processing stages, which can be used to develop analyses that are executed alongside the simulation run (in the same address space) by using C++ or Python programming languages. The library is designed to inter-operate with the VTK and ParaView and enables simulations to perform analysis in-line with a running simulation, output relevant data, and visualize concurrently as the simulation performs.

The following software libraries were used in the course of this thesis for specific purposes.

1. OpenGL Mathematics (GLM)

   GLM is a C++ mathematics library for graphics software based on the OpenGL Shading Language (GLSL) specifications [Riccio, 2014]. The library is header only library and platform independent. In this thesis, the library is used to efficiently calculate matrix transformations (such as translate, rotate, scale) and containers for vector and matrix.

2. Point Cloud Library (PCL)

   PCL is an open source library for 3D geometry processing and handling data represented as point clouds [Rusu and Cousins, 2011]. The library is mainly generated for robots to work in unstructured environments to be able to perceive the environment that they operate. However, it contains useful data structures and algorithms. It includes 3D processing algorithms that operate on point cloud data, including: filtering, feature estimation, surface reconstruction, model fitting, segmentation and so on. Furthermore, each data structure and algorithm is defined via base classes that attempt to integrate all the common functionality used via a processing pipeline architecture [Rusu and Cousins, 2011, Aldoma et al., 2012]. Finding all neighbors of a point inside a given radius is provided via a range search operation. Elseberg evaluated different strategies of nearest-neighbor-search operations and octrees implementations [Elseberg et al., 2012]. In this thesis, the library is used to create indexes, octree building and range search operations (search within a radius). PCL library is used for its optimized performance in octree implementation and nearest-neighbor-search operations.

3. Computer Vision Library (OpenCV)

   OpenCV is an open source computer vision and machine learning software library that provides extensive number of optimized algorithms for computer vision and image processing [Bradski et al., 2000]. In this thesis, the library is used to generate high-resolution videos from a series of still images.

5.3 Software Application

5.3.1 High Level Design

During the course of this thesis, an application has been developed that supports the basic requirements listed in subsection 5.1.1. The interaction method proposed as a part of this thesis is based on maintaining a single point of interest at the center of the screen, and bringing relevant features to that point of interest by translating the entire scene. As the user navigates, more details are brought into the interest areas on demand in order to generate visualization, such that when an interesting cluster, or set of structures, is identified on the screen, the system retrieves the traces of selected particles to show them as an animated sequence.

5.3.2 User Interface

The user interface consists of a main window that provides the visualization, and a side window with three helper viewports to provide guidelines. The main window consists of two components: the first is the background generated from particle samples that are randomly selected form a single time-step data file. The particles are visualized as points (that have same size and alpha values) by using the position values. The second component is the abstract spherical area of interest that is positioned in the center of the screen and maintained during the interaction. The side window contains three helper viewports. At the top is a view of the data space showing particle velocities, in the middle is an orthogonal side view of the point cloud, and at the bottom is a top-down view to provide spatial orientation within the overall data space. Figure 5.2 illustrates the elements of the interface.

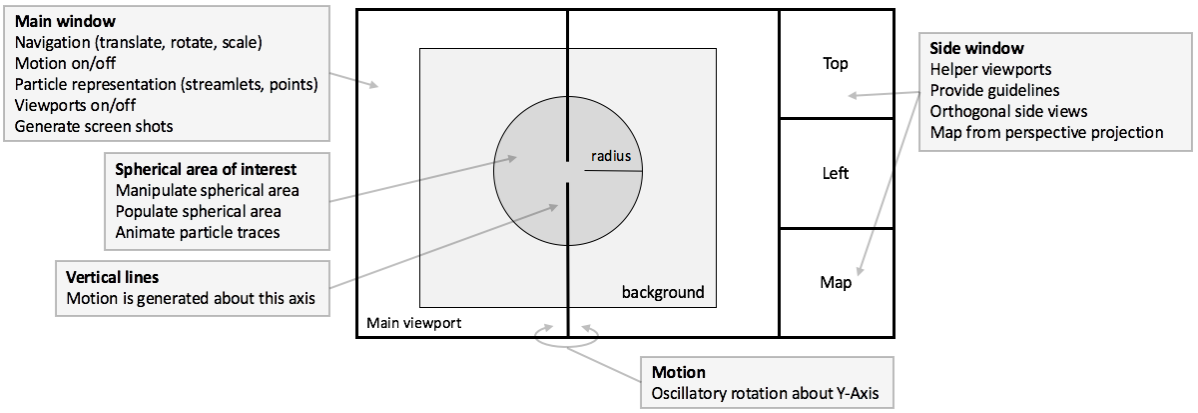


Figure 5.2: The User interface design to support the interactive exploration of the data set

5.3.3 Navigation

The center of data space is translated to a fixed point within the viewing frustum as the user navigates, and more particle data around the spatial center of interest is maintained by populating with particle data around it by retrieving more data during the interaction. The mouse and keyboard are used as input devices for the navigation.

While initializing the application, the center of the data space is positioned at the center of the frustum. This is realized by generating the frustum then translating the whole scene to a position, such that the center of the dataspace appears on the frustum center. The user interface provides a vertical line that appears on the center of the screen. This guideline is parallel to Y-axis and passes through the dataspace center, which is also the frustum center. This feature can be turned on and off during the interaction.

When the user clicks on the screen, the dataspace center is translated to that position. This is done by using composite transformation matrixes. First, the distance between the user click position and the screen center is measured. This point defines a ray from the viewpoint into the day space. The angular difference between this ray and a ray through the center of the workspace is used calculate a rotation angle. The whole scene is rotated, using a smooth animation so that the

ray through the cursor becomes aligned with the center of the screen. This means that any features behind the cursor position appear to move to the center of the screen. Further adjustment can possibly be done by scrolling the wheel of the mouse to change the depth with respect to the center of the workspace. This enables the feature of interest to be translated to the center of rotation. This change in depth value is applied to the translation matrix. Finally, a composite transformation matrix which is the product of these matrixes, is calculated and provided to the GPU for the transformation calculations. The zoom is done by scaling the whole world space and can be manipulated by user inputs from the keyboard. Figure 5.8a presents a screen capture of the application as the user interactively navigates within a single time-step by using the application interface.

5.3.4 Spherical area of interest

The spherical focal area is provided within the three-dimensional space as shown on Figure 5.3. As the user navigates, more particles are brought into this region, . The spherical shape is formed by adding extra particle samples within a given radius of 8 cm on the projection plane. The number of particles within the spherical area become more than the background surrounding it, therefore the spherical region become visible enough. The user can also interact with the spherical area to populate more particle samples from the same time step by translating a new center of interest to the center of the screen and changing the radius. Furthermore, the particle ids within the area can also be used to bring properties of particles from different time slices to observe animated particle traces over time. Moreover, visualization methods (such as representing particles as points, using velocities to generate streamlets, and animating particle traces in the temporal domain) are coupled with mouse and keyboard shortcuts that can be toggled during the interaction.

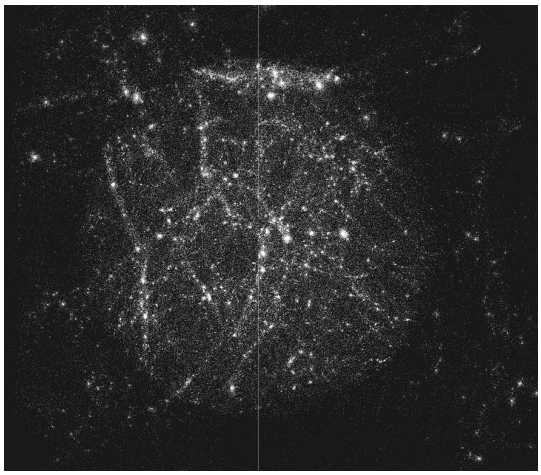


Figure 5.3: Spherical area of interest on three dimensional space. The number of points in the center spherical area of interest is updated on demand as the user navigates.

5.3.5 Motion

In order to benefit from the kinetic depth cue in perceiving data in point clouds, motion is generated by auto-rotation of the whole scene with an oscillatory (or rocking) fashion around the Y-axis. The period of oscillation defaults to 2 seconds, and the amplitude of oscillation is 15.0 degrees. The motion is smooth because all point and line rendering is done in the GPU. The user can enable or disable the auto-motion feature at any time during interaction from the keyboard.

5.3.6 Stereoscopic viewing

The application provides stereoscopic viewing conditions. The stereo viewing mode requires special hardware (monitor and shutter glasses). The application is developed to provide stereoscopic viewing specifically with a monitor capable of providing refresh ratesof 120Hz and above. The user also needs to wear special shutter glasses to benefit from this feature. The application mode (stereo or mono) is selected from user input (run the application binary with the –s switch) before the application starts.

5.3.7 Visualization Parameters

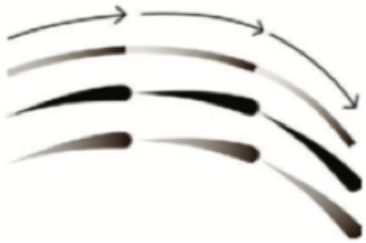
One purpose of the software is to visualize the orientation and direction of the particles throughout time by visual representation of velocities via a set of streamlets. Ware et al.'s strokes or streamlets used for this purpose [Fowler and Ware, 1989, Ware et al., 2014, Butkiewicz and Ware, 2011]. By using visualization parameters, such as streamlet opacity, streamlet line width, streamlet color, the attributes of the streamlets that are generated to visualization the particle direction were possibly controlled. The following subsections briefly describe the technique and the parameters used for this purpose.

