IEEE Visualization 2008 Tutorial Proposal

Title

Perception-Driven Techniques for Effective Visualization of Large Volume Data

Duration

The tutorial is half-day, including 180 minutes presentation and discussion, and 30 minutes coffee break.

Organizers

Chaoli Wang	University of California, Davis	wangcha@cs.ucdavis.edu
Han-Wei Shen	The Ohio State University	hwshen@cse.ohio.state.edu
Klaus Mueller	Stony Brook University	mueller@cs.sunysb.edu
Huamin Qu	Hong Kong University of Science and Technology	huamin@cse.ust.hk

Abstract

Data visualization is an iterative and exploratory process, which involves choices of parameters for queries of different types. Examples of visualization parameters include level-of-detail, color and opacity transfer function, camera position and path, lighting and so on. To reveal the important aspects of data, the users often have to go through a lengthy and expensive process to obtain a large ensemble of visualization results. With the ever-increasing size of volume data, manual data browsing through the immense, high-dimensional parameter space is no longer a viable solution. Efficient and effective solutions that search and narrow down the parameter space for assisting the users in their decision making become imperative.

In this tutorial, we introduce recent advances and emerging techniques in volume visualization towards perception-driven data analysis, rendering and presentation. The fundamental visual perception and cognitive principles are incorporated into the visualization process, thus enable presentation of relevant information for gleaning insights from the data. Selective topics include perception-informed color and highlighting, saliency-aware rendering techniques, perception-guided transfer function specification, quality enhancement of direct volume rendered images, view selection for three-dimensional and time-varying volume visualization, level-of-detail (LOD) selection for multiresolution visualization, and multiscale volume data quality assessment. The tutorial covers principles and practice of perception and cognition (such as color perception), mathematics and statistics (such as entropy theory, frequency-domain foundation, and conjoint analysis), as well as user study and evaluation. The tutorial also demonstrates the applications of those principles in visualization. The goal of this tutorial is to inform visualization researchers and practitioners the state-of-the-art technologies that leverage human perception for effective visualization of large volume data.

Topics

topic	instructor	time
Introduction	All	10 minutes
So Many Parameters, So Little Time: Guiding Users to Obtain the Best Visualization for the Given Data, Task, and Intent	Klaus Mueller	40 minutes
Perception-Guided Transfer Function Specification and Quality Enhancement	Huamin Qu	40 minutes
Break	—	30 minutes
View Selection for 3D and Time-Varying Volume Visualization	Han-Wei Shen	40 minutes
LOD Selection and Multiscale Volume Data Quality Assessment	Chaoli Wang	40 minutes
Discussion	All	10 minutes

Level

Beginner/Intermediate.

Prerequisite

An intermediate knowledge level of data visualization, specifically volume rendering algorithms, is required. However, prior knowledge of or background in perception and cognitive science is not necessary.

Audience

The intended audience includes visualization researchers and practitioners who are interested in automated techniques and solutions that apply perception knowledge in volume visualization towards effective data analysis, rendering and presentation.

History

Over the past decade, related tutorials of the IEEE Visualization Conference were centered on perception design, analysis and evaluation. Since 1996, eight tutorials have been organized in the Visualization Conference with topics including perceptual psychology, perception design and evaluation (Visualization '96, '97, '98, '02, and '07), and color in information display (Visualization '05, '06, and '07). All previous tutorials have been well-received by the audience, reflecting strong interest from graphics and visualization research community. As the field of visualization matures, there is a growing trend to move away from the use of ad-hoc, trial-and-error techniques to the adoption of automated algorithms that are based on the understanding of how people perceive visual representations of information. This tutorial shares the same theme of applied perception in visualization, with a focus on introducing new techniques developed in the context of volume visualization.

Importance

In the recommendations from the DOE/ASCR 2007 Workshop on Visual Analysis and Data Exploration at Extreme Scale, visualization experts around this country listed *quantitative metrics for parameter choices* as one of the key research areas for knowledge-enabling visualization and analysis. Visual data exploration is fundamental to our ability to analyze complex phenomena and discover new knowledge from the vast amounts of data. The process of visual data exploration involves choices of parameters for queries of different types. As the size of data keeps increasing at an astonishing rate, automated algorithms and techniques become imperative. Automatic searching and narrowing down the immense parameter space is critically important for facilitating pattern detection and situation assessment. Visual perception and cognition plays an essential role in data understanding and thus should be leveraged in visualization to streamline visual analysis.

We believe that the proposed tutorial will inform the audience of recent advances in perception-driven volume visualization techniques, and motivate researchers to incorporate principles and practice of perception into visualization towards effective analysis, rendering and presentation of data.

Description

The description and outline of each topic is in the following:

So Many Parameters, So Little Time:
Guiding Users to Obtain the Best Visualization for the Given Data, Task, and Intent

Abstract

The interactive rendering capabilities brought about by modern graphics hardware (GPUs) have been a blessing, but they have also been a curse. A positive aspect is that they enable interactive image generation even for the most elaborate visualization methods and datasets. However, there is also a negative aspect associated with this in that the task of traversing and exploring the associated abundant parameters space starts to hit the limits imposed by the curse of dimensionality. To cope, users need shortcuts and guidance to locate and navigate the parameter subspace most suited for the data, the task, the intent, and also their personal preferences (or those of the intended audience). This requires analyses in two domains, data space and observer space, which are mediated by the generated visualization given task and intent.

Accordingly, in this segment of the tutorial we will provide a survey of the latest research trends to measure saliency in the data, in terms of features, discussing an array of data analysis methods. Following, we will survey visualization methods that are sensitive to the results of these analyses, with a focus on how the various data features can be mapped to visual outcomes. Such knowledge can then be employed to reduce the parameter space, generating functional combinations of parameters to generate a single meta-parameter that can be conveniently controlled by a single slider. On the other hand, the visual expressions (manifestations) of the data features can be terms of rendered detail, color for highlighting, and also illumination and lighting. In that context, the great advantage that computer generated visualization imagery has is that it can "bend" the physics of illumination and light to bring out features in unfamiliar ways. However, the effects of these non-photorealistic renditions are widely unknown and must be studied to assess their success and impact, also in the context of the given task, intent, and target users. Since the human observer is often a black box to the visualization researcher, effective methods are sorely needed here, especially when faced with the vast number of visualization parameters and their settings. In this respect, as a final topic of this segment of the tutorial, we will present a statistical framework rooted in market research that allows for the analysis of parameters in a conjoint fashion, which greatly reduces the number of experiments required to produce a statistically significant and reliable study outcome.

Outline

- Introduction and motivation
  - Brief survey of state-of-the art visualization methods with a focus on rendering parameters
- Data analysis and parameter control
  - Survey of data analysis methods, rooted in computer vision and information theory
  - Reduction methods for visualization parameter space
- Focusing on the user

  - Survey of relevant work in psychology to assess human cognition and visual saliency
- Expressive, saliency-aware rendering methods
  - Survey of relevant aspects of illustrative rendering techniques (sparseness, color, lighting)
  - Fusion techniques to unify data from various sources
  - Lenses for effective screen and detail management
- Measuring impact on the user
  - Effective user study design and statistical evaluation (via conjoint analysis)
  - Tuning visualization parameters for task, intent, and user

Contribution

This segment of the tutorial surveys the latest research trends in data analysis, parameter control, and rendering methods to measure and express saliency in the data. The human aspect of visualization, including cognition assessment, user study design, and statistical evaluation, are also discussed in this talk.

Perception-Guided Transfer Function Specification and Quality Enhancement

Abstract

In this section, we introduce a framework for transfer function specification based on human perception of direct volume rendered images (DVRIs). We first present a transfer function design method based on editing direct volume rendered images. For many end users of visualization systems, it is more intuitive and convenient for them to directly work on DVRIs. Some DVRI editing operations, such as fusing features from different DVRIs, blending two DVRIs, and erasing unwanted features in DVRIs, may be very useful in practice. Thus, we introduce a novel framework based on image-similarity and genetic algorithms which allows users to directly edit features in DVRIs and interactively design transfer functions. A palette style intuitive interface is further developed to serve as the front-end of the framework. After that, we will describe how to enhance the quality of direct volume rendered images based on entropy correlation. The enhancement method is driven by the existing information in both the image and the volume. To deliver faithful DVRIs which can effectively convey the information in the volume, we proposed an image quality assessment scheme which measures the correlation between the entropy in the volume and the entropy in the DVRI. By adjusting the rendering parameters using genetic algorithm based on the quality metric, some more pleasing and informative DVRIs can be obtained. Finally, we present a set of quantitative effectiveness metrics including visibility, distinguishability, contour clarity, familiarity, and coherence for DVRIs. A transfer function design framework is then introduced to increase the effectiveness of DVRIs based on the proposed metrics.

Outline

- Introduction and motivation
  - Transfer function design
  - Object-based methods vs. image-based methods
  - Perception-guided transfer function design
- Transfer function design based on editing direct volume rendered images
  - Framework
  - Transfer function fusing
  - Blending and removing features in direct volume rendered images
  - Palette style interface
- Quality enhancement of direct volume rendered images
  - Entropy of volumes
  - Entropy of direct volume rendered images
  - Quality enhancement based on entropy correlation
- Transfer function design based on effectiveness metrics
  - Generic and quantitative effectiveness metrics
  - Visibility metric
  - Distinguishability metric
  - Contour clarity metric
  - Familiarity metrics
  - Coherence metrics
  - Transfer function design based on effectiveness metrics

Contribution

This part highlights how transfer function design can benefit from a thorough analysis of the human perception problems in direct volume rendered images. The methods introduced in the talk can automatically address the problems such as poor quality and low effectiveness existed in the direct volume rendered images and allows users to intuitively design transfer function solely in the image domain.

View Selection for 3D and Time-Varying Volume Visualization

Abstract
For three-dimensional volume visualization, suggestion of interesting viewpoints can improve both the speed and efficiency of data understanding. Automatic view selection becomes particularly useful when the visualization process is non-interactive - for example, when visualizing large data sets or time-varying sequences. In this talk, I will first present a viewpoint goodness measure based on the formulation of entropy from information theory. By taking into account the transfer function, the data distribution and the visibility of the voxels, this measure suggests interesting viewpoints and creates a partitioning of the view space based on view similarity to guide further exploration. Then, I will introduce a dynamic view selection algorithm that shows the maximum amount of information from a time-varying data set. The algorithm includes an improved view selection method for static data and a dynamic programming approach to select views for time-varying data. Finally, I will discuss a user interface, called the *LOD map*, for view selection and LOD comparison of multiresolution volume data. Based on an intuitive representation of LOD quality, the LOD map is a cost-effective visual interface for navigating multiresolution data exploration.

Outline

- Introduction
- Related work
- Information theory and entropy
- Data space approach
  - Data importance and noteworthiness
- Image space approach
  - Opacity distribution
  - Color distribution
  - Curvature distribution
- View space partitioning
  - View similarity
  - View likelihood and stability
- Dynamic view selection
  - View selection by dynamic programming
  - View smoothness
- Entropy and LOD selection

Contribution
This part of tutorial highlights the use of entropy for view selection in different volume visualization tasks, including static view selection, dynamic view selection and multiresolution data visualization. The suite of solutions covered in this talk enable the user to easily narrow down the visualization parameter space through automated algorithms or visual feedbacks, and select good choices of parameters that reveal a maximum amount of information.

LOD Selection and Multiscale Volume Data Quality Assessment

Abstract
As an important topic in the field of visualization, multiresolution techniques have proven to be a viable solution to address the challenges posted by data explosion. Building a hierarchy from the original data enables the user to select data of different resolutions for cost-effective analysis and visualization. In this talk, I introduce an interactive level-of-detail (LOD) selection algorithm using image-based quality metric and a reduced-reference, multiscale approach to volume data quality assessment.

The first part of this talk discusses quality-driven LOD selection and rendering of large volume data sets. An image-based quality metric is formulated based on an efficient way to evaluate the contribution of multiresolution data blocks to the final image. Techniques for interactive LOD decisions are proposed to ensure real-time update of the view-dependent information as well as adjust to transfer function changes. Compared with traditional metrics such as mean square error and signal-to-noise ratio, the LODs selected based on this image-based quality metric yield images of better visual quality.

In the second part, a reduced-reference volume data quality assessment algorithm is presented. The algorithm extracts important statistical information as feature from the original data in the wavelet domain. The feature incorporates visual importance of data in the visualization process. With extracted feature, we are able to identify and quantify the loss of quality in the reduced or distorted version of data, eliminating the need to access the original data again. The feature representation is naturally organized in the form of multiple scales, which facilitates quality assessment of data with different resolutions.

Outline

- Introduction and motivation
  - Large volume data and multiresolution visualization
  - Traditional solution vs. perception-driven solution

- Hierarchical data representation using wavelet transform
- Image-based quality metric
  - Importance value design
  - Multiresolution error evaluation
  - Run-time metric update
  - LOD selection and rendering
- Volume data quality assessment
  - Wavelet subband statistics
  - Voxel visual importance
  - Feature representation
  - Quality assessment and improvement

Contribution

This part of tutorial focuses on new multiresolution algorithms that incorporate perception reasoning towards a more effective LOD decision and data quality assessment. Technical details are discussed. Comparisons with traditional approaches are also provided.

Tutorial Notes

The tutorial notes will consist of the description of the tutorial, copies of the slides for each talk, and an extensive bibliography including specific references used in the tutorial as well as a general selection of relevant references.

Instructors

The background of each instructor is listed in alphabetical order.

Klaus Mueller
Stony Brook University

Klaus Mueller is currently an Associate Professor at the Computer Science Department at Stony Brook University, where he also holds co-appointments at the Biomedical Engineering and the Radiology Departments. He earned an MS degree in Biomedical Engineering in '91 and a PhD degree in Computer Science in '98, both from Ohio State University. His current research interests are computer and volume graphics, visualization, medical imaging, and computer vision. He won the NSF CAREER award in 2001 and has served as a program co-chair at various conferences, such the Volume Graphics Workshop, IEEE Visualization, and the Symposium on Volume Visualization and Graphics. He is the author of over 80 peer-reviewed journal and conference papers, has taught tutorials on various topics at IEEE Visualization every year since 2002, and he is a senior member of the IEEE.

Relevant Publications

N. Neophytou and K. Mueller
Color-Space CAD: Direct Gamut Editing in 3D
IEEE Computer Graphics & Applications, 28(3):88-98, 2008.

J. Giesen, K. Mueller, E. Schuberth, L. Wang, and P. Zolliker
Conjoint Analysis to Measure the Perceived Quality in Volume Rendering
IEEE Transactions on Visualization and Computer Graphics, 13(6):1664-1671, 2007.

H. Wong, H. Qu, U. Wong, Z. Tang, and K. Mueller
A Perceptual Framework for Comparisons of Direct Volume Rendered Images
In *Proceedings of IEEE Pacific-Rim Symposium on Image and Video Technology 2006*, pages 1314-1323, 2006.
Also in *Springer Lecture Notes in Computer Science, Advances in Image and Video Technology*, vol. 4319.

L. Wang, Y. Zhao, K. Mueller, and A. Kaufman
The Magic Volume Lens: An Interactive Focus+Context Technique for Volume Rendering
In *Proceedings of IEEE Visualization Conference 2005*, pages 367-374, 2005.

L. Wang and K. Mueller
Generating Sub-Resolution Detail in Images and Volumes Using Constrained Texture Synthesis
In *Proceedings of IEEE Visualization Conference 2004*, pages 75-82, 2004.

Huamin Qu
Hong Kong University of Science and Technology

Huamin Qu is an assistant professor at the Department of Computer Science and Engineering, the Hong Kong University of Science and Technology. His main research interests are in visualization and computer graphics. He has conducted research on sample-based rendering, volume visualization, graph visualization, visual analytics, and medical imaging. He received a BS in Mathematics from Xi'an Jiaotong University, China, an MS and a PhD in Computer Science from the Stony Brook

University.

Relevant Publications
Y. Wu and H. Qu
Interactive Transfer Function Design Based on Editing Direct Volume Rendered Images
IEEE Transactions on Visualization and Computer Graphics, 13(5):1027-1040, 2007.

M. Chan, Y. Wu, and H. Qu
Quality Enhancement of Direct Volume Rendered Images
In *Proceedings of IEEE/EG International Symposium on Volume Graphics 2007* (Cover Image), pages 25-32, 2007.

Y. Wu, H. Qu, and K. Chung
Quantitative Effectiveness Metrics for Direct Volume Rendering
Under Review, A shorter version is a best poster candidate at IEEE Visualization Conference 2007.

Han-Wei Shen
The Ohio State University

Han-Wei Shen is an Associate Professor at The Ohio State University. He received his BS degree from Department of Computer Science and Information Engineering at National Taiwan University in 1988, the MS degree in computer science from the State University of New York at Stony Brook in 1992, and the PhD degree in computer science from the University of Utah in 1998. From 1996 to 1999, he was a research scientist at NASA Ames Research Center in Mountain View California. His primary research interests are scientific visualization and computer graphics. Professor Shen is a winner of National Science Foundation's CAREER Award and US Department of Energy's Early Career Principal Investigator Award. He also won an Outstanding Teaching Award in the Department of Computer Science and Engineering at The Ohio State University.

Relevant Publications
U. D. Bordoloi and H.-W. Shen
View Selection for Volume Rendering
In *Proceedings of IEEE Visualization Conference 2005*, pages 487-494, 2005.

G. Ji and H.-W. Shen
Dynamic View Selection for Time-Varying Volumes
IEEE Transactions on Visualization and Computer Graphics, 12(5):1109-1116, 2006.

C. Wang and H.-W. Shen
LOD Map - A Visual Interface for Navigating Multiresolution Volume Visualization
IEEE Transactions on Visualization and Computer Graphics, 12(5):1029-1036, 2006.

Chaoli Wang
University of California, Davis

Chaoli Wang is a postdoctoral researcher in the Visualization and Interface Design Innovation (VIDI) research group, University of California, Davis. He received the PhD degree in computer and information science from The Ohio State University in December 2006. In his PhD work, he developed algorithms for managing and rendering large-scale three-dimensional and time-varying volume data sets. His current research focuses on importance-driven large data analysis and visualization.

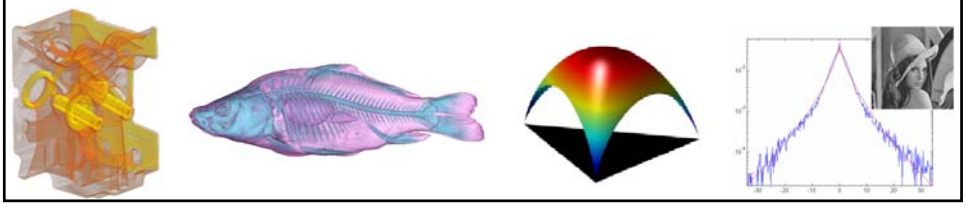
Relevant Publications
C. Wang, A. Garcia, and H.-W. Shen
Interactive Level-of-Detail Selection Using Image-Based Quality Metric for Large Volume Visualization
IEEE Transactions on Visualization and Computer Graphics, 13(1):122-134, 2007.

C. Wang and K.-L. Ma
A Statistical Approach to Volume Data Quality Assessment
IEEE Transactions on Visualization and Computer Graphics, 14(3):590-602, 2008.

Perception-Driven Techniques for Effective Visualization of Large Volume Data

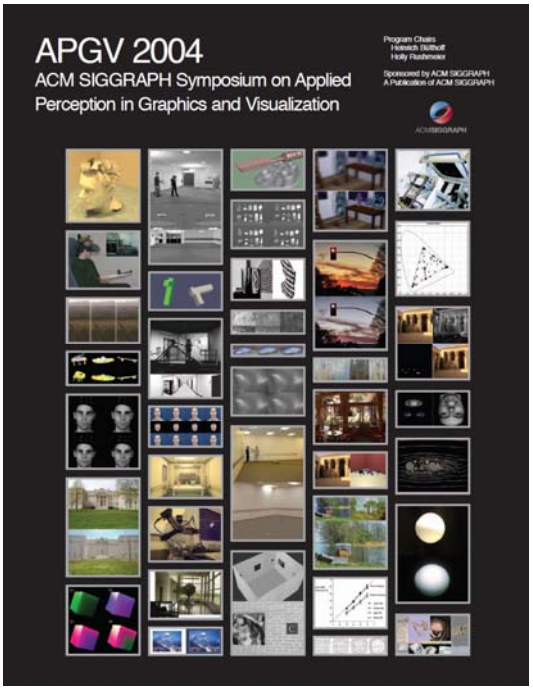
IEEE Visualization Tutorial

Columbus, OH, 19 Oct 2008

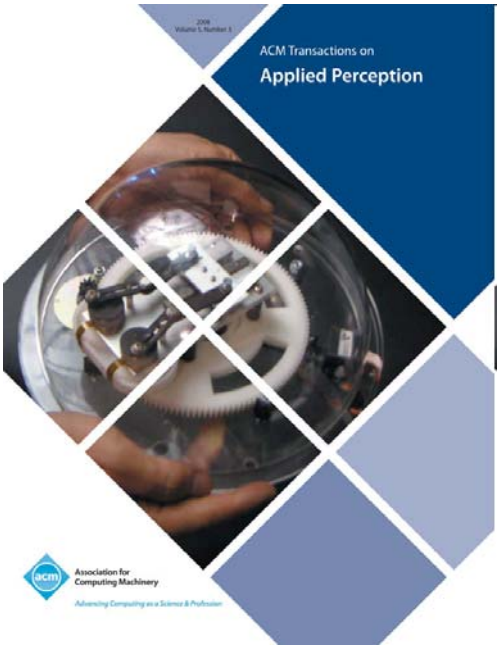


Speakers

- Klaus Mueller
  - Stony Brook University
- Huamin Qu
  - Hong Kong University of Science and Technology
- Han-Wei Shen
  - The Ohio State University
- Chaoli Wang
  - University of California, Davis



- ACM Symposium on Applied Perception in Graphics and Visualization
- Launched in 2004
- Topics:
  - Applied perception to visual, auditory, and haptic representation
  - Basic perception and cognition research



- ACM Transactions on Applied Perception
- Launched in July 2004
- Topics:
  - Visual
  - Auditory
  - Haptics
  - Sensorimotor
  - Multimodal rendering and multimodal interaction
  - Sensory integration

Perception in Volume Visualization

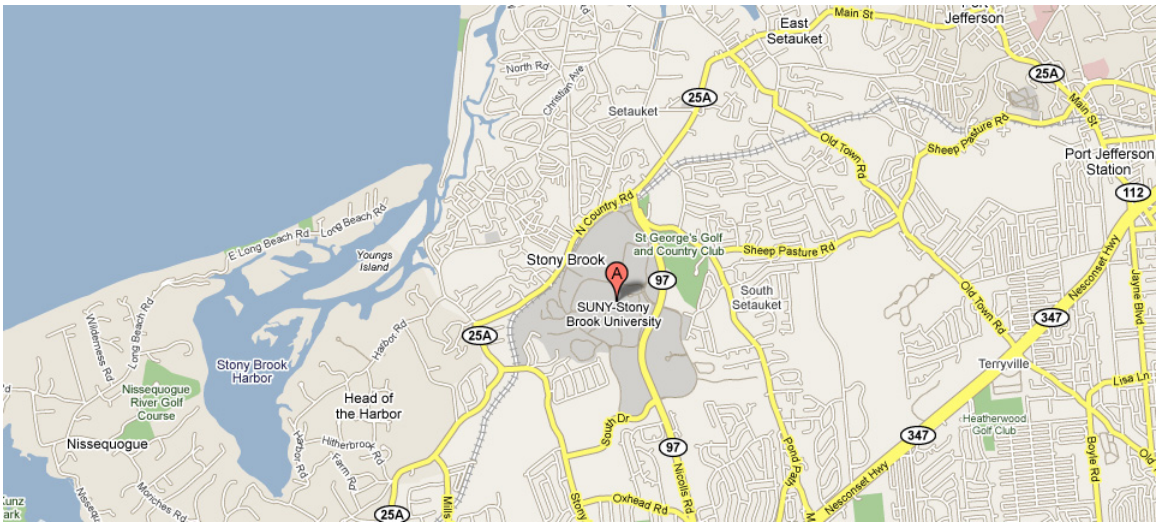
- Parameter selection for visual data exploration
  - Transfer function (color and opacity), lighting, camera position and path, …
- Large data analysis and visualization

Schedule

- 8:30-10:00
  - So Many Parameters, So Little Time: Guiding Users To Obtain Better Visualizations (Muller)
  - Perception-Based Transfer Function Design (Qu)
- 10:00-10:30
  - Break
- 10:30-12:00
  - Information and Visualization (Shen)
  - Perception-Driven Techniques for Large Volume Data Analysis and Visualization (Wang)

Klaus Mueller

Stony Brook University

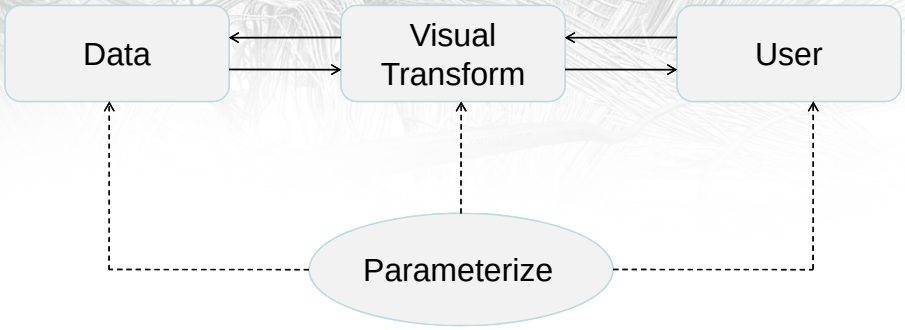


Perception-Driven Techniques for the Effective Visualization of Large Volume Data

So Many Parameters, So Little Time…. Guiding Users To Obtain Better Visualizations

Klaus Mueller
Department of Computer Science
Stony Brook University, USA
mueller@cs.sunysb.edu
http://www.cs.sunysb.edu/~mueller

The Conceptual Data Visualization Pipeline



- Parameterization of all pipeline components is essential
  - allows tuning and optimization of the visual transform given the data and the user
  - shall look at each pipeline component and then join them

What We Will NOT Talk About

- We shall not consider data arrangements here
  - such as grids, lattices, spatial dimensions, etc.
  - assume sampling is not an issue
  - assume interpolation and errors are understood
  - further assume that the methods generalize to higher spatial dimensions

What We Will NOT Talk About

- We shall not consider data arrangements here
  - such as grids, lattices, spatial dimensions, etc.
  - assume sampling is not an issue
  - assume interpolation and errors are understood
  - further assume that the methods generalize to higher spatial dimensions
- There is still plenty to if stuff to worry about ☺

Topic 1: The Data and Their Parameterization

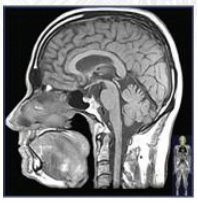
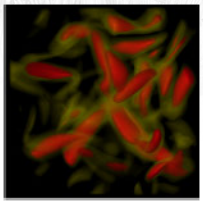
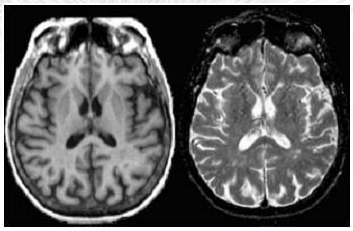
- Data may come as:
  - scalar data (densities)
  - multi-valued data (multi-variate)
  - vectors (vector fields)
  - and others
- Data parameterization = data characterization
- How can data be characterized?
  - their features
- What are these features?

Topic 1: The Data and Their Parameterization

- Data may come as:
  - scalar data (densities)
  - multi-valued data (multi-variate)
  - vectors (vector fields)
  - and others
- Data parameterization = data characterization
- How can data be characterized?
  - their features
- What are these features?
  - this is the hard part ☺

The Raw Data

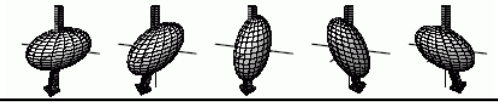
- Scalar density fields (topic of this tutorial)

medical scientific multi-modal or multi-variate (example: T1/T2 MRI)

- Vector fields

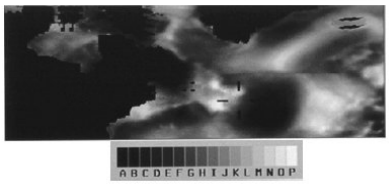
- Tensor fields
  - MRI DTI

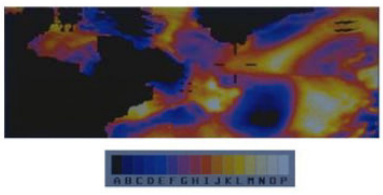


Direct Visualization Of Scalar Densities

- Contrast: the role of color
  - variations in brightness (grey levels) encode local contrast well
  - but the range of distinguishable grey levels is small (~100)

  → grey levels are good for local but not for global contrast



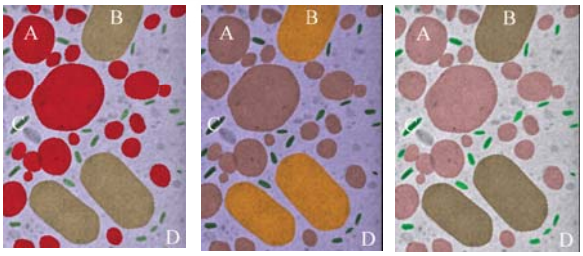


[Ware 04]

Direct Visualization Of Scalar Densities

- Color for highlighting
  - color is effective in guiding viewer attention to salient features

  → in particular, vividness (saturation) is important here

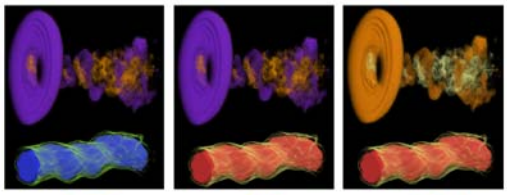


[Wang 08b]

Direct Visualization Of Scalar Densities

- Aesthetics
  - color can make a display more cheerful and pleasing
  - aesthetic design can also reduce stress in problem solving tasks
  - objects considered beautiful stimulate different areas in the brain than those considered unattractive [Kawabata 04]

  → this motivates the use of harmonized color schemes



non-harmonic T-harmonic V-harmonic

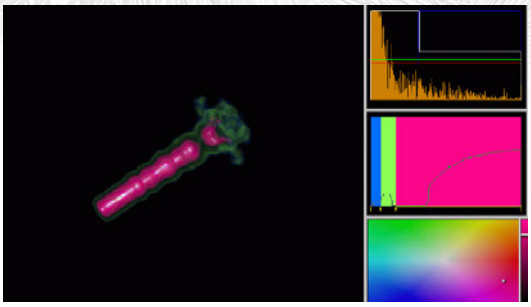
[Wang 08a]

Direct Visualization Of Scalar Densities

- At this point, we have done analysis only on a per pixel-basis
  - may have involved global scene analysis (e.g., for highlighting)
- One may map scalar densities to
  - other scalar densities: windowing of interesting ranges
  - colors
  - transparencies
- This mapping may be driven by functions of
  - importance
  - aesthetics
  - certainty
  - and others

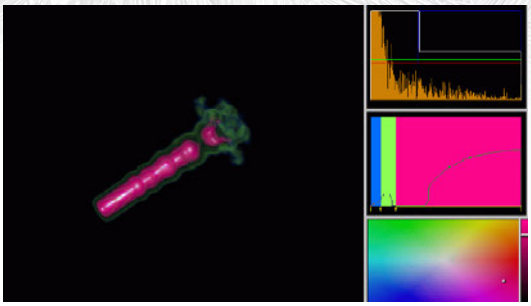
Direct Visualization Of Scalar Densities

- Essentially we get a 1-D transfer function: density→ color



Direct Visualization Of Scalar Densities

- Essentially we get a 1-D transfer function: density→ color

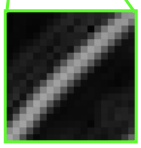


- Let us now look at more complex analyses
  - creating new, derived data

Accentuate Events In The Data

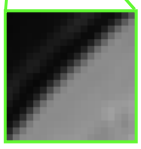
- Flat, uniform regions are not particularly interesting
- We are interested in events and critical points → the features
  - thus, accentuate discontinuities and variations in the data
- Visually convey these events by graphical techniques
- Can still use transfer functions for this
  - their complexity grows with the complexity of the event descriptor
- Distinguish between:
  - analytic feature detection via derivatives and moments
  - analytic feature detection looking for topology changes
  - statistical feature detection calculating histograms and variance

Derivatives

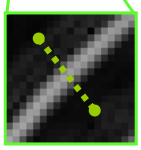


gradient magnitude

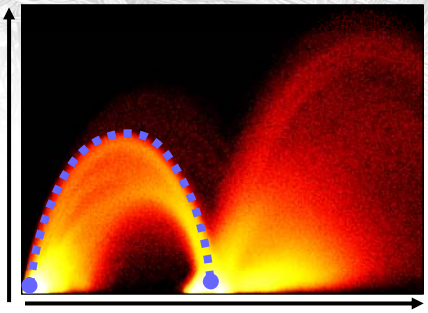
data (CT) value



Derivatives: Two-Dimensional Transfer Function



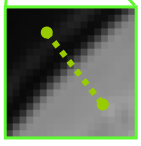
gradient magnitude



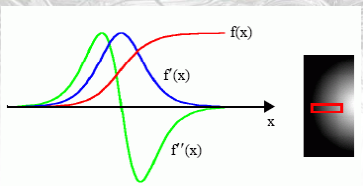
Boundaries in volume create arches in (value, gradient) domain [Kindlmann 98]

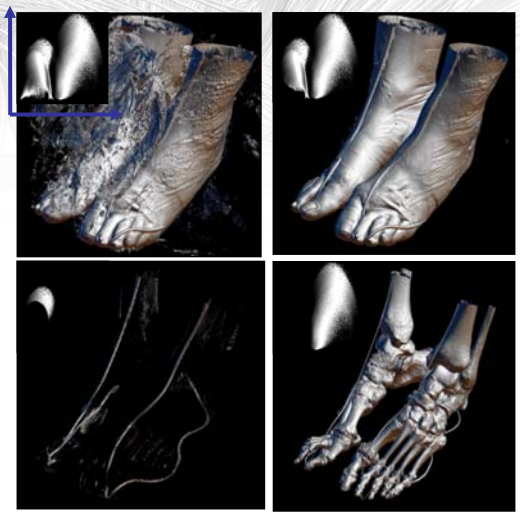
These arches can guide placement of opacity to emphasize material interfaces [Kniss 01]

data (CT) value



Three-Dimensional Transfer Function





Boundaries can be described in terms of:

- maximum in 1st derivative
- zero-crossing in 2nd derivative

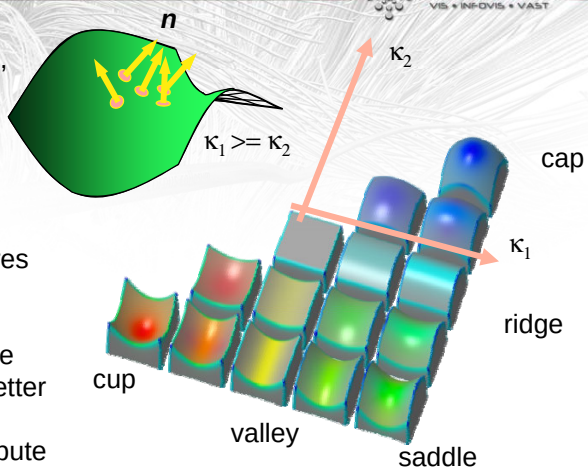
Semi-automatic classification possible in clean data

Transfer Function for Perceptual Enhancement

Add in additional properties, such as curvature

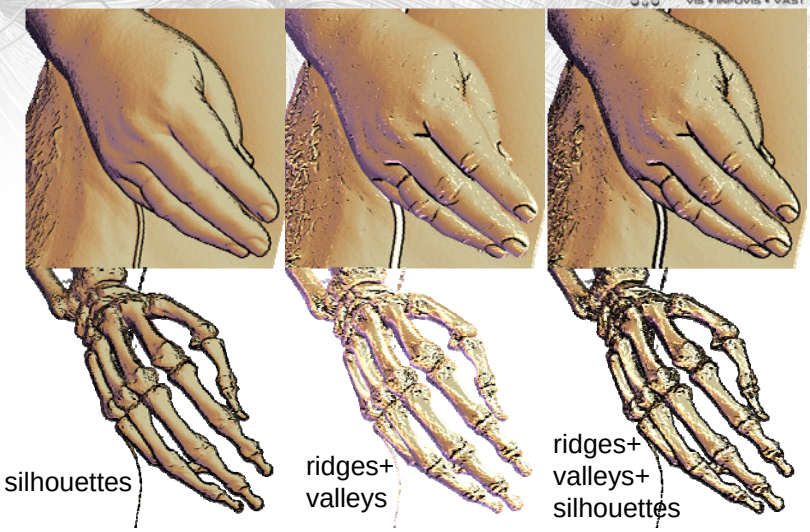
Curvature: how the change in surface position changes surface normal (***n***)

- principal curvature features (κ_1, κ_2) form the transfer function domain
- curvatures enable surface surface enhancement, better control over silhouettes
- convolution used to compute 1st and 2nd derivatives



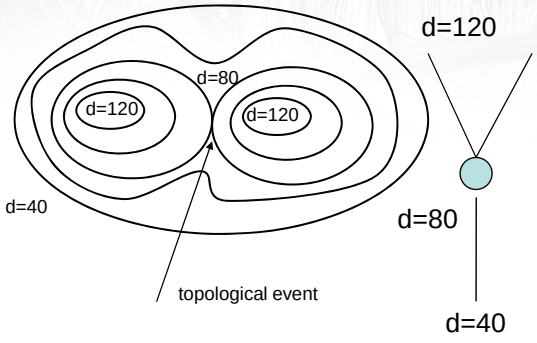
[Kindlmann 03]

Effects Of Curvature Enhancement



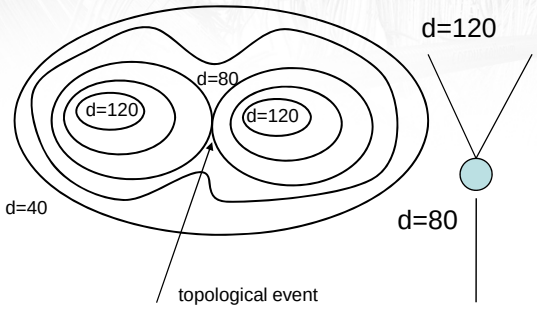
Analysis Of Level Set Topology

- Level-Set = Iso-contour
  - contours of equal interpolated scalar density



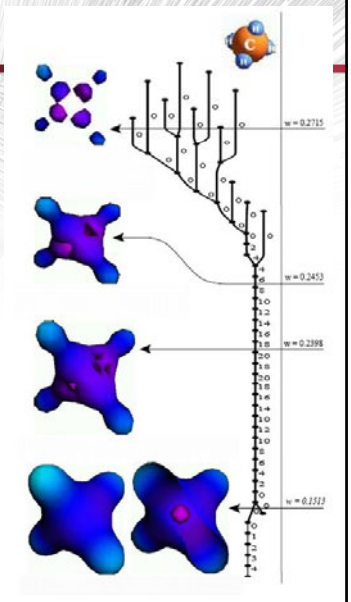
Analysis Of Level Set Topology

- Level-Set = Iso-contour
  - contours of equal interpolated scalar density



Contour tree
Reeb Graph

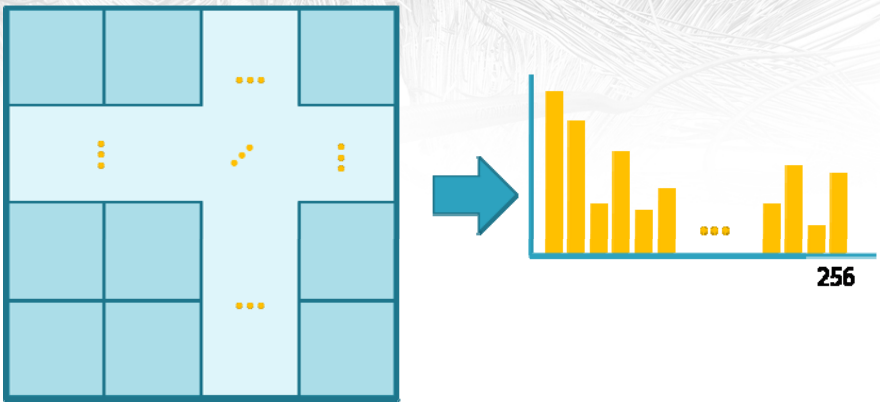
[Pascucci 03]



Statistical Features

- What to do when there are no concrete topological events or boundaries, yet the density field is not uniform?
  - simple example found in nature: smoke
- Assess the spectrum of density variations
  - density histograms
- Apply a descriptor rooted in human perception
  - humans most sensitive to 1st and 2nd spatial derivatives
  - already used in the transfer function context
  - now use in a statistical context

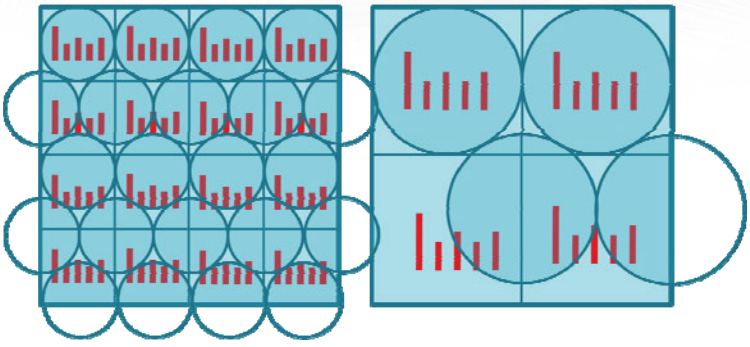
Density Global Histogram



this and following slides: [Nam 08]

Density Local Histogram

- Density signatures in local histograms at hierarchy of window sizes
- Detect density statistics at multiple levels of scales
- Representation to capture the essence of an object.