Streamlets

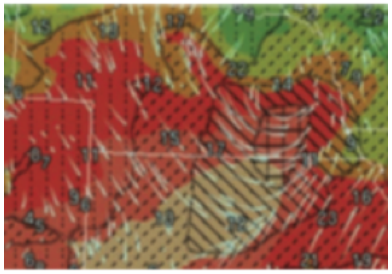
Research suggests that streamlets (streamline elements, or teardrop streamlets, that have a much stronger head than tail) are one of the best solutions to show a vector field map [Ware et al., 2014]. There are also studies that were primarily focused on determining values for the streamlet parameters, such as how to represent flow speed, the head and tail size, and the head and tail transparency of the streamlets [Mitchell et al., 2009]. For this purpose, we implemented a method for generating streamlets that have a much more salient head than tail. The streamlet design is shown on Figure 5.4. We used these methods to design streamlets for both static and animated versions to represent time series. The head tail difference of the streamlets generated by varying alpha value (opacity) from 0 to 1 within the length of the representation.

Streamlet colors

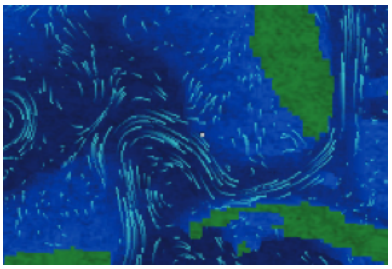
Color is used to help clarify the direction in which particles are traveling. The method used in this study is adapted from that used for coloring 3D line fields representing brain fiber tracts [Demiralp et al., 2009]. To assign colors to directions we use a color space where direction of travel with respect the XYZ axes are assign RGB colors. The amount of redness, greenness and



(a) Tear drop streamlet illustrations, salient head than tail strokes clearly shows the direction. (Ware et al.)

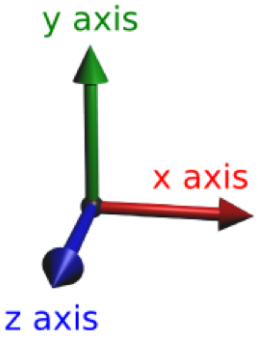


(b) A forecast weather map displays static and animated streamlets to represent wind patterns. (Ware et al.)

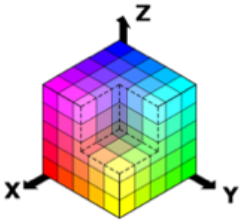


(c) An ocean current dataset represented as streamlets, revealing the current ring (Butkiewicz et al.)

Figure 5.4: Streamline elements, or teardrop streamlets, have a much stronger head than tail.



(a)



(b)

Figure 5.5: Particles are colored with a color a function of direction: The absolute value of particle velocities(vx, vy, vz) are corresponded with color(RGB), which resulted a color space.

blueness are a function of the absolute particle speed with the respect to the X, Y and X axes(Figure 5.5). Figure 5.7 shows velocities and the directions of the particles. The user can select colored and monochrome versions of the streamlets. These features can enabled both in the background and in the spherical area. The user can turn enable and disable by using keyboard inputs. Figure 5.7 shows a screen capture of velocity representations for the whole background, and Figure 5.8d shows particles within the spherical point of interest rendered as streamlets by using velocity and position values.

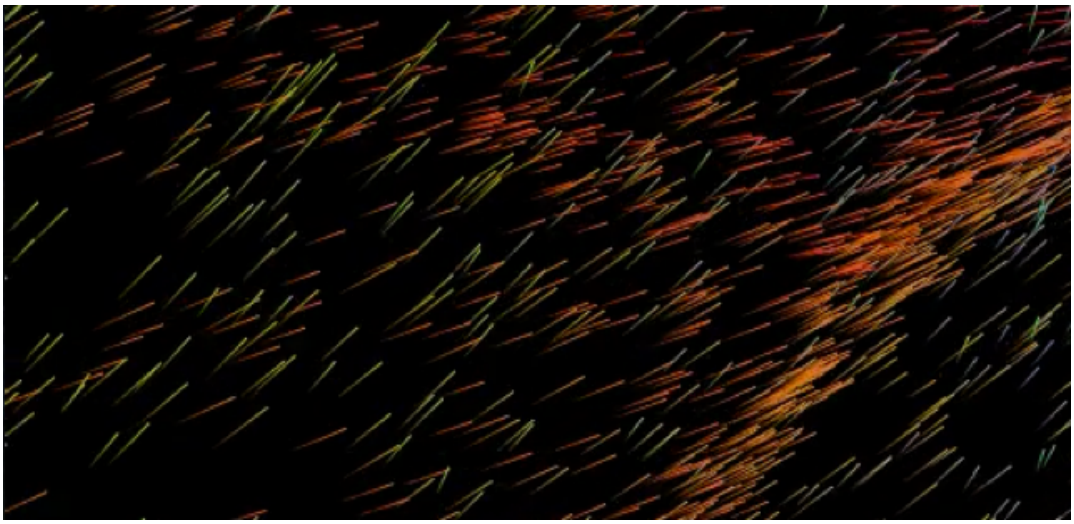


Figure 5.6: Streamlets present the speed and direction of particles

Streamlet width and length

Particle traces are represented as 2 pixel wide lines. In the static representation of streamlets, the length is calculated as a function of the position and velocity value of each particle. The streamlets start from the position (xx, yy, zz) coordinates and end at a position that is calculated by adding the velocity value of the particle (vx, vy, vz) to the position. This makes the length of the line segments proportional to the speed (Figure 5.6).

5.3.8 Tracing particles over time

The application provides a method for displaying particle traces over time from the beginning of the simulated universe to the present day. A spherical point of interest is maintained at the center of the workspace and as the user navigates, more detail is generated in this central region. The user also has the option of selecting a group of particles and requesting a time series animation within the spherical area. The particle IDs within the spherical area are used to retrieve properties of particles from multiple time step files to generate animated particle traces over time. The application brings in data from a number of time steps by using an internal index to facilitate search

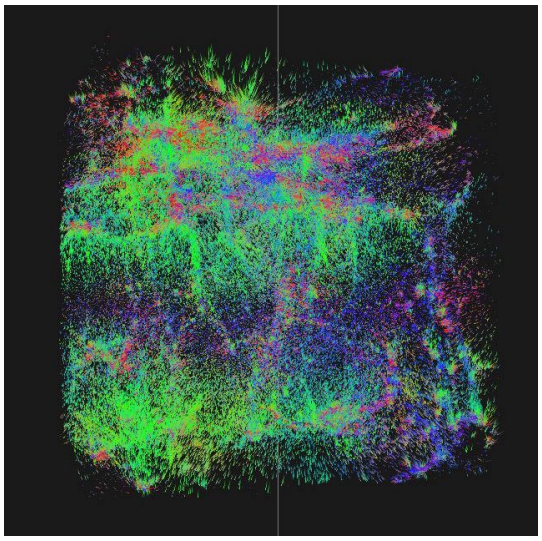


Figure 5.7: Velocity and directions

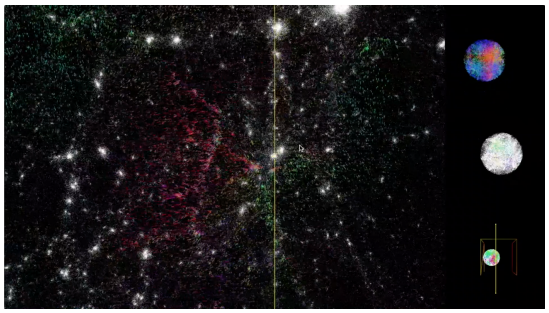
in the original data files. The software then animates the selected group of particles using streamlets or simply using transparent point clouds as a representation method. The streamlets are colored using the same method developed for the vectors shown in an individual time step. In this way it is possible to see how halo structures develop and migrate through the evolution of the universe.

Figure 5.8 shows sample images of an animation specifically generated by using every second of time steps (125 out of 250 available).