SIFT (Scale Invariant Feature Transform)

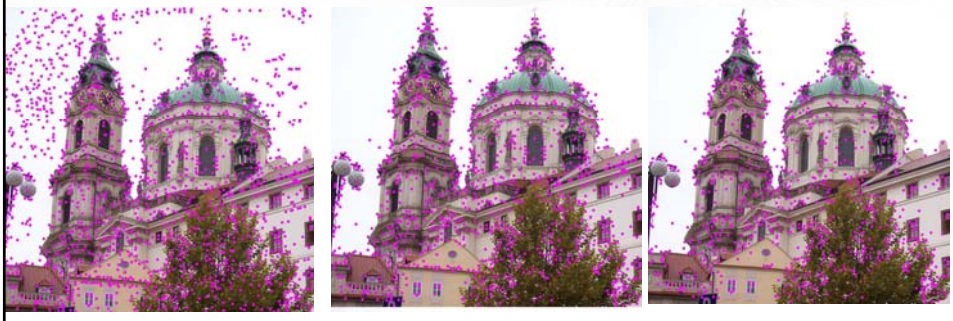
- Gradient histogram of local neighborhood
- Highly expressive of a local neighborhood's salient dynamics
- Invariant to scale, translation and rotation

- Algorithm
  - the detection of critical points (the *keypoints) in scale-space*
  - the encoding of these into *keypoint descriptors*

SIFT [Lowe 04]

SIFT (Scale Invariant Feature Transform)

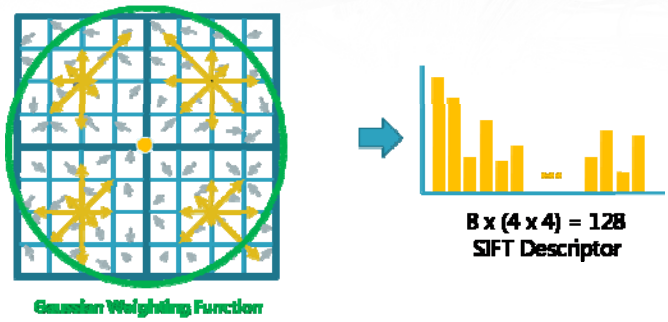
- Find keypoints
  - local extremas in a difference-of-Gaussians in multi-scale space
- Discard low contrast keypoints
- Filter out keypoints situated on edges



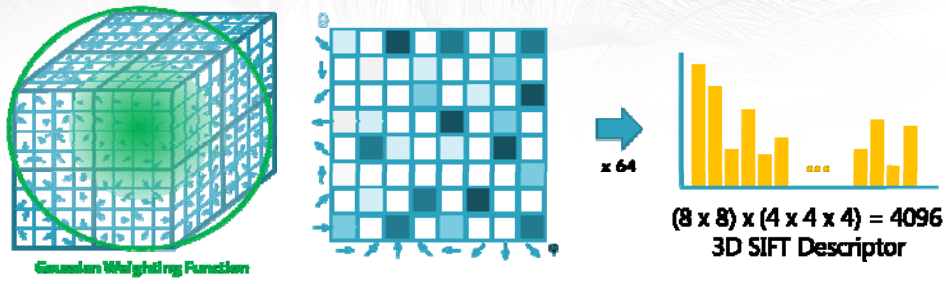
- Pictures from Wikepedia.org

SIFT (Scale Invariant Feature Transform)

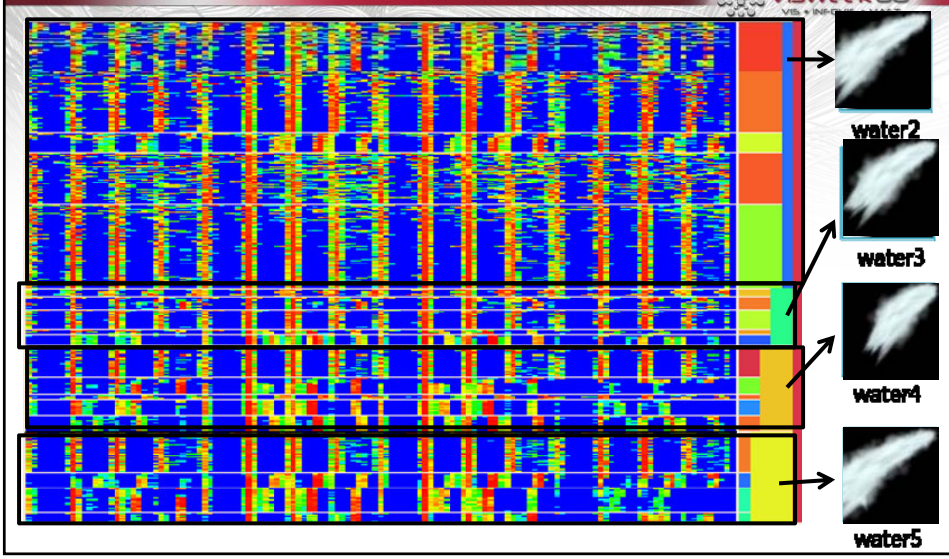
- Keypoint descriptor
  - the magnitude and orientation at each sample point around the keypoint location
  - weighted by a Gaussian function to achieve a certain level of smoothing.
  - aggregated into orientation histograms describing the neighborhood



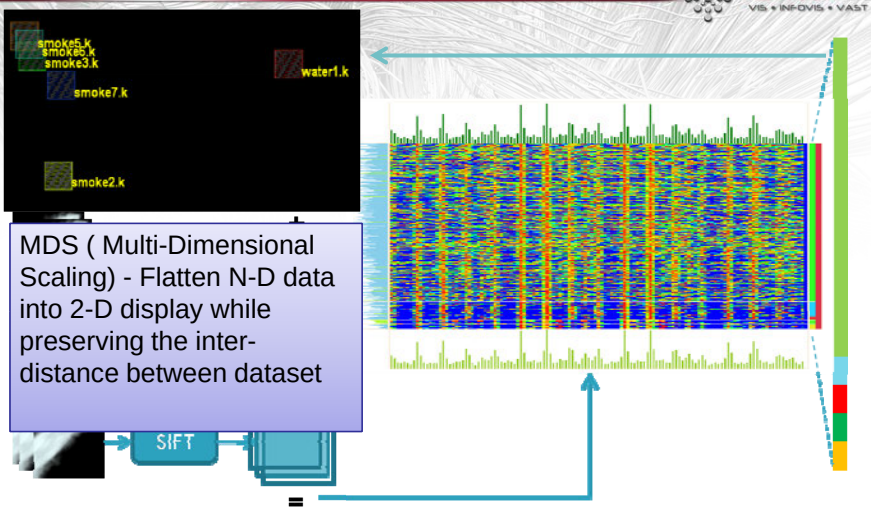
3D SIFT



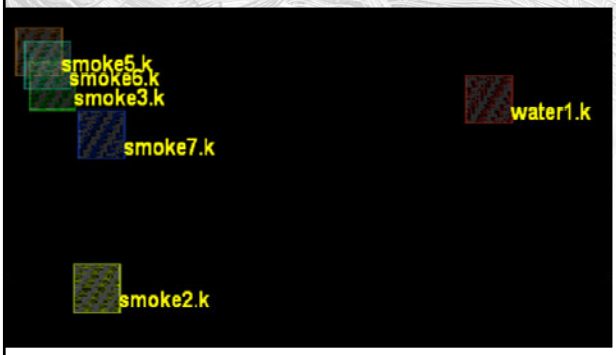
Cluster Analysis on Visual Contents



Feature Visualization in an Information Space



MDS Analysis: Categorization



1. Categorize different groups of flows
2. Distinguish different features within same category

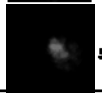
Water dropping from top-right to bottom-left corner

Smoke flow starting from the bottom-center

Smoke flow narrower than smoke2

Smoke flow similar with smoke3

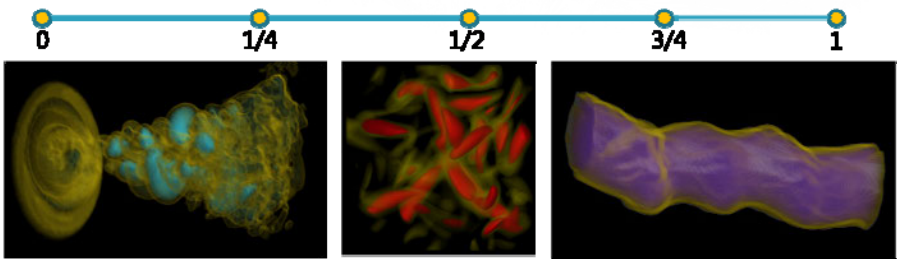
Smoke flow similar with smoke3 and 5



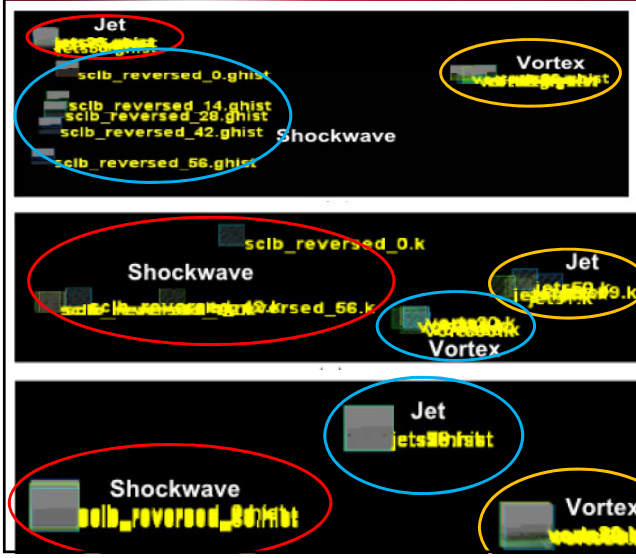
Smoke flow similar with smoke3,5, and 6 but different direction

Categorization Of 3D Flows

- 5 frames extracted from each series
- Features
  - Global histogram
  - Local histogram
  - 3D SIFT



MDS Analysis



Global histograms

3D SIFT

Local histograms

Conclusion: Data Features

- The more the data characteristics are understood the more specific the features will be (in most cases)
  - opposite extremes: feature templates vs. neural networks
  - others are somewhere in between
- Feature specification can be embedded in a data exploration process
  - neural networks require users to provide feature examples in the dataset
  - these may then be re-used in later visualizations

Topic 2: Visual Transform

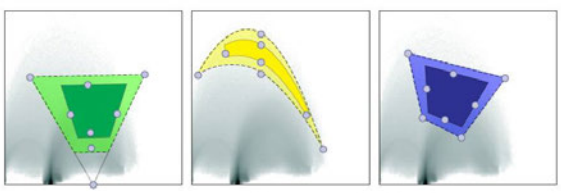
- Determines how features are expressed into visual manifestations = their visual appearance
- Features may control the rendering pipeline at various stages:
  - local color and opacity (mapping via transfer function)
  - scene composition (local sparseness, warping by lenses)
  - rendering style (lighting model, illustrative techniques)
  - iconic sprites (specific visual expression)

Topic 2: Visual Transform

- We can use transfer functions to maps feature parameters into visual transform parameters
  - what to do when parameter vector is large?
  - what to do when transfer function is complex?
- We have seen clustering/MDS as a way to visualize similar features
  - implicit parameterization is given by location in MDS cluster
- Can we make the parameterization more explicit?
  - detect parameter combinations sensitive to change
  - come up with templates given prior experiences

Example: Complex Transfer Function

A more elaborate value-gradient transfer function parameterization:



Typically, datasets typically deviate only modestly from this

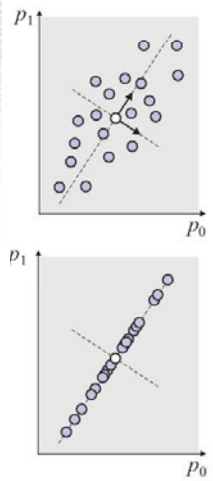
- but they do so in complex ways
  → lots of tedious tweaking is required

[Rezk-Salama 06]

Parameter Aggregation

We can learn these small deviations by observing a few datasets

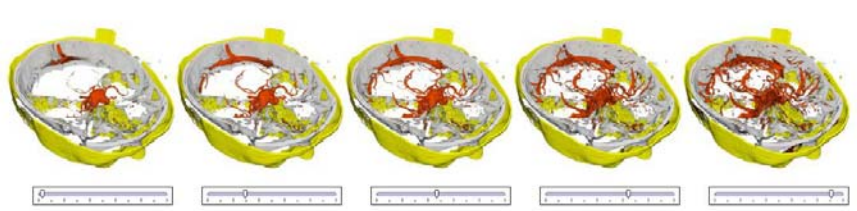
- encode the parameters into an N-D vector
- find the principal component of the vectors (the main Eigenvector)
- project all other vectors onto this Eigenvector
- the min and max then represent the min and max of the slider



[Rezk-Salama '06]

Transfer Function Simplification

Transformed aggregation enables transfer function simplification from N-D to 1-D



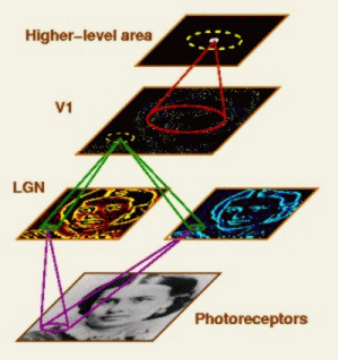
- works here since in CT usually only small deviations exist
- but these small deviations require complex interactions in the transfer function domain

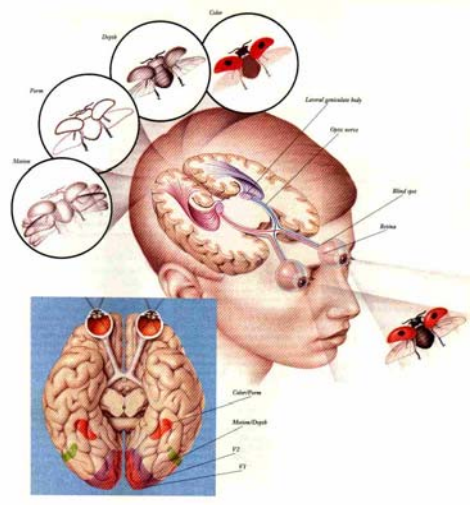
[Rezk-Salama '06]

Topic 3: The User, The Human Visual System

Visual cortex breaks input up into different aspects:

- color, shape, motion, depth





Visual Saliency

- Notion of visual importance

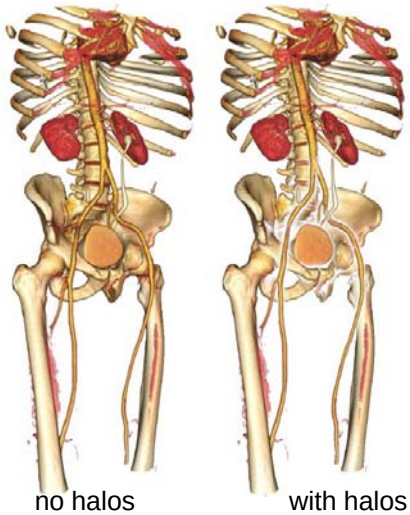
Visual Saliency

- Notion of visual importance
- Visual transform of data features to direct a viewer's attention
  - shape (edges, silhouettes)
  - surface (curvatures, suggestive contours)
  - size
  - intensity and color
  - texture

Visual Saliency

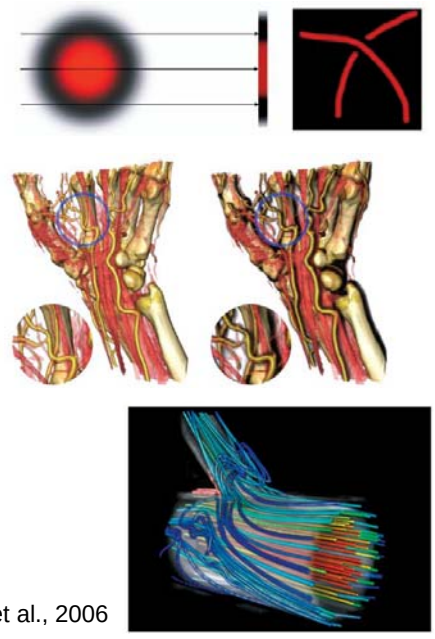
- Notion of visual importance
- Visual transform of data features to direct a viewer's attention
  - shape (edges, silhouettes)
  - surface (curvatures, suggestive contours)
  - size
  - intensity and color
  - texture
- Enhancement / suppression makes this more effective
  - opacity controls presence
  - rendering style and texture control expression and appearance
  - illumination controls shading
  - intensity and color control attention (by highlighting)
  - caricature controls shape
  - but these influences are typically mixed (and not exclusive)

Halos



no halos with halos

Bruckner et al., 2006



Wenger et al., 2006

Two Levels Of Abstraction

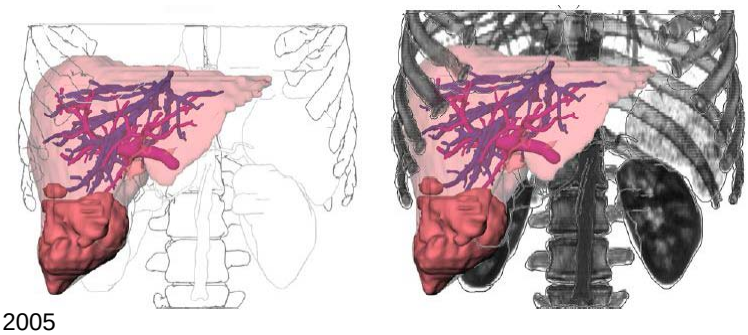
- Low-level abstraction:
  - concerned with **how** objects are represented
  - stylized depiction: silhouettes, contours, pen+ink, stippling, hatching, etc.

Two Levels Of Abstraction

- Low-level abstraction:
  - concerned with **how** objects are represented
  - stylized depiction: silhouettes, contours, pen+ink, stippling, hatching, etc.
- High-level abstraction
  - deal with what should be visible and recognizable and at what level of detail
  - this should be importance-driven, that is, the current visualization goal controls feature rendering style and visibility

Mixing Rendering Styles

- First, classify the scene:
  - *Focus Objects (FO):* objects in the center of interest are emphasized in a particular way
  - *Near Focus Objects (NFO):* important objects for the understanding of the functional interrelation or spatial location.
  - *Context Objects (CO):* all other objects (rendered e.g., as silhouettes)
  - *Container Objects (CAO):* one object that contains all other objects.
- Render these in a certain order to ensure visual consistency



Tietjen et al., 2005

Attention

VisWeek 08
VIS + INFOVIS + VAST

- The cognitive process of selectively concentrating on one thing while ignoring other things
  - detecting features in visual clutter (CAPTCHA, next slide)
  - detecting coherent speech in noisy environments (cocktail party effect)
  - ignore features while concentrating on others (recall gorilla)
  - can also have divided attention (example: cell phone + driving)
  - heavily studied in psychology and neuroscience
  - closely tied to perception

Attention

- The cognitive process of selectively concentrating on one thing while ignoring other things
  - detecting features in visual clutter (CAPTCHA, next slide)
  - detecting coherent speech in noisy environments (cocktail party effect)
  - ignore features while concentrating on others (Simon's Gorrilla)
  - can also have divided attention (example: cell phone + driving)
  - heavily studied in psychology and neuroscience
  - closely tied to perception
- Attention theory is important for visualization as well
  - in contrast to computer vision, WE design/create the scene
  - this design guides the attention of the viewer
  - guidance determined by visualization goals

Attention

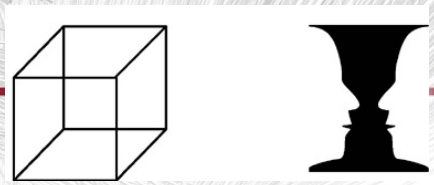
- The cognitive process of selectively concentrating on one thing while ignoring other things
  - detecting features in visual clutter (CAPTCHA, next slide)
  - detecting coherent speech in noisy environments (cocktail party effect)
  - ignore features while concentrating on others (Simon's Gorilla)
  - can also have divided attention (example: cell phone + driving)
  - heavily studied in psychology and neuroscience
  - closely tied to perception
- Attention theory is important for visualization as well
  - in contrast to computer vision, WE design/create the scene
  - this design guides the attention of the viewer
  - guidance determined by visualization goals
- Therefore it is important to understand mechanism of attention

Visual Recognition and Attention

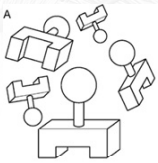
- Two opposing theories:
  - Gestalt
  - Feature integration
- Gestalt theory
  - top-down approach
  - proposes that the operational principle of the brain is holistic, parallel, and analog, with self-organizing tendencies
  - important in user interface design (button grouping, etc)
- Feature integration theory
  - bottom-up approach
  - primary visual features are processed and represented with separate **feature maps**
  - these are later integrated in a **saliency map** that can be accessed in order to direct attention to the most conspicuous areas

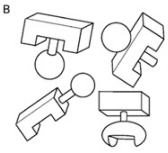
Gestalt Theory: Confirming Examples

Emergence

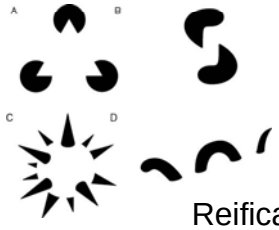


Multi-Stability

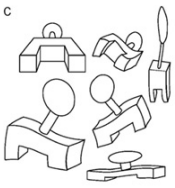


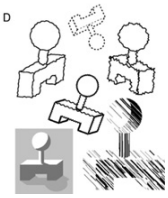


Invariance



Reification





Gestalt Theory: Opposing Examples

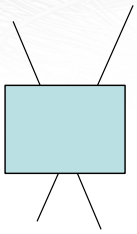
- **Selective-Encoding:**
  - involving one to distinguish what is important in a problem and what is irrelevant (i.e., filtering)
- **Selective-Comparison:**
  - identifying information by finding a connection between acquired knowledge and experience
- **Selective-Combination:**
  - identifying a problem through understanding the different components and putting everything together.

Feature Integration Theory

- One of the most influential psychological models of human visual attention in recent years
- Two types of visual search mechanisms
- Feature search
  - can be performed fast and pre-attentively for targets defined by primitive features (such as color, orientation, intensity, etc)
- Conjunction search
  - serial search for targets defined by a conjunction of primitive features
  - much slower
  - requires conscious attention
- Very promising technique for computer vision to detect partially occluded objects (SIFT)

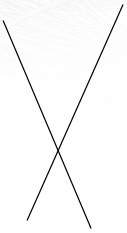
What Does It Mean For Visualization?

- Feature integration theory:
  - justifies enhancement of features
  - exploit this to guide attention
  - relatively “easy” since it involves mostly local enhancements
  - notion of feature saliency is important
- Gestalt theory:
  - justifies omission of detail to save space
  - viewers assume continuity of occluded lines
  - underlies ghosting techniques (mental feature completion)
  - silhouettes and contours for context objects
  - many techniques used now in illustrative rendering
  - recall also optical illusions



What Does It Mean For Visualization?

- Feature integration theory:
  - justifies enhancement of features
  - exploit this to guide attention
  - relatively “easy” since it involves mostly local enhancements
  - notion of feature saliency is important
- Gestalt theory:
  - justifies omission of detail to save space
  - viewers assume continuity of occluded lines
  - underlies ghosting techniques (mental feature completion)
  - silhouettes and contours for context objects
  - many techniques used now in illustrative rendering
  - recall also optical illusions



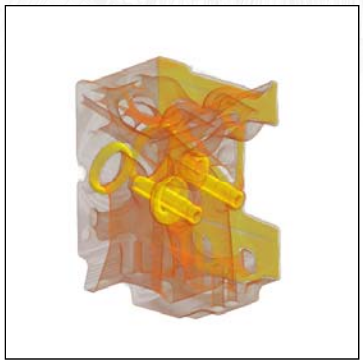
Ghosting

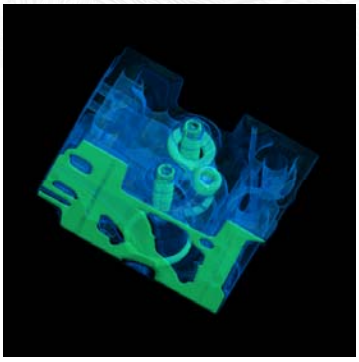
Topic 4: User Studies Are Important

- Some design rules exist, but combinations are often untested
- Also consider
  - user background (education, age, gender, profession, attitude, etc)
  - underlying task and application (medical, business, science, etc)
  - computational resources and level of interactivity sought
  - other factors
- User studies can reveal this insight
  - they allow, in some sense, a parameterization of the user
- An effective and efficient means for user studies is *conjoint analysis*
  - allows parameters to be tested in a conjoint fashion, via pair-wised comparison tests (or task-based tests)
  - subsequent statistical analysis then separates the sensitivities of these parameters

Sample Testing Scenario

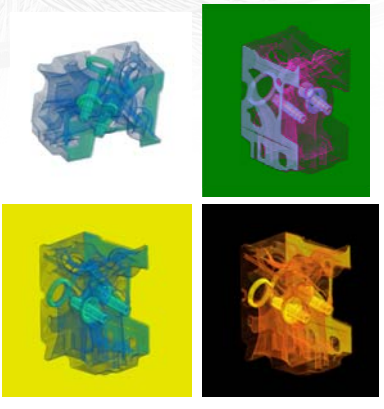
- Which color transfer function shows more detail?





Putting Conjoint Analysis to the Test

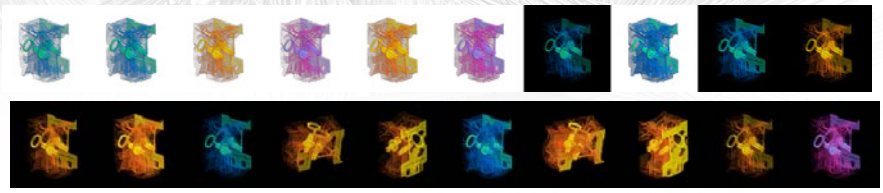
- Performed a user study on a multi-parametric visualization scenario
- On a set of 2700 images of engine blocks, we varied:
  - color transfer function (3)
  - rendering mode (5)
  - viewpoint (6)
  - image resolution (2)
  - ray step size (3)
  - background (5)
- Tested
  - 786 respondents
  - 20 pair-wise tests each



[Giesen '07]

User Study Results

- Top 10 (detail / aesthetics):



- Flop 10 (detail / aesthetics):

Wrap-Up

- Define the features that best characterize your visualization task
- Devise a suitable feature retrieval method
- Find a suitable mapping of these to salient visual representations
- Confirm and tune via user studies

References (1)

- [Bruckner 06] S. Bruckner, S. Grimm, A. Kanitsar, E. Gröller, “Illustrative Context-Preserving Exploration of Volume Data,” IEEE Trans. Vis. Comput. Graph., 12(6):1559-1569, 2006.
- [Giesen 08] J. Giesen, K. Mueller, E. Schuberth, L. Wang, and P. Zolliker, "Conjoint analysis to measure the perceived quality in volume rendering," IEEE Trans. Visualization and Computer Graphics, 13(6): 1664-1671, 2007.
- [Kawabata 04] H. Kawabata, S. Zeki, "Neural correlates of beauty," J. Neurophysiology, 91:1699–1705, 2004.
- [Kindlmann 98] G. Kindlmann and J. Durkin, "Semi-automatic generation of transfer functions for direct volume rendering," Symp. Volume Visualization '98, pp. 79-86, 1998
- [Kindlmann 03] G. Kindlmann, R. Whitaker, T. Tasdizen, T. Möller, “Curvature-Based Transfer Functions for Direct Volume Rendering: Methods and Applications,” IEEE Visualization, 513-520, 2003.
- [Kniss 02] J. Kniss, G. Kindlmann, and C. Hansen, "Multidimensional transfer functions for interactive volume rendering," IEEE Trans. Visualization and Computer Graphics, vol. 8, no. 3, pp. 270-285, 2002.

References (2)

- [Lowe 04] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," Intern. Journal of Computer Vision, 60(2):91-110, 2004.
- [Nam 08] J. Nam, M. Maurer, K. Mueller, "High-Dimensional Feature Descriptors to Characterize Volumetric Data, “ 2nd Workshop on Knowledge-Assisted Visualization (KAV), (to be presented), Columbus, OH, October, 2008.[Pascucci 03] V. Pascucci, K. Cole-McLaughlin, “Parallel Computation of the Topology of Level Sets. Algorithmica 38(1):249-268, 2003.
- [Rezk-Salama 06] C. Rezk-Salama M. Keller, and P. Kohlmann, “High-level user interfaces for transfer function design with semantics," IEEE Visualization ’06 (IEEE Trans. Visualization and Computer Graphics), 2006.
- [Tietjen 05] C. Tietjen, T. Isenberg, B. Preim, “Combining Silhouettes, Surface, and Volume Rendering for Surgery Education and Planning,” EuroVis, pp. 303-310, 2005.
- [Wang 08a] L. Wang, K. Mueller, "Harmonic Colormaps for Volume Visualization,“ *Volume Graphics Symposium*, Los Angeles, August, 2008.
- [Wang 08b] L. Wang, J. Giesen, K. McDonnell, P. Zolliker, K. Mueller, "Color Design for Illustrative Visualization," (to appear), *IEEE Transactions on Visualization and Computer Graphics, (Special issue IEEE Visualization Conference),* 2008.

References (3)

- [Ware 04] C. Ware. Information Visualization: Perception for Design. Morgan Kaufmann, 2nd edition, 2004.
- [Wenger 04] A. Wenger, D. Keefe, S. Zhang, D. Laidlaw, “Interactive Volume Rendering of Thin Thread Structures within Multivalued Scientific Data Sets,” IEEE Trans. Vis. Comput. Graph. 10(6): 664-672, 2004.

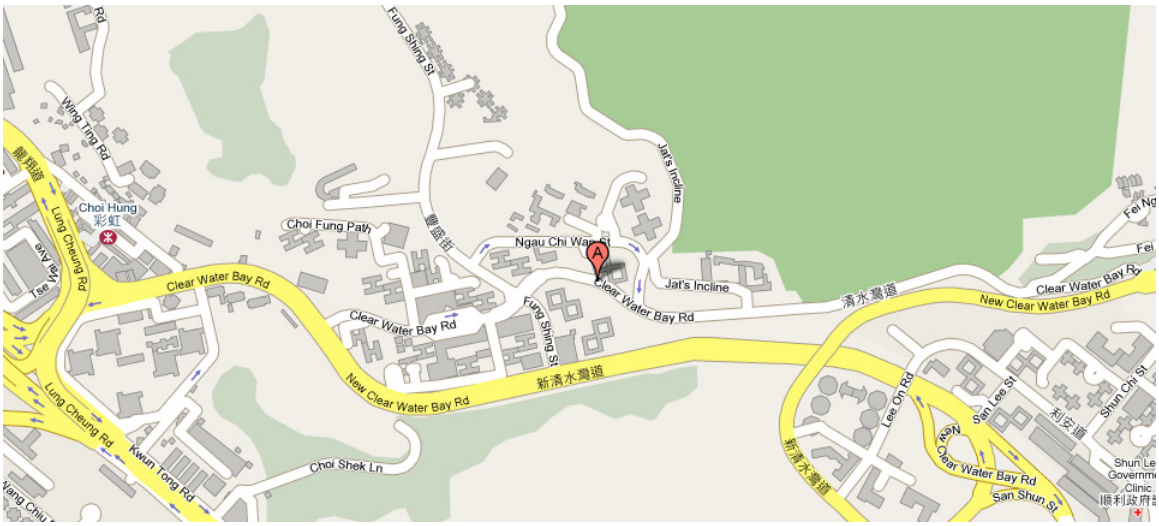
More and Up-To-Date Information

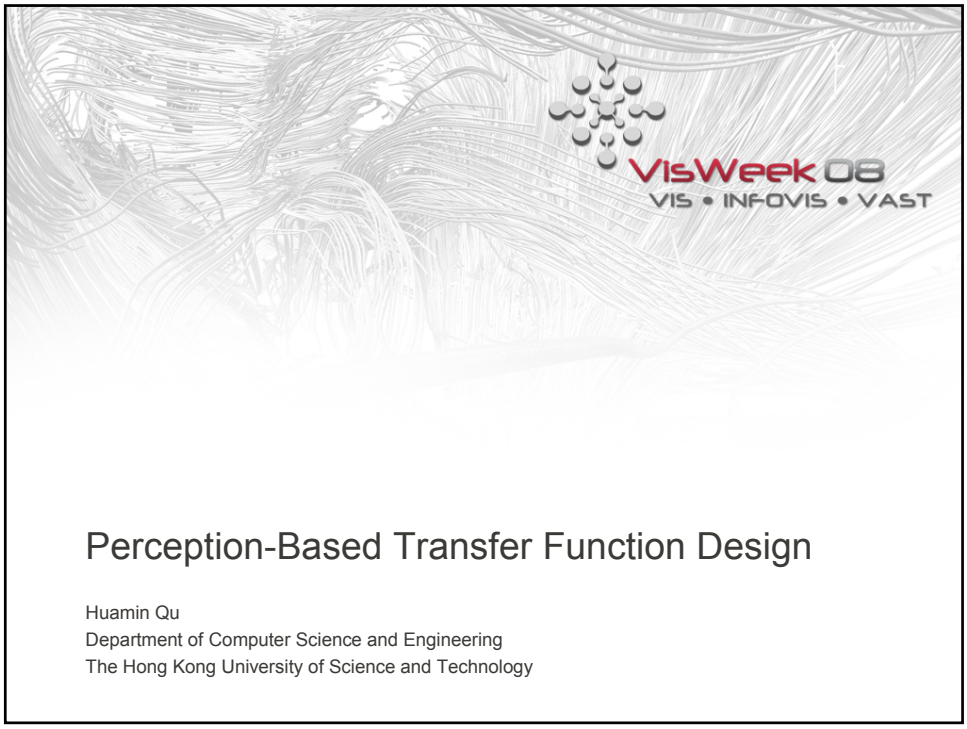
- Visit http://vis.cs.ucdavis.edu/~wangcha/vis08-tutorial.htm

- Support was provided by NSF grants ACI-0093157 and CCF-0702699, and NIH grant 5R21EB004099-02,

Huamin Qu

The Hong Kong University of Science and Technology





Perception-Based Transfer Function Design

Huamin Qu
Department of Computer Science and Engineering
The Hong Kong University of Science and Technology

Outline

- Introduction
- Transfer Function Design Based on Editing Direct Volume Rendering Images
- Quality Enhancement of Direct Volume Rendered Images
- Quantitative Effectiveness Metrics for Direct Volume Rendering

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

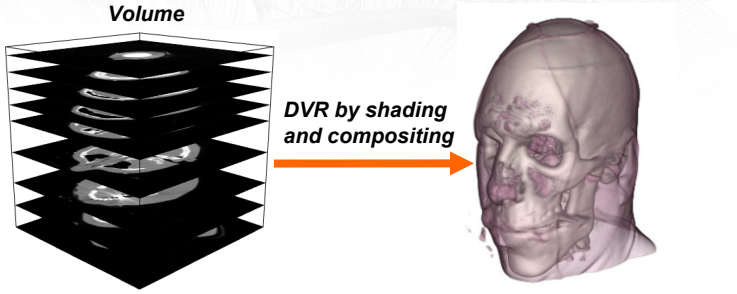
Outline

- Introduction
- Transfer Function Design Based on Editing Direct Volume Rendering Images
- Quantitative Effectiveness Metrics for Direct Volume Rendering
- Quality Enhancement of Direct Volume Rendered Images

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Direct Volume Rendering

- *Direct Volume Rendering* (DVR) is a powerful and flexible volume visualization tool and has been widely used in many fields



Direct Volume Rendered Image (DVRI)

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

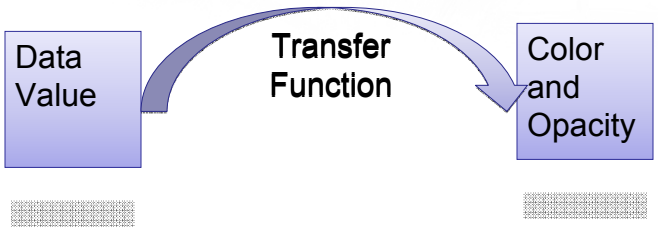
Object-Based Methods V.S. Image-Based Methods

- Direct Volume Rendering (DVR):
  - Object-order DVR (forward mapping)
  - Image-order DVR (backward mapping)

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

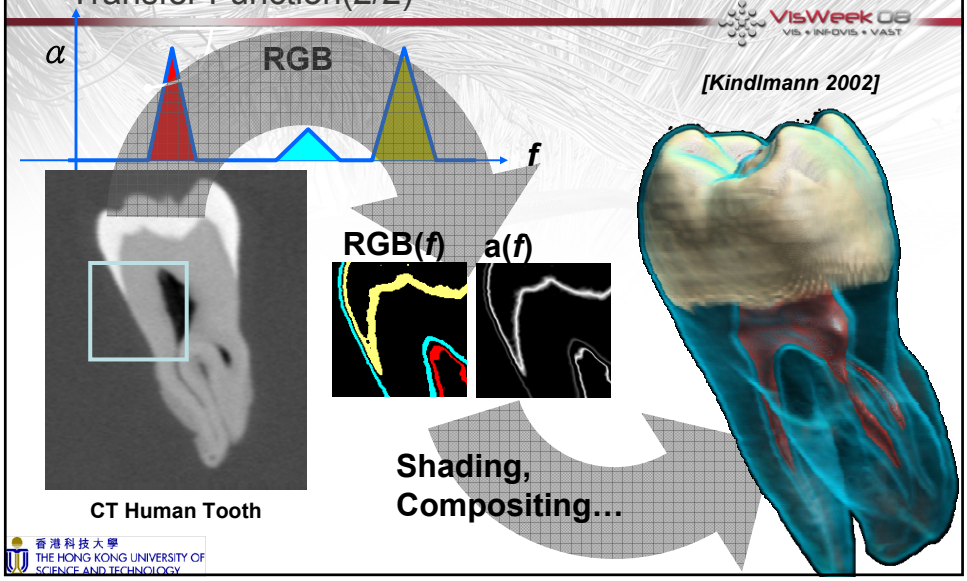
Transfer Functions (1/2)

- *Transfer functions* (TFs) assign opacity and color to the different features in the volume data
  - Emphasize the region of focus
  - Subjugate the unimportant details



香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Transfer Function(2/2)



香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Perception-Guided Transfer Function Specification

- The effectiveness of DVR largely depends on the TF used
  - Appropriate TFs allow users to reveal important features in the data
  - Inappropriate ones may obscure these features
- Finding appropriate TFs is difficult in practice
  - One major reason is that the search space for finding TFs is huge even for one dimensional TFs, not to mention multi-dimensional TFs
- We propose a perception-guided volume exploration framework
  - Transfer function design framework based on editing DVRIs and its front-end intuitive interfaces
  - Quality enhancement for the edited DVRIs
  - Quantitative effectiveness evaluation for direct volume rendering

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Outline

- Introduction
- Transfer Function Design Based on Editing Direct Volume Rendering Images
- Quality Enhancement of Direct Volume Rendered Images
- Quantitative Effectiveness Metrics for Direct Volume Rendering

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Motivation

- Physicians usually prefer to directly work on 2D slice images rather than in the TF domain
  - It is more straightforward for them to identify features in 2D slice images
- Some 3D structures can be more easily identified in direct volume rendered images (DVRIs) than in 2D gray-scale slice images
  - Therefore, it is more intuitive and convenient for users to directly work on DVRIs
- Usually, a number of partially good DVRIs can be easily generated by previous volume visualization methods, however, these DVRIs may only partially satisfy users' demands
  - Some context in one image may need to be removed
  - Some features in different image may need to be combined

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

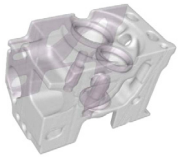
Two Straight Solutions

- Some DVRI editing operations, such as fusing features from different DVRIs, blending two DVRIs, and erasing unwanted features in DVRIs, may be very useful in practice
- There are two straightforward solutions for fusing:
  - Traditional 2D image editing operations
  - Linear combination of several TFs
- They both fail to achieve goals in most cases

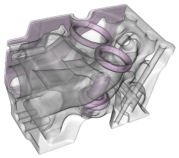
香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Traditional 2D image editing operations

- DVRIs are different from traditional 2D images
  - DVRIs are used to reveal information contained in 3D volume data so multi-layer transparent surfaces are usually displayed
  - Traditional images usually show objects in a real world setting, thus opaque surfaces are often presented
- Alpha Blending is not suitable for DVRIs
  - Lose depth cues and introduce misleading information



(a) Traditional Image created by blending two images

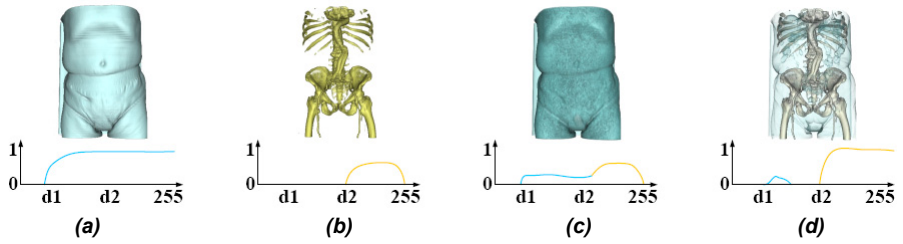


(b) DVRI

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Linear Combination of TFs

- Linear Combination of TFs
  - Nonlinear operations of the integration used in DVR makes it inappropriate for the general fusing problems



(a)-(b) Source images 1 and 2, and their TFs: TF1 and TF2; (c) DVRI rendered with a linearly combined TF: TF3 = α*TF1+ ß *TF2, where a = 0.3 and ß = 1; (d) DVRI rendered with our method by fusing (a) and (b)

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

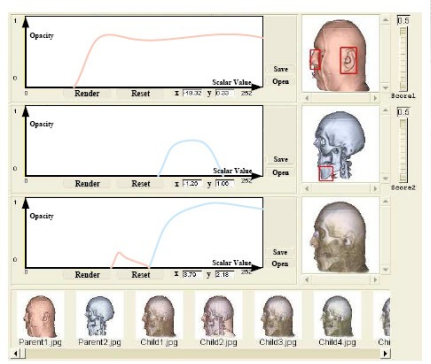
A Robust DVRI Editing Framework

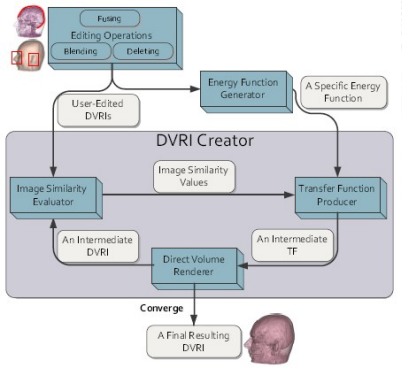
- We introduce a general and robust editing framework to solve the general DVRI editing problem
  - Allows users to directly manipulates features in DVRIs
  - Integrates user knowledge into the optimization process

- The uses of the framework are two-fold:
  - As an image editing tool
    - For users without expertise in TF, our system is a Photoshop-style editing tool for DVRIs while TFs are only used internally and will not be exposed to users.
  - As an interactive TF design method
    - For expert users, our system can show the generated TF, which can be further edited or manually fine-tuned by users. The system allows users to interactively and intuitively design TFs from simple to complex by gradually editing simple DVRIs into comprehensive ones

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

System Overview





(a) ***(b)***

(a) User interface which consists of source DVRIs and their TFs, target DVRI and its TF, and a history region; (b) System architecture which consists of energy function generator, transfer function producer, direct volume renderer, and image similarity evaluator

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Feature Selection

- Our approach allows users to select desired features in DVRIs to edit using rectangles or semi-automatic feature selection tools

Rectangles

Lazy Snapping

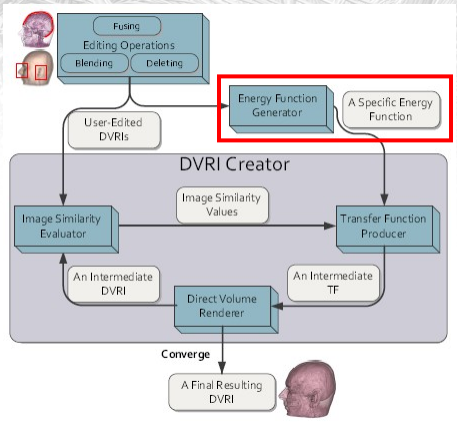
香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Editing Operations

- Fusing Operation
  - Combines multiple user selected features which appear in different DVRIs into a comprehensive one
- Blending Operation
  - Composites two DVRIs and generate a similar resulting image from alpha blending.
- Deleting Operation
  - Removes extra features from a DVRI

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Energy Function Generator



香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

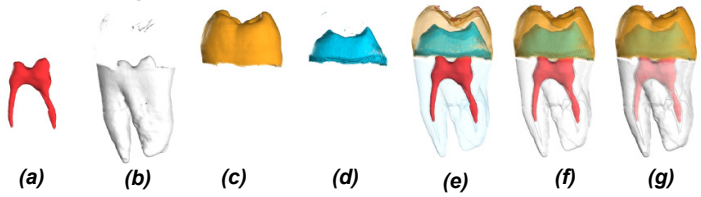
Energy Function

- An energy function for evaluating the fitness of candidate solutions should be formed after users specify editing operations in DVRIs
- The energy function is based on image similarity and editing operations to objectively evaluate the fitness of intermediate TFs
- It returns the measurement to the Genetic Algorithm (GA) module
  - The values are used to determine which genomes in the current population are more likely to be selected to survive

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Fusing Operation

- The energy function: $F_1 = \sum_{k=1}^{n} V_k * |V_k - S_k|$
  - n : number of source images to be fused
  - V_k: the vote (or user expected similarity value) given by users for the features in source image k
  - S_k: the computed image similarity value between the candidate image and source image k



(a) (b) (c) (d) (e) (f) (g)

The fusing operation: (a)-(d) Source DVRIs; (e) Target DVRI generated with V1 = 0.7, V2 = 0.3, V3 = 0.4, and V4 = 0.6; (f) Target DVRI generated with V1 = 0.5, V2 = 0.5, V3 = 0.5, and V4 = 0.6; (g) Target DVRI generated with V1 = 0.3, V2 = 0.7, V3 = 0.6, and V4 = 0.4

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Blending Operation

- The energy function: $F_2 = \alpha_1 * |\alpha_1 - S_1| + \alpha_2 * |\alpha_2 - S_2|$
  - α_1 and α_2: the alpha values used for blending
  - S_1 and S_2: the computed image similarity value between the candidate image and the source images to be blended

(a) (b) (c)

The blending operation: (a)-(b) Source images with bone and skin respectively; (c) Target image generated by blending (a) and (b) using our system

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Deleting Operation

- The energy function: $F_3 = S_1 + (1 - S_2)$
  - S_1 is the computed image similarity value between the candidate image and the source image within region A where a selected feature is to be removed, and S_2 is defined the same as S_1 but outside A

(a) (b) (c)

The deleting operation: (a) A DVRI where the skin is to be removed; (b) Mask created from (a) using lazy snapping; (c) Resulting DVRI after executing the deleting operation

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Mix Multiple Editing Operations

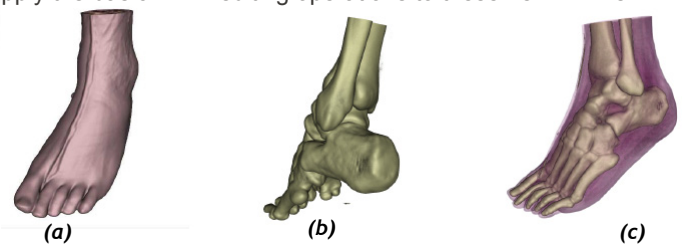
- Our system enables users to mix different basic operations together
  - Users can fuse multiple features in distinct DVRIs together and can meanwhile remove certain features from some DVRIs
- The energy function:

$$F = F_1 + F_2 + F_3$$

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Editing Features from Different Viewpoints

- Given *n* viewpoints and their corresponding DVRIs
  - Select a common good viewpoint
  - Re-render the user selected features from this viewpoint
  - Apply the basic DVRI editing operations to these new DVRIs

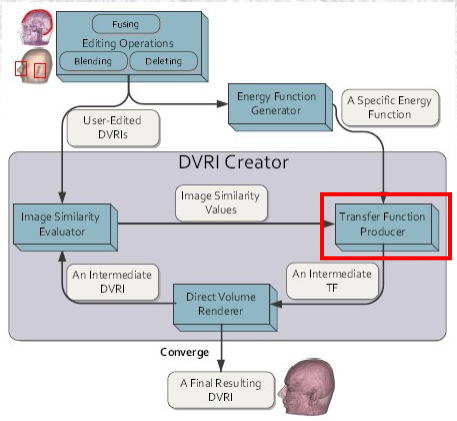


(a) (b) (c)

The editing operation on DVRIs generated from different viewpoints: (a)-(b) Source images generated from different viewpoints; (c) Resulting image generated by blending the (a) and (b)

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Transfer Function Producer



香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Genetic Algorithm

- A *Genetic Algorithm* (GA) is a search algorithm imitating the process of natural evolution
- It is particularly useful for searching solutions to optimization problems, especially when the search space is huge and unknown

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

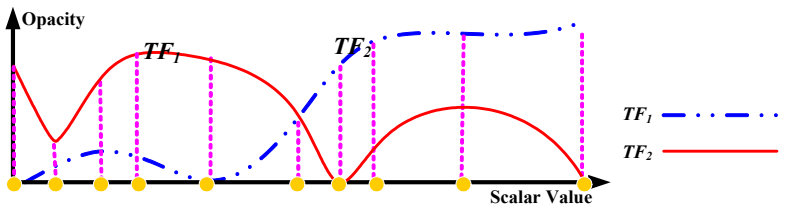
Solution Encoder/Decoder

- The solution encoder/decoder specifies the genome representation by analyzing the source TFs
  - 1D array of floating numbers
- The process:
  - Smoothens the source TFs
  - Samples TFs adaptively above the Nyquist frequency

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

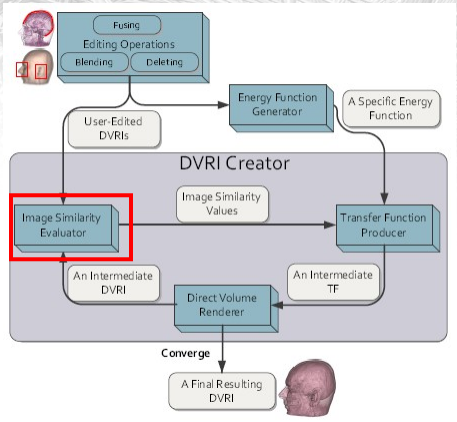
Genome Representation

- The samples are then used to specify the genome representation
  - They can be used to restrict the search space to improve the GA performance
  - The yellow points below are the genome representation



香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Image Similarity Evaluator



香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Image Similarity Metric

- A contour-based similarity metric is developed to compare two DVRIs
- The preprocessing:
  - Converts DVRIs into grey-scale images
  - Detects the edge images from the grey-scale images with *Canny edge detector*
  - Smoothes the edge images with a Gaussian filter

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

How to Compute Image Similarity

- The image similarity value S_k is:

$$S_k = \frac{\sum_{y=1}^{height} \sum_{x=1}^{width} Q(x,y)}{N_{source}}$$

$$Q(x,y) = \begin{cases} 1 & \text{if } P(x,y) < threshold \\ 0 & Otherwise \end{cases}$$

$$P(x,y) = |K_{(x,y)} - K'_{(x,y)}|$$

  - N_{source} : the number of all pixels on the edges of the source k's edge image
  - K and K' :are the Gaussian filtered target and source edge images with resolution (*width*, *height*)

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Supplement to Image Similarity

- Notice that we consider only the pixels on the source edge image k for S_k
  - S_k is only computed if $K'_{(x,y)} != 0$

$$S_k = \frac{\sum_{y=1}^{height} \sum_{x=1}^{width} Q(x,y)}{N_{source}}$$

$$Q(x,y) = \begin{cases} 1 & \text{if } P(x,y) < threshold \\ 0 & Otherwise \end{cases}$$

$$P(x,y) = |K_{(x,y)} - K'_{(x,y)}|$$

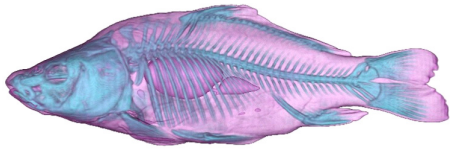
- If there are user-selected features in the DVRIs which are to be compared, our system considers only the pixels within these features
  - N_{source} becomes the number of pixels within the features on the edges of the source edge image

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Results for The Basic Editing Operations (Fusing)

- Create a new DVRI (d) by fusing the features in multiple DVRIs (a), (b), and (c) into a comprehensive one (d) with V_1=0.3, V_2=0.4, and V_3=0.3

(a) *(b)* *(c)*

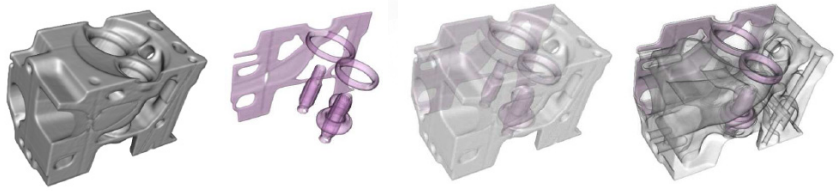


(d)

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Results for The Basic Editing Operations (Blending)

- Generate a new DVRI (c) by blending DVRIs (a) and (b)
  - (c) Obtained by traditional alpha-bending
  - (d) Created by our approach (better details)

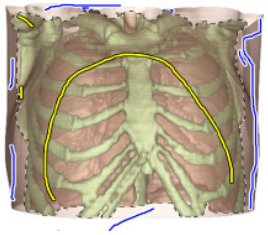


(a) *(b)* *(c)* *(c)*

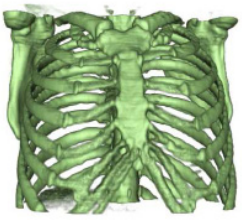
香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Results for The Basic Editing Operations (Deleting)

- Generate a new DVRI (b) by deleting feature (a) indicated by blue strokes while retaining the feature selected by yellow strokes



(a)

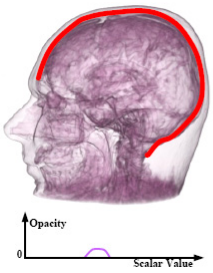


(b)

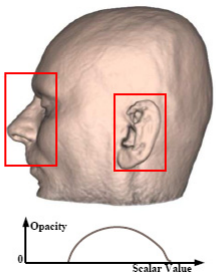
香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Examples for TF Design (1/2)

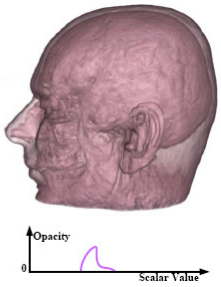
- Creating a DVRI of better quality (*i.e.*, clearer contours) by fusing the selected features in (a) and (b)



(a)



(b)

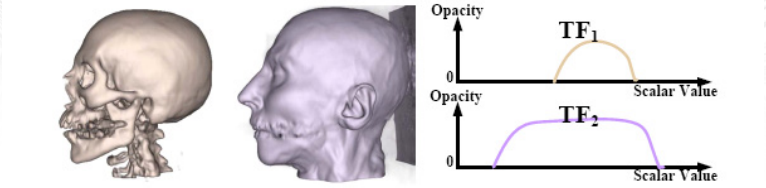


(c)

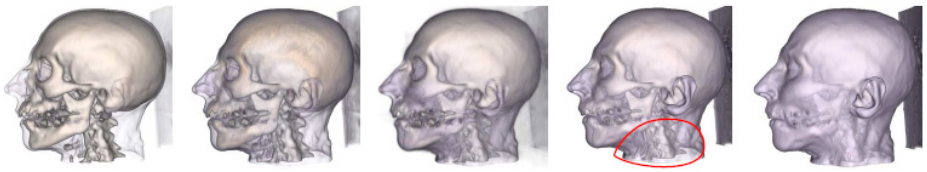
香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Examples for Generating Animations

- Generating intermediate frames for animations



(a) keyframe1 ***(b) keyframe2***

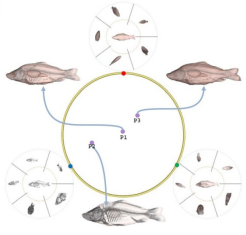


Intermediate frames created by the blending operation

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Palette-Style Interface and Radial Graph Interface

- A palette style intuitive interface is further developed to server as the front-end of the editing framework
- A radial graph interface arranges the resulting images based on viewpoints and image quality for detailed exploration



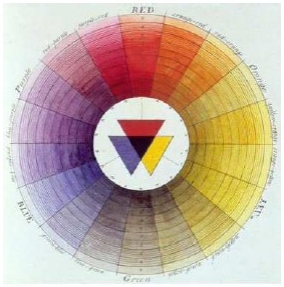
Palette-Style Volume Exploration Interface

Radial Graph

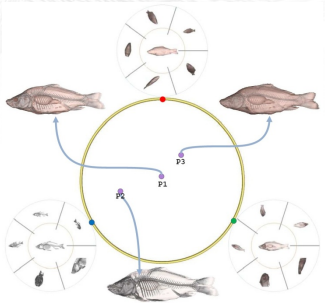
香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Palette-Style Interface

- A palette-style interface motivated by the color palette is proposed for intuitive DVRI generation to increase exploration intuitiveness, reduce exploration redundancy, and save and share exploration results



Moses Harris: the first color wheel to classify red, blue and yellow as the three primary colors



Palette-Style Volume Exploration Interface

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

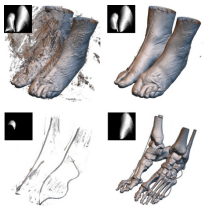
Visualization Process with The Interface

1. Primary opacity TFs and DVRIs (analogy of the primary color in the color palette) are created automatically or semi-automatically
2. More DVRIs of different opacity TFs can be created by fusing the primary DVRIs in the DVRI wheel
3. The system enables users to further explore the data using the created opacity TF with different parameters in a separate radial graph

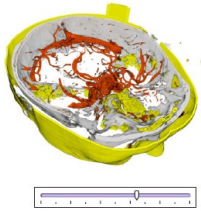
香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Primary DVRI Generation

- Primary DVRIs (analogy of primary colors in the color palettes) can be created by experts manually or semi-automatically which makes boundaries of structures in volumetric data visible
- For non-expert users, other high-level methods can be adopted for primary DVRI generation without the knowledge of TFs



Kindlmann and Durkin 1998

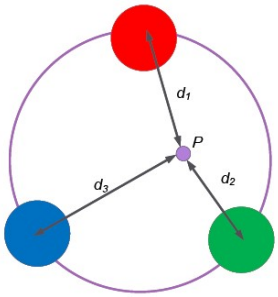


Salama et al. 2006

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Editing Operations on The DVRI Wheel

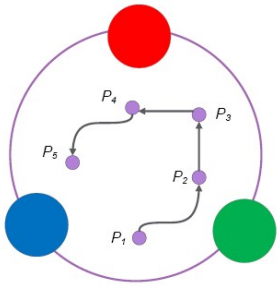
- After creating primary DVRIs, our system allows users to generate more DVRIs from the primary ones intuitively
- Users just need to select a point on the DVRI wheel and indicate how to fuse the DVRIs using the DVRI editing framework



香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Animation Generation

- Animation can be used to reveal 3D relationships between the different structures more effectively than a still image
- Users can create an animation for volume visualization using the DVRI wheel intuitively based on the fusing operation

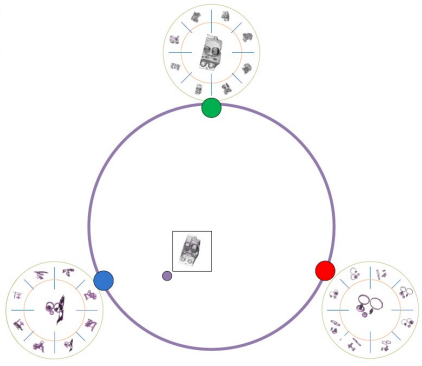


Animation operation

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Support for Multiple Viewpoints

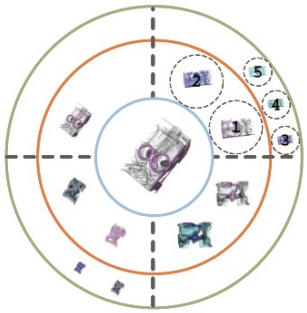
- Primary DVRIs can be generated from different viewpoints
  - However, image similarity can be computed only for DVRIs rendered from the same viewpoint



香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Radial Graph for Detailed Exploration

- Users may need to further explore the volume using the opacity TF with different lighting parameters, color TFs, and viewpoints
- A new radial graph style interface was proposed to arrange the resulting images based on viewpoints and image quality



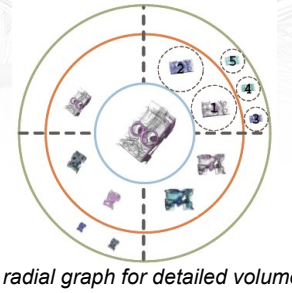
香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Graph Layout

- The graph layout consists of multiple concentric circles, and the radii for these concentric circles are $r_1 \ldots r_n$ from inside to outside and defined as

$$r_i = r_{i-1} + \frac{(r_{i-1} - r_{i-2})}{2}$$

  - where i >=3 and $r_2 = 2r_1$, and $r_1 = C$, *and* C is determined by users



A radial graph for detailed volume exploration with the same opacity TF

- The graph is further divided into multiple sectors for storing the DVRIs created from different viewpoints

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Image Quality Evaluation

- In each sector, the DVRIs are sorted in terms of image quality so that better DVRIs have larger sizes and are closer to the center
- The features and details in an image with a higher contrast are always better perceived by viewers
  - Contrast can be interpreted as the standard deviation of the pixel values in the image:

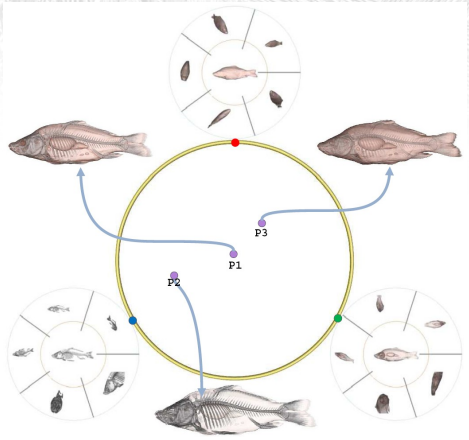
$$\sigma = \sqrt{\frac{1}{|\Omega|}\sum_{\Omega}(v(i)-\mu)^2}$$

where Ω is the image and $v(i)$ is the intensity of pixel i, and μ is the mean value of all pixels in the image

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Volume Exploration (1)

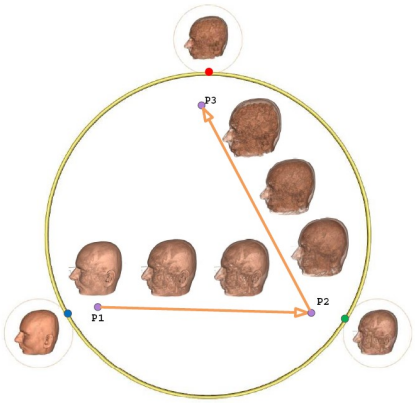
- Intuitive Volume Exploration



香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Volume Exploration (2)

- A palette-style DVRI wheel for creating animation
  - An animation was generated along the user selected path P1→P2→P3



香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Image Quality Evaluation

- A radial graph for detailed volume exploration
  - The radial graph was divided into multiple sectors for different viewpoints, and the DVRIs in each sector were sorted according to the image quality such that DVRIs with higher image quality lie closer to the center

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Outline

- Introduction
- Transfer Function Design Based on Editing Direct Volume Rendering Images
- Quality Enhancement of Direct Volume Rendered Images
- Quantitative Effectiveness Metrics for Direct Volume Rendering

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Introduction (1/3)

- Direct volume rendering for scientific visualization
  - Revealing different structures by specifying proper transfer functions
  - Allowing visual analysis on the volumetric data
- Quality of the DVRIs is an important issue
  - Ensuring features are clearly shown (enhanced features)
  - Preserving the information in the volume
  - Delivering a pleasing result (better contrast)

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Introduction (2/3)

- Quality enhancement in image domain
  - Commonly used image processing approaches, *e.g.,* contrast and feature enhancement
  - Formation of a new image with certain mapping of pixel values
  - Objective: easy interpretation of image information
  - Drawbacks: information may be missing in the image and cannot be restored

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Introduction (3/3)

- Our approach: working in transfer function and volume domains
  - Image measurement: evaluating the effectiveness of the image in conveying the volume information
  - Parameter refinement: adjusting the rendering parameters for better image quality
- Restoring the missing information due to poor lighting and rendering settings
- Revealing the information in the volume by analyzing the composition of the rays in the rendering process

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Limitations of Image-Based Approaches

- Limitations of image-based approaches
  - Cannot recover the missing details due to poor lighting
  - Cannot enhance the structure with respect to the topology and shape in the volumetric data
- Our method:
  - Taking the volume into consideration to preserve the details

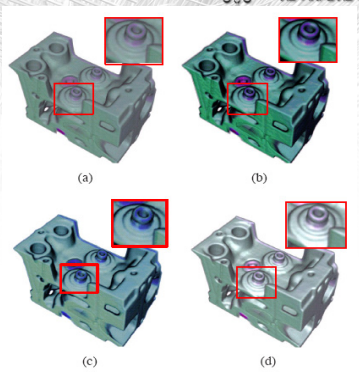
香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Typical Problems in DVRIs (1/2)

- Structures are not clearly shown due to poor lighting and rendering parameter settings
- Pixels in the DVRIs cannot give any implication on the existence of structure
  - Homogenous regions in DVRI may represent some fine features
  - They should demonstrate certain variations in the image to convey the information of the structures

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Typical Problems in DVRIs (2/2)



An example using the CT engine dataset: (a) shows the original image with a poor contrast; (b) and (c) are the images enhanced by Photoshop and manual adjustment using various image filters; (d) is the result generated by adjusting the rendering parameters.

- Idea: with the help of the volumetric data and the knowledge of the rendering process of DVRIs, we can
  - Further improve the image quality accordingly
  - Reinforce the hidden details about the volume in the image

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Image Quality Measurement

- The quality of DVRIs is defined as the effectiveness of the rendered images in presenting the information in the volumetric data
  - Determine whether the image can show a significant variation in regions where the rays carry different information
- To quantitatively analyze a DVRI, we establish several measurements for both image and volume data information

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Image Measure

- Variations / information in an image are interpreted as contrast
- Homogeneity measurement [Cheng et al. 03']
- Image standard deviation σ as $\sigma(x) = \sqrt{\frac{1}{|\omega|}\sum_{\omega}(v(i) - \mu_x)^2}$
- Image entropy h as $h(x) = -\frac{1}{log|\omega|}\sum_i p_i log p_i$
- Estimate the visual information in the image
- Final image measure: $M_I(x) = \sigma(x) \times h(x)$

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Ray Measure

- Each sample point contributes to the final image in different degrees and their contribution can be estimated by

$$\alpha(1 - \alpha_{accum})$$

- We estimate the information carried by the rays and their variations by considering those visible sample points along the rays
- Ray measure can be represented as

$$M_R(R) = -\sum_i p_i \sum_j p_i(j) log p_i(j)$$

  - Entropy term on the composition of the rays
  - Signifying the information variation among rays

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Composite Measure

- Compositing measure on the quality of an image

$$M_C = (1 + exp(-\frac{-M_I + M_R}{s}))^{-1}$$

- Indicating the deviation between the image and ray information at each pixel in the image
- Minimizing the overall information deviation - preserving the information of the volume in the image domain.

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Parameter Refinement

- Adjusting different rendering parameters for better results
  - Manual adjustment
    - Tedious and non-trivial task
  - Optimization of the parameters using a genetic algorithm
    - Image quality measure as the fitness measure
    - Parameters involved:
      - Reflection / illumination model
        - » Ambient, diffuse and specular coefficients
      - Transfer function
        - » HSV (brightness and saturation)
        - » YIQ (luma information)
        - » Only “safe” channels are modified to preserve the original color

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Genetic Algorithm

- Combinatorial optimization of parameters
- Efficient search of an optimal solution in the parameter space through the evolution process
- Process driven by the image quality measure to obtain a better result
- Advantages:
  - stochastic search - avoiding local optima
  - efficient

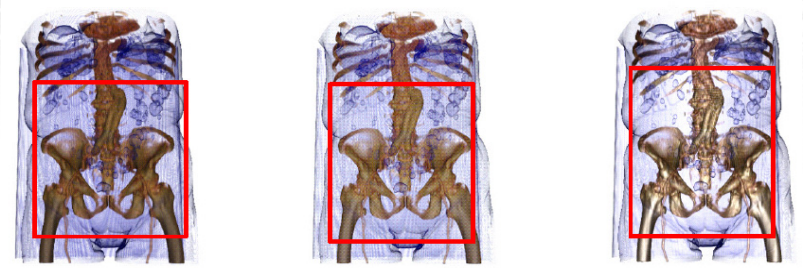
香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Adaptive Enhancement and User Interaction

- Adaptive enhancement
  - Preserve the under-enhanced details missing in the global configuration
  - Enhance and refine on certain parts of the DVRI
- User interaction
  - Highlight regions in the image
  - Select Structure in volume / intensity domain
  - Process on the selected regions and the corresponding rays

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

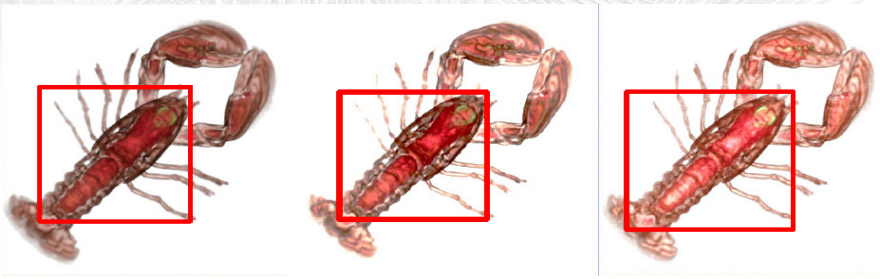
Experimental Results



(a) (b) (c)

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Experimental Results



(d) (e) (f)

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Outline

- Introduction
- Transfer Function Design Based on Editing Direct Volume Rendering Images
- Quality Enhancement of Direct Volume Rendered Images
- Quantitative Effectiveness Metrics for Direct Volume Rendering

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Visualization Tasks and Features of Interest

- The effectiveness of a visualization highly depends on the tasks that can be classified into two categories
  - Routine tasks
    - Users usually have prior knowledge of the features that they intend to visualize
  - Exploration tasks
    - Features of interest are unknown
      - Use other visualization techniques to explore the volume and gain some knowledge
      - Use our system in a divide-and-conquer manner
      - Estimate the useful information in the data automatically
- In summary, features of interest (or desired features) can be either specified by users or predicted by the system

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Effectiveness Metrics (1/3)

- It is very difficult to design an ideal visualization system which can automatically reveal desired features to users
  - Huge parameter space (e.g., transfer function, viewpoint, lighting)
- Merely displaying these desired features one by one is not enough
  - Users want to know context and spatial relations between the features
- Even if only one feature needs to be displayed, self occlusion may become an issue.
- Artifacts and illusions may be introduced because of inappropriate lighting or view angles

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Effectiveness Metrics (2/3)

- User interactions such as changing viewpoints and specifying transfer functions are needed
  - Routine tasks and exploration tasks
- Even if we know what users want, we may not have a perfect solution to present the information to users automatically so user interactions are still needed
- The interactions by non-expert users are often error prone and may introduce misleading information leading to unreliable conclusions
- A scheme is needed to let users know whether their fine tuned transfer functions or view angles are effective or not

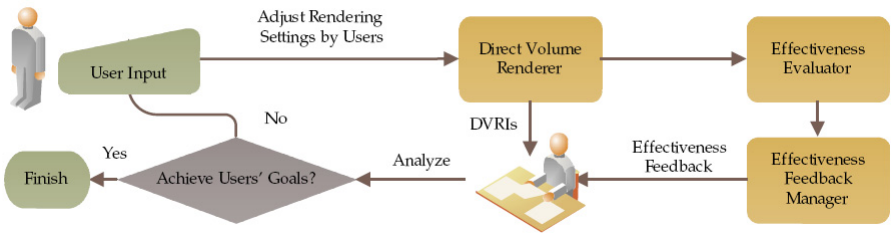
香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Effectiveness Metrics (3/3)

- There are two sets of effectiveness metrics
  - Adequate effectiveness metrics
    - If these metrics are satisfied then the visualizations tasks can be achieved
    - They are the holy grail of volume visualization and may not be possible for many applications in the near future
  - Necessary effectiveness metrics
    - If these metrics are not satisfied then visualization tasks cannot be achieved. However, even if these necessary metrics are satisfied, there is still no guarantee that the visualization tasks can be achieved
    - They are more practical and what we want to deal with

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

New Visualization Pipeline



香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

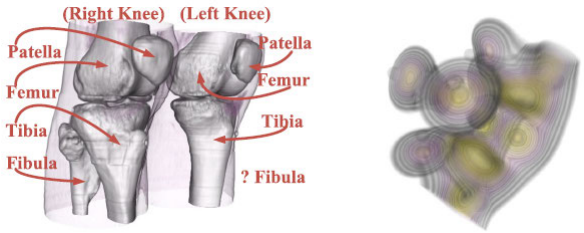
Effectiveness Evaluator

- The effectiveness evaluator is used to quantitatively and objectively assess the effectiveness of a DVRI or a whole visualization process based on three effectiveness metrics
  - Visibility metric
  - Distinguishability metric
  - Contour Clarity metric

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Visibility Metric (1/2)

- The visibility metric measures the visibility of an important feature in a volume by counting the visible voxels of the feature



Two common cases where the visibility metric is needed: (a) DVRI of the CT Knee where the fibula of the left knee is invisible; (b) DVRI of the simulated Neghip having large variance of the intensity values

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Visibility Metric (2/2)

- The visibility of voxels can be estimated in the process of full image-order volume rendering
  - Estimate visibility values for the sampling points along each ray

Visibility value V_i for sampling point i

$$C = (1-\alpha) * \alpha_i * C_i + C \quad (1)$$

$$\alpha = \alpha_i * (1-\alpha) + \alpha \quad (2)$$

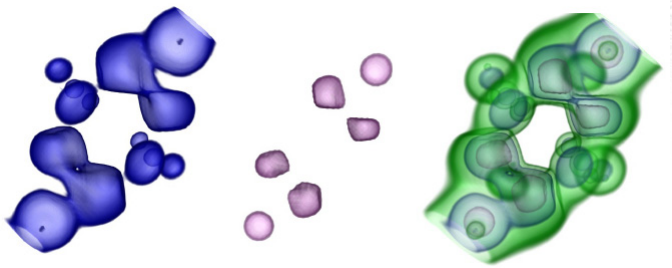
  - V_i would be distributed to the neighboring voxels based on the corresponding weights used in the interpolation

- The visibility value of a DVRI can then be estimate as $E = n_i / n$
  - n_i and n are the number of visible voxels and the number of all voxels inside the important feature, respectively

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Distinguishability Metric (1/2)

- The distinguishability metric evaluates how well a feature can be visually differentiated from its surroundings



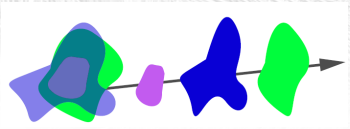
(a) *(b)* *(c)*

(a) Structure A in blue; (b) Structure B in purple; (c) Blending A and B as well as a green structure C into a new DVRI where A and B are indistinguishable

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Distinguishability Metric (2/2)

- The distinguishability metric aims at detecting the ambiguity caused by the blending effect



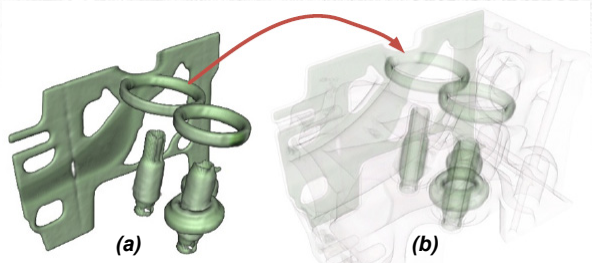
1. Segment a given DVRI into a number of fragments
2. Classify the fragments into two classes - fragments with the desired feature and fragments without the desired feature
3. If the fragment with the desired feature has color similar to any fragments without the feature, the metric will record the feature's voxels that contribute to the fragment as indistinguishable voxels

- The distinguishability value of a DVRI can then be estimated as the ratio of the distinguishable voxels to indistinguishable voxels

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Contour Clarity Metric (1/2)

- The contour clarity metric measures how clear the contours of a desired feature are presented in a DVRI



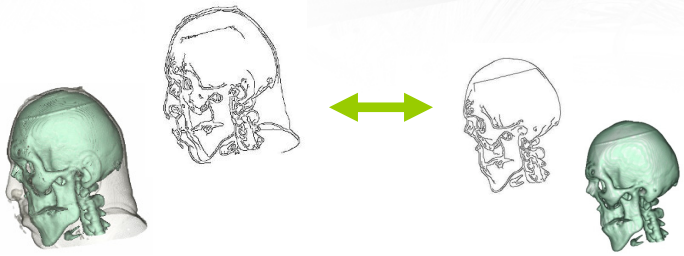
(a) ***(b)***

A common scenario in volume rendering where the contour clarity metric is needed: the contours of an important feature as shown in (a) are fuzzy and unclear in (b) although the feature is visible and distinguishable

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Contour Clarity Metric (2/2)

- The metric measures the contour clarity by estimating the similarity between the DVRI and the iso-surface of the salient feature
  1. Derive the edge images of both DVRI and the iso-surface
  2. Estimate the similarity between the two edge images



A given DVRI ***Salient iso-surface***

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

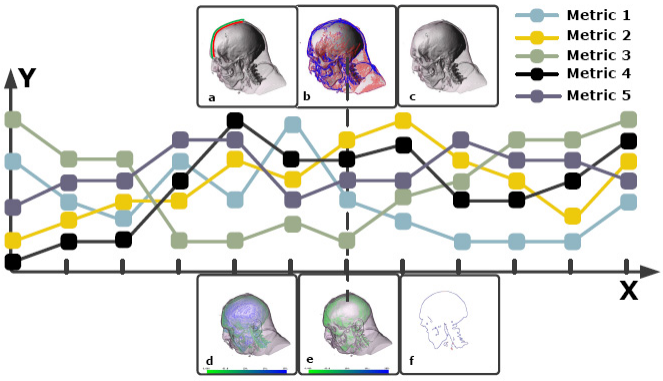
Effectiveness for A Whole Visualization Process

- For the visibility and distinguishability metrics, we accumulate each voxel's effectiveness values at each DVRI in the visualization process
  - The overall effectiveness values can then be measured as the percentage of the visible and distinguishable voxels after the accumulation to all voxels, respectively.
- For each of other metrics,
  - Collect all explored viewpoints in the visualization process
    - Each viewpoint stores a highest effectiveness value for the metric and treat it as the metric value at this viewpoint for the whole process
  - Select the lowest effectiveness value among all explored viewpoints as the metric's effectiveness value of the process

香港科技大學 THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Effectiveness Feedback Manager

- It organizes and presents the effectiveness values to the end-users in an intuitive manner



香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Conclusions

- Perception-based transfer function design and evaluation
- Transfer function design based on editing DVRIs
- Quality enhancement of DVRIs
- Effectiveness assessment of DVRIs

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

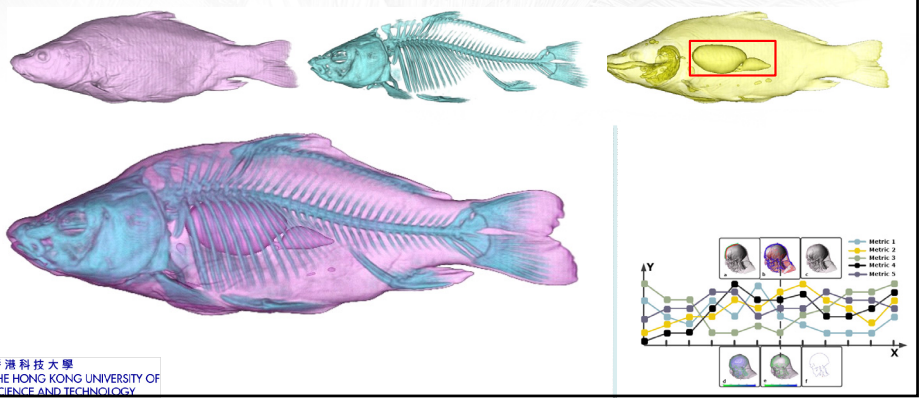
References

- Yingcai Wu, Huamin Qu. “Interactive Transfer Function Design Based on Editing Direct Volume Rendered Images", IEEE Transactions on Visualization and Computer Graphics (TVCG), Vol. 13, No.5, pp. 1027--1040, 2007
- Ming-Yuen Chan, Yingcai Wu, and Huamin Qu. "Quality Enhancement of Direct Volume Rendered Images", 6th IEEE/EG International Symposium on Volume Graphics (VG'07), pp. 25 - 32, 2007. (Cover Image)
- Yingcai Wu, Anbang Xu, Ming-Yuen Chan, Huamin Qu, and Ping Guo. "Palette-Style Volume Visualization", to appear in 6th IEEE/EG International Symposium on Volume Graphics (VG'07), pp. 33 - 40, 2007.
- Yingcai Wu, Huamin Qu, Ka-Kei Chung, Wai-Ho Mak, Anbang Xu. "Quantitative Effectiveness Metrics for Direct Volume Rendering". Poster, in the poster session of IEEE Visualization 2007 (Best Poster Candidate).

香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Q & A

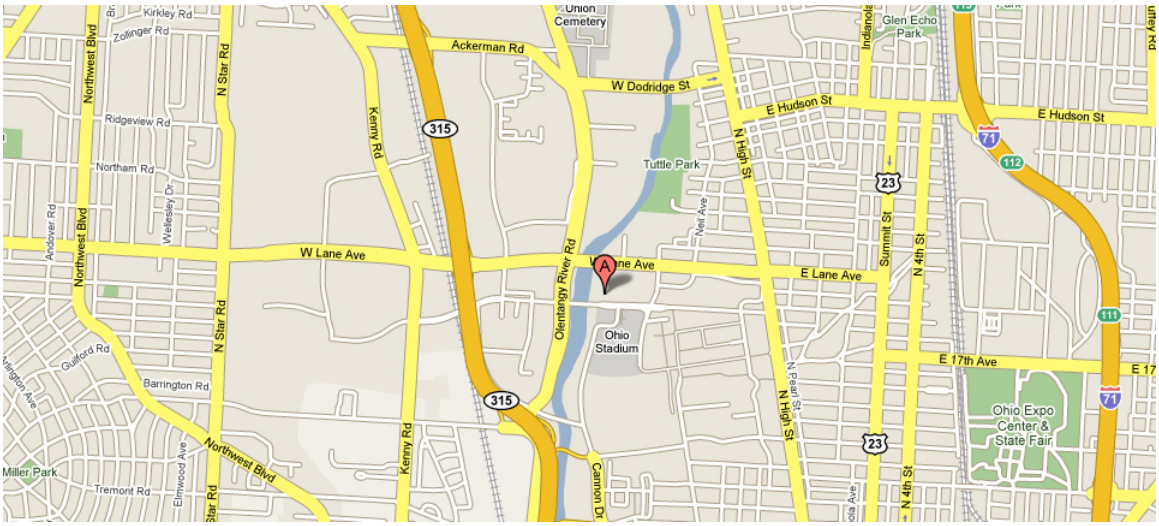
- Thank you for your attention!



香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Han-Wei Shen

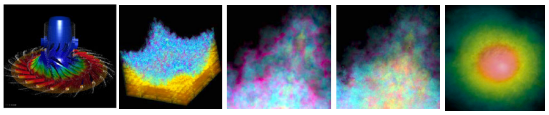
The Ohio State University



Information and Visualization

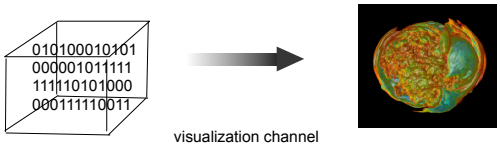
Han-Wei Shen

Department of Computer Science and Engineering
The Ohio State University



The Goal of Visualization

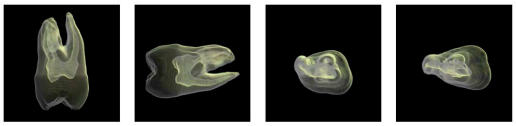
- The goal of visualization is to faithfully convey the maximal amount of information from the data through the display channel



How do we measure the information content?

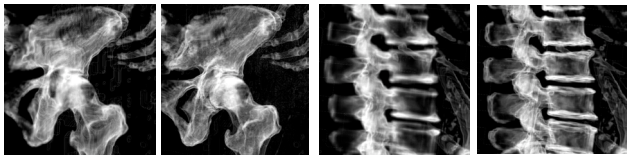
View Point Selection

- Due to 3D occlusion, images generated from different view will convey different amount of information
- How to choose views that can convey maximal amounts of information?



Multiresolution Visualization

- How do we measure and compare the quality of different LOD selections?
- Are the computation resources effectively utilized?