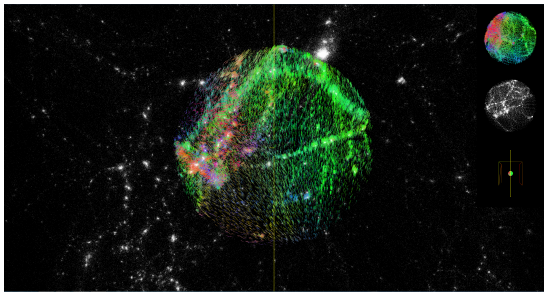
5.4 Implementation

5.4.1 Overview

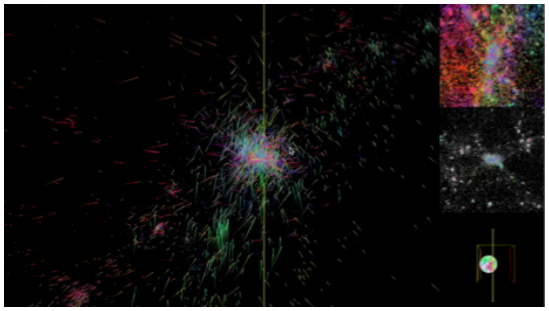
This section describes the software that has been developed to support interactive visualization as described in the previous sections. A large portion of the software was developed from scratch, and care was taken to comply with the pipeline design metaphor to handle the data and parameter flow between the different components of the application. The size of the complete source code



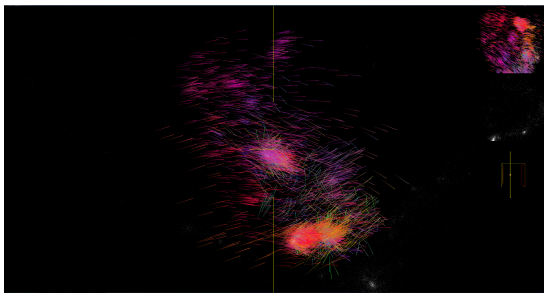
(a) The user can navigate within the point cloud dataset by using the interface



(b) Particles within the spherical point of interest are rendered as streamlets.

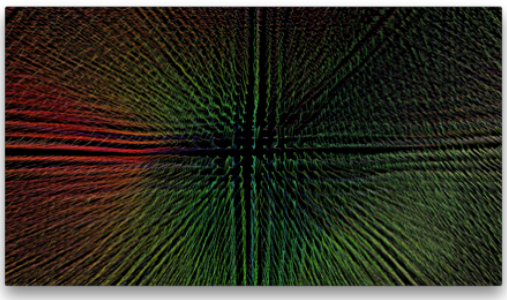


(c) The user observes a cluster by using the interactive interface

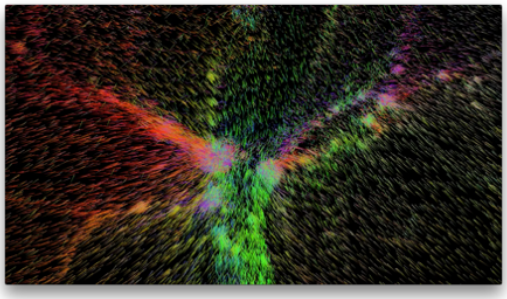


(d) A closer look at clustering of particles represented as streamlets.

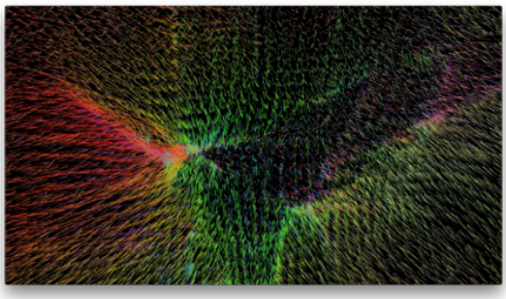
Figure 5.8: The user is guided via a vertical line that points targets the center of interest. Three helper view-ports are aligned on the right: first, top view of the particles as velocity representation, second left view as points , and third map view of the spherical area. More images can be found on the link provided in Appendix C of this thesis.



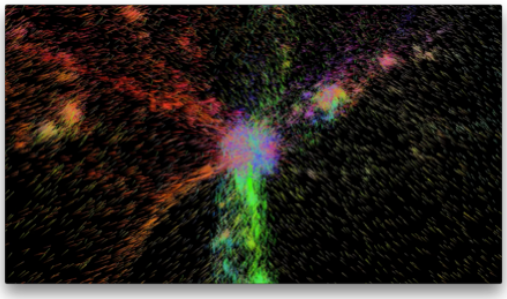
(a) time 0



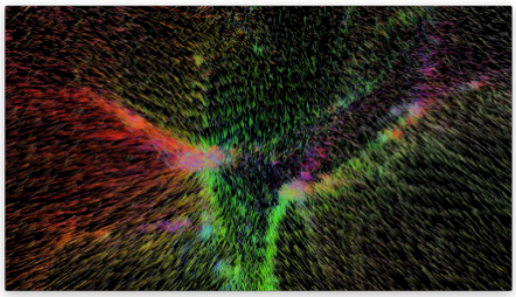
(d) time 150



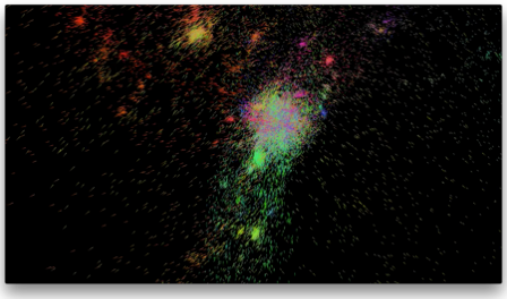
(b) time 50



(e) time 200



(c) time 100



(f) time 250

Figure 5.9: The particles initially arranged evenly on a 3D grid. As the simulation proceeds, the particles formed dense clusters and sparse regions. Starting from time step 100, the clusters become more visible. The animated version of this images can be found on the link provided in the Appendix C of this thesis.

grew to more than a two thousand lines of code, excluding external libraries and comments. No other parties contributed to the development of the application.

The design of the application is compatible with the software libraries commonly used for visualization in high performance computing, so that the outcomes of this thesis can be used in developing a new application. For example, the code developed so far can be compiled as a ParaView plugin.

5.4.2 Data Management (Spatial Partitioning and Range Search Operations)

The target data set consists of more than 10^{12} particle samples and stored in separate files, one for each time step. Each file includes 16.7 million particles with 32 bytes per particle (position, velocity and mass values) resulting in 537 MB per file.

In order to efficiently access particles (represented as points) stored on disk (or in memory), they have to be spatially ordered; this requires a data structure that organizes the particles in a form of level-of detail hierarchy. One of the best alternative for this purpose is the octree and this is the structure used here.

All points are read from the file into the octree data structure.

An octree is a tree-based data structure for managing sparse three dimensional data. Figure 5.10 illustrates the structure to partition three-dimensional space by recursively subdividing into octants, and shows the tree representation of an octree. Each internal node has exactly eight children. First, a point cloud structure is defined and instantiated as a data structure to contain the point data that will be read from the time step file. Then an octree instance is initialized with a specified resolution for its leaf nodes. The implementation of the octree data structure keeps a vector of point indices within its leaf nodes. The resolution parameter describes the length of the smallest voxels at the lowest octree level. The depth of the octree is therefore a function of the resolution as well as the spatial dimension of the point cloud data. As the data is ingested into

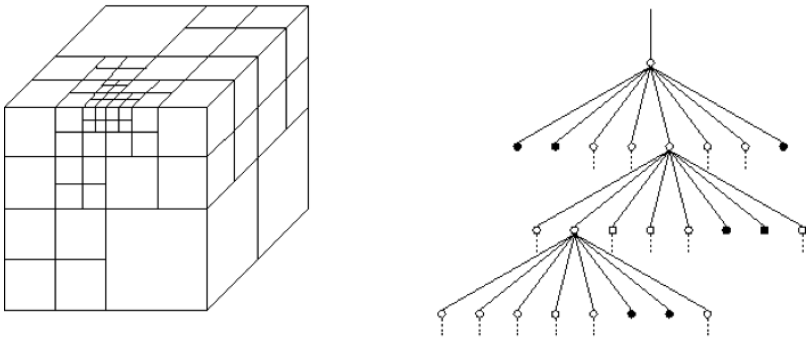


Figure 5.10: The three-dimensional space is recursively partitioned into Octants by using an Octree data structure. The subdivision of the space is representation with cubes on the left, tree illustration is on the right.

the data structure, the bounding box dimensions of the dataset is also calculated. Once the data is filled into the octree, it's possible to perform range search operations by using three parameters: center, radius and maximum number of points that limits the result set. The search operations assign the search point to the corresponding leaf node and returns a vector of point indices. These indices relate to points which fall within the same node or voxel. The distance between the search point and the search result depend on the depth parameter of the octree.

5.4.3 Approach

The original time step file is organized as an array of structures, in which multivariate data delineating a single particle (position, velocity and mass) is stored. Figure 5.11 illustrates the organization of the dataset and the logical particle data structure. There is also a unique particle identification number (pid) for each particle, so that particle properties can be reached by simply pointing to the correct position on the binary data file. For example, a method (seekg) from C++11 standard libraries can reach any properties of a particle.

The following approach is used to support the functional requirements. First, a complete time step file is sequentially scanned and partially loaded into a hash map consisting of tag position value in main memory. An index file is created from tag and position values, stored and saved.

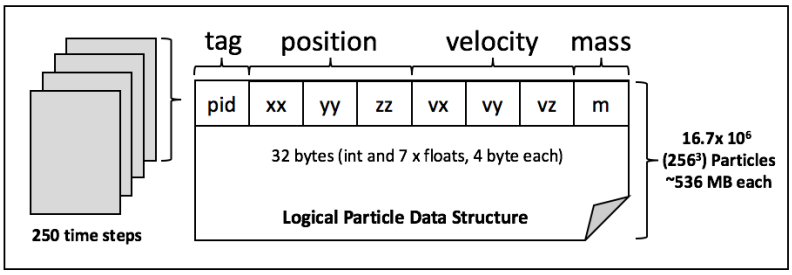


Figure 5.11: Logical Particle Data Structure

This saves the tag position pairs in a more compact form than the original file. Second, in order to support range search operations, an octree data structure is instantiated and populated. Building up an octree data structure requires key and value pairs. The spatial position (x, y and z) of the particle is used to build the spatial hierarchy but only particle ids are stored in the leaf nodes, thus, memory usage is minimized while supporting range search operations.