Low resolution High resolution Low resolution High resolution

Information Theory

- Study the fundamental limits to reliably transmit messages through a noisy channel
- Model the message as a random variable whose value is taken from a sequence of symbols
- Information content of the message is measured by Shannon's Information Entropy

Shannon Entropy

- The random variable takes a sequence of symbol $\{a_1, a_2, a_3, \ldots, a_n\}$ with probabilities $\{p_1, p_2, p_3, \ldots, p_n\}$
- The information contained in each symbol a_i is defined as:

$$\log(1/p_i) = -\log p_i$$

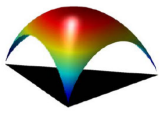
- The average amount of information expressed by the random variable is the entropy:

$$H(x) = -\sum_{i=1}^{n} p_i \log p_i$$

Properties of Entropy

- Entropy is to measure the average uncertainty of the random variable
- Entropy is a concave function, which has a maximum value when all outcomes are equally possible:

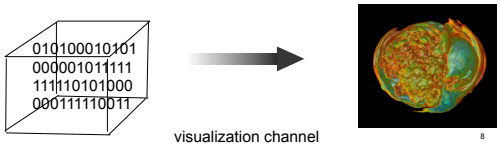
$$p_1 = p_2 = p_3 = ... = p_n$$



An example of three dimensional Probability vector {p1,p2,p3}

Information Theory and Visualization

- A data set can be considered as a random variable
- Each data point can be considered as an outcome for a random variable X
- We can measure the information content of the visualization output (image)

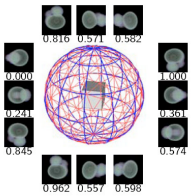


Finding a Good View

- Viewpoint entropy for a camera view:

$$H(x) = -\sum_{i=1}^{n} p_i \log p_i$$

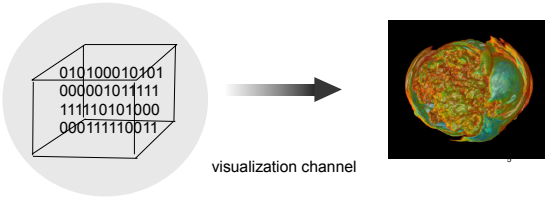
- Sample the view sphere and evaluate from multiple sample views



(Image courtesy: Takahashi et al)

Object Space Approach

- Consider that the visualization is generated from 3D data
- Evaluate the information content from each data point (voxel) to the screen



Information Content of a Voxel

- Each voxel's information content depends on its visibility and importance
  - visibility v: transparency from the camera to the voxel
  - importance w: the voxel's alpha value (defined by the transfer function)
- Visual probability for a voxel i:

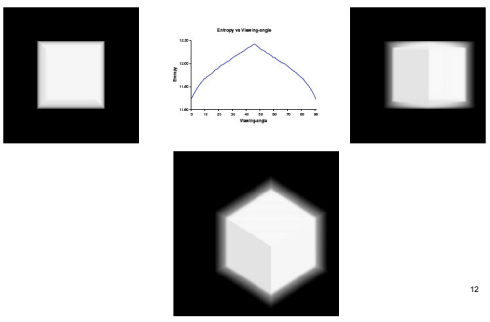
$$p_i = \frac{1}{\sigma} . \frac{v_i}{w_i} \quad , \text{ where } \quad \sigma = \sum_{i=0}^{n-1} \frac{v_i}{w_i}$$

- Information Content for the voxel:

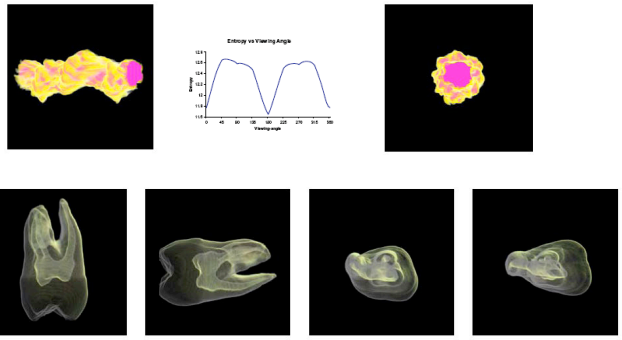
$$\log (1/p_i) = -\log p_i$$

Example

$$H(x) = -\sum_{i=1}^{n} p_i \log p_i$$



Examples



View Partitioning

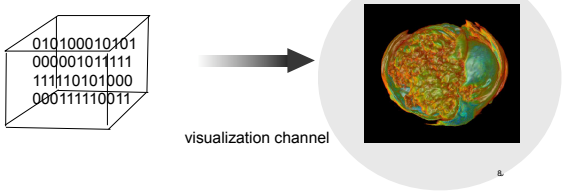
- Choose only one from the nearby similar view samples
- Use Jensen-Shannon (JS) divergence measure to estimate the distance between two view entropies q1 and q2

$$JS(q1,q2) = 2H(\frac{1}{2}q1 + \frac{1}{2}q2) - H(q1) - H(q2)$$

- Cluster the view samples based on the JS divergence

Image Space Approach

- Consider only the visualization output, i.e., the images
- Evaluate the information content based on the pixel value distribution in the image



View Selection Criteria

- Larger projection size
  - more voxels to be visible
- Even opacity distribution
  - opaque voxels do not clump together
- Even color distribution
  - data in different ranges can be seen
- More salient geometric features (curvature)

Projection & Opacity Distribution

- For an imagine contains n pixels with accumulated opacities $\{\alpha_0, \alpha_1, \alpha_2, ..., \alpha_{n-1}\}$, the probability of pixel i is defined as:

$$p_i = \frac{\alpha_i}{\sum_{i=0}^{n-1} \alpha_j}$$

- Background pixels are excluded
- The view entropy is defined in the same way:

$$H(x) = -\sum_{i=1}^{n} p_i \log p_i$$

Color Distribution

- A well designed transfer function should highlight salient features with attentive colors
- A good view should maximize the area of the salient colors while maintaining an even distribution
- Assuming there are n colors in the transfer function $\{C_0, C_1, ..., C_{n-1}\}$, with C_0 being the background
- For each pixel, we measure the perceptual color distance (using CIELUV) to the salient colors and classify the pixel

Entropy Calculation

- Calculate the area A_i for each color C_i
- The probability for C_i is defined as:

$$p_i = \frac{A_i}{T} \quad \text{where} \quad T = \sum_{i=0}^{n-1}(A_i)$$

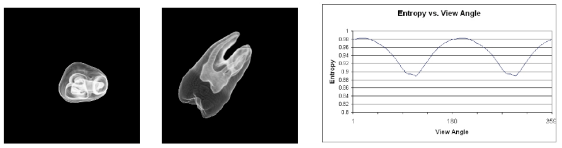
- The entropy for the image is calculated similarly:

$$H(x) = -\sum_{i=1}^{n} p_i \log p_i$$

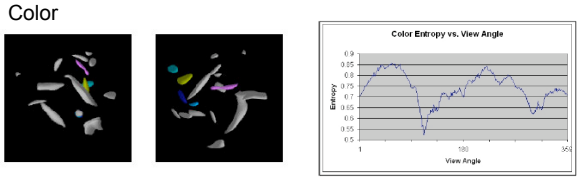
- The entropy of an image is a maximum when all salient colors are shown

Examples

Opacity and Projection Area

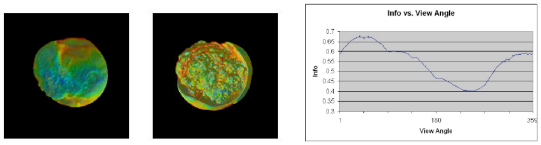


Color



Curvature Distribution

- Low curvatures imply flat areas and high curvatures mean highly irregular shapes.
- We can represent curvatures in a volume by color coding each voxel based on its curvature
- High luminance colors for high curvatures



The Final Utility Function

- Decide the best view based on opacity, color, and curvature

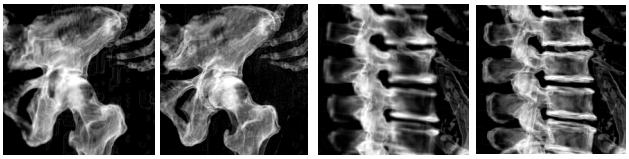
$$u(v) = \alpha.opacity(v) + \beta.color(v) + \gamma.curvature(v)$$

$$\alpha + \beta + \gamma = 1$$

- Normalize each component to [0,1]
- User can decide a different combination of weights and/or introduce new factors

Multiresolution Visualization

- How do we measure and compare the quality of different LOD selections?
- Are the computation resources effectively utilized?



Low resolution High resolution Low resolution High resolution

Global LOD Quality Metric

- Measure the amount of information contained in the selected LOD
  - Compare LODs
  - Decide whether the computation resources are distributed evenly to render-worthy blocks
  - LOD adjustment
- Approach: Information theory

LOD Entropy

$$H(X) = -\sum_{i=1}^{M} p_i \log p_i$$

- A LOD contains a sequence of blocks B_i at particular resolutions
- P_i, the 'probability' of a data block B_i at a particular resolution, is defined as:

$$P_i = \frac{C_i \times D_i}{S} \qquad S = \sum_{i=1}^{M} C_i \times D_i$$

- C_i and D_i are the block's contribution and distortion (if it is a low resolution block)

Contribution and Distortion

- Contribution: the block's color (μ), projection size (a), thickness (t), visibility (v)

$$C_i = \mu.t.a.v$$

- Distortion: the difference between the block's data values and those of a higher resolution block

$$d_{ij} = \sigma_{ij} \cdot \frac{\mu_i^2 + \mu_j^2 + C_1}{2\mu_i\mu_j + C_1} \cdot \frac{\sigma_i^2 + \sigma_j^2 + C_2}{2\sigma_i\sigma_j + C_2}$$

covariance luminance contrast

LOD Entropy

$$H(X) = -\sum_{i=1}^{M} p_i \log p_i$$

- Maximize the entropy function when P_i are all equal
- The entropy function prefers that the block's contribution matches its resolution

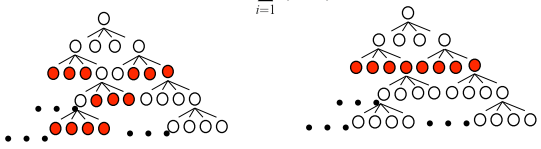
$$P_i = \frac{C_i \times D_i}{S}$$

Ci ⇨ Di : use high resolution

Ci ⇨ Di : use low resolution

LOD Comparisons using Entropy

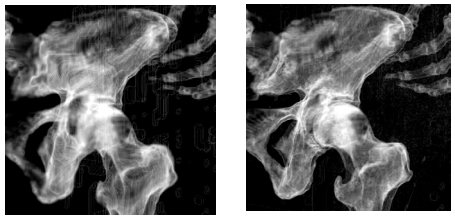
$$H(X) = -\sum_{i=1}^{M} p_i \log p_i$$



H1 H2

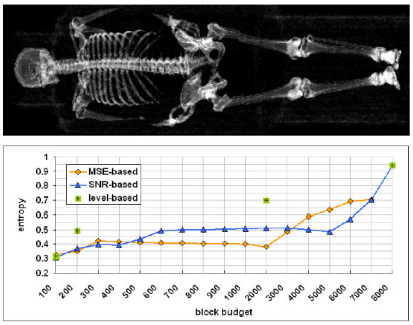
A higher entropy value indicates a balanced probability distribution, thus a better overall quality

Entropy vs. Quality



Entropy = 0.166 (34 blocks) Entropy = 0.316 (259 blocks)

Entropy vs. # of Blocks

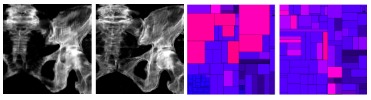


Visual Representation of LOD Quality

- An optimal selection of LOD is an NP complete problem
- Fine tuning of LOD selection is often necessary
- Can we visualize what are selected, and make adjustments if necessary?

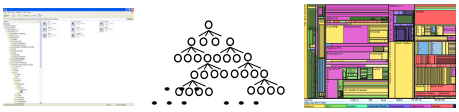
LOD Map

- A visual user interface to visualize the LOD selection
- Allow the user to see individual block's contribution vs. distortion, i.e., visualize the entropy terms



Treemap

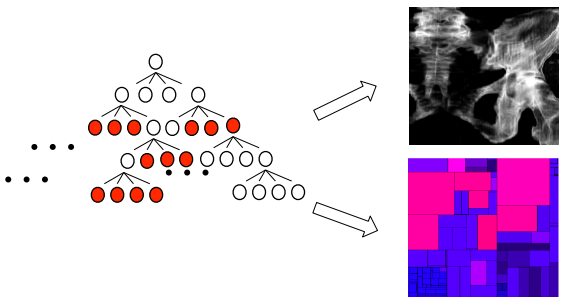
- A space-filling method to visualize hierarchical information [Shneiderman et al. 1992]
  - Recursive subdivision of a given display area
  - Information of each individual node
    - Color and size of its bounding rectangle



LOD Map

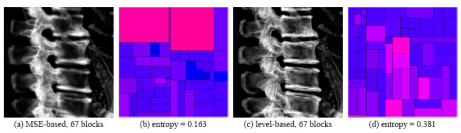
- Display the blocks belong to the selected LOD in a tree-map like manner
- Color (blue to red) is used to encode the block's distortion
- The contribution of the block (μ.t.a.v) is divided into two parts
  - The size of rectangle is to encode μ.t.a
  - The opacity of rectangle is to encode v

LOD Map



How Can LOD Map Help

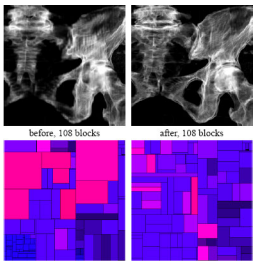
Comparisons of different LOD selection schemes



How Can LOD Map Help

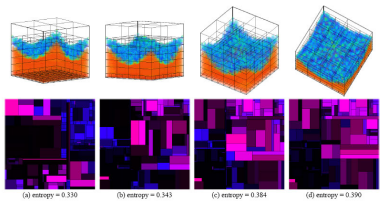
- Spot problematic regions in the current LOD
  - Large red rectangles – high contribution blocks rendered with low resolutions
    - Action: split the blocks and increase the resolutions
  - Small blue rectangles – low contribution blocks rendered with high resolutions
    - Action: join the blocks and reduce the resolutions
  - Dark rectangles – low visibility blocks
    - Action: join them and reduce the resolutions

LOD Adjustment



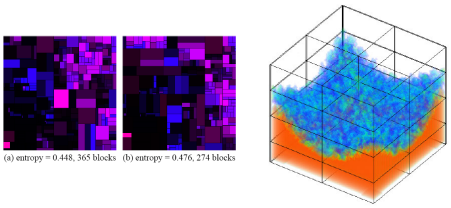
How Can LOD Map Help

- View selection on the fly - High entropy and brighter LOD map for better views



How Can LOD Map Help

• Budget Control - Render fewer blocks, i.e., lower Resolutions in certain regions, for the same entropy



Conclusions

- Entropy can be used to quantify the information content in a visualization
- Applications: view selection & LOD selection
- The goal of visualization is to reduce the uncertainty perceived by the viewer
- More applications of information theory are expected to quantify the goodness of visualization

Acknowledgements

- US DOE SciDAC Ultra Scale Visualization Institute
- NSF Career Award CCF-0346883
- DOE Early Career Principal Investigator Award DE-FG02-03ER25572
- NSF ITR grant ACI-0325934

View Selection for Volume Rendering

Udeepta D. Bordoloi* Han-Wei Shen†

The Ohio State Univesity

ABSTRACT

In a visualization of a three-dimensional dataset, the insights gained are dependent on what is occluded and what is not. Suggestion of interesting viewpoints can improve both the speed and efficiency of data understanding. This paper presents a view selection method designed for volume rendering. It can be used to find informative views for a given scene, or to find a minimal set of representative views which capture the entire scene. It becomes particularly useful when the visualization process is non-interactive – for example, when visualizing large datasets or time-varying sequences. We introduce a viewpoint "goodness" measure based on the formulation of entropy from information theory. The measure takes into account the transfer function, the data distribution and the visibility of the voxels. Combined with viewpoint properties like view-likelihood and view-stability, this technique can be used as a guide which suggests "interesting" viewpoints for further exploration. Domain knowledge is incorporated into the algorithm via an importance transfer function or volume. This allows users to obtain view selection behaviors tailored to their specific situations. We generate a view space partitioning, and select one representative view for each partition. Together, this set of views encapsulates the "interesting" and distinct views of the data. Viewpoints in this set can be used as starting points for interactive exploration of the data, thus reducing the human effort in visualization. In non-interactive situations, such a set can be used as a representative visualization of the dataset from all directions.

CR Categories: I.3.3 [Computer Graphics]: Picture/Image Generation—Viewing algorithms

Keywords: viewpoint selection, view space partitioning, volume rendering, entropy, visibility

1 INTRODUCTION

With the advent of faster hardware and better algorithms, volume rendering has become a popular method for visualizing volumetric data. The traditional challenge of speeding up the rendering to achieve interactive frame-rates has been overcome for small datasets, but large datasets still pose problems for users who do not have access to supercomputing facilities. Slow frame-rates, coupled with large datasets, result in an increase in time needed to gain insights from the data. The efficiency of the visualization process can be increased by guiding the user to more informative parts of the data (or parameters) and thus saving time she would have otherwise spent in a trial-and-error search. Another option is to show more information on the screen without having a negative effect (such as, due to occlusion or cluttering). For example, various alternative rendering techniques can be used to provide a more understandable picture to the user [5][8][4]. Users can be guided to interesting features, isosurfaces and transfer functions by methods that suggest such candidates [10][19].

In this paper, we present a novel view selection method for volume rendering that can help improve the effectiveness of visualization by guiding the user to views that convey more information. Such interesting viewpoints are helpful both for data exploration and data presentation. For complex datasets, it is very difficult to manually find a view that maximizes the visibility of the relevant part of the data and minimizes occlusion. Currently, users can only use subjective judgment to evaluate and compare views. Our view selection technique introduces a measure to evaluate a view based on the amount of information displayed (and not displayed). It gives the users the ability to objectively compare two different views. The algorithm can be used to generate viewing positions to be used as starting viewpoints for browsing. Such suggested starting camera positions prove very beneficial in rendering situations with non-interactive frame-rates. Because of the time-lag between frames, users do not want to, and should not be made to [17][7][2], search the whole view-space for desirable views. The algorithm can also be used when presenting data in a non-interactive setting. It creates a smart partitioning of the view space, and selects representative views from each view group for rendering.

For this paper, we assume that the dataset is centered at the origin and that the camera is always looking at the origin from a fixed distance. We will refer to this set of camera positions as the view sphere. To evaluate and compare viewpoints, we use three viewpoint characteristics associated with each view:

- **View goodness:** The view-goodness measure tries to capture how closely the voxel visibilities for a given view match a user-input importance function. We define a view to be good if more important voxels in the volume are highly visible, and vice versa. It is maximized when the voxel visibilities are proportional to their importance. When selecting viewpoints, it is desirable that they have high goodness scores.

- **View likelihood :** Intuitively, the view likelihood of a given view is the number of other viewpoints on the view sphere which yield a view that is similar (defined by a threshold) to the given view. We define the view similarities in terms of voxel visibilities and importance that are used for view goodness. A highly likely view is a good candidate for representing the dataset from different views. On the other hand, low likelihood views are interesting because they display information that is not seen from most other viewpoints, and hence is likely to be missed by users during an interactive search.

- **View stability :** View stability of a view denotes the maximal change in view that can occur when the camera position is shifted within a small neighborhood (defined by a threshold). A small change implies a stable view, and a large change would make a view unstable. Unstable views make good starting viewpoints during interactive visualization, because the user can see a large change in view with a small mouse movement.

The contribution of this paper is as follows. We introduce a goodness measure of viewpoints based on the information theory

*Email: udeepta at acm.org

†Email: hwshen at cse.ohio-state.edu

IEEE Visualization 2005
October 23-28, Minneapolis, MN, USA
0-7803-9462-3/05/$20.00 ©2005 IEEE.

concept of entropy, also called average information. We propose that good viewpoints are ones which provide higher visibilities to the more important voxels. This interpretation leads us to the formulation of viewpoint information presented in this paper. We utilize a property of our entropy definition which indicates that when the visibilities are close to their desired values, the viewpoint information is maximized. This measure allows us to compare different viewpoints and suggest the best ones to the user. The voxel importance can be specified by the users based on their domain knowledge and the desired output. Given a desired number N of views, our algorithm can be used to find the best N viewpoints over the view space. A GPU-based algorithm is used to find the visibilities at the exact voxel centers of the volume. We also use the framework to find similarity between views, which is then used to create a view space partitioning and to find the likelihood and stability of views. Representative views for each partition can be chosen either by taking the highly likely or highly unlikely views. In interactive situations, our method can suggest unstable viewpoints, so that a small change in the camera position will yield a large change in view. For time-dependent data, we present a modification of the goodness measure of a viewpoint by taking into account not only the static information but also the change in each time-step.

2 Related Work

The idea of comparing different views developed much before computer graphics and visualization matured. As early as 1976, Koenderink and van Doorn [11][12] had studied singularities in 2D projections of smooth bodies. They showed that for most views (called stable views), the topology of the projection does not change for small changes in the viewpoint. The topological changes between viewpoints can be stored in an aspect graph. Each node in the graph represents a region of stable views, and each edge represents a transition from one such region to an adjacent one. These regions form a partitioning of the view space, which is typically a sphere of a fixed radius with the object of interest at its center. The aspect graph (or its dual, the view space partition) defines the minimal number of views required to represent all the topologically different projections of the object. A lot of research has been done since the early papers, mainly in the field of computer vision, which extended the ideas to more complex objects. In the case of volume rendering, a similar topology based partitioning can not be constructed. Instead, we find a visibility based partitioning by comparing visibilities of voxels in neighboring views, and clustering together viewpoints that are similar.

Viewpoint selection has been an active topic of research in many fields. For instance, viewpoint selection solutions have been proposed for the problem of modeling a three-dimensional object from range data [22] and from images [6], for object recognition [1], and also for cinematography [9]. However, the topic has not been well investigated in the fields of computer graphics and visualization, possibly because applications in these domains have relied heavily on human control. Recently, Vázquez et al. [20][21] have presented an entropy based technique to find good views of polygonal scenes. They define an entropy for any given view, which is derived from the projected area of the faces of the geometric models in the scene. Their motivation is to achieve a balance between the number of faces visible and their projection areas. The entropy value is maximized when all the faces project to an equal area on the screen. In this conference, two solutions are being presented for selecting good viewpoints for volumetric data. Takahashi et al. [18] calculate the view entropy for different isosurfaces and interval volumes, and then find a weighted average of the individual components. The weights are assigned using the transfer function opacities. The viewpoint measure presented in this paper is also based on the entropy function, but is defined on voxels as opposed to geometric objects. Each voxel in the data is assigned a visual significance, and the entropy is maximized when the visibilities of the voxels approach the respective significance values.

3 Viewpoint Evaluation

The essential goal of this paper is to have a computer suggest 'good' viewpoint(s) to the user. This naturally leads us to the question: "what is a good viewpoint?", or, "what makes a viewpoint better than another?". The answer will depend greatly on the viewing context and the desired outcome. For example, a photographer will choose the view which best contributes to the chosen mood and visual effect. For this paper, the context is the process of volume rendering, which is being used to get visual information from the data. Hence, for our purposes, *a viewpoint is better than another if it conveys more information about the dataset*. In this section, we present a method for quantifying the information contained in a view using properties of the entropy function from information theory.

The information that is transferred from a volumetric dataset to the two-dimensional screen is governed by the optical model which is used for the projection. In this paper, we assume the popular absorption plus emission model [15]. The intensity Y at a pixel D is given by

$$Y(D) = Y_0 T(0) + \int_0^D g(s)T(s)ds \qquad (1)$$

where, $T(s)$ is the transparency of the material between the pixel D and the point s. We will refer to $T(s)$ as the visibility of the location s. The first term in the equation represents the contribution of the background, Y_0 being its intensity. The second term adds the contributions of all the voxels along the viewing ray passing through D. A voxel at point s has an emission factor of $g(s)$, and its effect on the pixel intensity is scaled by its visibility $T(s)$. If two voxels have the same emission factor, then the one with a higher visibility will contribute more toward the final image.

The emission factors of voxels are usually defined by the users. They set the transfer function to highlight the group of voxels they want to see, and to make the others more transparent. We use this fact to define a *noteworthiness* factor for each voxel (section 3.2), which captures, among other things, the importance of the voxel as defined by the transfer function. Users can also specify the noteworthiness by other means – for example, using a separate importance volume. Based on the preceding discussion, we have the following two (not necessarily disjoint) guidelines for defining a good viewpoint:

1. A viewpoint is good if voxels with high noteworthiness factors have high visibilities.

2. A viewpoint is good if the projection of the volumetric dataset contains a high amount of information.

In the following section, we present the details of our view information function and its properties.

3.1 Entropy and View Information

Consider any information source X which outputs a random sequence of symbols taken from the alphabet set $\{a_0, a_1, \ldots, a_{J-1}\}$. Suppose the symbols occur with the probabilities $\mathbf{p} = \{p_0, p_1, \ldots, p_{J-1}\}$. Alternatively, we can think of it as the random variable X which gets the value a_j with probability p_j. The information associated with a single occurrence of a_j is defined in information theory as $I(a_j) = -\log p_j$. The logarithm can be taken with base 2 or e, and the unit of information is bits or nats respectively. In a sequence of length n, the symbol a_j will occur

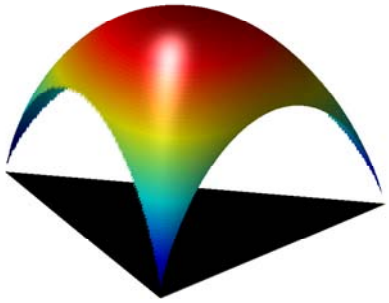


Figure 1: Entropy Function for three dimensional probability vectors $\mathbf{p} = \{p_0, p_1, p_2\}$. The function is defined only over the plane $p_0 + p_1 + p_2 = 1$, within the triangular region specified by $0 \le p_1, p_2, p_3 \le 1$. The maximum occurs at the point $p_0 = p_1 = p_2 = 1/3$, and the value falls as we move away from that point in any direction. So, increasing the entropy has the effect of making the probabilities more uniform.

np_j times, and will carry $-np_j \log p_j$ units of information. Then the *average information* of the sequence, also called its entropy, is defined as

$$H(X) \equiv H(\mathbf{p}) = -\sum_{j=0}^{J-1} p_j \cdot \log_2 p_j \qquad \text{bits/symbol} \qquad (2)$$

with $0 \cdot \log_2 0$ defined as zero [3]. Even though the entropy is frequently expressed as a function of the random variable X, it is actually a function of the probability distribution $\mathbf{p}$ of the variable X. We will use the following two properties of the entropy function in constructing our viewpoint evaluation measure:

1. For a given number of symbols J, the maximum entropy occurs for the distribution $\mathbf{p_{eq}}$, where $\{p_0 = p_1 = \ldots = p_{J-1} = 1/J\}$. (See figure 1, which gives an example of the entropy values for a three dimensional distribution.)

2. Entropy is a concave function, which implies that the local maximum at $\mathbf{p_{eq}}$ is also the global maximum. It also implies that as we move away from the equal distribution $\mathbf{p_{eq}}$, along a straight line in any direction, the value of entropy decreases (or remains the same, but does not increase).

We will use probability distributions associated with views to calculate their entropy (average information). For each voxel j, we define an importance factor W_j. We will call it the *noteworthiness* of the voxel, and it indicates the visual significance of the voxel. (More details about W_j are given in section 3.2). Suppose, for a given view V, the visibility of the voxel is $v_j(V)$. We are using the term 'visibility' to denote the transparency of the material between the camera and the voxel. It is equivalent to $T(s)$ in equation (1). Then, for the view V, we define the *visual probability*, q_j, of the voxel as

$$q_j \equiv q_j(V) = \frac{1}{\sigma} \cdot \frac{v_j(V)}{W_j} \qquad \text{where, } \sigma = \sum_{j=0}^{J-1} \frac{v_j(V)}{W_j} \qquad (3)$$

where the summation is taken over all voxels in the data. The division by σ is required to make all probabilities add up to unity. Thus, for any view V, we have a visual probability distribution $\mathbf{q} \equiv \{q_0, q_1, \ldots, q_{J-1}\}$, where J is the number of voxels in the dataset. Then, we define the entropy (average information) of the view to be

$$H(V) \equiv H(\mathbf{q}) = -\sum_{j=0}^{J-1} q_j \cdot \log_2 q_j \qquad (4)$$

The view with the highest entropy is then chosen as the best view. This satisfies the two guidelines presented earlier in section 3:

1. The best view has the highest information content (averaged over all voxels).

2. The visual probability distribution of the voxels is the closest (of all the given views) to the equal distribution $\{q_0 = q_1 = \ldots = q_{J-1} = 1/J\}$, which implies that the voxel visibilities are closest to being proportional to their noteworthiness.

To calculate the view entropy, we need to know the voxel visibilities and the noteworthiness factors. Visibilities can be queried through any standard volume rendering technique such as ray casting. The noteworthiness, described in the next section, is view independent, and needs to be calculated only once for a given transfer function.

3.2 Noteworthiness

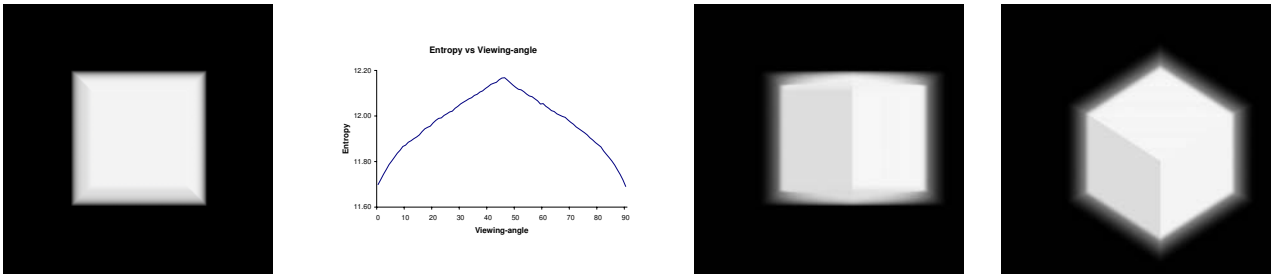
The noteworthiness factor of each voxel denotes the significance of the voxel to the visualization. It should be high for voxels which are desired to be seen, and vice versa. Considering the diverse array of situations volume rendering is used in, it is practically impossible to give a single definition of noteworthiness that satisfies expectations of all users. Instead, we can rely on the user-specified transfer functions to deliver us a definition which is tailor-made for the particular situation. The opacity of a voxel, as assigned by the transfer function, is part of the emission factor $g(s)$ in equation (1), and controls the contribution of the voxel to the final image. We use opacity as one element of the noteworthiness of the voxel. Another consideration is that voxels of a particular color can be more visually meaningful to the viewer than voxels of another color. Consider a scene with voxels of two different colors. If there are fewer voxels of the first color compared to the second, then it is possible that the second group of voxels completely occludes the first one. Also, Gestalt principles [16] suggest that the human mind extrapolates the larger object (called ground) behind the smaller one (called figure). Hence, other factors (such as opacities) being equal, we would want the voxels of the first color to have a higher importance than the others.

Based on these observations, we construct the noteworthiness W_j of the jth voxel as follows. We assign probabilities to voxels in our dataset by constructing a histogram of the data. All the voxels are assigned to bins of the histogram according to their value, and each voxel gets a probability from the frequency of its bin. The information I_j carried by the jth voxel is then $-\log f_j$, where f_j is its probability (bin frequency). Then, W_j for the voxel is $\alpha_j I_j$, where α_j is its opacity. We ignore voxels whose opacities are zero or close to zero, and these voxels are not included in the evaluation of equation (4). This reduces the computational and memory requirements for the entropy and similarity calculations (section 4.1).

The answer to what is interesting and what is not is very subjective. Our algorithm can be made to suit the goals of any particular visualization situation just by changing the noteworthiness factors. Domain specific knowledge can be readily incorporated into the framework – for example, using a separate importance volume which gives importance values for each voxel. It can be used in conjunction with, or in place of, the noteworthiness criteria described in this section. Irrespective of the method used to specify the interestingness of the voxels, maximizing the entropy serves to give better visibility to the more interesting voxels.

3.3 A Simple Example

To demonstrate our concept of view information, we constructed the test dataset shown in figure 2. The voxel opacities of the cube



(a) (b) (c) (d)

Figure 2: An illustration of the change in view entropy (equation 4) with camera position for a test dataset. Figure (a) shows the initial position of the camera. Figure (b) shows the behavior of entropy as the camera revolves around the dataset (around the vertical axis in the figure) at 1° increments. The entropy increases steadily and reaches a maximum for a movement of 45° (figure (c)), and then begins to decrease again. The maximum entropy for the whole view space is obtained for the view in figure (d).

dataset increase linearly with distance from the boundary of the cube. Figure 2(a) shows the volume rendering the dataset when the camera is looking directly at one of its faces. Next, we revolve the camera about the vertical axis of the dataset at 1° increments (or equivalently, rotate the dataset in the opposite direction about the vertical axis). The view entropy steadily increases (figure 2(b)) as more and more voxels on the side face start becoming visible. It reaches a maximum when camera has moved by 45°, which is the view that shows the two faces equally (figure 2(c)). Further movement of the camera results in greater occlusion of voxels near the first face, and the entropy begins to drop again. Upon evaluating the entropies for all camera positions around the dataset, the view in figure 2(d) results in the highest entropy. Clearly, this is one of the more informative views about the cube dataset for a human observer.

3.4 Finding the Good View

The view selection proceeds as follows. The dataset is centered at the origin, and the camera is restricted to be at a suitable fixed distance from the origin. This spherical set of all possible camera positions defines the view sphere, and represents all the view directions. The view space is then sampled by placing the camera at sample points on this sphere. We create a uniform triangular tessellation of the sphere and place the viewpoints at the triangle centroids. The camera position and the origin specify the eye and the center points respectively for the modelview transformation. Since the roll of the camera does not affect the visibilities, the up vector can be arbitrarily chosen.

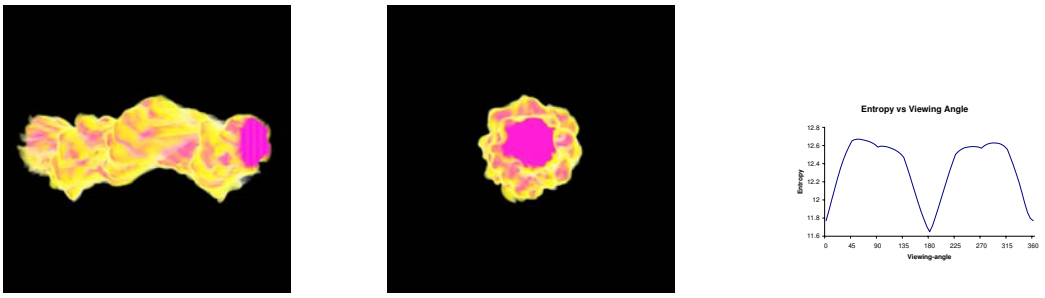
Next, the voxel visibilities are calculated for each sample view position. Our technique is not dependent on any particular volume rendering method, and both software and hardware renderers can be used by modifying them to output voxel visibilities. (Please note that the transparency or voxel visibility as given in equation (1) is numerically the same as the accumulated opacity subtracted from unity.) Since we will be comparing the visual probability distributions ($\mathbf{q}$) and the entropies ($H(\mathbf{q})$) of different views, it is desirable that we compute the visibilities at the same locations for all views (say, at the voxel centers). Most renderers, however, do not perform the opacity calculations exactly at the voxel centers. Ray-casters accumulate opacities along the rays, and texture based renderers accumulate opacities at frame-buffer pixel locations, neither of which are necessarily aligned with voxel centers. We use the GPU to calculate the visibilities at the exact voxel centers by rendering the volume slices in a front-to-back manner using a modified shear-warp algorithm. We give a brief description of our implementation below.

The object-aligned slicing direction is taken along the axis which is most perpendicular to the viewing plane. We use a floating point p-buffer with the same resolution as the volume slices. The first slice has no occlusion, so all the voxels in this slice have their visibilities set to unity. We iterate through the rest of the slices in a front-to-back order. In each loop, we calculate the visibilities of a slice based on the data opacities and the visibilities of the immediate previous slice. In each iteration, the following actions are performed. The frame-buffer (p-buffer) is cleared, and the camera set such that the current slice aligns perfectly with the frame-buffer. Then, the immediate previous slice is rendered with a relative shear[13], and a fragment program combines its opacities with its visibility values. The frame-buffer now contains the visibilities of the current slice, and these will be used in the next loop. Render to texture techniques are used to prevent a copy from the frame-buffer to a texture. After all the slices are processed, the entropy for the given view direction is calculated using equations (3) and (4).

Figure 3 shows a time-step of a 256-cube shockwave dataset. The camera was rotated about the vertical axis in a complete circle, with the dataset centered at the origin. Entropy was evaluated at 5° increments. The first figure shows the view at 55° rotation which was the view with the highest entropy. Figure(b) shows the worst view, which occurred at 180°. Figure 4 shows a $128 \times 128 \times 80$ tooth dataset. The view sphere was sampled at 128 points. Figures (a) and (b) have the highest view entropy values. Figures (c) and (d) have the lowest entropy, and not surprisingly, are highly occluded views. It is notable that the viewpoints for (c) and (d) are not very far apart, and that (a) and (b) show much of the same voxels. This shows that if the user wants a few (say, N) good views from the algorithm, returning the N highest entropy views might not be the best option. Instead we can try to find a set of good views whose view samples are well distributed over the view sphere. The next section presents such a solution.

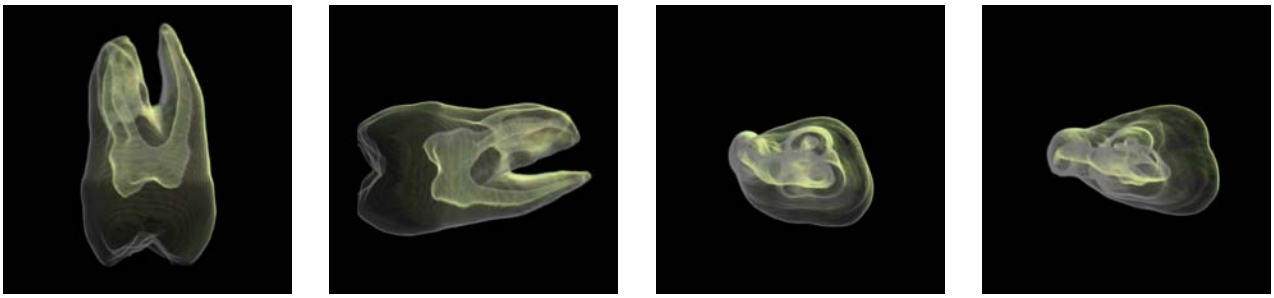
4 View Space Partitioning

The goodness measure presented in the previous section can be used as a yardstick to measure the information captured by different volume rendering views and select the best view. But the calculation of goodness considers information from only the given view, and ignores the information that might be contained in other views. In particular, neighboring viewpoints tend to have similar visibilities, and comparing a viewpoint with its neighbors can provide additional properties of the viewpoint. Also, for most datasets, a single view does not give enough information to the user. The user will almost certainly want to look at the dataset from another angle. Instead of a single view, it is desirable to present to the user a set of



(a) (b) (c)

Figure 3: The figure shows a time-step of a 256-cube shockwave dataset. The camera was rotated about the vertical axis in a complete circle, with the dataset centered at the origin. Entropy was evaluated at 5° increments. Figure (a) shows the view at 55° rotation which was the view with the highest entropy. Figure(b) shows the worst view, which occurred at 180°. Figure (c) plots the change of entropy with change in angle.



(a) (b) (c) (d)

Figure 4: The two highest entropy views for the tooth dataset are shown in (a) and (b), and the two worst ones in (c) and (d).

views such that, together, all the views in the set provide a complete visual description of the dataset. This can also be thought of as a solution to the best N views problem: given a positive number N, we want to find the best N views which together give the best visual representation of the dataset.

We propose to find the N views by partitioning the view sphere into N disjoint partitions, and selecting a representative view for each partition. A similar partitioning is defined by aspect graphs [11][12], where each node (aspect) of the graph represents a set of stable views. Each set shows the same group of features on the surface of the object. However, the aspect graph creation methods deal mostly with algebraic and polygonal models and their topology, and cannot be applied in a straightforward manner to volume rendering. Instead, we compute the partitioning by grouping similar viewpoints together.

4.1 View Similarity

To find the (dis)similarity of viewpoints, we use the visual probability distributions associated with each viewpoint (section 3.1). Popular measures for computing the dissimilarity between two distributions $\mathbf{p}$ and $\mathbf{p}'$ are the relative entropy (also known as the Kullback-Leibler (KL) distance), and its symmetric form (known as divergence) which is a true metric [3]. (Please note that some texts refer to the KL distance as divergence instead.)

$$D(\mathbf{p}\|\mathbf{p}') = \sum_{j=0}^{J-1} p_j \log \frac{p_j}{p'_j} \quad (5)$$

$$D_s(\mathbf{p},\mathbf{p}') = D(\mathbf{p}\|\mathbf{p}') + D(\mathbf{p}'\|\mathbf{p}) \quad (6)$$

Although these measures have some nice properties, there are some issues with these measures that make them less than ideal. If $p'_j = 0$ and $p_j \neq 0$ for any j, then $D(\mathbf{p}\|\mathbf{p}')$ is undefined. In our case, any voxel which is fully occluded (zero visibility) will get a visual probability q_j of zero (equation (3)). If it is visible in one view but occluded in the other, we cannot evaluate equation 5 for these views. Also, $D(\mathbf{p}\|\mathbf{p}')$ and $D_s(\mathbf{p},\mathbf{p}')$ do not offer any nice upper-bounds. To overcome these problems, we instead use the Jensen-Shannon divergence measure [14]:

$$JS(\mathbf{p},\mathbf{p}') = JS(\mathbf{p}',\mathbf{p}) = K(\mathbf{p},\mathbf{p}') + K(\mathbf{p}',\mathbf{p}) \quad (7)$$

$$\text{where, } K(\mathbf{p},\mathbf{p}') = D(\mathbf{p}\|(\frac{1}{2}\mathbf{p} + \frac{1}{2}\mathbf{p}')) \quad (8)$$

The distance between two views V_1 and V_2, with distributions $\mathbf{q}_1$ and $\mathbf{q}_2$, is then defined as $JS(\mathbf{q}_1,\mathbf{q}_2)$. This measure does not have the zero visual probability problem, since the denominator of the log term is zero iff the numerator is zero. It is also nicely bounded by $0 < JS(\mathbf{q}_1,\mathbf{q}_2) < 2$. Moreover, it can be expressed in terms of entropy [14], which allows us to reuse the view information calculations given in equation (4):

$$JS(\mathbf{q}_1,\mathbf{q}_2) = 2H(\frac{1}{2}\mathbf{q}_1 + \frac{1}{2}\mathbf{q}_2) - H(\mathbf{q}_1) - H(\mathbf{q}_2) \quad (9)$$

4.2 View Likelihood and Stability

We can now use the definition of view-distance given by equation (9) to define two additional characteristics of viewpoints – view

likelihood and view-stability. View likelihood of a view V is defined as the probability of finding another view anywhere on the view-sphere whose view-distance to V is less than a threshold. In our scenario, it is given the number of view samples on the view sphere that are within the threshold of V. If a view has a (relatively) high likelihood, it implies that the object or dataset projects a similar image for a (relatively) large number of views. On the other hand, a view with low likelihood provides information that is unique to a few views. This property is indirectly taken into consideration when we partition the set of all the view samples (that is, the view sphere). Large partitions have views with high likelihoods.

Sometimes it is not the view itself but the change in view that provides important information. If the view is changed from one viewpoint to another very similar view, the user is not shown much new information. But, if the rendering changes by a large amount, the user sees not just the new information in the visualization but also derives knowledge from the change that has occurred. Occlusion is one of the most important depth cues that is available to the user when visualizing three-dimensional renderings on a two-dimensional surface. A large change in occlusion implies a large change in visibilities, which results in a large JS distance between two viewpoints. View stability is a view property that captures this information and can be used to select viewpoints during interaction. It is defined as the maximal change that occurs when the viewpoint is moved anywhere within a given radius from its original position. The greater the change, the more unstable a viewpoint is. We calculate the (un)stability as the maximum view-distance between a view sample and its neighboring view samples in the triangular tessellation of the view-sphere. The 180° viewpoint in figure 3(b) is an unstable viewpoint for this particular viewpoint sequence.

4.3 Partitioning

Once the visual probability distributions ($\mathbf{q}$) and their entropies ($H(\mathbf{q})$) are calculated as described in section 3, we use the *JS*-divergence to find the (dis)similarities between all pairs of view samples. We then cluster the samples to create a disjoint partitioning of view sphere. The number of desired clusters can be specified by the user. Each partition represents a set of similar views, i.e., these views show the voxels at similar visibilities. If desired, the *JS*-measure can be weighted using the physical distance between the view samples to yield tight regional clusters.

The best (highest entropy) views within each partition are selected as representatives of the cluster and displayed to the user. Together, this set of images gives a good visualization of the dataset from many different viewpoints. Sometimes, it might happen that the selected representatives of two neighboring partitions lie on the common boundary and next to each other. If the view distance between two selected view samples is less than a threshold, we use a greedy approach and select the next best sample.

Figure 5 shows the results of a 5-way partitioning of the view space for the tooth dataset. 128 view samples were used with a *JS*-divergence measure. The largest partition contains 39 samples, while the smallest one has 18. The representative views from four of the partitions are shown. The view for the fifth partition is figure 4(a). We would like to point out that figures 4(a) and 4(b) both are in the same partition. In fact, the top ten high entropy viewpoints all belong to the same partition, illustrating the need for selecting representative views from different partitions.

5 Time Varying Data

Suggestion of good views becomes all the more useful in the case of time dependent data. The time required to compute a volume rendering animation of the dataset grows with the number of time steps. In an interactive setting, this creates a large lag between a viewpoint update and the completion of rendering all the frames. Moreover, it takes more tries by the user to find the desired viewpoint because the data changes with time, and the user has to consider not only the current time step but also the previous and future ones. The user's job is made harder by cases where an interesting view in a few time steps turns out to be a dull view in the rest.

In section 3, we discussed the notion of a good view and presented a measure of view information for a volume dataset. For time-dependent data, using equation (4) separately for each time-step is not the best solution – it can yield viewpoints in adjacent time-steps that are far from each other, thus resulting abrupt jumps of the camera during the animation. A natural solution is to constrain the camera, but it still does not guarantee the most informative viewpoint. For instance, it can result in a viewpoint which has a high information value for each individual time-step, but does not show any time-varying changes. It is contrary to what is expected from an animation – it should show both the data at each time-step, and also the changes occurring from one frame to the next. In the next section, we present an alternate version of viewpoint information tailored to capture the view information present in one time-step, taking into account the information present in the previous step.

5.1 View Information

Consider two random variables X and Y with probability distributions $\mathbf{r}$ and $\mathbf{s}$ respectively. If X and Y are related (not independent), then an observation of X gives us some information about Y. As a result, the information carried by Y, conditional on observing X, becomes $H(Y|X) \equiv H(\mathbf{s}|\mathbf{r})$. Then the information carried together by X and Y is $H(X,Y) = H(Y,X) = H(X) + H(Y|X)$, as opposed to $H(X) + H(Y)$. We will use this concept to create a modified viewpoint goodness measure for time dependent data.

Suppose there are n time-steps $\{t_1, t_2, \ldots, t_n\}$ in the dataset. For a given view V, we denote the entropy for time-step t_i as $H(V,t_i) \equiv H_V(t_i)$. The view entropy for all the time-steps together is $H_V(t_1, t_2, \ldots, t_n)$. We will assume a Markov sequence model for the data, i.e., the data in any time-step t_i is dependent on the data of the time-step t_{i-1}, but independent of older time-steps. Then the information measure for the view, for all the time-steps taken together, is given by equation (11). (10) is a standard relation [3], and (11) follows from the independence assumption.

$$\begin{aligned} H(V) &= H_V(t_1, t_2, \ldots, t_n) \\ &= H(t_1) + H(t_2|t_1) + \ldots + H(t_n|t_1, \ldots, t_{n-1}) \quad (10) \\ &= H(t_1) + H(t_2|t_1) + \ldots + H(t_n|t_{n-1}) \quad (11) \end{aligned}$$

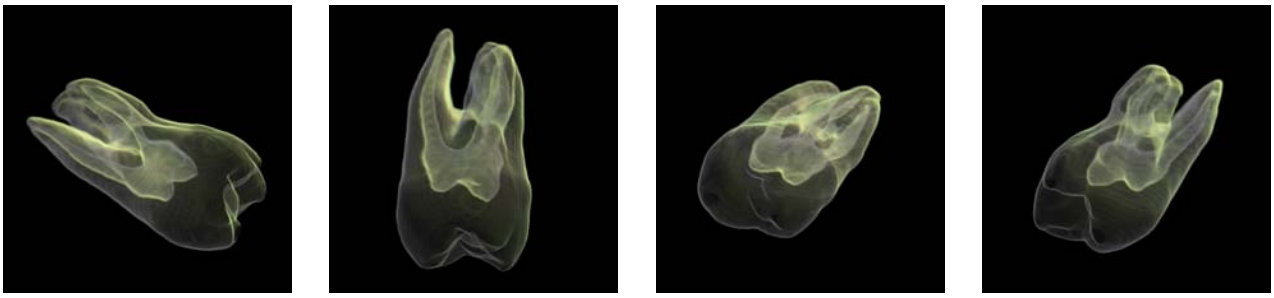
The conditional entropies will be defined following the same principles outlined in section 3. We consider a view to be good when the visibilities of the voxels are in proportion to their noteworthiness. But in the time-varying case, the significance of a voxel is derived not only from its opacity, but also from the change in its opacity from the previous time-step. For the time-step t_i, we then define the noteworthiness factor of the jth voxel as $W_j(t_i|t_{i-1}) =$

$$\{k \cdot |\alpha_j(t_i) - \alpha_j(t_{i-1})| + (1-k) \cdot \alpha_j(t_i)\} \cdot I_j(t_i) \quad (12)$$

where, $0 < k < 1$ is used to weight the effects of voxel opacities and the change in their opacities. A high value of k will highlight the changes in the dataset. Suppose the visibility of the voxel for the view V is $v_j(V,t_i)$. Then, the conditional visual probability, $q_j(t_i|t_{i-1})$, of the voxel is

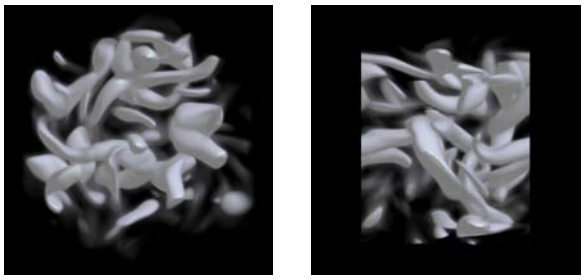
$$q_j(t_i|t_{i-1}) \equiv q_j(V,t_i|t_{i-1}) = \frac{1}{\sigma} \cdot \frac{v_j(V)}{W_j(t_i|t_{i-1})} \quad (13)$$

where, σ is the normalizing factor as in equation (3). The entropy of the view V is then calculated using equations (11) and (13). Voxels



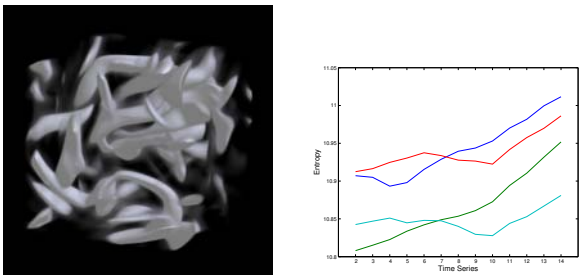
(a) (b) (c) (d)

Figure 5: Representative views for a 5-way partitioning of the view-sphere for the tooth dataset. The view for the fifth partition is figure 4(a).



(a) (b)

Figure 6: View Evaluation results for a 128-cube vortex dataset. Figure (a) shows the recommended view with a high entropy value, (b) shows a bad view for comparison.



(a) (b)

Figure 7: View Evaluation for time-varying dataset. (a) The best overall view for 14 time-steps. (b) The conditional entropies of four selected views for each of the 14 time-steps. The view in (a) is represented by the blue plot (highest curve, top-right corner).

with both low opacities and small changes (as defined by thresholds) are ignored for these calculations.

6 Results and Discussion

We have implemented our technique using a hardware-based visibility calculation (section 3.4). 128 sample views were used for each dataset. The camera positions were obtained by a regular triangular tessellation of a sphere with the dataset place at its center. The tests were run on a 2GHz P4, 8x AGP machine with a GeForce 5600 card. For a 128-cube dataset, the visibility calculations for all 128 views were completed in 42s, resulting in an average time per view of 0.33s. In case of a 256-cube dataset, the calculations for 128 views took 310s, with an average time per view of 2.42s.

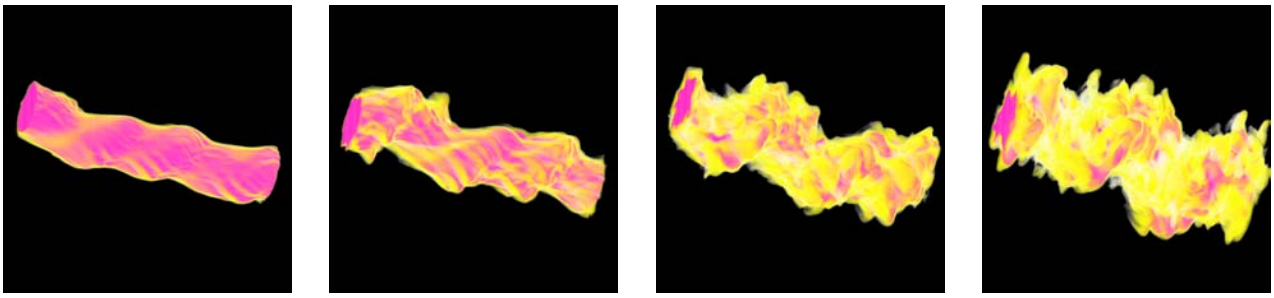
View selection results for the $128 \times 128 \times 80$ tooth dataset have been shown in figure 4. Figure 5 shows the results of a 5-way view space partitioning for the dataset using the *JS* divergence measure. The partitioning helps to avoid selection of a set of good views which happen to be similar to each other. Even though we have not considered the physical distance between the viewpoints during partitioning, it forces the selected viewpoints to be well distributed over the view sphere. Figure 6 shows view evaluation results for a 128-cube vortex dataset. Both high and low quality views are shown for comparison.

For time-varying data, we used the view information measure presented in section 5. A sequence of 14 time-steps of the 128-cube vortex data was used. The entropy for each view was summed over all the time-steps, as given by equation (11). The conditional entropy for each time-step was calculated with $k = 0.9$ in equation (12). A high value of k gives more weight to the voxels which are changing their values with time compared to high opacity voxels which remain relatively unaltered. Figure 7(a) shows the view with the best cumulative entropy for the time-series. Although the summed entropy gives a good overall view for the whole time-series, there might be other views which are better for particular segments of the time-series. Figure 7(b) plots the conditional entropies ($H(t_n|t_{n-1})$) for four selected views of the vortex dataset. The best overall view (figure 7(a)), which is represented by the blue curve (highest curve on the right edge), is not the best choice for the first half of the series. For long time sequences, it might be beneficial to consider different good views for different segments of time.

View entropy was calculated over fifty time-steps of the 256-cube shockwave dataset with 128 view samples. Figure 8 shows four time-steps from a viewpoint which had a good entropy using the time-varying criteria, and figure 9 shows the corresponding time-steps for a viewpoint which resulted in a bad score.

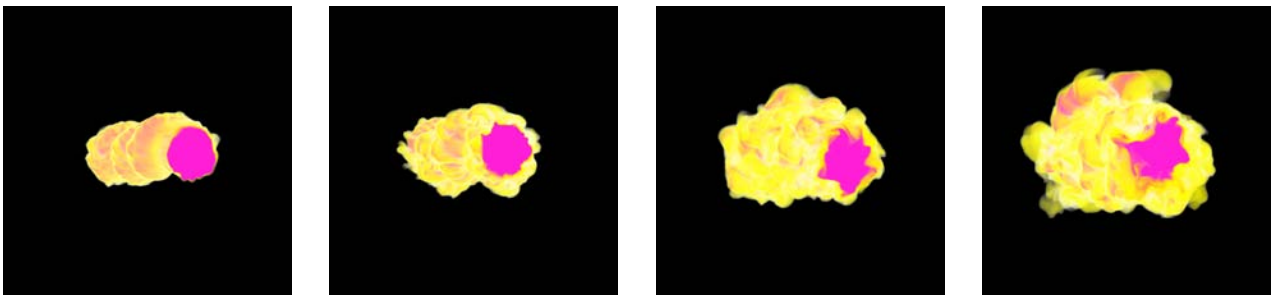
7 Conclusion and Future Work

We have presented a measure for finding the goodness of a view for volume rendering. We have used the properties of the entropy function to satisfy the intuition that good views show the noteworthy voxels more prominently. The user can set the noteworthiness of the voxels by specifying the transfer function, or by using an importance volume, or a combination of both. The algorithm can be used both as an aid for human interaction, and also as an oracle to present multiple good views in less interactive contexts. Furthermore, view sampling methods such as IBR can use the sample similarity information to create a better distribution of samples.



(a) (b) (c) (d)

Figure 8: View entropy results over 50 time-steps of the 256-cube shockwave dataset. Time steps 1, 16, 31 and 46 for a good view.



(a) (b) (c) (d)

Figure 9: Low entropy viewpoint for 50 time-steps of the 256-cube shockwave dataset. Time steps 1, 16, 31 and 46 are shown.

ACKNOWLEDGMENT

We would like to thank Antonio Garcia for his help with our implementation. This work was supported by NSF grant ACI-0329323, NSF ITR grant ACI-0325934, DOE Early Career Principal Investigator award DE-FG02-03ER25572, and NSF Career Award CCF-0346883.

REFERENCES

[1] T. Arbel and F. Ferrie. Viewpoint selection by navigation through entropy maps. In *Proc. of the 7th IEEE International Conf. on Computer Vision (ICCV-99)*, volume I, pages 248–254. IEEE, 1999.

[2] R. E. Barber and H. C. Lucas Jr. System response time operator productivity, and job satisfaction. *Comm. of the ACM*, 26(11):972–986, 1983.

[3] R. E. Blahut. *Principles and practice of information theory*. Addison-Wesley Publ. Co., 1987.

[4] S. Bruckner, S. Grimm, A. Kanitsar, and M. E. Gröller. Illustrative context-preserving volume rendering. In *Proc. of EuroVis '05*, pages 69–76, 2005.

[5] D. Ebert and P. Rheingans. Volume illustration: Non-photorealistic rendering of volume models. In *Proc. of IEEE Visualization '00*, pages 195–202, 2000.

[6] S. Fleishman, D. Cohen-Or, and D. Lischinski. Automatic camera placement for image-based modeling. *Computer Graphics Forum*, 19(2):101–110, 2000.

[7] J. L. Guynes. Impact of system response time on state anxiety. *Comm. of the ACM*, 31(3):342–347, 1988.

[8] M. Hadwiger, C. Berger, and H. Hauser. High-quality two-level volume rendering of segmented data sets on consumer graphics hardware. In *Proc. of IEEE Visualization '03*, pages 301–309, 2003.

[9] L-W. He, M. F. Cohen, and D. H. Salesin. The virtual cinematographer: A paradigm for automatic real-time camera control and directing. In *Proc. of SIGGRAPH '96*, pages 217–224, 1996.

[10] G. Kindlmann and J. W. Durkin. Semi-automatic generation of transfer functions for direct volume rendering. In *Proc. of IEEE Symposium on Volume Visualization '98*, pages 79–86, 1998.

[11] J. J. Koenderink and A. J. van Doorn. The singularities of the visual mapping. *Biological Cybernetics*, 24:51–59, 1976.

[12] J. J. Koenderink and A. J. van Doorn. The internal representation of solid shape with respect to vision. *Biological Cybernetics*, 32:211–216, 1979.

[13] P. Lacroute and M. Levoy. Fast volume rendering using a shear-warp factorization of the viewing transformation. In *Proc. of SIGGRAPH 1994*, pages 451–458. ACM, 1994.

[14] J. Lin. Divergence measures based on the shannon entropy. *IEEE Trans. on Information Theory*, 37(1):145–151, January 1991.

[15] N. Max. Optical models for direct volume rendering. *IEEE Trans. on Visualization and Computer Graphics*, 1(2):99–108, June 1995.

[16] L. Pessoa, E. Thompson, and A. Noë. Finding out about filling-in: A guide to perceptual completion for visual science and the philosophy of perception. *Behavioral and Brain Sciences*, 21(6):723–748, 1998.

[17] B. Shneiderman. Response time and display rate in human performance with computers. *ACM Computing Surveys*, 16(3):265–285, 1984.

[18] S. Takahashi, I. Fujishiro, Y. Takeshima, and Tomoyuki Nishita. A feature-driven approach to locating optimal viewpoints for volume visualization. In *Proc. of IEEE Visualization '05 (To Appear)*, 2005.

[19] S. Tenginakai, J. Lee, and R. Machiraju. Salient iso-surface detection with model-independent statistical signatures. In *IEEE Visualization '01*, pages 231–238, 2001.

[20] P. P. Vázquez, M. Feixas, M. Sbert, and W. Heidrich. Viewpoint selection using viewpoint entropy. In *Proc. of Vision, Modelling, and Visualization '01*, pages 273–280, 2001.

[21] P. P. Vázquez, M. Feixas, M. Sbert, and W. Heidrich. Automatic view selection using viewpoint entropy and its application to image-based modeling. *Computer Graphics Forum*, 22(4):689–700, 2003.

[22] L. Wong, C. Dumont, and M. Abidi. Next best view system in a 3-d modeling task. In *Proc. of International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, pages 306–311, 1999.

Dynamic View Selection for Time-Varying Volumes

Guangfeng Ji and Han-Wei Shen

Abstract—Animation is an effective way to show how time-varying phenomena evolve over time. A key issue of generating a good animation is to select ideal views through which the user can perceive the maximum amount of information from the time-varying dataset. In this paper, we first propose an improved view selection method for static data. The method measures the quality of a static view by analyzing the opacity, color and curvature distributions of the corresponding volume rendering images from the given view. Our view selection metric prefers an even opacity distribution with a larger projection area, a larger area of salient features' colors with an even distribution among the salient features, and more perceived curvatures. We use this static view selection method and a dynamic programming approach to select time-varying views. The time-varying view selection maximizes the information perceived from the time-varying dataset based on the constraints that the time-varying view should show smooth changes of direction and near-constant speed. We also introduce a method that allows the user to generate a smooth transition between any two views in a given time step, with the perceived information maximized as well. By combining the static and dynamic view selection methods, the users are able to generate a time-varying view that shows the maximum amount of information from a time-varying data set.

Index Terms—Static view selection, image based method, dynamic view selection, information entropy, optimization.

1 Introduction

Visualization of time-varying data has been a challenging problem due to the large size and the time varying nature of the underlying datasets. Previously, researchers have proposed various techniques [11, 30, 10, 21, 18] to allow for a better understanding of the time-dependent features and their evolutions through high dimensional projection, feature tracking, and illustration. However, the most general and commonly used method for visualizing time-varying data is still animation, which is created by rendering each static volume data in the time sequence. One problem for producing animations for time-varying data is that the features of interest often evolve over time, with their shapes, positions, and orientations changing continuously. To provide the user with the best visualization of those features in an animation, it is very important to select dynamic views that can follow those features so that a maximum amount of information throughout the time sequence can be perceived. As a time-varying dataset is usually large in size and time-consuming to render, selecting views by hand can be a daunting task if it is simply done by trial-and-error. To ensure that the large scale time-varying dataset can be explored in an efficient and effective way, the process of view selection should be done automatically as much as possible.

In the context of data visualization, researchers have considered ways to automate the process of view selection [22, 4, 25, 26]. However, their focuses had not been on time-varying data, which requires special treatments in order to maximize the amount of information embedded in the whole time sequence. In addition, certain important factors when selecting a good view for static data such as the perceived colors, curvatures, and opacities in the final image were not considered in their algorithms. In this paper, we first present an improved static view selection technique to address some issues that were not previously considered, and then use the new static view selection method and a dynamic-programming optimization approach to find the best time-varying view. The goal of identifying the optimal time-varying view is to maximize the amount of information the user can perceive from the rendering sequence, with constraints on movement of the views to ensure a smooth viewing path. Our static view method measures the quality of a view based on the opacity, color and curvature images generated by a volume rendering technique. The contribution of the paper is as follows:

- An optimization approach that finds the best time-varying view in a polynomial time within a search space of exponential size. The approach also takes into account the constraints of the movement of the views.
- We properly design the probability function for the opacity distribution and incorporate it into the opacity entropy evaluation. Our opacity entropy prefers an image with a large projection area with an even opacity distribution. This technique avoids some problems that can be encountered in [4].
- The color transfer function conveys important information for volume rendering. We explicitly take into account the color information by properly designing a probability function and incorporating it into the color entropy evaluation.
- The curvature of the dataset contains essential geometric information about the dataset. We explicitly take the curvature into account during the static view selection.

In this paper, we assume that all the view points are located on the surface of a viewing sphere. At each view point the user looks at the center of the sphere, where the volume is located. During view selection, the view moves on the sphere, which means the distance between the view and the volume center is fixed. We also assume the viewing and projection parameters are appropriately set up so that the projection of the volume from any view will not fall outside the window.

The organization of the paper is as follows. In section 2, we discuss the related work. In section 3, we introduce our static view selection method, which includes the evaluation of opacity entropy, color entropy and information from curvatures. We also discuss how to incorporate all the three factors into a utility function. In section 4, we introduce our optimization method to perform time-varying view selection in a polynomial time from an exponential-size search space. We also give a method to select a dynamic path between any two views which also maximizes the perceived information. In section 5, we present results to prove the effectiveness of our method.

2 Related Work

The study of view point evaluation can be dated back to 1976, when Koenderink and van Doorn [14, 15] introduced the idea of *aspect graph* to partition the viewing regions surrounding an object. The node of the aspect graph is a stable view, around which the topology of the

- *Guangfeng Ji is with The Ohio State University, E-mail: ji.15@osu.edu.*
- *Han-Wei Shen is with The Ohio State University, E-mail: hwshen@cse.ohio-state.edu.*

Manuscript received 31 March 2006; accepted 1 August 2006; posted online 6 November 2006.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

1077-2626/06/$20.00 © 2006 IEEE Published by the IEEE Computer Society

object projection does not change within a small region. The edge of the aspect graph represents a transition from a stable view to an adjacent one. The aspect graph defines the minimum number of views required to represent all the topologically different projections of the object. After its introduction, aspect graph has been studied intensively in computer vision, where many researchers used aspect graph for object recognition [5, 7, 2].

In computer graphics, several methods have been proposed to locate the optimal views for polygonal meshes. Kamada and Kawai [12] defined a view to be optimal if it minimizes the number of degenerated faces under orthogonal projection. Barral *et al.* [3] extended the idea to cope with perspective projection. In [25, 26], Vazquez *et al.*utilized the concept of information entropy from Information Theory [19] to evaluate the quality of a viewpoint. The relative visibility of each face is defined as its probability, and the optimal view is found by maximizing the probability distribution using the entropy function. Vazquez *et al.* [24] also introduced techniques to accelerate the viewpoint entropy calculation for molecular models based on different OpenGL features. Recently, Takahashi *et al.* [22] discussed view selection in the context of volume visualization. They decompose the volume into a set of feature interval volume components, and use the surface-based view point selection method suggested in [25, 26] to find the optimal view for each of the components. Then they calculate the globally optimal view by a compromise between the locally optimal views of all the feature components. In [4], Bordoloi and Shen took a volume rendering approach and proposed that in a good view point, the visibility of a voxel should be proportional to the noteworthiness value of the voxel. The noteworthiness value, or the weight of the voxel can be determined by factors such as the opacity and color of the voxel. They also discussed view similarity and how to partition the view space.

There is a rich literature in computer graphics and animation about dynamic view selection [20, 1, 29, 9, 23]. Applicable techniques range from direct orientation interpolation [20] to complex view planning for complicated 3D scenes. Andujar *et al.* [1] proposed a camera path planning method for walkthrough of complex scene models. Their method is based on identifying the free-space structure of the scene and an entropy-based measurement of the relevance of a viewpoint. Wernert and Hanson [29] discussed the camera path planning based on a personal "guide" that keeps the user oriented in the navigation space which also points to interesting subject area. Barral *et al.* [3] presented a method for automatic exploration of static scenes. In their method, the quality of a view is computed by defining a new importance function that depends on the visible pixels of each polygon. Hong *et al.* [9] studied how to select camera path to navigate in the human colon. van Wijk and Nuij [23] introduced an elegant method to generate a smooth animation from one view to the other by zooming and panning. There are two major differences between our work and the previous work. First we deal with the problem of view selection for time-varying data where the underlying phenomena are changing over time. Second our problem involves a different scene setting from the previous work, where our views move on a viewing sphere and look at the center of the sphere. Our goal is to maximize the information perceived from the time-varying data while following the view movement constraints. In [4], Bordoloi and Shen considered the problem of finding a good viewpoint for time-varying dataset. However, their method is to find a static view point throughout the animation so that the user can perceive the maximum summation of conditional entropy from the time series. The conditional entropy is the relative entropy of a datastep based on its previous step. Compared with the method, our method tries to find a dynamic viewing path.

It is also worth mentioning that information entropy has been utilized in lighting design and shape analysis [8, 27, 17]. Gumhold [8] designed lighting for static scenes and placed light sources at locations where the illumination information is maximized. The illumination information is measured by the entropy function, which is calculated based on the pixel brightness values. Based on the user perception study, Gumhold further refined the illumination entropy definition by perceptually binning the brightness values and incorporating an importance weight based on surface curvature measured in the image. Vazquez *et al.* [27] improved the method of Gumhold by defining the information entropy over regions with similar colors measured in the CIELUV color space, rather than only based on the brightness values. In [17], Page *et al.* measured the shape complexity for 2D image contours and 3D triangle meshes by a shape information metric, and they proposed an algorithm to compute the metric based on curvature estimates for both discrete curves and surfaces.

3 Static View Selection

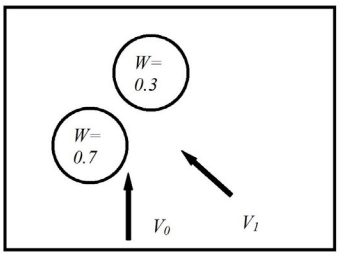


Fig. 1. The figure to illustrate that the result from [4] should be improved.

The essential problem any view selection technique tries to solve is to find a good view point through which the users are able to perceive the maximum amount of information from the underlying scene. In the context of volume visualization, Takahashi *et al.* [22] proposed a surface-based view point optimization algorithm where the geometric properties of interval volumes faces are considered. Their method produces good static views for data that can be decomposed into different interval volumes. Bordoloi and Shen [4] took a direct volume rendering approach without the need of intermediate geometry. Their method generates good views in general with the exception of some cases. For example, in Figure 1, there are two voxels in the scene, one with a weight of 0.7 and the other 0.3. Since their method prefers views from which the visibility of the voxel is proportional to its weight, the voxel with weight 0.7 has to occlude the other voxel to some degree in order to achieve a higher score for their entropy formula. However, these two voxels are readily visible through some views such as V_1. From this example, we can see that if the visibility of a voxel can be maximized, it does not have to be proportional to its weight. To remedy this problem and consider additional important properties of the data, we propose an image-based view selection method. Our method measures the quality of a static view not only based on its opacity and projection size (which is the primary criterion of some of the previous algorithms), but also explicitly considers the color and curvature distribution of the rendered images. Our motivation comes from the fact that color and curvature convey very important information about the underlying phenomenon in many applications.

3.1 Measurement of Opacity Distribution and Projection Size

Imagine a user is visualizing a volumetric dataset using a volume rendering technique. Some voxels in the volume have higher opacity values, meaning these voxels are more important. Less important voxels are assigned with smaller opacities. Initially the user may choose a view through which many opaque voxels are aligned in the viewing direction and hence more occlusion occurs. In this case, some pixels in the final image will have very high opacity values, while the opacity values at other pixels are low. The user realizes that this is not a good view, so s/he changes to a view where less occlusion occurs in the volume, so that the user can see many voxels more clearly. In this case, the opacity value in the image will be more evenly distributed. Besides this, the user may also generally prefer a rendering image with a larger projection area. From this example, it can be seen that an important factor that contributes to the selection of good views is the distribution of opacity values and the size of the projection area in the resulting image. An image with an even opacity distribution and a large projection area should be more favorable than one with an uneven opacity

distribution and/or a small projection area. A function is desired to reflect the property. The Shannon entropy function [19] can be utilized to perform the measurement.

In Information Theory, the Shannon entropy function is used to measure the amount of information contained in a random sequence of symbols. Suppose the symbols occur in the set $\{a_0, a_1, ..., a_{n-1}\}$ with the occurrence probability $\{p_0, p_1, ..., p_{n-1}\}$, the average information of the sequence, called entropy, is defined as

$$H(x) = -\sum_{i=0}^{n-1} p_i \cdot log_2(p_i) \quad (1)$$

One nice property of the entropy function is that it is a concave function. It only has one local maximum value, which is also the global maximum value. It reaches this maximum value $log_2 n$ when $p_0 = p_1 = ... = p_{n-1} = 1/n$, that is, the distribution of the probability is perfectly even among all the symbols. As the probability moves away from the perfectly even distribution along a straight line in any direction, the probability becomes less and less evenly distributed, and the value of the entropy function will also decrease.

The Shannon entropy function can be utilized to measure the information contained in an opacity image. We now explain how the probability is designed so that the entropy function gives a higher value when the opacity value is more evenly distributed and the projection area is larger, while it gives lower values otherwise. Given an opacity image which contains n pixels with opacity value $\{\alpha_0, \alpha_1, ..., \alpha_{n-1}\}$, we define the probability p_i of the *ith* pixel as

$$p_i = \frac{\alpha_i}{\sum_{j=0}^{n-1} \alpha_j} \quad (2)$$

The image entropy is calculated by equation 1. Although the entropy is evaluated over all the image pixels, the background pixels actually do not contribute to the entropy. The reason is that the opacity value of any background pixel is 0, so it will not affect the probability and entropy contribution of any foreground pixel. Furthermore, since $0 \cdot log_2 0$ is defined as 0, background pixels will not contribute to the final entropy value of the whole image. Therefore, we can define the image entropy just over the foreground area. The image entropy gets the maximum value when all the foreground pixels occur in the same probability, that is, all the foreground pixels have the same opacity values.

The entropy function also takes into account the size of the projection area, which is the foreground of the image. The reason is that the maximum entropy value of an image is $log_2 f$, where f is the size of the foreground. Therefore, the entropy of an image with a large foreground area and even distribution gets a higher value than one with smaller foreground areas. In summary, our opacity entropy function prefers an image with a large projection area with an even opacity distribution.

3.2 Measurement of Color Distribution

Opacity is just one factor that influences the selection of good views. Another important factor that determines the quality of a view is color. In volume rendering, colors are often assigned to voxels by using a color transfer function. A well-designed color transfer function should highlight salient features by using perceptually attentive colors, and map unimportant voxels to some less attentive colors. The measurement of a view's quality should keep the fidelity of the color transfer function. This means that in the color-mapped volume, even though some colors (the less attentive colors assigned to unimportant voxels, for example) may occur more frequently than some other colors (attentive colors assigned to salient features, for example), the less frequently salient feature colors actually carry more information. Therefore a good volume rendering image should contain more of these colors and thus more information about the salient features. Furthermore, we always want to highlight as many salient features as possible in the limited screen area. If the volume contains multiple salient features, these features should be mapped to the final images equally, i.e, the projected areas for different colors should be as even as possible among all the salient features. Based on the analysis, it can be seen that a good view should maximize the area of the salient colors while maintaining an even distribution among these colors.

To measure the color distribution of the volume rendering image, we also utilize the Shannon entropy function. The entropy function and the probability evaluation should be designed so that the entropy function gives a higher value for an image with more evenly distributed and larger areas of salient colors, while giving lower values for images with less evenly distributed and/or smaller areas of salient colors. Suppose there are n colors $\{C_0, C_1, ...C_{n-1}\}$, where $C_1, C_2, ...C_{n-1}$ occurs in the color transfer function and C_0 is the background color (actually C_0 can be a spectrum of colors, which includes every pixel of the image which is not perceptually similar to any of $C_1, C_2, ...C_{n-1}$). Given any pixel in the rendered image, we can determine which feature it belongs to by measuring the perceptual color distance between the pixel color and the feature color. If it does not belong to any feature (either the feature it should belong to is highly occluded, or it comes from unimportant voxels), it will be assigned to C_0. Please note that a perception-based color space should be used during the process. We choose the CIELUV color model [6] since it provides a perceptually equal color space, i.e., the distance in CIELUV space reflects the perceptual color difference. Suppose the total window area is T and the color areas of $C_1, C_2, ...C_{n-1}$ are $A_1, A_2, ...A_{n-1}$ respectively. The area for C_0 is then $A_0 = T - \sum_{i=1}^{n-1}(A_i)$. The probability is defined as

$$p_i = \frac{A_i}{T} \quad (3)$$

It is a probability definition since $T = \sum_{i=0}^{n-1}(A_i)$. The color entropy function is defined as in equation 1. We can see that the entropy reaches its maximum value when $A_0 = A_1 = ... = A_{n-1}$, that is, all the color areas are even. Due to the inclusion of A_0, large background area will incur small total salient color area, and thus uneven probability distribution and small entropy value accordingly. Therefore, the entropy function and our probability definition prefer larger total salient color area and more even distribution among all salient colors. It should be noted that the probability definition can lead to a small undesired effect. This happens when we see each of the salient colors and the background with the same area, which reaches the maximum of the entropy. The entropy will get smaller if the area of salient colors is enlarged, and this is undesired. However, this is less likely to happen in practice since the background area for any given view is usually large enough so that the volume rendering images from all the views can be projected into the window. We can also intentionally increase the window size to avoid the problem. Furthermore, even if the error occurs, it can be as large as $log(n) - log(n-1)$, which is a negligible number for a relatively large n. A similar approach has been used in [25] to deal with the background issue.

It is also noteworthy to mention that we choose a lighting model which involves only ambient and diffuse lighting calculation. Specular lighting is not included since it can alter the color of pixel by the color of the light. The color entropy evaluation works well for a well-designed color transfer function where colors are used to highlight different features (for example, colors are used to depict different components in a segmented volume). If a color transfer function just simply assigns gray-scale or rainbow colors according to different values, the color entropy may not reflect the feature information contained in the view.

3.3 Measurement of Curvature Information

Opacity and color are two important factors that measure the quality of a view. In addition to opacity and color, there are other properties that also contribute to the information provided in a volume rendering image. One of such properties is the curvature. Previously, Lee *et al.* [16] utilized curvature to identify the mesh importance information. They introduced the idea of mesh saliency as a measure of regional importance for graphics meshes, and the mesh saliency is defined in a scale-dependent manner using a center-surround operator on Gaussian-weighted mean curvatures. In our method, we notice that low curvatures imply flat areas and high curvatures mean highly

irregular surfaces, which often contain more information (If the volume is noisy, a smoothing operation should be performed beforehand). Therefore, it is important to take the curvature information into account during the selection of good views.

One problem of considering curvature information in view selection is how to present the curvatures in a volume rendering image. We achieve this with two steps. First we calculate the curvature at each voxel position of the volume, using the method proposed by Kindlmann *et al.* [13]. When the volume is rendered, the color of a voxel is determined by its curvature. Voxels with high curvature are assigned with high intensity colors, while voxels below a certain low-curvature threshold are assigned with the color $(0,0,0)$. The opacity of the voxel is determined independently, which can be based on its original data value, or some other properties such as the gradient. After the rendering is performed and the image is generated, the intensity of the image reflects the amount of curvature perceived from the visible part of the volume, that is, an image with high intensity means that the user can see many high-curvature voxels from that view.

3.4 The Final Utility Function

Opacity, color and curvature all contribute to the information perceived from a rendering of the volume. We need a function to incorporate all the factors. This utility function [28] u from a view v should have the following basic form:

$$u(v) = \alpha \cdot opacity(v) + \beta \cdot color(v) + \gamma \cdot curvature(v) \qquad (4)$$

where $\alpha + \beta + \gamma = 1$. One problem with the utility function is that the opacity, color and curvature contributions are not normalized. We should normalize each of the factors into $[0,1]$ before the summation. The maximum value of the entropy function of an image with a projection size of n is $log_2 n$. So if we find the maximum projection size M of the images among all the views, each of the entropies can be normalized by dividing over $log_2 M$. The maximum value of the color entropy is $log_2 n$, where n is the number of colors (see section 3.2). Therefore, the color entropy can be easily normalized by a division over $log_2 n$. The normalization of the curvature contribution can also be easily done by a division over the maximum projection size M, since the maximum intensity of each pixel is 1.

If we possess any prior knowledge of the volume, it is often desirable to give different weights to different factors. One scenario is that people often design very sophisticated opacity transfer function, but use a simple gray-scale or rainbow color transfer function. In this case, it is desirable to put more weight into $opacity(v)$ than $color(v)$, since opacity conveys more information. However, in another case where different colors are used to highlight different features in a segmented volume, it is desirable to put large weight to $color(v)$. In practice, we can choose proper weight for every factor based on the characteristic of the data and transfer function and the nature of the application.

4 Dynamic View Selection

In this section, a dynamic view selection algorithm is presented. The goal of dynamic view selection is to allow the user to find a viewing path which shows the maximum amount of information from the time-varying dataset, and the path should show near-constant angular velocity (all the views lie on the surface of a viewing sphere). We formulate this into the following three principles that a good dynamic viewing path should follow:

- The view should move at a near-constant speed.
- The view should not change its direction abruptly.
- The information perceived from the time-varying data should be maximized among all the viewing paths.

In the following subsections, we first discuss the issue of how to select time-varying views that follow the three principles. Then we present a method that allows the user to find a path between any two views in a given timestep that maximizes the perceived information while obeying the other two principles.

4.1 Time-Varying View Selection

The problem of time-varying view selection is that given a view at $t = 0$, among all the possible paths along which the view can move smoothly to the final timestep at a near-constant angular velocity, find the path that gives the maximum perceived information. If in average a view can move to one of n possible views at the next timestep, and there are total t timesteps, the complexity of the problem can be n^t. This search space is exponentially large. It is impractical to try all these paths and find the optimal one.

To solve the problem more efficiently, we can employ the dynamic programming approach. Let's first consider selecting time-varying views with the first and third principles in mind, that is, we want to find a time-varying view that moves at a near-constant speed, and the information perceived from that path is maximized out of all possible paths. Suppose the camera is moving with speed V, with $V_{min} \le V \le V_{max}$. V_{min} and V_{max} are used to bound the speed of the view so that when V_{min} is close to and V_{max}, the view moves at a near-constant speed. We use $P_{i,j}$ to denote the position of the jth view at $t = i$, and $MaxInfo(P_{i,j})$ is the maximum amount of information perceived from $P_{i,j}$ to some view at the final timestep. The following recursive function holds:

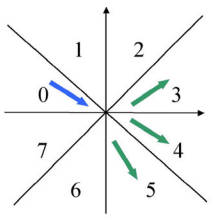
$$MaxInfo(P_{i,j}) = \max_{k=0}^{NumofViews-1}\{u(P_{i,j}) - Cost(P_{i,j}, P_{i+1,k}) + MaxInfo(P_{i+1,k})\}$$

where $u(P_{i,j})$ measures the information perceived at the view $P_{i,j}$. $Cost(P_{i,j}, P_{i+1,k})$ measures the cost to move from $P_{i,j}$ to $P_{i+1,k}$. If the jth view point and the kth view are within $[V_{min}, V_{max}]$, the cost is 0, otherwise the cost is $+\infty$. The equation basically says the maximum amount of information perceived from $P_{i,j}$ to some view point at the final timestep will be equal to the sum of the information perceived at $P_{i,j}$, and the maximum information perceived from $P_{i+1,k}$ to some view at the final time step. $P_{i+1,k}$ represents a view point at $t = i+1$ that can be reached within $[V_{min}, V_{max}]$ distance from $P_{i,j}$. We will consider all the views $P_{i+1,k}$ at timestep $i+1$. The following C-style code performs the calculation of all the $MaxInfo(P_{i,j})$.

```
for (i=0; i<NumofViews; i++)
   MaxInfo[NumofTimeSteps-1, i]=
   u[NumofTimeSteps-1, i];

for (i=NumofTimesteps-2; i>=0; i--)
   for (j=0;j<NumofViews; j++)
   {
      MaxInfo[i, j]=0;
      for (k=0; k<NumofViews; k++)
      {
         double Info=u[i, j]-Cost(j,k)
         +MaxInfo(i+1, k);
         if (Info>MaxInfo[i, j])
         {
            MaxInfo[i, j]=Info;
            NextViewIndex[i, j]=k;
         }
      }
   }
```

The initial condition is $MaxInfo(P_{n-1,i}) = u(P_{n-1,i})$ for $i \in [0..NumofViews-1]$. The dynamic programming process calculates all the $MaxInfo\{P_{i,j}\}$ backwards in time, according to the recursive function. $NextNodeIndex\{P_{i,j}\}$ records the view index at the next timestep that gives the maximum information from $P_{i,j}$ to some view at the final timestep, and it can be used to recover the time-varying path. The dynamic programming process finishes all the computation in $O(n \cdot v^2)$ time, where n is the number of total timesteps, and v is the number of total views. This process only takes a polynomial time complexity.