Finally, the visualization is generated following two sampling operations. The first sampling operation involves randomly picking particle samples from the data set that has been loaded. The particle ids are stored in a set data structure. These points provide a background for more focused aspects of the visualization. The number of particles is sampled down to 1.67M particle samples which is 10% of a single time slice. The 1.67M particles offer sufficient information to provide spatially awareness inside the dataset during the navigation, especially when kinetic depth and/or stereoscopic viewing conditions are in use. The second sampling operation supports the visualization of a specific region of interest within the scene. The radius of this region remains constant in user coordinates. However, it changes in data coordinates as the user scales the scene up and down around the region of interest. A range search operation is performed supported by the octree data structure. The result set of this operation is also stored in the main memory as a set data structure.

The elements of two sets (initial sampling and the result of the range search operation) are used to generate the visualization. The upper bound for range search result is set to 0.35 x 10^6 particle

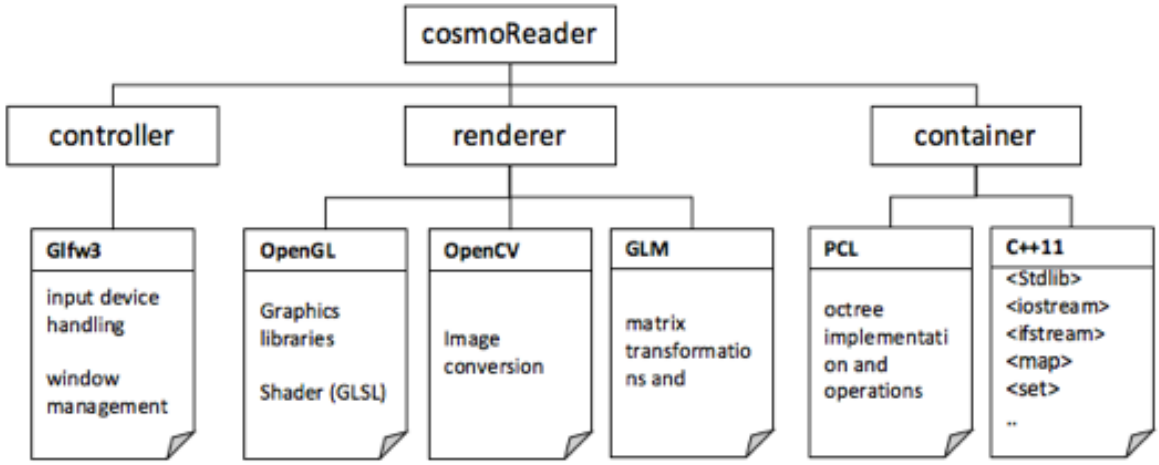


Figure 5.12: The main libraries and dependencies are represented as a class diagram. OpenGL, OpenCV, PCL, and GLM are external dependencies.

samples and consequently, the number of particles that can be displayed for visualization is the sum of the background particles and the focal area, which is always less than 2.02 x 10^6.

Particle sample size to generate the background (10% of a single time step)	:	1.67 x 10^6
Upper bound for range search result set (max number of points within the spherical area of interest)	:	0.35 x 10^6
Maximum number of particles to create visualization	:	2.02 x 10^6

Table 5.1: Particle sample size to create the background, search result limit in the spherical area of interest, and maximum number of particles that can possibly be used to create visualization.

5.4.4 Design

In Figure 5.12, the main classes and dependencies of the application are shown as a UML class diagram.

The main application consists of a main window that is derived from a C++ class. All other windows of the application are registered as children of the main window. The application is depen-

dent on PCL for the management of the different objects in the 3D scene. A C++ class called *Container* is developed is responsible for collecting the data and organizing it for fast retrieval operations such as time slices and rendering from the different view-ports. The point clouds are all part of a single class node, which is intended to render a single scene graph object. The *Renderer* class, however, manages all point clouds on its own and does the point rendering with direct OpenGL calls, so it does not use dependent methods or libraries for rendering.

Graphics library related methods that require initialization, like setting up OpenGL states and drawing methods, loading and settings parameters for the shaders, managing vertex buffer object (VBO), and so on, are collected in a separate class called *Renderer*. Mouse and Keyboard interactions are collected in a separate class called *Controller*.

5.4.5 Development

The application is developed in C++ compliant with the v11 standards (C++11). OpenGL (version 3.2) is used as the graphics library. OpenGL Shading language (GLSL) is used for GPU manipulations. Glfw3, an open source library for creating cross-platform windowing, reading user input from keyboard and mouse and handling the events, is used. Rendering the scenes into images files tasks are done by using the OpenCV image processing library. The point cloud library (PCL) was used for its optimized IO and Octree operations. The source code is mostly developed in Sublime-text editor on sMac OSX and Windows based computers. CMake and GNU Make was used to manage the multi-platform compilation process. The application is tested with Microsoft Windows (Win7x64), Mac OSX (Sierra), and Linux (Red hat Enterprise 6.5) based operating systems

5.4.6 Rendering

The Stereoscopic view requires the generation of pairs of images and the projection of those images seperately into each eye of the observer. The method used to set up stereoscopic display

uses parallel axis asymmetric frustum perspective projection, which is also simply recognized as asymmetric frustum [Jones et al., 2001, Bourke, 2002, Maupu et al., 2005].

In this method, the view vectors for each eye remain parallel. However, by considering the distance between the two eyes of the observer (disparity or eye offset), two distinct frustums are structured to generate individual images for each eye. The graphics library used in this thesis (OpenGL) offers a frustum method that simply be created by six parameters, which are basically focal distance, the eye separation, the actual dimensions of the screen used, distance between the observer and the screen, near and far clipping planes. Table 5.2 shows the list of parameters and the values used to generate stereo pairs in this thesis. Figure 5.13 illustrates the frustum metaphor and the description of the parameter used to generate it. The eye separation parameter used was 6.5 cm.

Left	:	-29.65 cm	Right	:	29.65 cm
Bottom	:	-16.80 cm	Top	:	16.80 cm
zNear	:	75.00 cm	zFar	:	300.0 cm
zDepth	:	112.50 cm	EyeOffset	:	3.25 cm

Table 5.2: Frustum Parameters

The frustum pairs are not identical. A delta value, calculated as a function of eye separation, frustum width, focal distance and aperture of the camera, is used in the formulate the difference and results the asymmetry in the frustum geometry. Pairs of images are generated by using the asymmetric frustum along similar lines with Bourke et al. Figure 5.14 shows the asymmetric frustum technique. The parameter descriptions and the code used to calculate the stereo pairs is available via the link provided in Appendix C.

Shutter glasses are used to present different images to the two eyes. The method is based on synchronized filtering of an eye with the refresh cycle of the display, so that the other eye (the one that is not blocked) can receive corresponding projection of the environment. The refresh rate per eye is maintained at 60 frame per second by using a 120 Hz refresh rate monitor. The application