The above dynamic programming calculates an optimal path based on the restriction that the view should move with the speed within



(a)

{{0, 0, 0, 1, 1, 1, 0, 0},
{0, 0, 0, 0, 1, 1, 1, 0},
{0, 0, 0, 0, 0, 1, 1, 1},
{1, 0, 0, 0, 0, 0, 1, 1},
{1, 1, 0, 0, 0, 0, 0, 1},
{1, 1, 1, 0, 0, 0, 0, 0},
{0, 1, 1, 1, 0, 0, 0, 0},
{0, 0, 1, 1, 1, 0, 0, 0}}

(b)

Fig. 2. An example of a partition of a view point's local tangent plane and one of the possible allowed turns encoded in matrix.

$[V_{min}, V_{max}]$. But it does not prohibit the view from making sharp turns, which is undesirable when viewing the animation. It is also impossible to use the information stored at *NextViewIndex* to find the optimal path that does not make sharp turns, since *NextViewIndex* only records the optimal paths that move at a near-constant speed. To address this problem, at each view point on the viewing sphere, we partition its local tangent plane into many different regions, and restrict the allowed turns. Figure 2 illustrates a partition of eight regions and a matrix that encodes the allowed turns. We use $MaxInfo(P_{i,j,r})$ to denote the maximum amount of information perceived from $P_{i,j}$ to some view point at the final timestep, and $P_{i,j}$ was entered from region r from its previous view. Then the following recursive function holds:

$$MaxInfo(P_{i,j,r}) = \max_{t=0..NumofRegions-1, k \in Regiont} \{u(P_{i,j}) - Cost(P_{i,j}, P_{i+1,k}) + MaxInfo(P_{i+1,k,t})\}$$

The following C-like code calculates all the $MaxInfo(P_{i,j,r})$:

```
for (i=0; i<NumofViews; i++)
   for (j=0; j<NumofRegions; j++)
      MaxInfo[NumofTimeSteps-1, i, j]=
      u[NumofTimeSteps-1, i];

for (i=NumofTimesteps-2; i>=0; i--)
   for (j=0; j<NumofViews; j++)
      for (r=0; r<NumofRegions; r++)
      {
         MaxInfo[i, j, r]=0;
         for (all ts and each k in region t)
         {
            int o=FindRegionNum(k, j);
            if (!AllowedTurn[r, o])
               continue;
            double Info=u[i, j]-Cost(j, k)
            +MaxInfo(i+1, k, t);
            if (Info>MaxInfo[i, j, r])
            {
               MaxInfo[i, j, r]=Info;
               NextViewIndex[i, j, r]=k;
               NextRegionIndex[i, j, r]=t;
            }
         }
      }
```

where o is the region number leaving the jth view, and o can be easily determined based on the the projection to local tangent plane at the jth view. *NextViewIndex* and *NextRegionIndex* record the view and region index at the next timestep that offers the maximum information to some view at the final timestep. These two data structures can be used to recover the path. The dynamic programming process finishes all the computation in $O(n \cdot r \cdot v^2)$ time, where n is the number of timesteps, v is the number of views, and r is the number of regions. This process only takes a polynomial time complexity. After the dynamic programming is done, given the initial view at $t = 0$, the results stored at $MaxInfo$, *NextViewIndex* and *NextRegionIndex* can be used to find the maximum perceived information and the optimal time-varying view associated with the initial view.

4.2 Viewing Path Between any Two Views in a Given Timestep

Another case of dynamic view selection is to find a viewing path between any two viewpoints in a given timestep. This viewing path should also follow the three principles, i.e., moves between these two viewpoints smoothly with a near-constant angular velocity, and maximizes the perceived data information at the same time. This technique can be very useful to showcase a static dataset. When generating an animation, keyframes are usually specified by the user, and intermediate frames are generated by interpolation. If different viewpoints are assigned in the different keyframes, spherical linear interpolation (SLERP) is a common technique to interpolate the intermediate view positions. SLERP does give a viewing path with constant angular velocity, but it does not take the perceived information into consideration. Next we will explain how we maximize the perceived information and take all three principles into consideration.

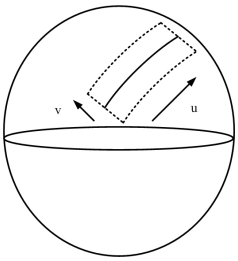
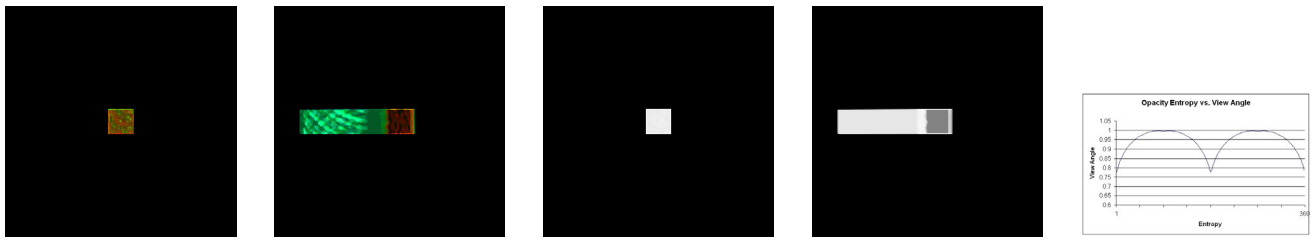


Fig. 3. The solid curve is the SLERP path. Our algorithm will consider all the neighbors of the SLERP path that lie within the dotted area. All the neighbors are parameterized by u and v.

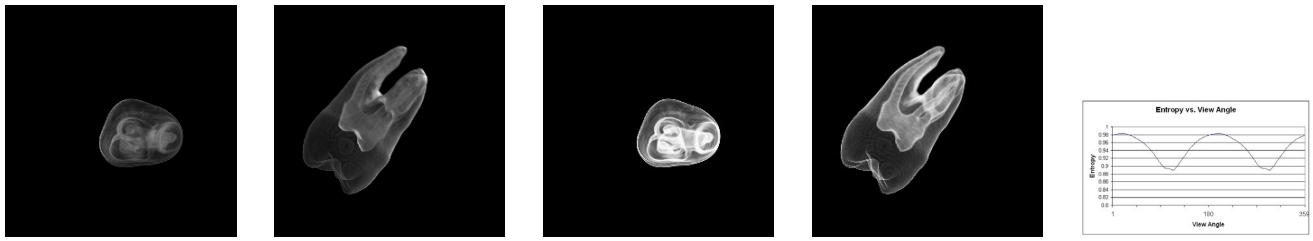
Given any two views on a viewing sphere, there are an infinite number of paths that connect these two views. One factor in our design of the dynamic path is that it should follow the general direction of the SLERP path, since the SLERP path is the shortest path that connects the two points with constant angular velocity. Therefore, we only allow the view to move at the neighboring views of the SLERP path (as shown in Figure 3). We also need to put restriction on the direction of the allowed movement so that the view will not go back and forth in a circular manner. We achieve this by parameterizing all the neighbors relative to the SLERP path, as illustrated in Figure 3. A movement is allowed only if the u parameter is increasing and the v parameter difference is within a threshold. We call these paths *monotonic paths*. We can also enforce the direction change by adopting the local coordinates and the admissible turn matrix in Figure 2. When evaluating the quality of different paths, the summation of information should not be used, since some paths can go through more view points than others. One good criterion can be the average information. The pseudo code below illustrates how to use the propagation method similar to the single-source shortest path algorithm to find the optimal path.

```
ActiveSet={Source viewpoint S};
PathLength=0;
PathInfo[S, PathLength]=u(S);

Initialize all the other PathInfos to a
minimum value;
NextActiveSet=empty;

while(ActiveSet is not empty)
{
   PathLength++;
   for each view V in ActiveSet
      for each neighbor N of V
```



(a) (b) (c) (d) (e)

Fig. 4. The figure shows the static view selection results based on opacity entropy for the shockwave dataset. (a) shows the worst view, (b) is the best view, and (c) and (d) are the opacity images for (a) and (b) respectively. (e) plots the change of opacity entropy with respect to different viewing angles where the shockwave is rotated around the Y axis in a full circle.



(a) (b) (c) (d) (e)

Fig. 5. The figure shows the static view selection results based on opacity entropy for the tooth dataset. (a) shows the worst view, (b) is the best view, and (c) and (d) are the opacity images for (a) and (b) respectively. (e) plots the change of opacity entropy with respect to the viewing angle when the tooth is rotated around the X axis in a full circle.

```
      if (the movement from V to N is
          monotonic)
      {
         PathInfo[N, PathLength]=max(
         PathInfo[N, PathLength],
         u(V)+PathInfo[V, PathLength-1]);
         Put N in NextActiveSet;
      }
      ActiveSet=NextActiveSet;
}

For all the PathInfo[D, n] where D is the
destination
{
   Find the one with the maximum average
   information and it will be the optimal
   path.
}
```

Notice the above process only runs on the neighborhood of the SLERP path. If the neighborhood vertices and edges among the vertices are stored in an adjacency matrix, the algorithm takes $O(V^2)$ time. If the vertices and edges are stored in an adjacency list, the algorithm takes $O(E+VlogV)$ time, where V is the number of vertices, and E is the number of edges.

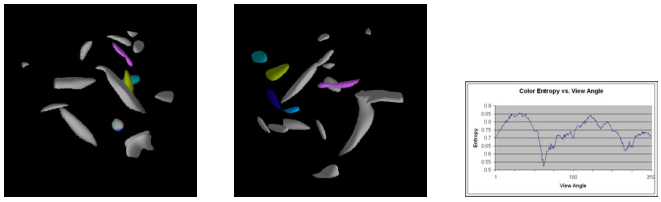
5 Results and Discussion

We have implemented and tested both the static and dynamic view selection algorithms on a Pentium IV 1.4GHz machine with an nVidia GeForce 6800 graphics card. Our view selection algorithms take as input the opacity, color and curvature images rendered from the dataset, which can be generated by any volume rendering technique. In our implementation, we choose a hardware-based volume slicing technique with 3D texture mapping to generate those images. 256 sample views were used for each dataset, and these views are evenly distributed on the viewing sphere.

The test result for the $512 \times 64 \times 64$ shockwave dataset is shown in Figure 4. The opacity entropy value is used during the test to show its effectiveness in determining view quality. Figure 4 (a) shows the worst view which has the smallest opacity entropy, and Figure 4 (b) shows the best view with the highest opacity entropy. Figures 4 (c) and (d) illustrate the opacity images of the worst and best views respectively. It took 6.92 seconds to compute the opacity entropy values for the 256 views and find the best and worst views, and the size of all the images is 256×256. By using entropy and the proposed probability function, our opacity entropy evaluation takes both the opacity distribution and the projection area into consideration, and the opacity entropy prefers an image with an even opacity distribution and a larger projection area. To illustrate how the opacity entropy varies according to different viewing angles, the view is rotated along the Y axis in a complete circle. Figure 4 (e) plots the change of opacity entropy with respect to different views.

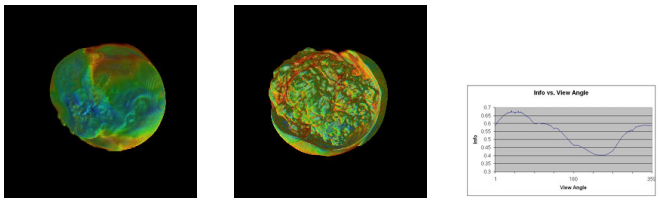
We also used the $128 \times 128 \times 80$ tooth data to test the view selection algorithm based on the opacity entropy, and the result is shown in Figure 5. Figure 5 (a) shows the worst view with the smallest opacity entropy, and Figure 5 (b) shows the best view with the largest opacity entropy. Figures 5 (c) and (d) are their opacity images. It took 7.18 seconds to compute the opacity entropy values for the 256 views and find the best and worst views, and the size of all the images is 256×256. The variation of opacity entropy with respect to different views is also plotted in the Figure 5 (e), where the viewing angle is rotated incrementally around the X axis.

We used the 128^3 vortex dataset to show the effectiveness of the color entropy function. The data set contains many components and we use the color transfer function to highlight components which may go through topological changes in future timesteps. Other components are assigned a gray-scale color. Figure 6 (a) shows the worst view with the smallest color entropy, and Figure 6 (b) shows the best view. It can been easily seen that Figure 6 (b) conveys more information about the five topologically important features than Figure 6 (a). In Figure 6 (a), the total projection area of the five highlighted features is small, and the projection area ratio among the highlighted features is very



(a) (b) (c)

Fig. 6. The figure shows the static view selection results based on color entropy for the vortex dataset. (a) shows the worst view, (b) is the best view, and (c) plots the change of color entropy with respect to different viewing angles when the vortex is rotated around the Y axis in a full circle.



(a) (b) (c)

Fig. 7. The figure shows the dynamic view selection results for the TSI dataset. (a) shows the worst view, (b) is the best view, and (c) plots the change of the final information with respect to the viewing angle when the TSI dataset is rotated around the Y axis in a full circle.

uneven. This leads to a very small color entropy value. In contrast, in Figure 6 (b), the five highlighted features have a large projection area and an even projection area distribution, and therefore a large value for the color entropy. It took 16.3 seconds to compute the color entropy values for the 256 views and find the best and worst views, and the size of the color images is 256×256. Figure 6 (c) plots the change of color entropy with respect to different views where the viewing angle is rotated incrementally around the Y axis.

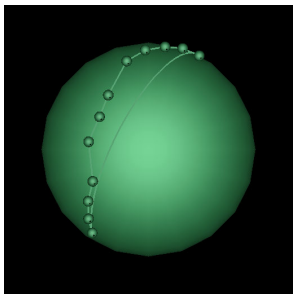


Fig. 8. The figure shows two paths which move from one view point to the other. The right path is generated by SLERP interpolation with an average information of 0.51. The left path is generated by our method. The path is smooth and gives an average information of 0.56.

Figure 7 gives the view-selection result for the Terascale Supernova Initiative (TSI) dataset. The dataset modelled the core collapse of supernovae and was generated by collaboration among Oak Ridge National Lab and eight universities. In the paper, we visualize the entropy scalar component of the dataset, which is derived from pressure and density scalar values. When exploring the dataset, we used the rainbow color transfer function. Therefore, in our view selection test, color information is not considered. Two factors, curvature and opacity, are considered in the calculation of view information. We want to design a utility function which puts more weight for views that show more jagged area. In our design, we set the coefficients for curvature and opacity to 0.8 and 0.2 respectively. Figure 7 (a) shows the worst view, and Figure 7 (b) is the best view. It is obvious that Figure 7 (b) shows more detailed information about the jagged area than Figure 7 (a). It took 18.7 seconds to evaluate the curvature information and opacity entropy for all the 256 views and find the best and worst views, and the size of the images is 256×256. To show how the view utility function varies, Figure 7 (c) plots the change of utility value with respect to different views, where the view is rotated incrementally around the vertical (Y) axis.

We also used the TSI dataset to test our dynamic view selection algorithm. The supernova is a very dynamic phenomenon where the features are morphing and rotating rapidly in space. Our previous static view selection algorithm shows that at a given timestep, very little information about the phenomenon can be perceived if the volume is viewed from some bad views. If the view for an animation is fixed, much of the phenomenon would be occluded for many timesteps (see Figure 9 (f)-(i)). Recall that the goal of our algorithm is to find a viewing path with the maximum amount of information, which also follows the constraint that the camera moves at a near-constant angular velocity. We used our static view selection to calculate the view information of every view point at every timestep and used our dynamic programming algorithm to find the best path. All the timesteps use the same view point set on the sphere. Figure 9 (a) shows the best path in which viewpoint $P_{0,0}$ moves in time with the speed within (0.9, 1.2) (The radius of the viewing sphere is 1). Although the supernova phenomenon is morphing rapidly, we still perceive a maximum amount of information following our dynamic viewing path. It took 4.31 seconds for the dynamic programming process to find the optimal path. Figure 9 (a) shows part of the path, which demonstrates near-constant angular velocity (the distance in Figure 9 (a) is distorted). Furthermore, following the path, the overall information perceived from the time-varying data is maximized. Figures 9 (b)-(e) show four snapshots of the time-varying dataset captured by the time-varying view path, and Figures 9 (f)-(i) show the images seen from the original view at the timesteps corresponding to (b)-(e) respectively. The user can apparently see more turbulent side of the phenomenon all the time from the time-varying views.

We also used the TSI dataset to show a viewing path selected from any two views in a given timestep. The TSI dataset at $t = 0$ is used, and Figure 8 shows both the SLERP and the optimized paths. It took 0.08 seconds to find the optimized path. The average information perceived by the SLERP path is 0.51, while the optimized path gives 0.56.

6 Conclusion and Future Work

In this paper, we present methods for both static and dynamic view selection. Our static view selection algorithm analyzes opacity, color and curvature images generated from different view points. We properly design the probability functions and use entropy to evaluate opacity and color distributions. Our algorithm also prefers a view which shows high curvature information. Depending on the characteristic of the data set and the opacity and color transfer function, and the nature of the application, we can design different utility functions to assign different weights to the three factors. Based on our static view selection and dynamic programming, our dynamic view selection method maximizes the information perceived from the time-varying dataset following a near-constant angular velocity path. The optimization is achieved in a polynomial time. Our results show the effectiveness of the static and dynamic view selection.

In addition to dynamic view point planning, another important parameter for animation would be lighting design. Gumhold [8] discussed light source placement for static polygonal meshes. We would like to conduct the research for lighting design for time-varying polygonal and volumetric data in our future work.

Acknowledgements

This work was supported by NSF ITR Grant ACI-0325934, NSF RI Grant CNS-0403342, DOE Early Career Principal Investigator Award DE-FG02-03ER25572, NSF Career Award CCF-0346883, and Oak

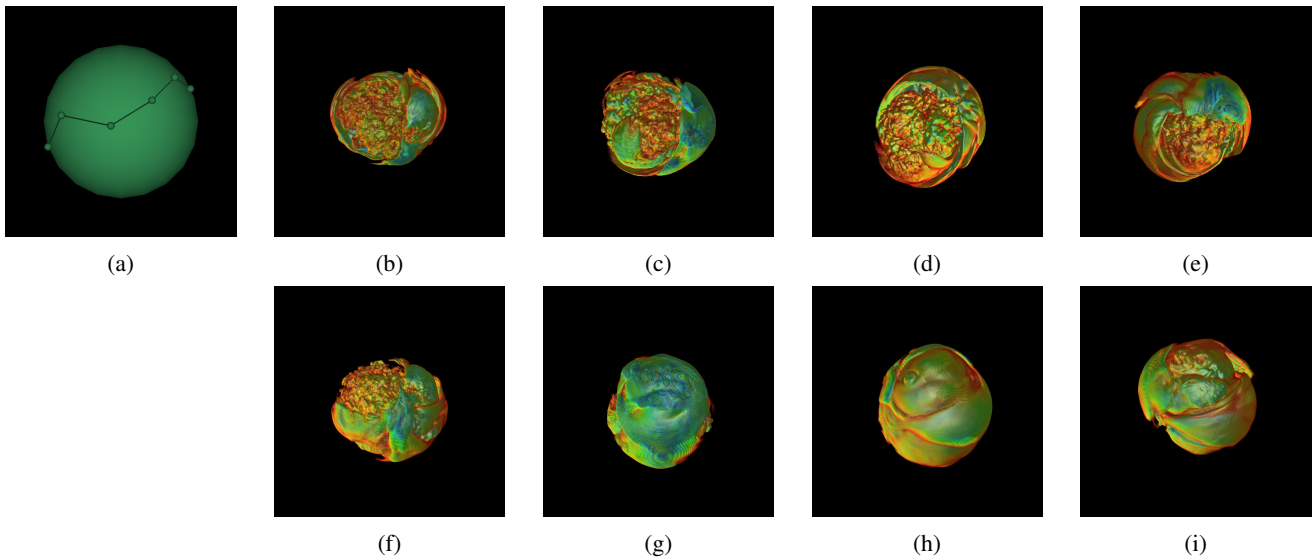


Fig. 9. The figure shows the dynamic view selection result for the TSI dataset. (a) shows the path of the time-varying view, which exhibits constant angular velocity. (b)-(e) show four snapshots captured by our time-varying view. (h)-(i) show the image from the original static view at the timestep corresponding to (b)-(e) respectively.

Ridge National Laboratory Contract 400045529. The TSI data set was provided by John M. Blondin (NCSU), Anthony Mezzacappa (ORNL), and Ross J. Toedte (ORNL). Thanks go to Kwan-Liu Ma for providing the the vortex data set made available through the NSF ITR project.

REFERENCES

[1] C. Andujar, P. Vazquez, and M. Fairen. Way-finder: Guided tours through complex walkthrough models. *Computer Graphics Forum*, 23(3):488–508, 2004.

[2] T. Arbel and F. Ferrie. Viewpoint selection by navigation through entropy maps. In *Proceeding of International Conference on Computer Vision*, pages 248–254, 1999.

[3] P. Barral, G. Dorme, and D. Plemenos. Scene understanding techniques using a virtual camera. In *Proceeding of Eurographics 2000*, 2000.

[4] U. D. Bordoloi and H.-W. Shen. View selection for volume rendering. In *IEEE Visualization Conference 2005*, pages 487–494, 2005.

[5] C. Cyr and B. Kimia. 3d object recognition using shape similarity-based aspect graph. In *Proceeding of International Conference on Computer Vision*, pages 254–261, 2001.

[6] M. D. Fairchild. *Color Appearance Models*. John Wiley and Sons, 2005.

[7] K. Gremban and K.Ikeuchi. Planning multiple observation for object recognition. *International Journal of Computer Vision*, 12(2/3):137–172, 1994.

[8] S. Gumhold. Maximum entropy light source placement. In *IEEE Visualization Conference 2002*, pages 275–282, 2002.

[9] L. Hong, S. Muraki, A. Kaufman, D. Bartz, and T. He. Virtual voyage: Interactive navigation in the human colon. *Computer Graphics*, 31:27–34, 1997.

[10] G. Ji, H.-W. Shen, and R. Wenger. Volume tracking using higher dimensional isocontouring. In *IEEE Visualization Conference 2003*, pages 209–216, 2003.

[11] A. Joshi and P. Rheingans. Illustration-inspired techniques for visualizing time-varying data. In *IEEE Visualization Conference 2005*, pages 86–93, 2005.

[12] T. Kamada and S. Kawai. A simple method for computing general position in displaying three-dimensional objects. *Proceeding of International Conference on Computer Vision*, 41(1):248–254, 1988.

[13] G. Kindlmann, R. Whitaker, T. Tasdizen, and T. Moller. Curvature-based transfer functions for direct volume rendering: Methods and applications. In *IEEE Visualization Conference 2003*, pages 513–520, 2003.

[14] J. Koenderink and A. van Doorn. The sigularities of the visual mapping. *Biological Cybernetics*, 24:51–59, 1976.

[15] J. Koenderink and A. van Doorn. The internal representation of solid shape with respect to vision. *Biological Cybernetics*, 32:211–216, 1979.

[16] C. H. Lee, A. Varshney, and D. W. Jacobs. Mesh saliency. In *Proceedings of ACM SIGGRAPH*, pages 659–666, 2005.

[17] D. Page, A. Koschan, S. Sukumar, B. Abidi, and M. Abidi. Shape analysis algorithm based on information theory. In *Proceeding of the International Conference on Image Processing*, pages 229–232, 2003.

[18] R. Samtaney, D. Silver, N. Zabusky, and J. Cao. Visualizing features and tracking their evolution. *IEEE Computer*, 27(7):20–27, 1994.

[19] C. E. Shannon. A mathematical theory of communication. In *Bell System Technical Journal*, pages 379–423 & 623–656, 1948.

[20] K. Sheomake. Animation with quaternion curves. *Computer Graphics*, 19:245–254, 1985.

[21] D. Silver and X. Wang. Volume tracking. In *IEEE Visualization Conference 1996*, pages 157–164, 1996.

[22] S. Takahashi, I. Fujishiro, Y. Takeshima, and T. Nishita. A feature-driven approach to locating optimal viewpoints for volume visualization. In *IEEE Visualization Conference 2005*, pages 495–502, 2005.

[23] J. J. van Wijk and W. A. Nuij. Smooth and efficient zooming and panning. In *IEEE Symposium on Information Visualization 2003*, pages 15–23, 2003.

[24] P.-P. Vaquez, M. Feixas, M. Sbert, and A. Llobet. Realtime automatic selection of good molecular views. *Computers and Graphics*, 30:98–110, 2006.

[25] P.-P. Vazquez, M. Feixas, M. Sbert, and W. Heidrich. Viewpoint selection using viewpoint entropy. In *Vision Modeling and Visualization Conference 2001*, pages 273–280, 2001.

[26] P.-P. Vazquez, M. Feixas, M. Sbert, and W. Heidrich. Automatic view selection using viewpoint entropy and its application to image-based modeling. *Computer Graphics Forum*, 22(4):689–700, 2003.

[27] P.-P. Vazquez and M. Sbert. Perception-based illumination information measurement and light source placement. In *Proceeding of ICCSA*, pages 306–316, 2003.

[28] J. von. Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944.

[29] E. Wernert and A. Hanson. A framework for assisted exploration with collaboration. In *IEEE Visualization Conference 1999*, pages 241–248, 1999.

[30] J. Woodring, C. Wang, and H.-W. Shen. High dimensional direct rendering of time-varying voulmes. In *IEEE Visualization Conference 2003*, pages 417–424, 2003.

LOD Map - A Visual Interface for Navigating Multiresolution Volume Visualization

Chaoli Wang, *Student Member, IEEE*, and Han-Wei Shen

Abstract—In multiresolution volume visualization, a visual representation of level-of-detail (LOD) quality is important for us to examine, compare, and validate different LOD selection algorithms. While traditional methods rely on ultimate images for quality measurement, we introduce the *LOD map* - an alternative representation of LOD quality and a visual interface for navigating multiresolution data exploration. Our measure for LOD quality is based on the formulation of entropy from information theory. The measure takes into account the distortion and contribution of multiresolution data blocks. A LOD map is generated through the mapping of key LOD ingredients to a treemap representation. The ordered treemap layout is used for relative stable update of the LOD map when the view or LOD changes. This visual interface not only indicates the quality of LODs in an intuitive way, but also provides immediate suggestions for possible LOD improvement through visually-striking features. It also allows us to compare different views and perform rendering budget control. A set of interactive techniques is proposed to make the LOD adjustment a simple and easy task. We demonstrate the effectiveness and efficiency of our approach on large scientific and medical data sets.

Index Terms—LOD map, knowledge representation, perceptual reasoning, multiresolution rendering, large volume visualization.

1 Introduction

The field of visualization is concerned with the creation of images from data. In many cases these images are observed by humans in an effort to test hypotheses and discover insights. Visualization is, therefore, an iterative and exploratory process. Human-computer interactions, such as parameter tweaking, are often involved in a bid to create better representations. This scenario works well for small data. With the advances in graphics hardware, the interactivity can be guaranteed even with a straightforward implementation. For larger data, the response time of a visualization system could become uncomfortably long. Slower responses result in an increase in time needed to obtain insights from the data. This poses a great challenge for visualization to be *effective* and *efficient* [22].

In this paper, we focus on the subject of multiresolution rendering in the context of large volume visualization. A central theme for multiresolution volume rendering is the selection of data resolution, or *level-of-detail* (LOD). Quite often, such LODs are automatically determined by user-specified error tolerances and viewing parameters [27, 10, 5]. Many of the metrics, such as mean square error (MSE) and signal-to-noise ratio (SNR), are data-centric in the sense that they measure the distortion between low and high resolution blocks in the data space (Geometric error metrics, on the other hand, are often used in surface rendering, where the geometry or mesh is known. In this paper, we do not consider this case.). Although those metrics have specific meanings and are simple to compute, they are not effective in predicting the quality of rendered images due to the lack of correlation between data and image [25]. In fact, finding the best LOD is a NP-complete optimization problem [3], so most algorithms take a greedy approximation strategy instead. In this case, data blocks are selected according to their priority values till certain constraints, such as the block budget, are met. Rarely, we have a mechanism to examine the quality of LODs from those greedy selections, and decide whether it is possible to further improve the quality through either automatic methods or user interactions.

Another important but non-trivial issue is the validation of existing LOD selection algorithms. In computer graphics and visualization, validation is usually done through side-by-side comparison of images created from different techniques. However, it may not be effective for multiresolution volume visualization. This is because direct rendering of 3D volumes to 2D images involves light attenuation, blending, and transfer function mapping, where much information about a LOD may be hidden or lost. Without the complete information, it could be insufficient to assess the improvements of new algorithms over existing ones. Moreover, a large data set is often too complex to be understood from a single image. Inspecting images from different views requires rendering large amount of data, which makes it very difficult for us to perform the validation efficiently.

From the problems mentioned above, it can be seen that there is a great need to devise techniques for multiresolution volume visualization that allow the user to examine, compare, and validate the quality of LOD selection and rendering. To fulfill this need, we present a new measure for LOD quality based on the formulation of entropy from information theory. Our quality measure takes into account the quality of each individual data block in a LOD and the relationships among them. This entropy measure allows us to map key ingredients of the LOD quality to a treemap representation [17], called the *LOD map*, for further visual analysis. The LOD map brings us a novel approach for navigating the exploration of large multiresolution data. By "navigating", we mean the user is guided with immediate visual feedback when making decisions about *where to go*, *what to do*, and *when to stop*. The user is provided with cues that leads her quickly to interesting parts of the data. She would readily know what actions to take to adjust the LOD, clearly see the gain or loss of the adjustment, and be informed when the refinement process may be stopped. Leveraging a heuristic optimization algorithm and a set of interactive techniques designed for the LOD map, making LOD adjustment and rendering budget control becomes a simple and easy task. Experimental results on large data sets demonstrate that this visual representation of LOD quality is light-weighted yet quite effective for interactive LOD comparison and validation.

2 Related Work

The past few years witnessed the rapid growth of data. Scientific simulations are producing petabytes of data as opposed to gigabytes or terabytes a couple of years ago. Building a multiresolution data hierarchy from large amount of data allows us to visualize data at different scales, and balance image quality and computation speed. Examples of hierarchical data representation for volume data include the *Laplacian pyramid* [4], multi-dimensional trees [27], and octree-based hierarchies [10]. Muraki introduced the use of wavelet transforms for vol-

• *C. Wang and H.-W. Shen are with the Department of Computer Science and Engineering, The Ohio State University, 395 Dreese Laboratories, 2015 Neil Avenue, Columbus, OH 43210.*
E-mail:{wangcha, hwshen}@cse.ohio-state.edu.

Manuscript received 31 March 2006; accepted 1 August 2006; posted online 6 November 2006.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

umetric data [16]. Westermann [26] proposed a framework for approximating the volume rendering integral using multiresolution spaces and wavelet projections. Guthe et al. [5] presented a wavelet-encoded hierarchical representation, called the *wavelet tree*, that supports interactive walkthroughs of large data sets on a single commodity PC. In this paper, we use the wavelet tree as an illustration for our presentation.

The goal of visual data exploration is to facilitate the extraction of new insights from the data. For instance, Marks et al. [14] presented the *Design Gallery* system that treats image and volume rendering as a process of exploring a multidimensional parameter space. Using an image difference metric, the system arranges renderings from various parameter settings, from which the user selects the most appealing one. Bajaj et al. [1] introduced the *contour spectrum*, a user interface component that improves qualitative user interaction and provides real-time exact qualification in the visualization of isocontours. A set of well-designed manipulation widgets for multidimensional transfer function design was given by Kniss et al. [9]. By allowing direct interaction in the spatial and transfer function domains, those widgets make finding transfer functions an intuitive and efficient process.

On the other hand, the process of visual data exploration contains a wealth of information - parameters, results, history, as well as relationships among them. To learn lessons and share experiences, the process itself can be stored, tracked, and analyzed. It can also be incorporated into, and thus becomes a part of the user interface of a visualization system. Such examples are *image graphs* [13], and interfaces with spreadsheet [6] and parallel coordinate [21] styles. A general model of the visualization exploration process was formalized by Jankun-Kelly et al. [7].

Over the last decade, a number of information visualization techniques have been developed to support the visual exploration of large and high-dimensional data sets [8]. Parallel coordinates and treemaps are two most widely-used techniques among them. The *treemap* [17] is a space-filling method for visualizing large hierarchical information. It works by recursively subdividing a given display area based on the hierarchical structure, and alternating between vertical and horizontal subdivisions. The information of an individual node is presented via visual attributes, such as color and size, of its bounding rectangle. Originally designed to visualize files on a hard drive, treemaps have been applied to a wide variety of domains [19].

One of the notable commercial applications of treemaps is the SmartMoney "Map of the Market", a simple yet powerful tool for the general public to spot investment trends and opportunities. The map shows approximately 600 publicly traded companies grouped by sector and industry. The size of each company in the map corresponds to its market capitalization. The color mapping is natural for most users to look for visually-striking features: green for gain, red for loss, and dark for neutral. A recursive algorithm was devised to reduces overall aspect ratios of the rectangles for more readable display. All these factors contributes the success of the Map of the Market. Inspired by this application, we strive for simplicity and effectiveness in search of our solution for navigating multiresolution data exploration.

3 LOD Analysis

In the visualization of multiresolution volume data, a given LOD consists of a sequence of data blocks at various resolutions. Intuitively, the LOD quality can be analyzed by investigating the quality of each individual block as well as the relationships among them: Each block may contain a different *distortion* because of the resolution and data variation. It may also convey a different optical content if a color and opacity transfer function is applied. Furthermore, the sequence of data blocks are rendered to the screen. Each block may have a different *contribution* to the image depending on its projection and how much it is occluded by other blocks. To optimize the total information received on the image, one may attempt to either adjust the LOD or change the view. The concept of *entropy* from information theory provides us a way to measure the LOD quality in a quantitative manner.

3.1 Entropy

In information theory, the entropy gives the average information or the uncertainty of a random variable. The Shannon entropy of a discrete random variable X with values in the finite set $\{a_1, a_2, ..., a_M\}$ is defined as:

$$H(X) = -\sum_{i=1}^{M} p_i \log p_i \tag{1}$$

where p_i is the probability of a variable to have the value a_i; $-\log p_i$ represents its associated information. The unit of information is call a *bit*. The logarithm is taken in base 2 or e. For continuity, variable of probability zero does not contribute to the entropy, i.e., $0 \log 0 = 0$. The entropy achieves its maximum of $\log M$ when the probability distribution is uniform.

To evaluate the quality of a LOD, we first define the probability of a multiresolution data block b_i as:

$$p_i = \frac{\mathcal{C}_i \cdot \mathcal{D}_i}{S}, \qquad \text{where } S = \sum_{i=1}^{M} \mathcal{C}_i \cdot \mathcal{D}_i \tag{2}$$

where $\mathcal{C}_i$ and $\mathcal{D}_i$ are the contribution and distortion of b_i respectively (more details about these two terms are given in Section 3.2); M is the total number of blocks in the multiresolution data hierarchy. The summation S is taken over all data blocks, and the division by S is required to make all probabilities add up to unity. Eqn. 2 states that the probability of a data block in a LOD is high if it has large distortion and high contribution. The entropy of a LOD then follows the definition in Eqn. 1.

Note that for any LOD, it is impossible for all the data blocks in the hierarchy to have the equal probability of $1/M$, i.e., a LOD could not achieve an entropy value of $\log M$. This is because in any LOD, if a parent block is selected, then none of its descendant blocks will be selected. Any block which is not included in the LOD receives zero probability and does not contribute to the entropy. Ideally, since a higher entropy indicates a better LOD quality, the best LOD (with the highest information content) could be achieved when we select all the leaf nodes in the data hierarchy. However, this requires rendering the volume data at the original resolution, and defeats the purpose of multiresolution rendering.

In practice, a meaningful goal is to find the best LOD under some rendering budget, such as a certain block budget, which is usually much smaller than M. Accordingly, the quality of a LOD could be improved by splitting data blocks with large distortion and high contribution, and joining those blocks with small distortion and low contribution. The split operation aims at increasing the entropy with a more balanced probability distribution. The join operation is to offset the increase in block number and keep it under the budget. In addition, adjusting the view could improve the quality of LOD too, since the contributions of data blocks vary when the view changes.

3.2 Distortion and Contribution

In a multiresolution data hierarchy such as a wavelet tree, a block having larger distortion or variation is more likely to receive a higher priority for LOD refinement. The distortion of a multiresolution data block captures this intrinsic property. Let us first consider two data blocks b_i and b_j, where b_j is an immediate child of b_i. We define the distortion between b_i and b_j as follows:

$$d_{ij} = \sigma_{ij} \cdot \frac{\mu_i^2 + \mu_j^2 + C_1}{2\mu_i\mu_j + C_1} \cdot \frac{\sigma_i^2 + \sigma_j^2 + C_2}{2\sigma_i\sigma_j + C_2} \tag{3}$$

where σ_{ij} is the covariance between b_i and b_j; μ_i and μ_j are the mean values of b_i and b_j respectively; σ_i and σ_j are their standard deviations. We include small constants C_1 and C_2 to avoid instability when $\mu_i\mu_j$ and $\sigma_i\sigma_j$ are very close to zero.

Eqn. 3 consists of three parts, namely, *covariance*, *luminance distortion*, and *contrast distortion*. Collectively, these three parts capture the distortion between the two blocks. The luminance distortion and

contrast distortion are originally from the image quality assessment literature [25], and have been shown to be consistent with the luminance masking and contrast masking features in the human visual system respectively.

In a wavelet tree, a parent node has eight immediate children. Thus, we add distortions between the parent block and its eight child blocks. We also take into account the maximum distortion of the child blocks, as an approximation of the distortion between the parent block and the original full-resolution data it represents. Written in equation:

$$\mathcal{D}_i = \sum_{j=0}^{7} d_{ij} + \max\{\mathcal{D}_j|_{j=0}^{7}\} \tag{4}$$

where $\mathcal{D}_i$ and $\mathcal{D}_j$ are the distortions of blocks b_i and b_j respectively. As a special case, if b_i is associated with a leaf node in the hierarchy, we define $\mathcal{D}_i = C_3$, where C_3 is a small constant. The distortion for each tree node can be calculated as we build up the multiresolution data hierarchy. They are then normalized for our use.

To evaluate the contribution of a multiresolution block b_i to the image, we treat the coarse-grained data block as a whole and approximate its contribution as follows:

$$\mathcal{C}_i = \mu \cdot t \cdot a \cdot \nu \tag{5}$$

where μ is the mean value of b_i; t is its thickness (the average length of the viewing ray segment inside b_i); a is the screen projection area of the block, and ν is its estimated visibility. Here, similar to the well-known composition equation [12], $(\mu \cdot t \cdot a)$ approximates the emission of b_i, and ν accounts for the occlusion.

Depending on our need, μ, σ, and σ_{ij} in Eqn. 3 and 5 can be evaluated directly in the scalar data space, or in the perceptually-adapted CIELUV color space (see [23] for more detail). In Section 7.2, we will describe our pre-computation and real-time update techniques for calculating $\mathcal{D}$ and $\mathcal{C}$ of multiresolution data blocks, which ensure a quick update of the entropy value for a LOD.

3.3 Heuristic Algorithm

Based on the entropy measure, we propose a three-stage heuristic algorithm to adjust a LOD automatically. The given LOD could come from any LOD selection methods. The motivation is to balance the probability distribution among all data blocks in the LOD, and improve its entropy. Our three-stage algorithm is as follows:

1. **Join**: Locate the data block b_l with the lowest probability. Check if joining b_l with its siblings would decrease the entropy or not. If not, join b_l with its siblings. Otherwise, mark b_l out. This process then repeats on all unmarked data blocks until all blocks are scanned. The first stage would potentially free some block budget for use in the second stage.

2. **Split**: Locate the data block b_h with the highest probability. Check if splitting b_h would increase the entropy or not. If yes, split b_h. Otherwise, mark b_h out. This process then repeats on all unmarked data blocks until either the block budget is met or all blocks are scanned.

3. **Join-Split**: Consider a pair of data blocks (b_l, b_h) in the LOD, where b_l is the block with the lowest probability, and b_h is the block with the highest probability. Check if joining b_l with its siblings and splitting b_h would increase the entropy or not. If yes, join b_l with its siblings and split b_h. Otherwise, mark the pair (b_l, b_h) out. This process then repeats on all unmarked pairs until all pairs are scanned.

Note that join or split operations in our algorithm will only increase the entropy, if possible, but never decrease. This monotonic condition guarantees the convergence of the algorithm. Our heuristic algorithm could also be used to generate a balanced LOD (in terms of probability distribution) given the constraint of block budget. The algorithm refines the LOD starting from the root of the data hierarchy until the block budget is met.

4 LOD MAP

Although various efforts have been made to choose proper LODs that condense data and preserve features [28, 10, 5], little work has been done to represent and validate this decision-making process. Actually, the information derived for data selection is *knowledge* that guides the user through the entire solution space. Such knowledge saves time on decision and computation, which the user would have otherwise spent on a trial-and-error search. The time saved may be used to improve frame rates or alternatively, on other techniques to better understand the data.

The formulation of LOD quality in Section 3 tells us that a LOD is good if it has a high entropy value. Thus, a good LOD not only indicates a good quality of rendered images, but also a balanced probability distribution for all the data blocks in the LOD. This includes information of both what we can perceive (data blocks not occluded) and can not (data blocks occluded) from the rendered image. Therefore, a suitable visual representation for LOD quality should reveal not only what is visible, but also what is not. Such information is important because it can help us answer questions such as "do we waste the budget on those blocks which are occluded?", or, "can we generate an image of similar quality with a reduced block budget?". In order to avoid possible occlusion, we rule out the option of any 3D representation of LODs. Actually, a 2D treemap provides a simple yet effective representation of large hierarchical structure. The key issues involved in this knowledge representation are information mapping and layout design.

4.1 Information Mapping

In this paper, we call the treemap representation of LOD the *LOD map*. As a treemap, the LOD map is formed by recursively subdividing a given area. Each data block in the LOD is represented as a rectangle in the LOD map. Treemaps can effectively visualize data with two attributes, in which one attribute is typically mapped to the size of the rectangles, and the other attribute to the color of the rectangles. The critical question to ask is what information should be displayed in the generated rectangles for the LOD map.