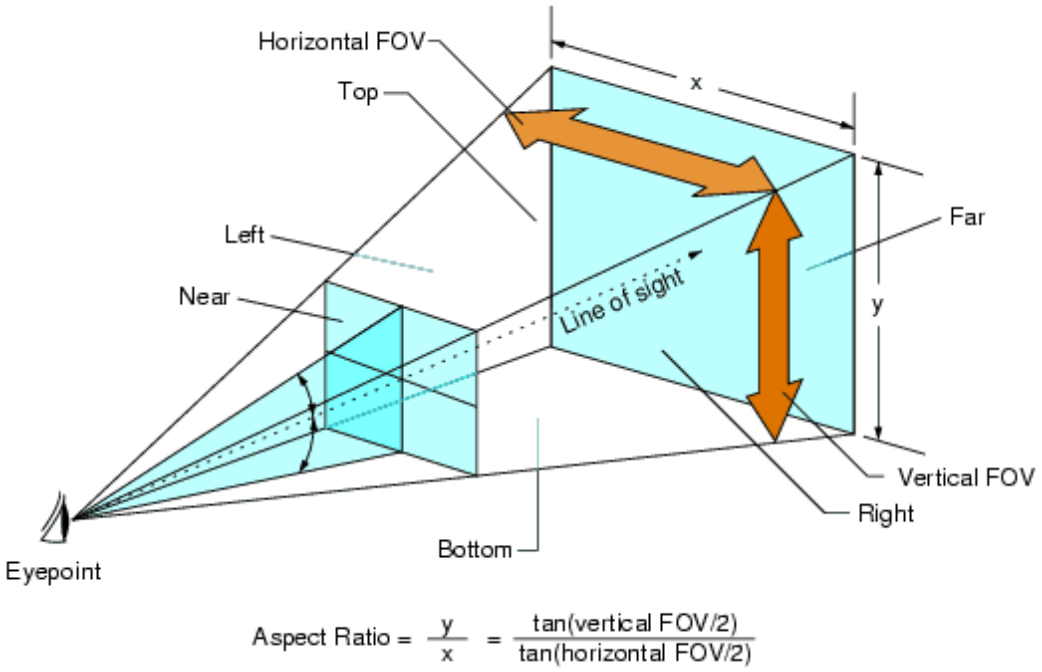


Figure 5.13: frustum

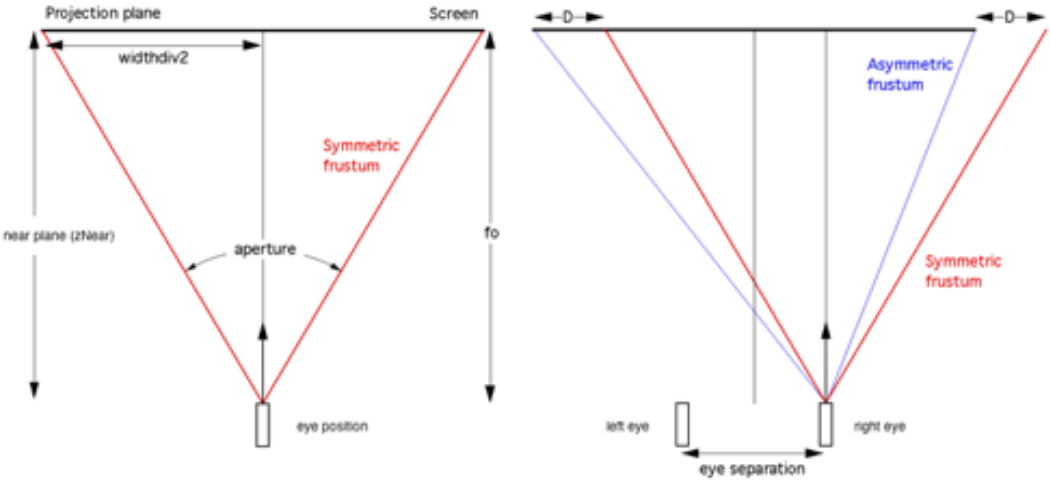


Figure 5.14: Pairs of images are generated by using the Asymmetric frustum along the similar lines with [Bourke, 2002]

also uses the oscillatory motion around a fixed axis (as described in Chapter 4) to provide kinetic depth perception, while preserving the view point. The period of the oscillation is 2 seconds, and the amplitude is about 15 degrees.

5.4.7 Rendering into image files

The user interface has multiple viewports, and each viewport is sequentially rendered by considering individual frustum parameters (eye position, direction, orthogonal or perspective projection, and so on) with the cost of interactivity performance (in terms of frame per second) as well as the quality of animation during the user experience. To generate better imagery, the application includes an image output mechanism that directly renders each frame to high resolution image files by using either lossless or lossy compression formats, such as png for lossless, and jpg for lossy compression. For example, the application can output full size images of any viewport for each time step, or angular step motion. The user can enable writing into image files via keyboard shortcuts. Sample images and videos can be found on the link provided in Appendix C. The images are saved by this feature, and videos are generated using sequence of frames created with same approach.

Chapter 6

Conclusion

6.1 Summary

Particle-based simulations are used across various science domains. These models utilize high performance computing platforms and produce massive amounts of data to analyze. The data generation rates are much greater than it can be handed by current methods. Consequently, data reduction through various forms of sampling has become a common practice. One form of data reduction is achieved by converting and storing the simulation into an imagery database. This in-situ solution is incorporated into the Cinema package [Ahrens et al., 2014]. In Cinema, a series of images from predefined set of viewing conditions (e.g., camera positions, angles, resolution) are generated during the actual simulation run and stored. The visualization is then generated by using these the stored images as a part of the post analysis stage.

The first major goal of this thesis was to enhance the image based in-situ solution to support the visualization of point clouds. Cinema requires a series of images to generate the visualization, and fewer images are more desirable from a storage perspective. If the images are the basis for motion parallax information, it is important to know how few can be used in order to still get a benefit from kinetic depth perception. The most useful perceptual cues for 3D point cloud data is motion parallax and stereoscopic depth. However, the relative advantages of stereo and kinetic depth have not been studied for point cloud data, and neither have the optimum viewing parameters been investigated. Therefore, in this thesis, two experiments were designed and conducted to

explore kinetic and stereo viewing parameters.

Two formal experiments concerning the optimal parameter for the visualization of point cloud data are presented in Chapter 4. In the first experiment, the number of discrete views was varied to determine the extent to which smooth motion is needed. The results show that kinetic depth to be more beneficial than stereo viewing as long as the motion is smooth. The second experiment varied the amplitude of the oscillatory motion. The results showed an increase in detection rate with amplitude with the best amplitudes being four degrees and greater. Furthermore, stereoscopic viewing conditions requires pairs of images, and in order to support stereoscopic depth perception, it is necessary to save a series of images based on a set of viewpoints orbiting a feature of interest.

The second major goal of this thesis was to develop methods for rapidly exploring and visualizing a point cloud data set. In Chapter 5, an application designed to support the interactive exploration of a cosmology data set is described. The data set used was a significantly smaller subset of the original HPC simulation output, which consisted of multivariate particle data (order of 10^6) stored in individual files for each of 250 time steps. The interaction method is based on a spherical point of interest maintained at the center of the workspace. As the user navigates more detail is generated in this central region. A number of data visualization techniques have been implemented to support the specific requirements of the data. The basic method involves showing particles as transparent point clouds with motion parallax and stereoscopic viewing to support understanding of 3D structure. As an alternative the particles can be shown as short line segments to reveal their direction of travel. In this mode of viewing color is used to help differentiate groups of particles travelling in different directions in the 3D space.

The user also has the option of selecting a relatively small group of particles and requesting a time series animation. In this case the software brings in data from a number of time steps by using an internal index to facilitate search in the original data files. The software then animates the selected group of particles using streamlets as a representation method. These streamlets are

also colored using the same method developed for the vectors shown in an individual time step. In this way it is possible to see how halo structures develop and migrate through the evolution of the universe.

6.2 Future Work

The two studies of stereo and motion for the visualization of point clouds have only begin to explore the possibilities. More work is needed here. The value of active versus passive motion remains to be explored for point cloud data. In addition, it would be useful to discover if the same findings apply to data having other kinds of spatial characteristics. For example, the point clouds generated by LIDAR or Multibeam Sonars are mostly defined by surfaces with outliers that may be invalid data points. There are tools for cleaning this kind of data that might benefit from better 3D spatial visualization.

The Cinema package requires storing set of images, however, the image formats and the compression schemas used to store can be in variety forms. One group of image compression is lossless, and others sacrifice image quality to benefit storage space. However, the parameters that can be used in compression methods could change the quality of the images and may result in loss of essential features of the simulation. The application code currently can generate fundamental image creation and conversion methods.

The visualization package developed for this thesis might be used in several applications, since the design considered the pipeline architecture, which is common practice in visualization software libraries. For example, the method we proposed to create motion and the stereoscopic viewing conditions can be integrated into ParaView as a renderer module. The data management method (container) that is implemented as a part of this thesis can also be used for data editing and cleansing purposes, which is commonly needed in point cloud data sets in other domains, such as laser mapping technologies.

Appendix A

IRB Approval Form

University *of* New Hampshire

Research Integrity Services, Service Building
51 College Road, Durham, NH 03824-3585
Fax: 603-862-3564

03-Sep-2015

Ware, Colin
Ctr For Coastal&Ocean Mapping
Chase Ocean Eng
Durham, NH 03824-3515

IRB #: 3013
Study: Perceptual Optimization for Data Visualization
Review Level: Expedited
Approval Expiration Date: 26-Aug-2016

The Institutional Review Board for the Protection of Human Subjects in Research (IRB) has reviewed and approved your request for time extension for this study. Approval for this study expires on the date indicated above. At the end of the approval period you will be asked to submit a report with regard to the involvement of human subjects. If your study is still active, you may apply for extension of IRB approval through this office.

Researchers who conduct studies involving human subjects have responsibilities as outlined in the document, *Responsibilities of Directors of Research Studies Involving Human Subjects.* This document is available at http://unh.edu/research/irb-application-resources or from me.

If you have questions or concerns about your study or this approval, please feel free to contact me at 603-862-2003 or Julie.simpson@unh.edu. Please refer to the IRB # above in all correspondence related to this study. The IRB wishes you success with your research.

For the IRB,

Julie F. Simpson
Director

cc: File

Figure A.1: IRB Approval

Appendix B

Participant Consent Form

CONSENT FORM FOR PARTICIPATION IN A RESEARCH STUDY

PERCEPTUAL OPTIMIZATION FOR DATA VISUALIZATION

A study carried out by Erol Aygar for Dr. Colin Ware in the Data Visualization Research Lab at the University of New Hampshire.

WHAT IS THE PURPOSE OF THIS STUDY?

The purpose of this study is to find more effective ways of displaying numerical data. There will be approximately 14 participants.

WHAT ARE THE POSSIBLE RISKS OF PARTICIPATING IN THIS STUDY?

There are no known risks associated with this research.

WHAT ARE THE POSSIBLE BENEFITS OF PARTICIPATING IN THIS STUDY?

The potential benefit of this research is that data displays will be better designed in the future.

WILL YOU RECEIVE ANY COMPENSATION FOR PARTICIPATING IN THIS STUDY?

You will receive $18 for each session in this study.

DO YOU HAVE TO TAKE PART IN THIS STUDY?

Your consent to participate in this research is entirely voluntary. If you refuse to participate, you will not experience any penalty or negative consequences.

CAN YOU WITHDRAW FROM THIS STUDY?

If you consent to participate in this study, you may refuse to answer any question and/or stop your participation in the study at any time without any penalty or negative consequences.

HOW WILL THE CONFIDENTIALITY OF YOUR RECORDS BE PROTECTED?

I seek to maintain the confidentiality of all data and records associated with your participation in this research and only averaged data will be published. The data will be stored on a password protected computer.

WHOM TO CONTACT IF YOU HAVE QUESTIONS ABOUT THIS STUDY

If you have any questions pertaining to the research you can contact Colin Ware (862-1138, cware@ccom.unh.edu) to discuss them.

If you have questions about your rights as a research subject you can contact Dr. Julie Simpson in UNH Research Integrity Services, 603-862-2003 or Julie.simpson@unh.edu to discuss them.

I, _____ CONSENT/AGREE to participate in this research study

_____ _____

Signature of Subject Date

Figure B.1: Participant consent form

Appendix C

Supplementary Materials

The experiments code can be found online in the following code repository. The interactive interface can be downloaded. The parameter descriptions and the code used to calculate the stereo pairs are also available in the repository. Code repository link is https://github.com/aygar. Sample images and videos can be found on the following web page. https://erolaygar.com

Appendix D

Manual

NAME

cosmoReader - an interactive interface for datasets in cosmo file format.

SYNOPSIS

cosmoReader [options [-id index-file |−−index-file=file] [-s stereo] [-o folder |-output-folder=folder] [-jpg |−−output-format=jpg]] [-f file-list |file ..]

DESCRIPTION

This application reads a list of timesteps stored in cosmo file format. The application provides interactive features to navigate within the dataset. Various data visualization methods are applied.

OPTIONS

file ..

A list of filenames. This can consist of multiple filenames, which are considered to be separated by spaces. Each file is expected to be in cosmo file format. The last file is considered to be the main timestep of the simulation. If multiple files is not provided, or neither this parameter nor a switch is specified, the program will abort with an error message.

-f, −−filelist

Provide a file that consists of file names to be used as time-steps. Each line should have single filename.

-i, −−info

Prints the information about the data file. e.g., file size, number of particles, dimensions and center point of the dataset. A file with .idx extension will be generated.

-g, −−generate

Generates an index file.

-h, −−help

Prints the help and usage.

-s, −−stereo

Start the program with the stereoscopic viewing mode. This mode requires specific hardware and parameters on the control panel to enable stereo mode and shutter glasses type and mode should be set correctly. If this switch is not provided, the application will run in non-stereo mode.

-id, −−index-file

Use the file provided as the index file. If this is not given, the first time-step file will be used to generate an index file.

-o folder, −−output-folder

Specify the path or a folder for storing images.

-jpg −−output-format

Specify the format of the images. Default output is the lossless PNG format

INTERACTION

KEY A	Zoom in.
KEY Z	Zoom out.
KEY W	Update the spatial point of interest.
KEY E	Increase the radius of the spherical area.
KEY D	Decrease the radius of the spherical area.
KEY UP	Navigate forward on z-axis.
KEY DOWN	Navigate backward on z-axis.
KEY G	Enable/disable guidelines.
KEY V	Enable/disable velocity representations.
KEY PERIOD	Stop/start animation.
KEY 3	Enable/disable spherical point of interest.
KEY 0	Enable/disable background.
KEY 8	Enable/disable alpha channel.
KEY ESC	Quit application.

version 1.0 2016-11-02 cosmoReader(1)

Table D.1: Manual

Bibliography

[Abram, 1995] Abram, G. (1995). An Extended Data-Flow Architecture for Data Analysis and Visualization. *Computer Graphics*, (May):17—-21.

[Ahern et al., 2011] Ahern, S., Shoshani, A., Ma, K., and Choudhary, A. (2011). Scientific discovery at the exascale. Report from the DOE ASCR 2011 Workshop on Exascale Data Management. *Analysis, and Visualization*, (February).

[Ahrens et al., 2005] Ahrens, J., Geveci, B., and Law, C. (2005). ParaView: An end-user tool for large-data visualization. In *Visualization Handbook*, pages 717–731.

[Ahrens et al., 2014] Ahrens, J., Jourdain, S., O'Leary, P., Patchett, J., Rogers, D. H., and Petersen, M. (2014). An Image-Based Approach to Extreme Scale in Situ Visualization and Analysis.

[Ahrens et al., 2010] Ahrens, J. L. A. N. L., Hendrickson, B. S. N. L., and Long, G. A. N. L. (2010). Data Intensive Science in the Department of Energy. 836.

[Aldoma et al., 2012] Aldoma, A., Marton, Z. C., Tombari, F., Wohlkinger, W., Potthast, C., Zeisl, B., Rusu, R., Gedikli, S., and Vincze, M. (2012). Tutorial: Point cloud library: Three-dimensional object recognition and 6 DOF pose estimation. *IEEE Robotics and Automation Magazine*, 19(3):80–91.

[Arsenault and Ware, 2004] Arsenault, R. and Ware, C. (2004). The importance of stereo and eye coupled perspective for eye-hand coordination in fish tank VR. *Presence: Teleoperators and Virtual Environments*, 13(5):549–559.

[Ayachit et al., 2015] Ayachit, U., Leary, P. O., Bauer, A., Moreland, K., Mauldin, J., Geveci, B., and Fabian, N. (2015). ParaView Catalyst : Enabling In Situ Data Analysis and Visualization. *Proceedings of the First Workshop on In Situ Infrastructures for Enabling Extreme-Scale Analysis and Visualization (ISAV 2015)*, pages 25–29.

[Aygar and Ware, 2016] Aygar, E. and Ware, C. (2016). The Contribution of Stereoscopic and Motion Depth cues to the Perception of Structures in 3D Point Clouds. *Transactions on Visualization and Computer Graphics*, pages 1–14.

[Bacim et al., 2013] Bacim, F., Kopper, R., and Bowman, D. A. (2013). Design and evaluation of 3D selection techniques based on progressive refinement. *International Journal of Human-Computer Studies*, 71(7-8):785–802.

[Bacim et al., 2014] Bacim, F., Nabiyouni, M., and Bowman, D. A. (2014). Slice-n-Swipe: A free-hand gesture user interface for 3D point cloud annotation. *IEEE Symposium on 3D User Interfaces 2014, 3DUI 2014 - Proceedings*, 185:185–186.

[Barfield et al., 1997] Barfield, W., Hendrix, C., and Bystrom, K. (1997). Visualizing the structure of virtual objects using head tracked stereoscopic displays. In *Virtual Reality Annual International Symposium, 1997., IEEE 1997*, pages 114–120. IEEE.

[Bavoil et al., 2005] Bavoil, L., Callahan, S. P., Crossno, P. J., Freire, J., Scheidegger, C. E., Silva, C. T., and Vo, H. T. (2005). VisTrails: Enabling interactive multiple-view visualizations. *Proceedings of the IEEE Visualization Conference*, (Dx):18.

[Bhattacharya et al., 2013] Bhattacharya, S., Habib, S., Heitmann, K., and Vikhlinin, A. (2013). Dark Matter Halo Profiles of Massive Clusters: Theory Versus Observations. *The Astrophysical Journal*, 766(1):32.

[Bourke, 2002] Bourke, P. (2002). 3d stereo rendering using opengl (and glut). *Published Online http://astronomy.swin.edu.au/ pbourke/opengl/stereogl*.

[Bowman et al., 2001] Bowman, D. A., Kruijff, E., LaViola, J. J., and Poupyrev, I. (2001). An Introduction to 3-D User Interface Design. *Presence: Teleoperators and Virtual Environments*, 10(1):96–108.

[Bradshaw et al., 2000] Bradshaw, M. F., De Bruyn, B., Eagle, R. A., and Parton, A. D. (2000). The task-dependent use of binocular disparity and motion parallax under natural viewing conditions. *Perception*, 26(Supplement):48.

[Bradski et al., 2000] Bradski, G. et al. (2000). The opencv library. *Doctor Dobbs Journal*, 25(11):120–126.

[Butkiewicz and Ware, 2011] Butkiewicz, T. and Ware, C. (2011). Multi-touch 3D exploratory analysis of ocean flow models. *Oceans 2011*, pages 1–10.

[Cabral et al., 2014] Cabral, M., Montes, A., Belloc, O., Ferraz, R., Teubl, F., Doreto, F., Lopes, R., and Zuffo, M. (2014). Bi-manual gesture interaction for 3D cloud point selection and annotation using COTS. *IEEE Symposium on 3D User Interfaces 2014, 3DUI 2014 - Proceedings*, pages 187–188.

[Demiralp et al., 2009] Demiralp, a., Hughes, J. F., and Laidlaw, D. H. (2009). Coloring 3D Line fields using Boy's real projective plane immersion. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1457–1463.

[Dyer, 1990] Dyer, D. S. (1990). A Dataflow Toolkit for Visualization.

[Elseberg et al., 2012] Elseberg, J., Magnenat, S., Siegwart, R., and Andreas, N. (2012). Comparison of nearest-neighbor-search strategies and implementations for efficient shape registration. *Journal of Software Engineering for Robotics (JOSER)*, 3(1):2–12.