Since the treemaps are commonly used for representing hierarchical structures. A first thought along this direction could lead us to use the size of rectangle to encode the resolution of different data blocks in the LOD map. As in [24], the size of the rectangle can be determined by the level of the data hierarchy the data block lies on. Although this way of representing LODs is natural, the mapping only yields limited usefulness. This is because which level of the hierarchy a data block lies on may not give hints on its actual quality. A data block at a low level (low resolution) may contain empty space or have a uniform value. In contrast, another data block at a high level (high resolution) may still exhibit much variation or distortion. Therefore, this kind of coding does not grant insights, and wastes important channels which could be used to encode more essential LOD information.

The entropy characterizes the quality of a LOD. To reveal this key information, we map its ingredients, i.e., the distortion $\mathcal{D}$ and contribution $\mathcal{C}$ of a data block in the LOD, to the color and size of its bounding rectangle in the LOD map. More specifically:

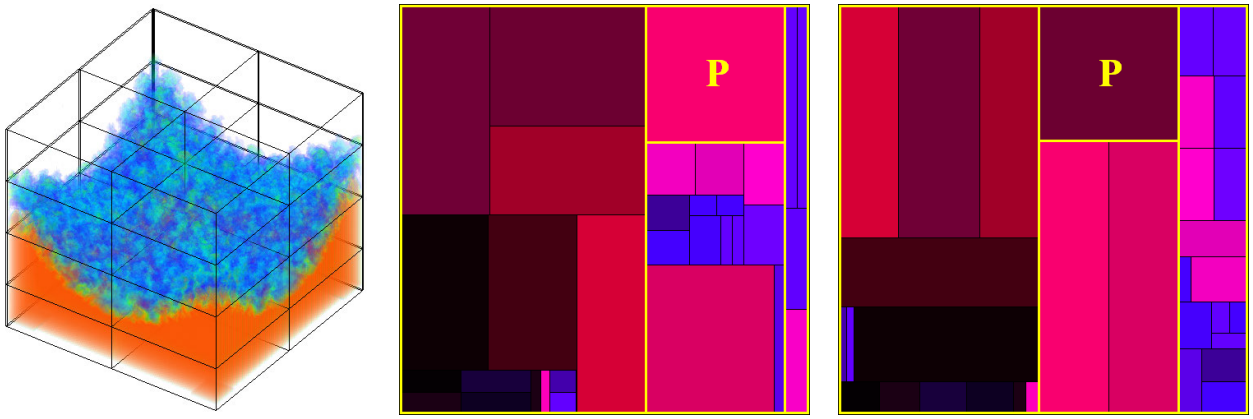
- The color assignment is based on the distortion $\mathcal{D}$: red for large distortion, magenta for medium, and blue for small.

- The contribution $\mathcal{C}$ is split into two parts: $(\mu \cdot t \cdot a)$ is mapped to the size of the rectangle, while ν is assigned to its opacity.

This color and size coding scheme makes it easy for the user to look for "hot spots" - data blocks with large distortion and high contribution in the LOD map. The motivation for separating visibility ν from $\mathcal{C}$ is intuitive too: more visible blocks in the LOD should appear brighter in the LOD map, and less visible ones darker.

4.2 Layout Design

Another key issue for the generation of LOD map is layout design. In the early 1990s when treemaps were first prototyped, a straightforward



(a) entropy = 0.238 (b) pivot-by-size (c) pivot-by-middle

Fig. 1. (a) shows a rendering of the RMI data set at low resolution with 36 blocks. (b) and (c) are its two LOD map layouts. In each layout, the "P" indicates the first pivot rectangle, and the yellow boundaries show the first subdivision. The normalized entropy is low because only a small number of data blocks are rendered.

slice-and-dice algorithm was used to generate the treemap layout. A problem with this standard method is that it often produces rectangles with high aspect ratios. As a result, such skinny rectangles can be difficult to see, compare, and select. Over the years, several algorithms have been proposed to improve the aspect ratios of rectangles in the treemap [18]. However, they introduce instability over time in the display of dynamically changing data, and fail to preserve an ordering of the underlying data. These drawbacks are critical in our LOD map representation because:

- The information of a LOD keeps changing whenever the user makes LOD adjustment or changes the view. If the LOD map layout has dramatic changes that causes unattractive flickering, it is hard to select and track data blocks in the LOD map.
- A LOD often comes with the back-to-front viewing order, in which neighboring blocks are close to each other in the LOD. Preserving this order in the LOD map layout is important for locating neighboring blocks and observing group patterns.

The *ordered treemap* introduced by Shneiderman and Wattenberg [18] uses layout algorithms that offer a good tradeoff among stable updates, order preserving, and low aspect ratios. In this paper, we adopt this layout design for our LOD map representation. The layout algorithm is similar to the idea of finding a 2D analogue of the `QuickSort` algorithm. The inputs are a rectangle R to be subdivided and a list of items that are ordered by an index and have given areas. The crux of the algorithm is to choose a special item, the *pivot*, which is placed at the side of R. The remaining items in the list are then assigned to three large rectangles that make up the rest of the display area. Finally, the algorithm is applied recursively to each of these rectangles. See [18] for a more detailed description of the algorithm. Variations of the algorithm are the choices of pivot. The pivot could be the item with the largest area in the list (*pivot-by-size*), or the middle item of the list (*pivot-by-middle*). In Section 7.1, we will discuss two scenarios where each of these choices of pivot should be used for smooth update.

4.3 Put It All Together

Having discussed how to define and represent the quality of LODs, it is time to put it all together and show how our scheme can be used. We experimented our idea with the Richtmyer-Meshkov Instability (RMI) data set [15]. The 7.5GB RMI data set has the spatial dimension of $2048 \times 2048 \times 1920$, from which we built a wavelet tree with a tree depth of six. Fig. 1 shows a LOD rendering of the RMI data set at the third tree level and its LOD map layouts. The viewing order is roughly preserved, as we can see that darker/brighter rectangles (data blocks with lower/higher visibility) are close to each other in the LOD map. Since a LOD map encodes the entropy information of a LOD, if there is an unbalanced probability distribution among data blocks in the LOD, we can easily perceive this through color and size attributes of rectangles in its LOD map. Then, as demonstrated in Fig. 1 (b) or (c), several directions for optimizing the LOD quality can be immediately followed:

- Look for "hot spots" - large rectangles with bright red colors. They are highly-visible data blocks that have high contribution and large distortion. Splitting these blocks will most likely increase the entropy and improve the LOD quality.
- Small blue rectangles are data blocks that have low contribution and small distortion. If they cluster in a local region, joining these blocks may save the block budget without decreasing the entropy.
- Dark rectangles are data blocks with the lowest visibility. If many of them cluster together, joining these blocks may also save the block budget without sacrificing the LOD quality.

The next section introduces a set of interactive techniques that allows the user to perform LOD adjustment efficiently.

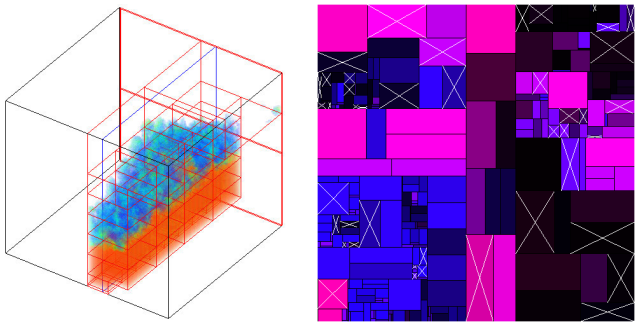
5 Interactive Techniques

In computer graphics and visualization, *brushing* has been used as a method for selecting subsets of data for further operations, such as highlighting, deleting, or analysis. In our case, brushing is used to select a subset of data blocks from a LOD for join or split operations. We provide brushing in both views: the rendering window and the LOD map. The result of selection is highlighted in both views, as they are linked together and updated dynamically whenever one of the views changes. This technique helps the user detect correspondences between the two different visual representations. We allow the user to perform brushing in the following ways:

- Direct brushing in the LOD map by clicking a rectangle, or specifying a rectangular region to select multiple rectangles simultaneously via mouse.
- Brushing in the rendering window by specifying the brush coverage as 1D point, 2D plane, or 3D box in the data space via slider.
- Brushing some parameter (such as visibility ν) or combination of parameters by specifying its range via slider.

Direct transformation is provided for interactivity and examination of local details. The user can translate, scale, and rotate the 3D volume in

the rendering window, and translate and scale the 2D LOD map. Data blocks selected in the LOD are highlighted with red boundaries and white crosses in the rendering window and the LOD map respectively. The user then proceeds to join or split these blocks. A *join* operation merges a selected block with its siblings into its parent block. A *split* operation breaks a selected block into its eight child blocks. For multiple selected data blocks, they are put into a queue and processed in sequence. We also provide the "undo" function so that the user can roll back to the previous LOD if the operation just performed is not desired.



(a) brushing with 2D plane (b) LOD map

Fig. 2. Brushing the RMI data set by specifying a 2D cutting plane. (a) shows the rendering of data selection and (b) is the corresponding LOD map.

Fig. 2 gives an example of brushing manipulation with a 2D cutting plane. Only the data blocks intersecting the plane (drawn in blue) are selected and rendered. In the LOD map, the selected data blocks are highlighted with white crosses.

data set (type)	RMI (byte)	VisWoman (short)
volume dimension	$2048 \times 2048 \times 1920$	$512 \times 512 \times 1728$
block dimension	$128 \times 128 \times 64$	$32 \times 32 \times 64$
volume (block) size	7.5GB (1MB)	864MB (128KB)
# non-empty blocks	10499	9446
compression ratio	5.60 : 1	2.37 : 1

Table 1. The RMI and VisWoman data sets.

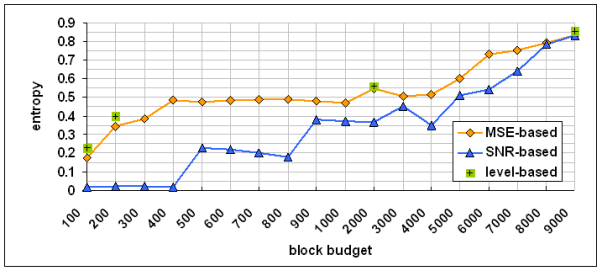


Fig. 4. LOD quality comparison on the RMI data set with the same fixed view as in Fig. 1 (a). The plot shows how the entropy changes with different block budgets for three LOD selection methods. There are a total of 10499 non-empty blocks in the data hierarchy.

6 Results

As listed in Table 1, we experimented with our LOD map on two data sets: the RMI data set (also mentioned in Section 4.3) from scientific simulation, and the visible woman (VisWoman) data set from medical application. For both data sets, the Haar wavelet transform with a lifting scheme was used to construct the wavelet tree data hierarchy. A *lossless* compression scheme was used with the threshold set to zero

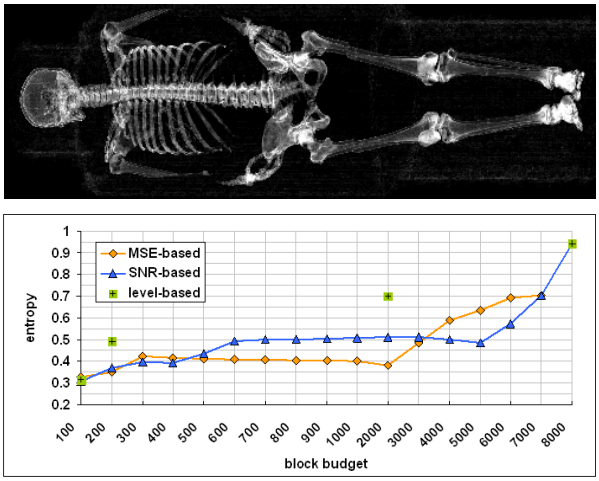


Fig. 5. LOD quality comparison on the VisWoman data set with the fixed front view shown above. The plot shows how the entropy changes with different block budgets for three LOD selection methods. There are a total of 9446 non-empty blocks in the data hierarchy.

to compress the wavelet coefficients. To ensure seamless rendering, we extended one voxel overlapping boundaries between neighboring blocks in each dimension when breaking the original volume data into blocks. As a result, both hierarchies have a tree depth of six. All tests were performed on a 3.0GHz Intel Xeon processor with 3GB main memory, and an nVidia GeForce 7800 GT graphics card with 256MB video memory.

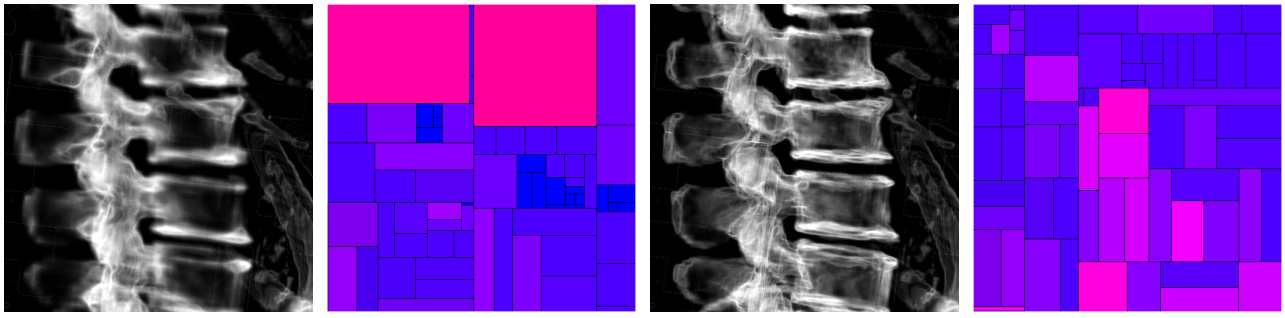
6.1 LOD Comparison

For LOD comparison, we took three commonly-used LOD selection methods, i.e., level-based, MSE-based, and SNR-based, for our test. The level-based method selects a particular resolution level in the multiresolution data hierarchy. For the MSE-based (SNR-based) method, we specify the MSE (SNR) error tolerance to determine the LOD (we followed Eqn. 4 where d_{ij} is the MSE (SNR) of blocks b_i and b_j). Fig. 3 gives an example where we compared the LOD quality of the MSE-based and level-based methods under the same block budget. Clearly, we can see that Fig. 3 (b) contains two "hot spots" (large rectangles with bright red colors) and some small blue rectangles clustered together. These are indications of bad distribution of data resolution. On the contrary, Fig. 3 (d) exhibits a much better distribution. It follows that the level-based method achieves a better LOD quality than the MSE-based one. Their entropy values and rendered images in Fig. 3 (a) and (c) also confirm this.

Fig. 4 and 5 show the change of entropy values on the two data sets under a series of block budgets. For the level-based method, the block budget increases eight folds when the level increases by one. Therefore, isolated data points rather than polylines are illustrated in both figures. For both data sets, we can see that the MSE-based and SNR-based LOD methods could not outperform the straightforward level-based method in terms of the tradeoff between LOD quality and block budget, although the level-based method does not allow flexible block budgets. For the RMI data set, the MSE-based method performs consistently better than the SNR-based one. However, this is not the case for the VisWoman data set, as the two polylines intertwine with each other in Fig. 5. Another finding is that the entropy does not always increase (actually decreases sometimes) with the increase of block budget. This does not indicate the deterioration in rendered image quality, but rather, a less balanced distribution of probability among all the data blocks. In fact, we can imagine this potentially leaves room for us to improve the LOD quality (see Section 4.3) using the LOD map.

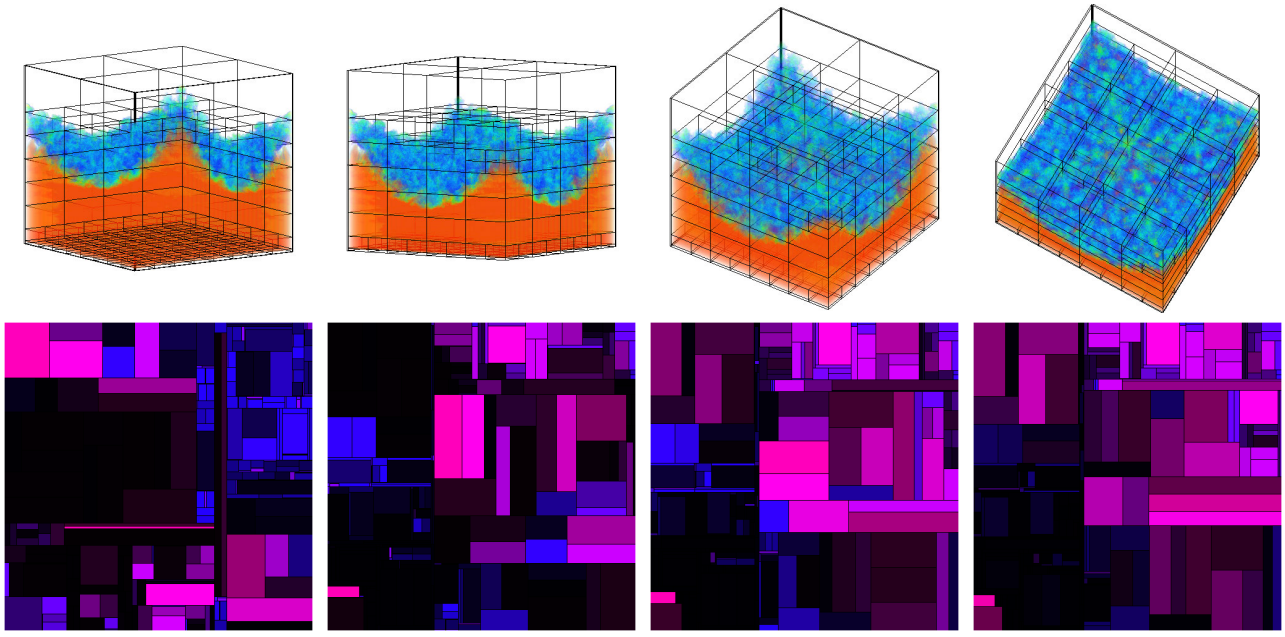
6.2 View Comparison

The quality of a given LOD can be improved by adjusting the view. Similar to the ideas of view selection presented in [2, 20], the entropy



(a) MSE-based, 67 blocks (b) entropy = 0.163 (c) level-based, 67 blocks (d) entropy = 0.381

Fig. 3. LOD comparison (fixed view, different LODs). A zoom to part of the spine of the VisWoman data set. (a) and (c) show the rendering of two different LODs with the same block budget. (b) and (d) are the LOD maps for (a) and (c) respectively.



(a) entropy = 0.330 (b) entropy = 0.343 (c) entropy = 0.384 (d) entropy = 0.390

Fig. 6. View comparison (fixed LOD, different views). Four snapshots from a rendering sequence of the RMI data set. The LOD remains the same with 246 blocks while the entropy increases from (a) to (d). The pivot-by-middle layout is used to maintain relative stable update of the LOD map throughout the rendering sequence. The dramatic change in the LOD map layout from (a) to (b) is due to the abrupt change of the viewing order.

and the LOD map can help us find good views for a certain LOD. Fig. 6 gives an example where we compare different views for a fixed LOD of the RMI data set. The entropy increases as more information content is received from Fig. 6 (a) to (d). Looking at the sequence of LOD maps, we can observe that the visibility of the LOD gets improved since the entire LOD map is getting brighter. Another finding from the LOD map sequence is that smaller blocks at the bottom of the volume corresponds to blue rectangles (small distortion) at the upper-right part of Fig. 6 (a), and larger blocks at the top of the volume corresponds to red rectangles (large distortion) in the LOD maps. The relative stable update of the LOD map allows us to detect such correspondences between the rendered blocks and the rectangles. This information is useful when it comes to LOD adjustment.

6.3 LOD Adjustment

The goal of LOD adjustment is to improve the quality of LOD. By splitting data blocks with large distortion and high contribution, and joining data blocks with small distortion and low contribution, we achieve a more balanced probability distribution among all data blocks, and therefore, increase the entropy of the LOD. Fig. 7 show two examples of LOD adjustment on the RMI and VisWoman data sets within fixed block budgets. For the RMI data set, we clearly see some "hot spots" (large rectangles with bright red colors) and a cluster of dark rectangles (occluded data blocks) in the LOD map of Fig. 7 (a). In this example, we used the heuristic algorithm (Section 3.3) to optimize the LOD automatically. The LOD map after adjustment in Fig. 7 (b) shows a more balanced result. For the VisWoman data set, hot spots are popping out in the LOD map of Fig. 7 (c). Although there are no dark rectangles, many small blue rectangles cluster at the lower-left corner of the LOD map. The LOD quality is improved by splitting those hot spots and joining small blue rectangles. With the assist of a set of brushing techniques (Section 5), selecting such target rectangles in the LOD map and performing LOD adjustment becomes a simple and efficient task.

6.4 Budget Control

As demonstrated in Fig. 4 and 5, the commonly-used MSE-based and SNR-based LOD selection methods do not perform well (in terms of improving the quality of LOD) with the increase of block budget. This gives us opportunities to control the block budget. That is, if such a LOD selection algorithm could not optimize the quality of LOD under a certain block budget, could we instead give a LOD of similar quality but with a reduced block budget? Generating images of similar quality with smaller budgets is appealing for large data visualization because

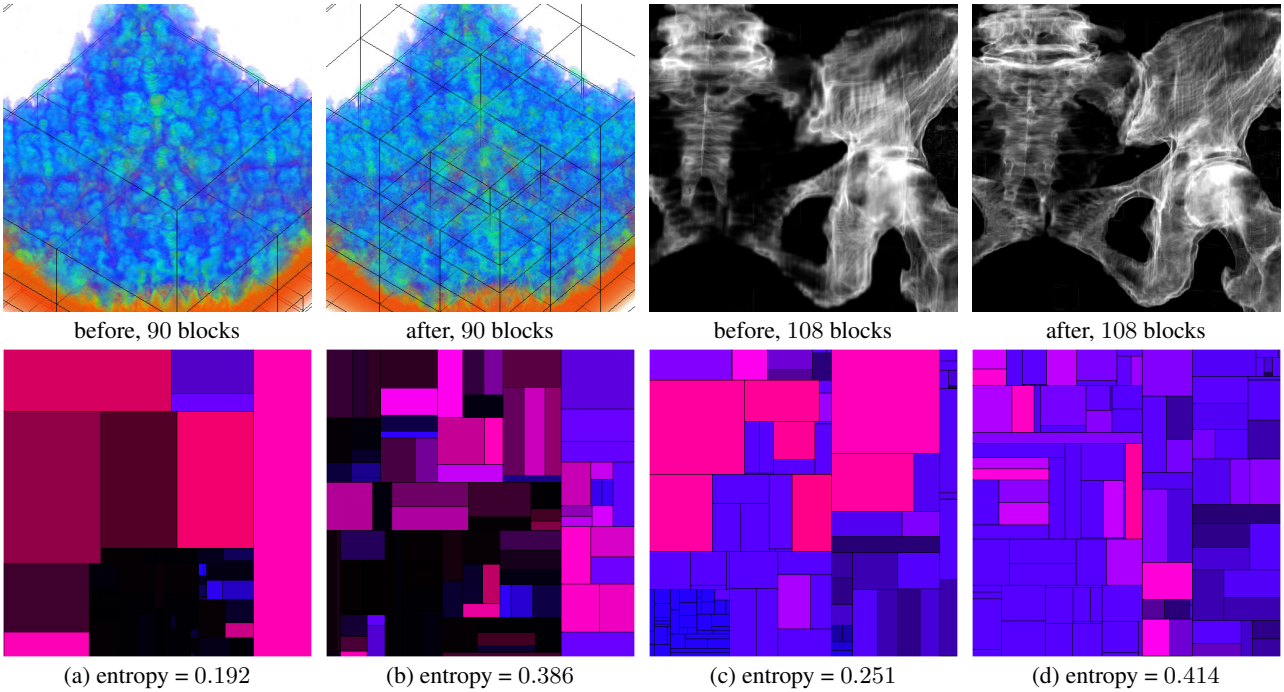


Fig. 7. LOD adjustment. A zoom of the RMI data set: (a) shows a given LOD based on the MSE, and (b) shows the result after the LOD adjustment. A zoom to the left pelvis of the VisWoman data set: (c) shows a given LOD based on the SNR, and (d) shows the result after the LOD adjustment. The pivot-by-size layout is used to maintain relative stable update of the LOD map during the adjustment process.

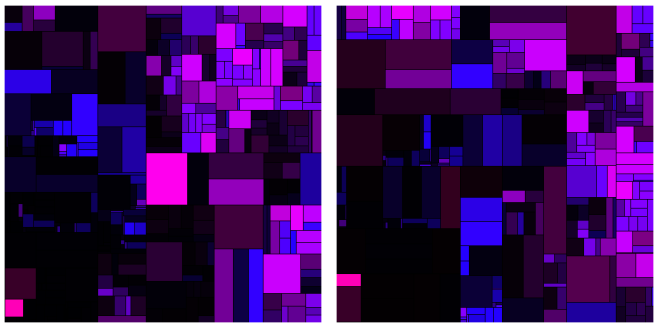


Fig. 8. Budget control on the RMI data set. A LOD based on the MSE is given, where the view is the same as in Fig. 1 (a). (a) and (b) are the LOD maps before and after the budget control. The percentage of blocks with visibility less than 0.20 decreases from 48% in (a) to 29% in (b).

it could save us limited resources. Fig. 8 shows an example of budget control on the RMI data set. By joining data blocks with low visibility (dark rectangles) and improving the overall distribution of resolution, we reduce the block budget by 25% while increasing the entropy of the LOD. This budget control does not affect the quality of rendered image, as the renderings before and after the budget control are almost identical.

7 Discussion

As a reflection of what we state at the beginning of the paper, we discuss the effectiveness and efficiency of our LOD map approach to multiresolution volume visualization.

7.1 Effectiveness

Using entropy, our LOD quality measure becomes a *global* quality index, i.e., it not only indicates the quality of rendered images, but also the probability distribution of all multiresolution data blocks in the LOD. The probability takes into account the distortion and contribution of data blocks. Therefore, a high entropy (good distribution) implies a balanced distribution of resources (e.g. data resolution), which is not captured by traditional LOD selection methods. Through the mapping of key LOD ingredients to a treemap representation, we create the LOD map as a visual interface for LOD comparison and validation. The LOD map is an intuitive representation of LOD quality, since key ingredients of a LOD are mapped to pre-attentive attributes (color and size of rectangles) in the LOD map. Integrating a heuristic algorithm and a set of interactive techniques, it also serves as a convenient user interface for visual data exploration. Note that the LOD map can be manipulated independently, and the actual rendering of data could be deferred until a desired LOD of good quality is achieved. The results in Section 6 demonstrate that this visual interface is light-weighted and quite effective for multiresolution volume visualization.

For the LOD map, we utilize both pivot-by-size and pivot-by-middle layouts for relative stable update when the view or LOD changes. The choice of which layout should be used depends on which layout is more likely to keep the pivot unchanged. For example, if the view changes, the middle item in the LOD list is more likely to remain the same, as opposed to the item with the largest area. Therefore, the pivot-by-middle layout should be used. On the other hand, if a LOD undergoes any join or split operations, then the pivot-by-size layout should be used, since the item with the largest area in the LOD list is more likely to keep unchanged rather than the middle item. This simple rule proves very effective in maintaining smooth update of the LOD map.

7.2 Efficiency

In order to generate the LOD map in real time, we need quick update of the distortion $\mathcal{D}$ and contribution $\mathcal{C}$ for multiresolution data blocks. In this paper, we calculate $\mathcal{D}$ in the roughly perceptually-uniform CIELUV color space (note that in this case, $\mathcal{D}$ depends on the input color and opacity transfer function). Similar to error calculation in [11], we pre-compute and store summary tables to ensure real-time update of the distortion $\mathcal{D}$. The summary tables are the histogram table (frequency of quantized scalar values) and correspondence table

(frequency of pairs of quantized scalar values in the parent and its child blocks) for each data block in the multiresolution data hierarchy. A zigzag run-length encoding scheme is used to reduce the storage overhead and facilitate the table lookup at run time. During the rendering, we can recompute the distortion within seconds for large data sets (such as the 7.5GB RMI data set) whenever the transfer function changes.

On the other hand, the quick update of the contribution $\mathcal{C}$ requires real-time estimate of the visibility ν (Eqn. 5). Here, the essential question is how to calculate the visibility of many data blocks quickly, given an input occlusion map. Our solution is based on summed area tables (SATs), which take the occlusion map as the input. We utilize the GPU to estimate the average visibility. The GPU implementation builds the SATs in passes with the support of framebuffer objects (`GL_EXT_framebuffer_object`). Getting the average visibility for a block is performed by a fragment program that looks up four corners of its projection in the SATs. In this way, the visibility for all the data blocks can be evaluated interactively at run time. For instance, it only takes around 0.2 second to update the visibility of thousands of data blocks for the RMI data set. We refer readers to [23] for the algorithm, implementation, and performance of our summary table scheme and GPU-based visibility estimation.

The efficiency of our LOD map is thus ensured with a real-time update of the distortion $\mathcal{D}$ and contribution $\mathcal{C}$. Our experiments show that at run time when we change the view or perform LOD adjustment, the entropy and the LOD map can be updated interactively for a LOD consisting of up to thousands of data blocks (a typical magnitude for our multiresolution data hierarchies).

8 Conclusion and Future Work

We have presented a new LOD quality measure using entropy and its visual representation using the LOD map for multiresolution volume visualization. Leveraging this quality measure and visual interface, it becomes not only feasible, but also effective and efficient for us to examine, compare, and validate the quality of LOD selection and rendering. We believe that this technique could be generalized, and applied to explore other solution spaces that exhibit similar properties as the LOD optimization problem.

The LOD map could carry more customized information for a LOD. For example, if the user wants to inspect how the transfer function contents of data blocks vary from their scalar field contents, we can add shading to the rectangles in the LOD map to indicate this. Texture could also be applied to the LOD map to represent some other information of interest. User study along this direction, such as investigating how many channels the user can perceive easily in the LOD map without information overload, is necessary. In the future, we also would like to extend our approach to multiresolution visualization of large-scale time-varying data.

Acknowledgements

This work was supported by NSF ITR Grant ACI-0325934, NSF RI Grant CNS-0403342, DOE Early Career Principal Investigator Award DE-FG02-03ER25572, NSF Career Award CCF-0346883, and Oak Ridge National Laboratory Contract 400045529. The RMI data set is courtesy of Mark A. Duchaineau at Lawrence Livermore National Laboratory, and the VisWoman data set is courtesy of The National Library of Medicine. Special thanks to Carrie Casto for story narration and Hana Kang for voice editing of the accompanying video. Finally, the authors would like to thank the anonymous reviewers for their helpful comments.

References

[1] C. L. Bajaj, V. Pascucci, and D. R. Schikore. The Contour Spectrum. In *Proc. of IEEE Visualization '97*, pages 167–173, 1997.

[2] U. D. Bordoloi and H.-W. Shen. View Selection for Volume Rendering. In *Proc. of IEEE Visualization '05*, pages 487–494, 2005.

[3] T. A. Funkhouser and C. H. Séquin. Adaptive Display Algorithm for Interactive Frame Rates During Visualization of Complex Virtual Environments. In *Proc. of ACM SIGGRAPH '93*, pages 247–254, 1993.

[4] M. H. Ghavamnia and X. D. Yang. Direct Rendering of Laplacian Pyramid Compressed Volume Data. In *Proc. of IEEE Visualization '95*, pages 192–199, 1995.

[5] S. Guthe, M. Wand, J. Gonser, and W. Straßer. Interactive Rendering of Large Volume Data Sets. In *Proc. of IEEE Visualization '02*, pages 53–60, 2002.

[6] T. J. Jankun-Kelly and K.-L. Ma. Visualization Exploration and Encapsulation via a Spreadsheet-Like Interface. *IEEE Trans. on Visualization & Computer Graphics*, 7(3):275–287, 2001.

[7] T. J. Jankun-Kelly, K.-L. Ma, and M. Gertz. A Model for the Visualization Exploration Process. In *Proc. of IEEE Visualization '02*, pages 323–330, 2002.

[8] D. A. Keim. Information Visualization and Visual Data Mining. *IEEE Trans. on Visualization & Computer Graphics*, 7(1):100–107, 2001.

[9] J. Kniss, G. Kindlmann, and C. D. Hansen. Interactive Volume Rendering Using Multi-Dimensional Transfer Functions and Direct Manipulation Widgets. In *Proc. of IEEE Visualization '01*, pages 255–262, 2001.

[10] E. LaMar, B. Hamann, and K. I. Joy. Multiresolution Techniques for Interactive Texture-Based Volume Visualization. In *Proc. of IEEE Visualization '99*, pages 355–362, 1999.

[11] E. LaMar, B. Hamann, and K. I. Joy. Efficient Error Calculation for Multiresolution Texture-Based Volume Visualization. In *Hierarchical & Geometrical Methods in Scientific Visualization*, pages 51–62, 2003.

[12] M. Levoy. Efficient Ray Tracing of Volume Data. *ACM Trans. on Graphics*, 9(3):245–261, 1990.

[13] K.-L. Ma. Image Graphs - A Novel Approach to Visual Data Exploration. In *Proc. of IEEE Visualization '99*, pages 81–88, 1999.

[14] J. Marks, B. Andalman, P. A. Beardsley, W. Freeman, S. Gibson, J. Hodgins, T. Kang, B. Mirtich, H. Pfister, W. Ruml, K. Ryall, J. Seims, and S. Shieber. Design Galleries: A General Approach to Setting Parameters for Computer Graphics and Animation. In *Proc. of ACM SIGGRAPH '97*, pages 389–400, 1997.

[15] A. A. Mirin, R. H. Cohen, B. C. Curtis, W. P. Dannevik, A. M. Dimits, M. A. Duchaineau, D. E. Eliason, D. R. Schikore, S. E. Anderson, D. H. Porter, P. R. Woodward, L. J. Shieh, and S. W. White. Very High Resolution Simulation of Compressible Turbulence on the IBM-SP System. In *Proc. of ACM/IEEE Supercomputing '99*, 1999.

[16] S. Muraki. Approximation and Rendering of Volume Data Using Wavelet Transforms. In *Proc. of IEEE Visualization '92*, pages 21–28, 1992.

[17] B. Shneiderman. Tree Visualization with Tree-Maps: A 2D Space-Filling Approach. *ACM Trans. on Graphics*, 11(1):92–99, 1992.

[18] B. Shneiderman and M. Wattenberg. Ordered Treemap Layouts. In *Proc. of IEEE Information Visualization '01*, pages 73–78, 2001.

[19] Shneiderman, B. Treemaps for Space-Constrained Visualization of Hierarchies (http://www.cs.umd.edu/hcil/treemap-history/).

[20] S. Takahashi, I. Fujishiro, Y. Takeshima, and T. Nishita. A Feature-Driven Approach to Locating Optimal Viewpoints for Volume Visualization. In *Proc. of IEEE Visualization '05*, pages 495–502, 2005.

[21] M. Tory, S. Potts, and T. Möller. A Parallel Coordinates Style Interface for Exploratory Volume Visualization. *IEEE Trans. on Visualization & Computer Graphics*, 11(1):71–80, 2005.

[22] J. J. van Wijk. The Value of Visualization. In *Proc. of IEEE Visualization '05*, pages 79–86, 2005.

[23] C. Wang, A. Garcia, and H.-W. Shen. Interactive Level-of-Detail Selection Using Image-Based Quality Metric for Large Volume Visualization. *IEEE Trans. on Visualization & Computer Graphics*. Accepted for publication, 2006.

[24] C. Wang and H.-W. Shen. Hierarchical Navigation Interface: Leveraging Multiple Coordinated Views for Level-of-Detail Multiresolution Volume Rendering of Large Scientific Data Sets. In *Proc. of Information Visualisation '05*, pages 259–267, 2005.

[25] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Trans. on Image Processing*, 13(4):600–612, 2004.

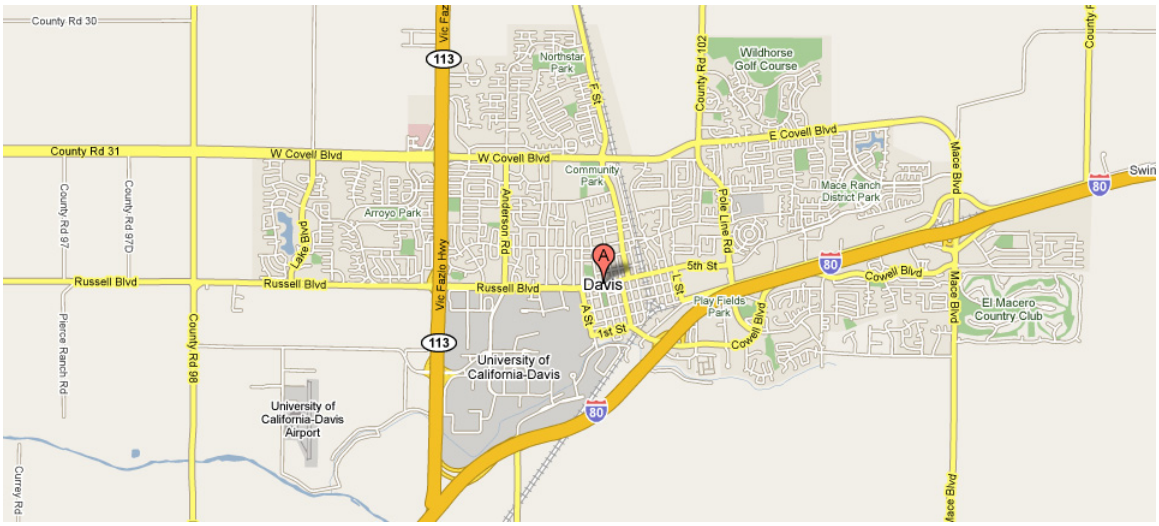
[26] R. Westermann. A Multiresolution Framework for Volume Rendering. In *Proc. of IEEE Volume Visualization '94*, pages 51–58, 1994.

[27] J. Wilhelms and A. van Gelder. Multi-Dimensional Trees for Controlled Volume Rendering and Compression. In *Proc. of IEEE Volume Visualization '94*, pages 27–34, 1994.

[28] Y. Zhou, B. Chen, and A. E. Kaufman. Multiresolution Tetrahedral Framework for Visualizing Regular Volume Data. In *Proc. of IEEE Visualization '97*, pages 135–142, 1997.

Chaoli Wang

University of California, Davis



Perception-Driven Techniques for Large Volume Data Analysis and Visualization

Chaoli Wang
University of California, Davis

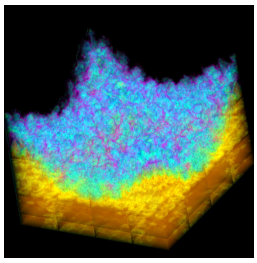
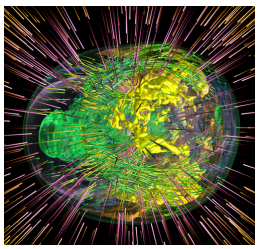
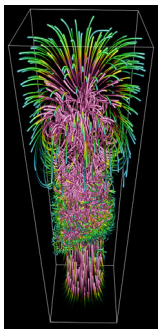
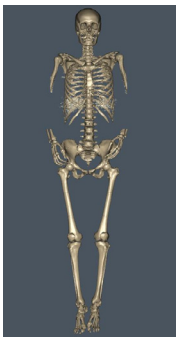
IEEE Visualization Tutorial
19 Oct 2008

Outline

- Introduction and motivation
  - Large data sets
  - Multiresolution visualization
  - Traditional solution vs. perception-driven solution
- Background
  - Wavelet transform
  - Hierarchical data representation
- Image-based quality metric
- Volume data quality assessment

Large Data Sets

- Scientific, medical, engineering, …
- Spatial, temporal, variable, …
- Gigabyte, terabyte, petabyte, exabyte, …

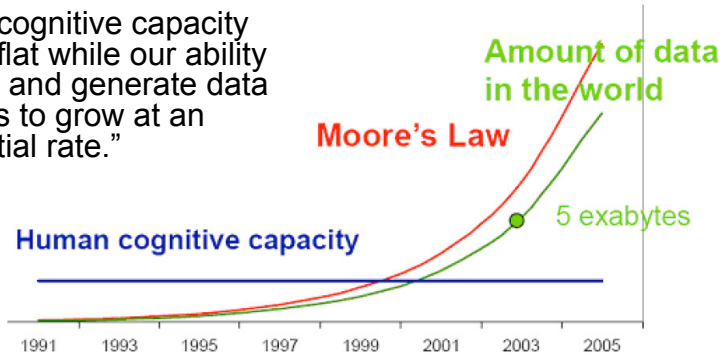





Multiresolution Visualization

- Large data sets make interactive visualization difficult
  - High (main + video) memory requirement
  - Slow I/O, slow rendering
- Multiresolution volume visualization
  - Adaptive data exploration
  - "*Overview first, zoom and filter, and then details-on-demand*" [SHNEIDERMAN 92]

Cognitive Capacity vs. Data Growth

- "Human cognitive capacity remains flat while our ability to collect and generate data continues to grow at an exponential rate."