[Foulser, 1995] Foulser, D. (1995). IRIS Explorer: a framework for investigation. *Computer Graphics*, 29(2):13–16.

[Fowler and Ware, 1989] Fowler, D. and Ware, C. (1989). Strokes for representing univariate vector field maps. *Proceedings of Graphics Interface*, 89:249–253.

[Frohlich and Plate, 2000] Frohlich, B. and Plate, J. (2000). The cubic mouse: a new device for three-dimensional input. *Sigchi*, 1(April):526–531.

[Geist II et al., 1997] Geist II, G. A., Kohl, J. A., and Papadopoulos, P. M. (1997). CUMULVS: providing fault tolerance, visualization, and steering of parallel applications. *International Journal of Supercomputer Applications*, 11(3):224–235.

[Griest, 2002] Griest, K. (2002). WIMPs and MACHOs. *Nature Publishing Group*, (785998):1–6.

[Grossman and Balakrishnan, 2004] Grossman, T. and Balakrishnan, R. (2004). Pointing at Trivariate Targets in 3D Environments. *Proceedings of the 2004 conference on Human factors in computing systems - CHI '04*, 6(1):447–454.

[Haan et al., 2005] Haan, G. D., Koutek, M., and Post, F. H. (2005). IntenSelect: Using Dynamic Object Rating for Assisting 3D Object Selection. *In Virtual Environments 2005*, pages 201–209.

[Habib et al., 2009] Habib, S., Pope, A., Lukić, Z., Daniel, D., Fasel, P., Desai, N., Heitmann, K., Hsu, C.-H., Ankeny, L., Mark, G., Bhattacharya, S., and Ahrens, J. (2009). Hybrid petacomputing meets cosmology: The Roadrunner Universe project. *Journal of Physics Conference Series*, 180:2019.

[Hand, 1997] Hand, C. (1997). A Survey of 3D Interaction Techniques. *Computer Graphics Forum*, 16(5):269–281.

[Haroz and Heitmann, 2008] Haroz, S. and Heitmann, K. (2008). Seeing the difference between cosmological simulations. *IEEE Computer Graphics and Applications*, 28(5):37–45.

[Haroz et al., 2008] Haroz, S., Kwan-Liu, M., and Heitmann, K. (2008). Multiple uncertainties in time-variant cosmological particle data. *IEEE Pacific Visualisation Symposium 2008, PacificVis - Proceedings*, pages 207–214.

[Hassaine et al., 2010] Hassaine, D., Holliman, N. S., and Liversedge, S. P. (2010). Investigating the performance of path-searching tasks in depth on multiview displays. *ACM Transactions on Applied Perception (TAP)*, 8(1):8:1–8:18.

[Henderson, 2004] Henderson, A. (2004). The ParaView Guide: A Parallel Visualization Application.

[Hey, 2012] Hey, T. (2012). *The Fourth Paradigm – Data-Intensive Scientific Discovery*, pages 1–1. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Hils, 1991] Hils, D. (1991). Datavis: a visual programming language for scientific visualization. *Proceedings of the 19th annual conference on ...*, pages 439–448.

[Howard and Rogers, 1995] Howard, I. P. and Rogers, B. J. (1995). *Binocular vision and stereopsis*. Oxford University Press, USA.

[Hubona et al., 1997] Hubona, G. S., Shirah, G. W., and Fout, D. G. (1997). The effects of motion and stereopsis on three-dimensional visualization. *International journal of human-computer studies*, 47(5):609–627.

[IJsselsteijn et al., 2001] IJsselsteijn, W., de Ridder, H., Freeman, J., Avons, S. E., and Bouwhuis, D. (2001). Effects of stereoscopic presentation, image motion, and screen size on subjective and objective corroborative measures of presence. *Presence*, 10(3):298–311.

[Jablonowski et al., 1993] Jablonowski, D. J., Bruner, J. D., Bliss, B., and Haber, R. B. (1993). Vase: The visualization and application steering environment. In *Supercomputing'93. Proceedings*, pages 560–569. IEEE.

[Jarosik et al., 2011] Jarosik, N., Bennett, C. L., Dunkley, J., Gold, B., Greason, M. R., Halpern, M., Hill, R. S., Hinshaw, G., Kogut, A., Komatsu, E., Larson, D., Limon, M., Meyer, S. S., Nolta, M. R., Odegard, N., Page, L., Smith, K. M., Spergel, D. N., Tucker, G. S., Weiland, J. L., Wollack, E., and Wright, E. L. (2011). Seven-Year Wilkinson Microwave Anisotropy Probe (WMAP) Observations: Sky Maps, Systematic Errors, and Basic Results. *The Astrophysical Journal Supplement Series*, 192(2):14.

[Johnson et al., 1999] Johnson, C., Parker, S., Hansen, C., Kindlmann, G., and Livnat, Y. (1999). Interactive Simulation and Visualization. *Computer*, 32(12):59–65.

[Jones et al., 2001] Jones, G. R., Lee, D., Holliman, N. S., and Ezra, D. (2001). Controlling perceived depth in stereoscopic images. *Proc. SPIE*, 44(11):42–53.

[Kopper et al., 2011] Kopper, R., Bacim, F., and Bowman, D. A. (2011). Rapid and accurate 3D selection by progressive refinement. *3DUI 2011 - IEEE Symposium on 3D User Interfaces 2011, Proceedings*, pages 67–74.

[Langley and Stanford, 1995] Langley, P. A. T. and Stanford, L. (1995). Stages in the Process of Scientific Discovery. page 1995.

[LANL, 2016] LANL (2016). www.lanl.gov.

[Lapenta, 2012] Lapenta, G. (2012). Particle simulations of space weather. *Journal of Computational Physics*, 231(3):795–821.

[Lazzaro et al., 1998] Lazzaro, G., Swift, D., Hamlin, G., and Faris, S. (1998). Stereoscopic 3-d viewing system and glasses having electrooptical shutters controlled by control signals produced using horizontal pulse detection within the vertical synchronization pulse period of computer generated video signals. US Patent 5,821,989.

[Liang and Green, 1994] Liang, J. and Green, M. (1994). Jdcad: A highly interactive 3d modeling system. *Computers & graphics*, 18(4):499–506.

[Light, 1989] Light, O. I. Y.-g. (1989). United States Patent [191. 6.

[LLNL, 2005] LLNL, L. L. N. L. (2005). VisIt User's Manual. *Contract*, (October):356.

[Ma et al., 2007] Ma, K.-L., Wang, C., Yu, H., and Tikhonova, A. (2007). In-situ processing and visualization for ultrascale simulations. *Journal of Physics: Conference Series*, 78:012043.

[Markevitch et al., 2002] Markevitch, M., Gonzalez, A. H., David, L., Vikhlinin, A., Murray, S., Forman, W., Jones, C., and Tucker, W. (2002). A Textbook Example of a Bow Shock in the Merging Galaxy Cluster 1E 0657-56. *Apjl*, 567:L27–L31.

[Maupu et al., 2005] Maupu, D., Van Horn, M. H., Weeks, S., and Bullitt, E. (2005). 3D stereo interactive medical visualization. *IEEE Computer Graphics and Applications*, 25(5):67–71.

[Mine, 1995] Mine, M. R. (1995). Collaborative virtual environment for feature based modeling. *Proceedings of the ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry*, pages 120–126.

[Mitchell et al., 2009] Mitchell, P., Ware, C., and Kelley, J. (2009). Investigating flow visualizations using interactive design space hill climbing. *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, pages 355–361.

[Moreland, 2013] Moreland, K. (2013). A survey of visualization pipelines. *IEEE Transactions on Visualization and Computer Graphics*, 19(3):367–378.

[Müller et al., 2003] Müller, M., Charypar, D., and Gross, M. (2003). Particle-Based Fluid Simulation for Interactive Applications. *Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, (5):154–159.

[Naepflin and Menozzi, 2001] Naepflin, U. and Menozzi, M. (2001). Can movement parallax compensate lacking stereopsis in spatial explorative search tasks? *Displays*, 22(5):157–164.

[Norman et al., 1995] Norman, J. F., Todd, J. T., and Phillips, F. (1995). The perception of surface orientation from multiple sources of optical information. *Perception & psychophysics*, 57(5):629–636.

[Palmer, 1999] Palmer, S. E. (1999). *Vision Science : Photons to Phenomenology.* A Bradford Book.

[Parker and Johnson, 1995] Parker, S. G. and Johnson, C. R. (1995). SCIRun: a scientific programming environment for computational steering. *Supercomputing 95*, page 1.

[Pfister et al., 2000] Pfister, H., Zwicker, M., van Baar, J., and Gross, M. (2000). Surfels. *Proceedings of the 27th annual conference on Computer graphics and interactive techniques - SIGGRAPH '00*, pages 335–342.

[Pierce et al., 1997] Pierce, J. S., Forsberg, A. S., Conway, M. J., Hong, S., Zeleznik, R. C., and Mine, M. R. (1997). Image plane interaction techniques in 3D immersive environments. *ACM i3D*, pages 39–ff.

[Pope et al., 2010] Pope, A., Habib, S., Lukic, Z., Daniel, D., Fasel, P., Desai, N., and Heitmann, K. (2010). The Accelerated Universe. *Computing in Science & Engineering*, 12(4):17–25.

[Ramachandran and Varoquaux, 2011] Ramachandran, P. and Varoquaux, G. (2011). Mayavi: 3D visualization of scientific data. *Computing in Science and Engineering*, 13(2):40–51.

[Remondino, 2003] Remondino, F. (2003). From point cloud to surface: the modeling and visualization problem. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, XXXIV:24–28.

[Riccio, 2014] Riccio, C. (2014). Opengl mathematics (glm). *URL: http://www. g-truc. net/project-0016. html (2014-07-16)*.

[Rogers and Springmeyer, 2011] Rogers, D. and Springmeyer, B. (2011). Visualization and Data Analysis at the Exascale. pages 1–3.

[Rubin et al., 1985] Rubin, V. C., Burstein, D., Ford, W. K., J., and Thonnard, N. (1985). Rotation velocities of 16 SA galaxies and a comparison of Sa, Sb, and SC rotation properties. *The Astrophysical Journal*, 289:81.

[Rusu and Cousins, 2011] Rusu, R. B. and Cousins, S. (2011). 3D is here: point cloud library. *IEEE International Conference on Robotics and Automation*, pages 1 – 4.

[Scheiblauer, 2014] Scheiblauer, C. (2014). Interactions with Gigantic Point Clouds. 2014.

[Schickore, 2014] Schickore, J. (2014). Scientific discovery. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Spring 2014 edition.

[Schroeder et al., 1996] Schroeder, W., Martin, K., and Lorensen, B. (1996). The Visualization Toolkit An Object Oriented Approach to 3D Graphics, 3rd Edition - Kitware Inc.pdf.

[Schroeder et al., 1992] Schroeder, W. J., Lorensen, W. E., Montanaro, G., and Volpe, C. R. (1992). VISAGE: an object-oriented scientific visualization system. *IEEE Computer Society Press*, Proceeding:219—-226.

[Schroeder and Martin, 2005] Schroeder, W. J. and Martin, K. M. (2005). The visualization toolkit. In *Visualization Handbook*, pages 593–614.

[Shan et al., 2012] Shan, G., Xie, M., Li, F. A., Gao, Y., and Chi, X. (2012). Interactive Visual Exploration of Halos in Large Scale Cosmology Simulation.

[Sheiman, 1988] Sheiman, D. (1988). Stereoscopic viewing system and glasses. US Patent 4,744,633.

[Shirley and Tuchman, 1990] Shirley, P. and Tuchman, A. (1990). A polygonal approximation to direct scalar volume rendering. *ACM SIGGRAPH Computer Graphics*, 24(5):63–70.

[Sollenberger and Milgram, 1993] Sollenberger, R. L. and Milgram, P. (1993). Effects of stereoscopic and rotational displays in a three-dimensional path-tracing task. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 35(3):483–499.

[Squillacote et al.,] Squillacote, A. H., Ahrens, J., Law, C., Geveci, B., King, B., and Kitware, P. The ParaView Guide. *Writing*.

[Steed and Parker, 2004] Steed, A. and Parker, C. (2004). 3D Selection Strategies for Head Tracked and Non-Head Tracked Operation of Spatially Immersive Displays. *8th International Immer- sive Projection Technology Workshop.*, pages 1–8.

[Sweet and Ware, 2004] Sweet, G. and Ware, C. (2004). View direction, surface orientation and texture orientation for perception of surface shape. In *Proceedings of Graphics Interface 2004*, pages 97–106. Canadian Human-Computer Communications Society.

[Tarini et al., 2006] Tarini, M., Cignoni, P., and Montani, C. (2006). Ambient occlusion and edge cueing to enhance real time molecular visualization. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1237–1244.

[Teunissen and Ebert, 2014] Teunissen, J. and Ebert, U. (2014). Controlling the weights of simulation particles: Adaptive particle management using k-d trees. *Journal of Computational Physics*, 259:318–330.

[Tuchman et al., 1991] Tuchman, A., Jablonowski, D., and Cybenko, G. (1991). Run-time visualization of program data. *Proceeding Visualization '91*, pages 255–261.

[Upson et al., 1989] Upson, C., Faulhaber, T.a., J., Kamins, D., Laidlaw, D., Schlegel, D., Vroom, J., Gurwitz, R., and Dam, a. V. (1989). The application visualization system: a computational environment\nfor scientific visualization. *IEEE Computer Graphics and Applications*, 9(4).

[Valkov et al., 2011] Valkov, D., Steinicke, F., Bruder, G., and Hinrichs, K. (2011). 2D Touching of 3D Stereoscopic Objects. *Proceedings of the 2011 annual conference on Human factors in computing systems - CHI '11*, page 1353.

[Van Beurden et al., 2010] Van Beurden, M. H. P. H., Kuijsters, A., and IJsselsteijn, W. A. (2010). Performance of a path tracing task using stereoscopic and motion based depth cues. *2010 2nd International Workshop on Quality of Multimedia Experience, QoMEX 2010 - Proceedings*, pages 176–181.

[Wallach et al., 1953] Wallach, H., O'Connell, D. N., and Neisser, U. (1953). The memory effect of visual perception of three-dimensional form. *Journal of Experimental Psychology*, 45(5):360–8.

[Wand et al., 2007] Wand, M., Berner, A., Bokeloh, M., Fleck, A., Hoffmann, M., Jenke, P., Maier, B., Staneker, D., and Schilling, A. (2007). Interactive Editing of Large Point Clouds. *Eurographics Symposium on Point-Based Graphics*, pages 37–45.

[Wanger et al., 1992] Wanger, L. R., Ferwerda, T. A., and Greenberg, D. P. (1992). Perceiving Spatial Relationships in Computer-Generated Images.

[Ware, 2012] Ware, C. (2012). *Foundation for a Science of Data Visualization*.

[Ware et al., 1993] Ware, C., Arthur, K., and Booth, K. S. (1993). Fish Tank Virtual Reality. *InterCHI*, pages 37–42.

[Ware and Franck, 1996] Ware, C. and Franck, G. (1996). Evaluating stereo and motion cues for visualizing information nets in three dimensions. *ACM Transactions on Graphics (TOG)*, 15(2):121–140.

[Ware et al., 1998] Ware, C., Gobrecht, C., and Paton, M. A. (1998). Dynamic adjustment of stereo display parameters. *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans.*, 28(1):56–65.

[Ware et al., 1996] Ware, C., Hui, D., and Franck, G. (1996). Evaluating stereo and motion cues for visualizing information nets in three dimensions. *ACM Transactions on Graphics (TOG)*, 15(2):121–140.

[Ware et al., 2014] Ware, C., Kelley, J. G., and Pilar, D. (2014). Improving the Display of Wind Patterns and Ocean Currents. *Bulletin of the American Meteorological Society*, 95(10):1573–1581.

[Ware and Mitchell, 2005] Ware, C. and Mitchell, P. (2005). Reevaluating stereo and motion cues for visualizing graphs in three dimensions. In *Proceedings of the 2nd symposium on Applied perception in graphics and visualization*, pages 51–58. ACM.

[Ware et al., 2016] Ware, C., Rogers, D., Petersen, M., Ahrens, J., Aygar, E., and Mapping, O. (2016). Optimizing for Visual Cognition in High Performance Scientific Computing. pages 1–9.

[White et al., 2009] White, M., Pope, A., Carlson, J., Heitmann, K., Habib, S., Fasel, P., Daniel, D., and Lukic, Z. (2009). Particle mesh simulations of the Lyman-alpha forest and the signature of Baryon Acoustic Oscillations in the intergalactic medium. 3:48–49.

[Wiebel et al., 2012] Wiebel, A., Vos, F. M., Foerster, D., and Hege, H.-C. (2012). Wysiwyp: what you see is what you pick. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2236–2244.

[Wikipedia, 2016] Wikipedia (2016). Polarized 3d system — wikipedia, the free encyclopedia. [Online; accessed 16-September-2016].

[Woodring et al., 2011] Woodring, J., Ahrens, J., Figg, J., Wendelberger, J., Habib, S., and Heitmann, K. (2011). In-situ sampling of a large-scale particle simulation for interactive visualization and analysis. *Computer Graphics Forum*, 30(3):1151–1160.

[Yu et al., 2012] Yu, L., Efstathiou, K., Isenberg, P., and Isenberg, T. (2012). Efficient structure-aware selection techniques for 3D point cloud visualizations with 2DOF input. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2245–2254.

[Yu et al., 2016] Yu, L., Efstathiou, K., Isenberg, P., and Isenberg, T. (2016). CAST: Effective and Efficient User Interaction for Context-Aware Selection in 3D Particle Clouds. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):886–895.

[Zeleznik et al., 1997] Zeleznik, R. C., Forsberg, A. S., and Strauss, P. S. (1997). Two Pointer Input For 3D Interaction. *Acm I3D*, pages 115–120.

Appendix E

Vita

Candidate's full name	:	Erol Aygar
Address	:	71 Dione Dr. Manchester, NH 03102
Education	:	University Of New Hampshire, Manchester, NH MS Information Technology (2013 - 2016) Bosphorus University, Istanbul, TR MS Engineering Management (2003 - 2006) Mimar Sinan University, Istanbul, TR BS Urban Planning (1994 - 1999)
Publications	:	[Aygar and Ware, 2016, Ware et al., 2016]