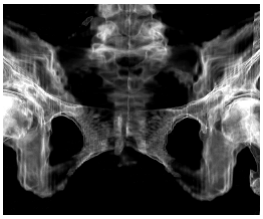
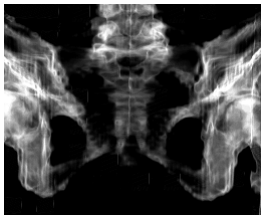
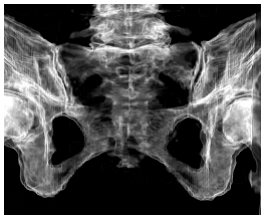


- Reference: Visualization and Knowledge Discovery: Recommendations from the DOE/ASCR Workshop on Visual Analysis and Data Exploration at Extreme Scale, 2007. (Image courtesy Jeffrey Heer)

Perception-Driven Techniques

- Quantitative metrics for parameter choices
  - LOD selection and rendering
  - Image-based data quality estimation
  - Present visually importance information
- Extract statistical information from the data
  - Volume data quality evaluation
  - Feature representation in multiscale manner
  - Incorporate perceptual reasoning

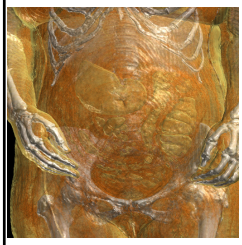
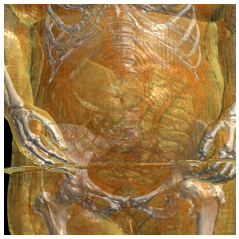
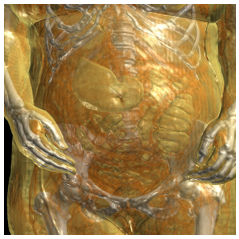
Image-Based Quality Metric

overview MSE, 8.6% SNR, 8.5% image, 8.3%

full resolution, 929

Multiscale Quality Assessment


(a) mean shift (b) voxel misplacement (c) averaging filter (d) salt-and-pepper noise

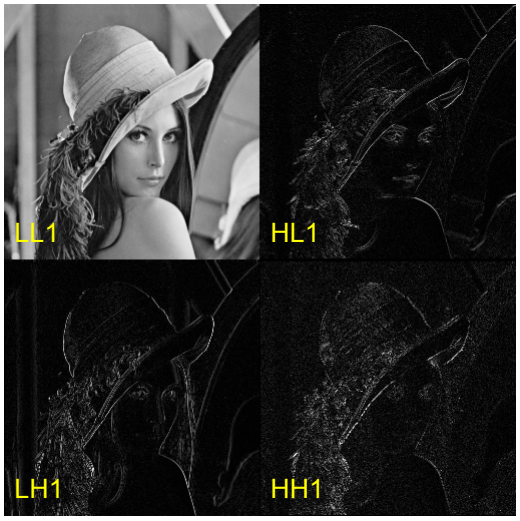
Ours: best worst

MSE/PSNR: best worst

Wavelet Transform and Hierarchical Data Representation

Wavelet Transform on 2D Image

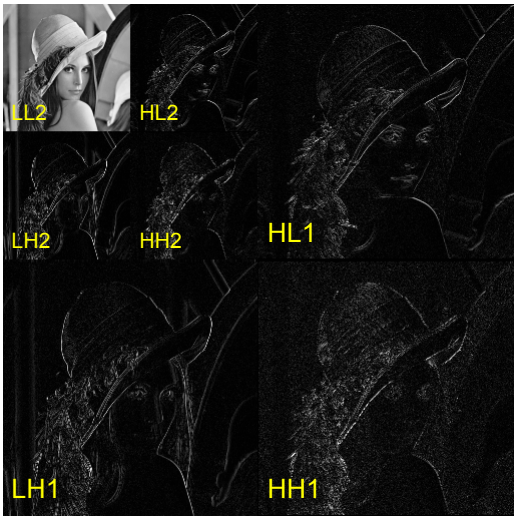
Wavelet Transform on 2D Image



1st level
L: low-pass filtered
H: high-pass filtered

LL1
LH1
HL1
HH1

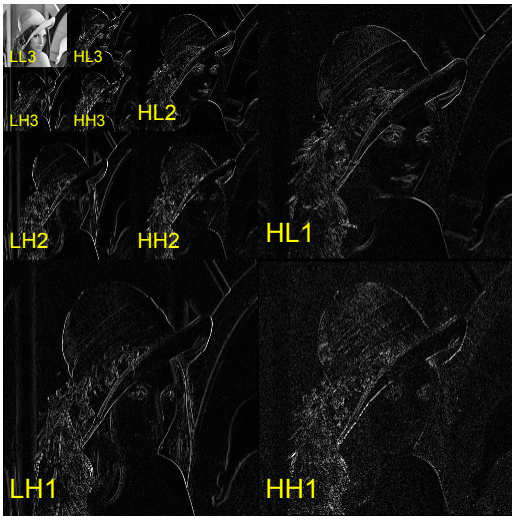
Wavelet Transform on 2D Image



2nd level
L: low-pass filtered
H: high-pass filtered

LL1
→
LL2
LH2
HL2
HH2

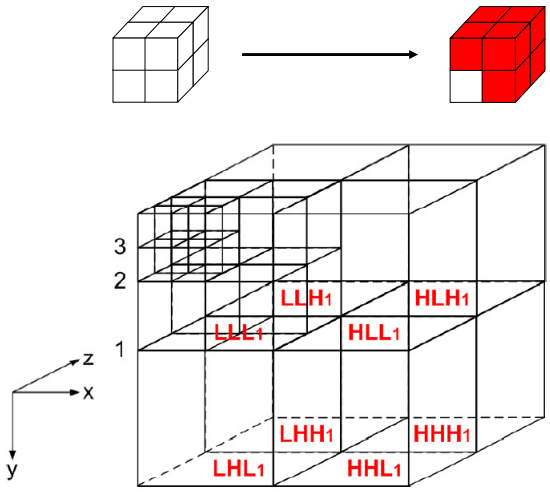
Wavelet Transform on 2D Image



3rd level
L: low-pass filtered
H: high-pass filtered

LL2
→
LL3
LH3
HL3
HH3

Wavelet Transform on 3D Volume



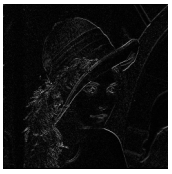
L: low-pass filtered
H: high-pass filtered

LLL
LLH
LHL
HLL
LHH
HLH
HHL
HHH

[MURAKI 92]

Wavelet Transform

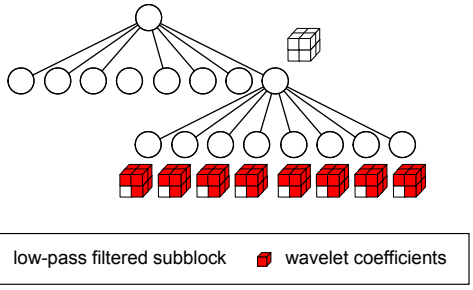
- Spatial domain → spatial-frequency domain
- Separable wavelet transform
- Wavelet compression
  - Low-pass filtered data: *summary* information
  - Wavelet coefficients: *detail* information
  - Coefficients are “sparse”, thus can be utilized in compression
  - Lossy and lossless compression
- Wavelet reconstruction



Hierarchical Data Representation

- Image and video
  - Laplacian pyramid [BURT et al. 83]
  - Multiresolution video [FINKELSTEIN et al. 96]
- 3D volume data
  - Laplacian pyramid [GHAVAMNIA et al. 95]
  - Octree hierarchy [LAMAR et al. 99]
  - Wavelet tree [GUTHE et al. 02]
- Time-varying volume data
  - Time-space partitioning (TSP) tree [SHEN et al. 99]
  - 4D hierarchy [LINSEN et al. 02]
  - Wavelet-based TSP tree [WANG et al. 04]

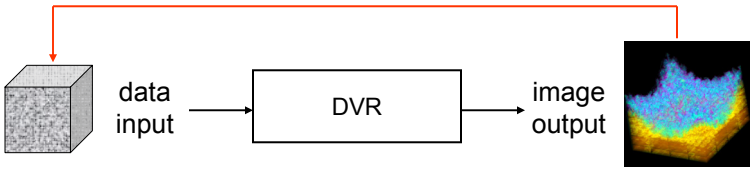
Wavelet Tree



- Octree-based space partition
- Block-wise wavelet transform and compression
- Error metric calculation

Image-Based Quality Metric for LOD Selection

Outline



- Importance values of data blocks
  - Emission (of a data block)
  - Occlusion (among data blocks)
  - Distortion (of low and high resolution data blocks)
  - Perceptually-uniform CIELUV color space
- Real-time update of quality metric
  - Summary table scheme
  - GPU-based visibility estimation

Volume Rendering Integral

- Volume rendering integral [MAX 95]

$$I_r = \underbrace{\int_0^D \tilde{c}(s(\vec{x}(\lambda)))}_{\text{(a)}} \underbrace{\exp(-\int_0^\lambda \tau(s(\vec{x}(\lambda'))) d\lambda') d\lambda}_{\text{(b)}}$$

- Discretized volume rendering integral

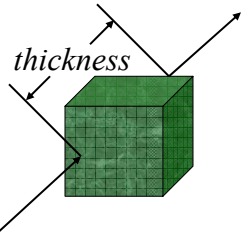
$$I_r = \underbrace{\sum_{i=0}^{n} c(s_i)\alpha(s_i)}_{\text{(a)}} \underbrace{\prod_{j=0}^{i-1}(1-\alpha(s_j))}_{\text{(b)}}$$

(a) emission
(b) attenuation

Importance Value Design

$$I_b = \underbrace{(c(\mu)\alpha(\mu)\cdot t \cdot a)}_{\text{(a)}} \cdot \underbrace{v}_{\text{(b)}} \cdot \underbrace{\varepsilon}_{\text{(c)}}$$

(a) emission
(b) attenuation
(c) distortion



μ : mean scale data value
$c(\mu)\alpha(\mu)$: color and opacity transfer function
t : average thickness
a : screen projection area
v : estimated visibility
ε : distortion of block *b* and its child blocks

Multiresolution Error Evaluation

$$\varepsilon_{ij} = \underbrace{\tilde{\sigma}_{ij}}_{\text{(a)}} \cdot \underbrace{\frac{\tilde{\mu}_i^2 + \tilde{\mu}_j^2 + C_1}{2\tilde{\mu}_i\tilde{\mu}_j + C_1}}_{\text{(b)}} \cdot \underbrace{\frac{\tilde{\sigma}_i^2 + \tilde{\sigma}_j^2 + C_2}{2\tilde{\sigma}_i\tilde{\sigma}_j + C_2}}_{\text{(c)}}$$

(a) covariance
(b) luminance distortion
(c) contrast distortion
structural similarity index
[WANG et al. 04]

$\tilde{\sigma}_{ij}$: covariance between b_i and b_j
$\tilde{\mu}$: mean value; $\tilde{\sigma}$: standard deviation
C_1 and C_2 : small constants; N : # of voxels in the block

$$\tilde{\sigma}_{ij} = \frac{1}{N-1}\sum_{k=1}^{N}(\tilde{x}_{ik} - \tilde{\mu}_i)(\tilde{x}_{jk} - \tilde{\mu}_j)$$

$$\tilde{\sigma}_i = \frac{1}{N-1}\sum_{k=1}^{N}(\tilde{x}_{ik} - \tilde{\mu}_i)^2 \qquad \tilde{\sigma}_j = \frac{1}{N-1}\sum_{k=1}^{N}(\tilde{x}_{jk} - \tilde{\mu}_j)^2$$

Multiresolution Error Evaluation

$\tilde{x}$ and $\tilde{\mu}$: CIELUV color values

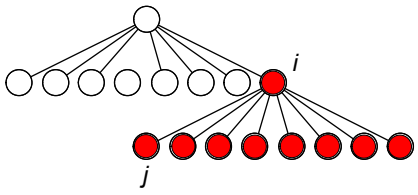
$$\tilde{x} - \tilde{\mu} = \Delta E(f(c_{rgb}(x)\alpha(x)), f(c_{rgb}(\mu)\alpha(\mu)))$$

ΔE : CIELUV color difference

$$\Delta E = \sqrt{\Delta L^{*2} + \Delta u^{*2} + \Delta v^{*2}}$$

ε_i : multiresolution error

$$\varepsilon_i = \sum_{j=0}^{7} \varepsilon_{ij} + \max\{\varepsilon_j \mid_{j=0}^{7}\}$$

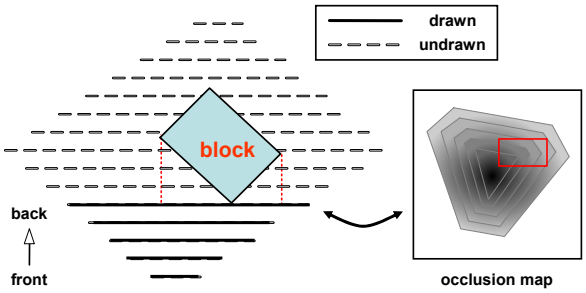


Summary Table Scheme

- Update metric when transfer function changes
  - Size of data range << # of voxels in the volume [LAMAR et al. 03]
  - Count frequencies of unique error terms: x_i, x_j, and (x_i, x_j)
  - Store histogram and correspondence tables
  - Runtime table lookup

data set	space (overhead)	update time
VisWoman	9.22MB (1.07%)	5s
RMI	44.1MB (0.57%)	13s

Visibility Estimation



- Evaluate approximate visibility of data blocks
  - Render low resolution data
  - Draw front-to-back view-aligned slices
  - $v = 1 - \alpha$, where α is the average opacity on the occlusion map

CPU vs. GPU Solutions

- CPU solution
  - Read framebuffer when drawing slices
  - Iterate through alpha channel
  - Framebuffer reads become bottleneck
- GPU solution
  - Utilize summed area tables (SATs)
  - `GL_EXT_framebuffer_object` (FBO)
  - 3~4 times faster than CPU solution

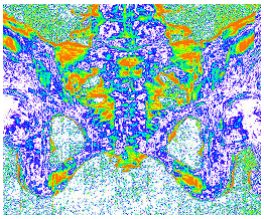
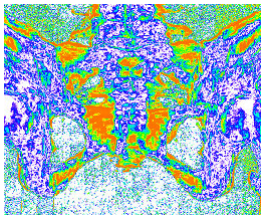
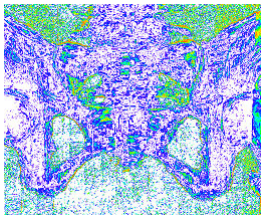
LOD Selection

- User specifies the block budget
- Update importance values
  - ν per view
    - Only update a certain percentage of blocks
    - Postpone update if the view changes slightly
  - ε per transfer function
- Priority queue for LOD refinement
- A list of blocks identified from greedy selection

Results – Timing

data set (type)	VisWoman (short)	RMI (byte)
volume dimension	512 * 512 * 1728	2048 * 2048 * 1920
volume size	864MB	7.5GB
block dimension	32 * 32 * 64	128 * 128 * 64
block size	128KB	1MB
# non-empty blocks	9446	10499
compression ratio (lossless)	2.37:1	5.60:1
visibility (GPU, 512^2 image)	0.151s	0.185s
prioritization (all blocks)	0.343s	0.563s
transfer function (256 levels)	5s	13s
3.0GHz CPU, 3GB memory, nVidia GeForce 7800 GT graphics card		

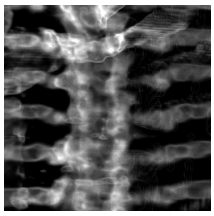
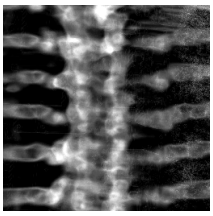
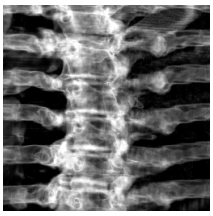
Results – VisWoman Data Set

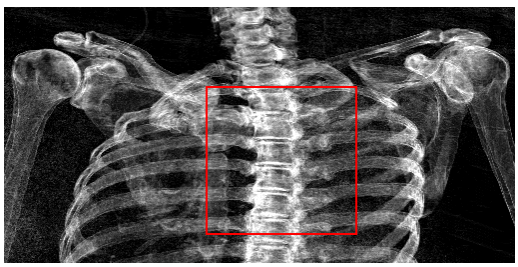
overview MSE, 80, 8.61% SNR, 79, 8.50% image, 77, 8.29%


full resolution, 929 color map

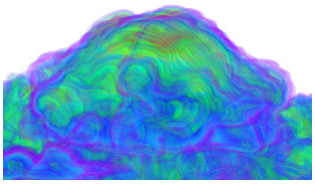
Results – VisWoman Data Set

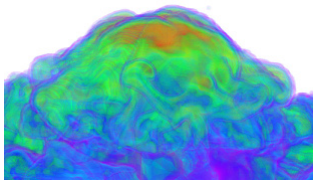
MSE, 36 blocks SNR, 37 blocks image, 34 blocks



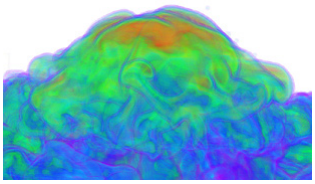
Results – RMI Data Set



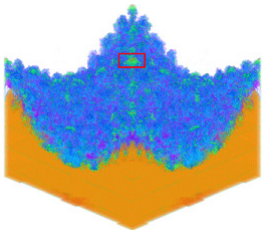
MSE, 55 blocks



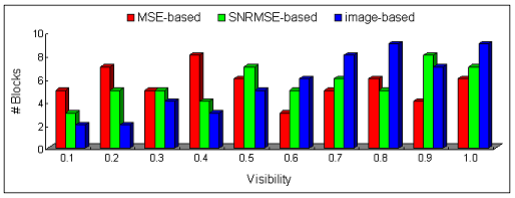
SNR, 55 blocks



image, 55 blocks

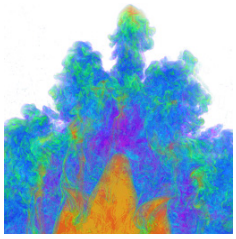


overview

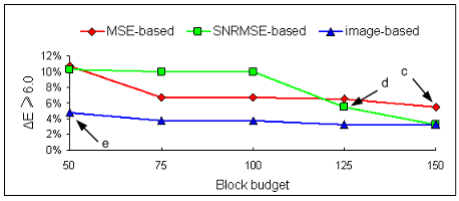


visibility

Results – RMI Data Set



full resolution, 1237



pixel difference percentage

Multiscale Volume Data Quality Assessment

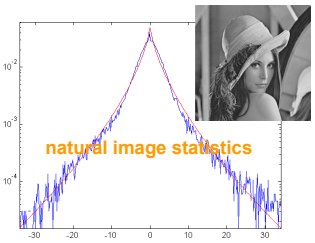
Motivation

- We may use any type of non-original data
  - Quantized (e.g., floating → byte/short)
  - Compressed (e.g., lossy compression)
  - Filtered (e.g., Gaussian smooth/blur)
  - Reduced (e.g., down sampling)
  - Distorted (e.g., noise)
  - Corrupted (e.g., lost in transmission)
- How to measure data quality loss introduced in different versions of data?

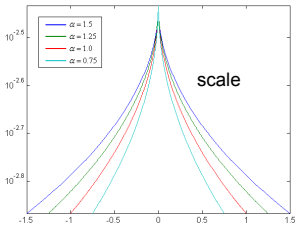
Solution

- Extract features from the original data in the wavelet domain
  - Multiscale wavelet decomposition
  - Wavelet subband analysis – *global* information
  - Collect important coefficients – *local* information
  - Define distance metrics
- Use features for quality assessment
  - Features as “carry-on” information
  - Reduced-reference approach

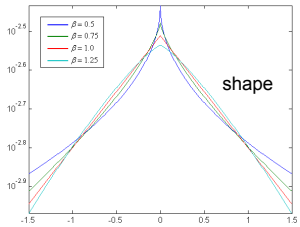
Generalized Gaussian Density



Lena image, HL_1 subband



β = 0.5



α = 1.5

$$p(x) = \frac{\beta}{2\alpha\Gamma\left(\frac{1}{\beta}\right)} \exp\left(-\left(\frac{|x|}{\alpha}\right)^{\beta}\right)$$

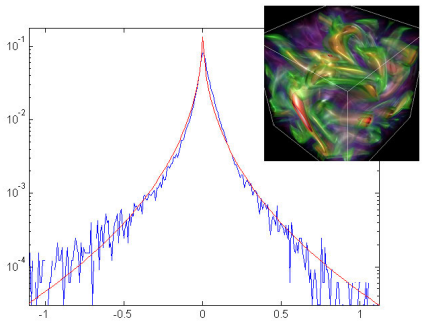
[MALLAT 99]

Γ Gamma function

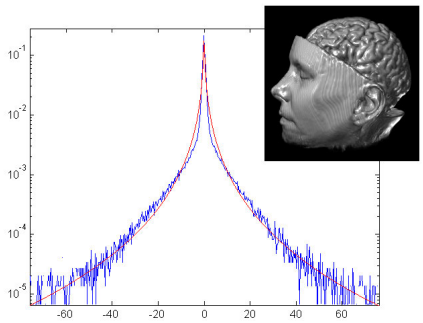
α scale parameter

β shape parameter
= 2, Gaussian distribution
= 1, Laplacian distribution

Generalized Gaussian Density



vortex data set, HHL_2 subband



brain data set, HLL_1 subband

Kullback-Leibler Distance

- Quantify the difference of wavelet coefficient distribution between the distorted and the original data

$$d(p \| q) = \sum_{i=1}^{M} P(i) \log \frac{P(i)}{Q(i)}$$

P: wavelet subband coefficient histogram approximated with GGD parameter(α , β)

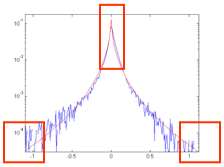
Q: wavelet subband coefficient histogram of the distorted data

$$D = \log(1 + \sum_{i=1}^{B} d(p^i \| q^i))$$

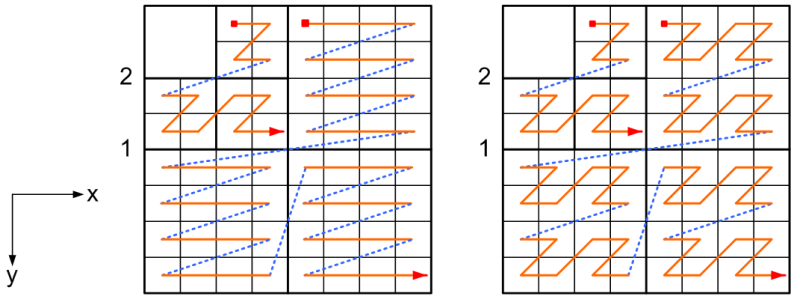
D: the KLD between the distorted and original data

Wavelet Coefficient Selection

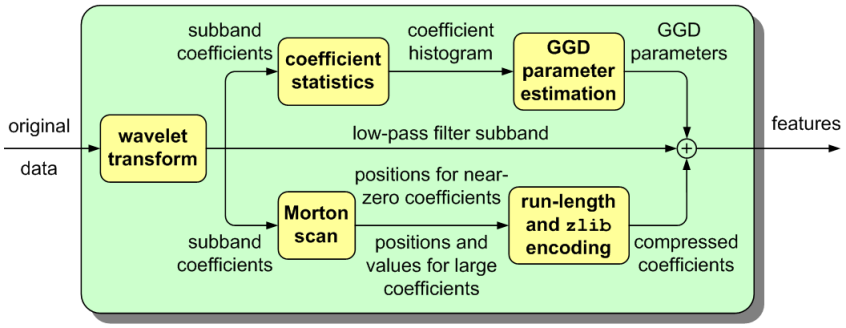
- Coefficients of large magnitude
  - Correspond to abrupt features like edges or boundaries
  - Along the tails of the marginal coefficient distribution
- Neighboring near-zero coefficients
  - Correspond to homogeneous regions
  - Close to the zero peak of the marginal coefficient distribution
- Modulated by visual importance
  - Consider opacity and visibility
  - Approximate used low-resolution data



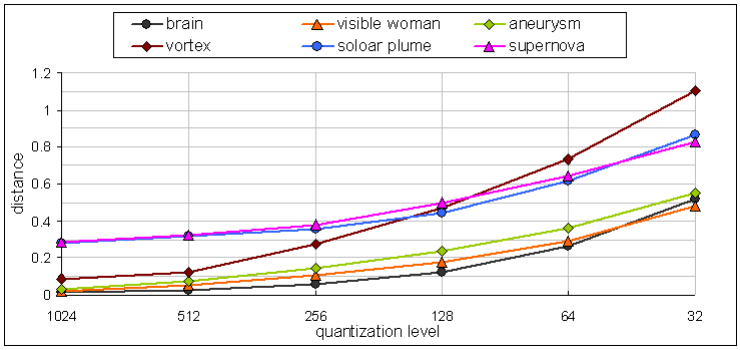
Coefficient Scan Order



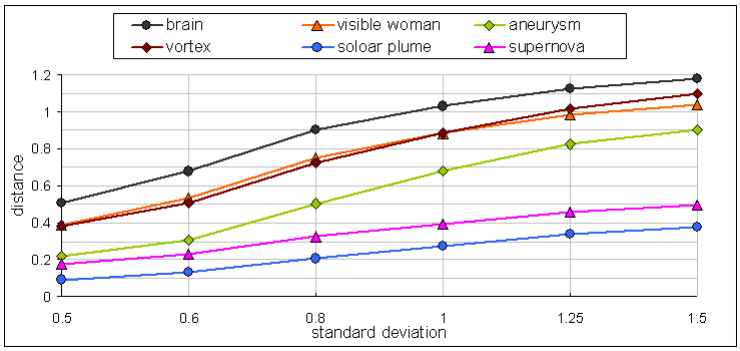
Feature Representation



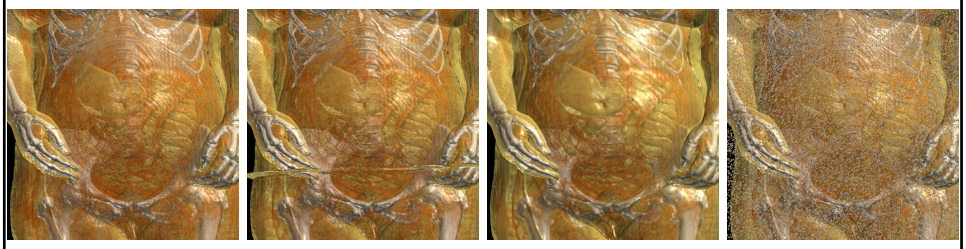
Quality Assessment – Quantization



Quality Assessment – Gaussian Filter



Quality Assessment – Cross Comparison



(a) mean shift (b) voxel misplacement (c) averaging filter (d) salt-and-pepper noise

Ours: best worst

MSE/PSNR: best worst

type	D1	D2	D3	D	rank	MSE	PSNR	rank
mean shift	8.8428e-5	6.9770e-7	2.3914e-2	0.0239	4	1.8691e+3	48.1648	3
misplacement	1.5366e-2	1.1612e-1	4.6497e-3	0.1223	3	1.5397e+3	49.0066	4
averaging	1.6449e+0	5.4139e-1	1.4596e-3	0.7073	2	1.9289e+5	28.0279	1
noise	1.7343+0	7.8530e-1	9.7468e-3	0.9685	1	1.2152e+4	40.0347	2

Summary

- Applied perception in visualization
  - Image-based quality metric
    - Backward approach (from image to data)
    - Evaluate data contribution in rendering
    - Precompute summary tables
    - Runtime update visibility for LOD decision
  - Volume data quality assessment
    - Multiscale approach (in the wavelet domain)
    - Use GGD to capture wavelet coefficient distribution
    - Select visually important coefficients
    - Quantify data quality loss in different versions

References

- BURT, P. J., AND ADELSON, E. H. The Laplacian Pyramid as a Compact Image Code. *IEEE Transactions on Communications*, 31(4):532–540, 1983.
- MALLAT, S. A Theory for Multiresolution Signal Decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674-693, 1989.
- SHNEIDERMAN, B. Tree Visualization with Tree-Maps: A 2D Space-Filling Approach. *ACM Transactions on Graphics*, 11(1):92–99, 1992.
- MURAKI, S. Approximation and Rendering of Volume Data Using Wavelet Transforms. *Proc. IEEE Visualization '92*, pages 21–28, 1992.
- MAX, N. Optical Models for Direct Volume Rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995.
- GHAVAMNIA, M. H., AND YANG, X. D. Direct Rendering of Laplacian Pyramid Compressed Volume Data. *Proc. IEEE Visualization '95*, 192–199, 1995.
- FINKELSTEIN, A., JACOBS, C. E., AND SALESIN, D. H. Multiresolution Video. Proc. *ACM SIGGRAPH '96*, pages 281–290, 1996.
- LAMAR, E., HAMANN, B., AND JOY, K. I. Multiresolution Techniques for Interactive Texture-Based Volume Visualization. *Proc. IEEE Visualization '99*, pages 355–362, 1999.
- SHEN, H.-W., CHIANG, L.-J., AND MA, K.-L. A Fast Volume Rendering Algorithm for Time-Varying Fields Using a Time-Space Partitioning (TSP) Tree. *Proc. IEEE Visualization '99*, pages 371–377, 1999.
- DO, M. N., AND VETTERLI, M. Wavelet-Based Texture Retrieval Using Generalized Gaussian Density and Kullback-Leibler Distance. *IEEE Transactions on Image Processing*, 11(2):146–158, 2002.
- LINSEN, L., PASCUCCI, V., DUCHAINEAU, M. A., HAMANN, B., AND JOY, K. I. Hierarchical Representation of Time-Varying Volume Data with $\sqrt[4]{2}$ Subdivision and Quadrilinear B-Spline Wavelets. *Proc. Pacific Graphics '02*, pages 346–355, 2002.

References

- GUTHE, S., WAND, M., GONSER, J., AND STRAßER, W. 2002. Interactive Rendering of Large Volume Data Sets. *Proc. IEEE Visualization '02*, pages 53–60, 2002.
- LAMAR, E., HAMANN, B., AND JOY, K. I. Efficient Error Calculation for Multiresolution Texture-Based Volume Visualization. *Hierarchical & Geometrical Methods in Scientific Visualization*, pages 51–62, 2003.
- WANG, C., AND SHEN, H.-W. A Framework for Rendering Large Time-Varying Data Using Wavelet-Based Time-Space Partitioning (WTSP) Tree. *Technical Report OSU-CISRC-1/04-TR05*, Department of Computer and Information Science, The Ohio State University, 8 pages, 2004.
- GUTHE, S., AND Straßer, W. Advanced Techniques for High-Quality Multi-Resolution Volume Rendering. *Computers & Graphics*, 28(1):51–58, 2004.
- WANG, Z., BOVIK, A. C., SHEIKH, H. R., AND SIMONCELLI, E. P. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.
- LJUNG, P., LUNDSTROM, C., YNNERMAN, A., AND MUSETH K. Transfer Function Based Adaptive Decompression for Volume Rendering of Large Medical Data Sets. *Proc. IEEE Volume Visualization '04*, pages 25–32, 2004.
- WANG, Z., WU, G., SHEIKH, H. R., YANG, E.-H., AND BOVIK, A. C. Quality-Aware Images. *IEEE Transactions on Image Processing*, 15(6):1680–1689, 2006.
- WANG, C., GARCIA, A., AND SHEN, H.-W. Interactive Level-of-Detail Selection Using Image-Based Quality Metric for Large Volume Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13(1):122-134, 2007.
- WANG, C., AND MA, K.-L. A Statistical Approach to Volume Data Quality Assessment. *IEEE Transactions on Visualization and Computer Graphics*, 14(3):590-602, 2008.

Acknowledgements

- Collaborators
  - Antonio Garcia, Han-Wei Shen, Kwan-Liu Ma
- Funding
  - National Science Foundation
    - ACI-0325934, CCF-0346883, CCF-0634913, CNS-0551727, OCI-0325934, and CCF-9983641
  - Department of Energy
    - DE-FG02-03ER25572, DE-FC02-06ER25777, DOE-FC02-01ER41202, and DOE-FG02-05ER54817
  - Oak Ridge National Laboratory
    - Contract 400045529

Interactive Level-of-Detail Selection Using Image-Based Quality Metric for Large Volume Visualization

Chaoli Wang, *Student Member, IEEE,* Antonio Garcia, and Han-Wei Shen

***Abstract*— For large volume visualization, an image-based quality metric is difficult to incorporate for level-of-detail selection and rendering without sacrificing the interactivity. This is because it is usually time-consuming to update view-dependent information as well as adjust to transfer function changes. In this paper, we introduce an image-based level-of-detail selection algorithm for interactive visualization of large volumetric data. The design of our quality metric is based on an efficient way to evaluate the contribution of multiresolution data blocks to the final image. To ensure real-time update of the quality metric and interactive level-of-detail decisions, we propose a summary table scheme in response to run-time transfer function changes, and a GPU-based solution for visibility estimation. Experimental results on large scientific and medical data sets demonstrate the effectiveness and efficiency of our algorithm.**

***Index Terms*— Data compaction and compression, perceptual reasoning, viewing algorithms, interaction techniques, hierarchical image representation, volume visualization.**

I. Introduction

DIRECT volume rendering with hardware texture mapping has become a standard technique for visualizing three-dimensional scalar fields from scientific and medical applications. An increasing number of these applications are now producing large-scale data sets, ranging from gigabytes to terabytes. One example is the Visible Woman (VisWoman) data set with resolution of $512 \times 512 \times 1728$ from The National Library of Medicine, generated as part of the Visible Human Project. Another example is the Richtmyer-Meshkov Instability (RMI) simulation performed at Lawrence Livermore National Laboratory. The simulation was executed on a $2048 \times 2048 \times 1920$ rectilinear grid, and it produced 7.5 gigabytes of data at each time step.

While it is common for the domain scientists to generate enormous amount of data, the size of video memory in the state-of-the-art high-end graphics hardware is limited to only several hundred megabytes. This disparity severely challenges brute-force conventional hardware-texturing based volume rendering approaches. New visualization systems that can scale adequately and ensure a high level of interactivity are needed. Among several alternatives, multiresolution volume rendering [13], [17], [31] is a solution that can reduce the rendering cost dramatically. To perform interactive rendering, a multiresolution data hierarchy composed of multiple spatially partitioned blocks is first created. At run time, as the user navigates through the hierarchy, various amounts of data at different levels of detail can be extracted and used for the rendering.

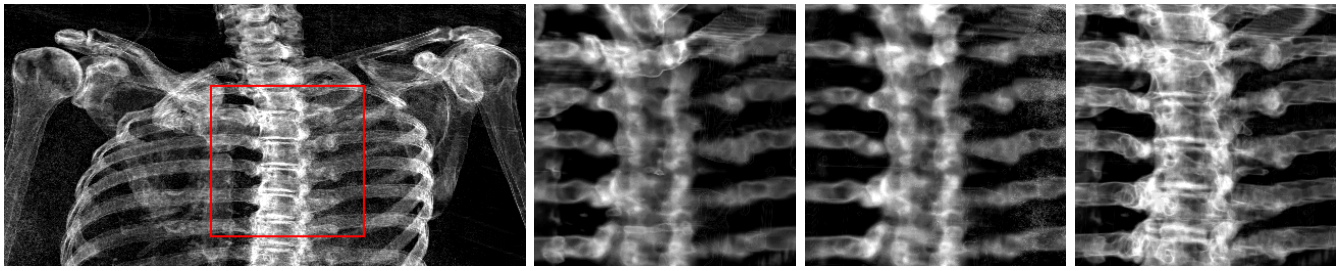
Often, such a level-of-detail (LOD) is determined by various user-specified parameters, such as the tolerance of errors based on certain data-dependent metrics [1], [19], [29], different view-dependent parameters [17], [21], or both [12], [13], [22], [32]. In general, these metrics can be classified as *data-based* and *image-based* metrics. Data-based metrics measure the distortion between low and high (or full) resolution data blocks in the volume. The most widely used data-based metrics are *mean square error* (MSE), L^2*-norm*, and *signal-to-noise ratio* (SNR). These metrics have clear physical meanings and are simple to compute. However, they are usually not effective in predicting the quality of the rendered images due to the lack of correlation between data and image, as indicated in [7], [15], [22], [27], [30]. Image-based metrics focus on the ultimate images the user perceives, and strive to capture the quality loss in the rendered images introduced by rendering low resolution data. These metrics are intrinsically view-dependent and more difficult to develop in conjunction with interactive LOD selections for large volume visualization. The major challenge lies in designing an image-based metric for quality-driven LOD selection, and updating the metric fast enough as not to harm the interactivity.

In this paper, we present an interactive LOD selection and rendering algorithm using an image-based quality metric for visualizing large volumetric data. The main contributions of our paper are:

- We introduce an image-space model for the quality metric design, based on an efficient way to evaluate the importance values of coarse-grained multiresolution data blocks on the final image. Fig. 1 shows a comparison of the LOD rendering of the VisWoman data set using our image-based quality metric and the MSE-based and SNRMSE-based (MSE of SNR) metrics. Unlike traditional LOD selection algorithms using data-based metrics, our LOD selection algorithm captures the structural distortion of the data and generates images of better visual quality under similar block budgets.
- Our method is adaptive to changes of the input transfer function. We utilize a zigzag run-length encoding scheme to store summary tables of data blocks in the mul-

C. Wang and H.-W. Shen are with the Department of Computer Science and Engineering, The Ohio State University, 395 Dreese Laboratories, 2015 Neil Avenue, Columbus, OH 43210. E-mail:{wangcha, hwshen}@cse.ohio-state.edu.

A. Garcia is with Intel Corporation, 2700 156th Avenue NE, Suite 300, Bellevue, WA 98007. E-mail:antonio.garcia@intel.com.



(a) (b) (c) (d)

Fig. 1. (a) shows a zoom to the upper skeleton of the VisWoman data set, rendered with full resolution. (b) (MSE-based, 36 blocks), (c) (SNRMSE-based, 37 blocks), and (d) (image-based, 34 blocks) show a closer zoom to the spine while rendered in low resolution.

tiresolution hierarchy with very small storage overhead (around 1% of the original volume data). During the run-time rendering, we can update the quality metric within seconds for large data sets whenever the transfer function changes at run time.

- Based on *summed area tables* (SATs), we propose a GPU reduction scheme that can efficiently perform the visibility estimation for multiresolution data blocks, ensuring real-time update of view-dependent information and interactive LOD selection.

The remainder of the paper is structured as follows. First, we present background and review related work in Section II. In Section III, we briefly introduce our multiresolution data representation for large three-dimensional data sets. In Section IV, we describe our multiresolution LOD selection and rendering algorithm in detail. Experimental evidence showing the visual quality gain using our image-based LOD selection over data-based ones is provided in Section V. The paper is concluded in Section VI with future work for our research.

II. Background and Related Work

A. Background

Multiresolution Data Representation: Building a multiresolution data hierarchy allows the user to visualize data at different scales, and balance image quality and computation speed. A number of techniques have been developed to provide hierarchical data representation for volumetric data, such as the *Laplacian pyramid* [9], multi-dimensional trees [32], and octree-based hierarchies [1], [17]. Muraki [25] first introduced the use of wavelet transforms for volumetric data. Westermann [31] presented a framework for approximating the volume rendering integral using multiresolution spaces and wavelet projections. More recently, Guthe et al. [13] presented a wavelet-encoded hierarchical representation for large volume data sets that supports interactive walkthroughs on a single commodity PC.

Image-Based Quality Measurement: The lack of correlation between the type of error in an image and the response of the *human visual system* (HVS) to different types of errors prompted researchers to develop image-based metrics. Jacobs et al. [15] introduced an image-query metric for searching in an image database using a query image similar to the intended target. The metric makes use of multiresolution wavelet decompositions of the query and database images, and operates on the coefficients of these decompositions. Gaddipati et al. [7] presented a wavelet-based metric which captures the change in images wrought by operators and the image synthesis algorithms. Sahasrabudhe et al. [27] proposed a quantitative technique which accentuates differences in images and data sets through a collection of partial metrics. A study of different image comparison metrics, categorized into spatial domain, spatial-frequency domain, and perceptually-based metrics, was presented in [33]. Alternatively, Wang et al. [30] proposed the use of structural similarity for the design of image quality measures. Experimental results show that their *Structural Similarity Index* simulates the response of the HVS with low computation cost.

In the context of large volume visualization, an image-based metric is difficult to incorporate because of the following reasons: First, image-based metrics need to consider run-time information, such as the viewing, projection, and occlusion. Unlike most data-based metrics which can be easily computed in a preprocessing stage, to get view-dependent occlusion information for a large data set, one has to resort to either sophisticated precomputation with considerable overhead [8], or run-time calculation with rough approximation [12]. Next, the user may adjust the transfer function during the rendering in order to reveal different features. Image-based metrics should be adaptive to run-time transfer function changes. Previous work on large data visualization usually assumes the input transfer function is fixed, or is limited to a family of transfer functions consisting of a series of basis functions [8]. Last, the human observer plays a central role in perceiving the image quality. Therefore, image-based metrics should also take into account human perception [28] in the visualization process. The factors need to be considered include the perception of distance, coverage, shape, color, occlusion, texture, and lighting. In this paper, we integrate an image-based quality metric into the multiresolution LOD selection and rendering framework.

Visibility Computation: To accelerate the process of image generation, visibility culling has long been employed [2] in rendering large polygonal models as well as volumetric data sets. Klosowski and Silva [16] introduced the time-critical *Prioritized-Layered Projection* (PLP) algorithm for fast rendering of high depth complexity scenes, using a solidity-based metric for visibility estimation. A similar approach that

TABLE I
THE COMPARISON OF THE THREE APPROACHES.

approach	Guthe et al. [12]	Ljung et al. [22]	our approach
data hierarchy	octree-based	flat block-based	octree-based
error evaluation	RGB, use max error	CIELUV, MSE-based	CIELUV, image quality measure
block projection	projection size	not considered	luminance, projection size, thickness
visibility estimation	assume uniform opacity occlusion only raycasting, CPU	use simplified histogram occlusion only raycasting, CPU	use low resolution data emission + occlusion SAT, GPU
transfer function	not adaptive	adaptive, 12-level simplified histogram	adaptive, 256-level histogram

integrates occlusion culling with view-dependent rendering was given in [4]. Gao et al. [8] proposed a *Plenoptic Opacity Function* (POF) scheme, which encodes the view-dependent opacity of a volume block for visibility culling of large volume data. Utilizing visibility information for multiresolution volume rendering has not been widely studied, nor has the potential of using programmable graphics hardware for visibility estimation been fully explored.

B. Related Work

Two recent methods have been proposed for error measurement and visibility calculation within the multiresolution volume rendering framework. Guthe et al. [12] presented several improvements on compression based multiresolution rendering of large data sets to speed up the volume rendering process. The screen-space error was measured as the maximum error produced by each block multiplied by its projection size. The maximum error was calculated by simply adding the differences of RGB color and opacity components, rather than more correct opacity-modulated color differences. The visibility estimation was performed for empty space skipping and occlusion culling to speed up the rendering. They took a conservative way of visibility testing, and assumed a uniform opacity for each data block for very rough approximation. Ljung et al. [22] focused on incorporating transfer function into adaptive decompression of volume data for multiresolution volume rendering. Similar to [21], they took a flat block-based volume decomposition approach. At the compression stage, they calculated the meta-data for each block: the average scalar value, the *root-mean-square* (RMS) wavelet coefficients, and a simplified histogram. The error metric was based on a simplified version of MSE in the CIELUV space. To account for occlusion, a low resolution ray-casting renderer was used to estimate the average opacity of each block, followed by empirical tests for approximating the simplified discrete rendering equation with no emission factor.

In our work, the goal is to incorporate an image-based quality metric for multiresolution volume rendering of large data sets. Thus, our main focus is on quality-driven interactive LOD selection, rather than compression based rendering [12], or transfer function based decompression [22]. To achieve this goal, we propose much more refined solutions at each step of our algorithm. Our image-based quality metric takes into account the emission as well as the occlusion of the multiresolution data blocks, and is more accurate than the simplified ones in [12] and [22]. We present a summary table scheme to account for the run-time transfer function change with much higher precision (256-level histogram) than the one (12-level simplified histogram) in [22]. Using simplified histogram has the risk of missing important details in the data. Therefore, our refined solution is more suitable for large data sets with high dynamic range. Our scheme allows one to update the errors for a large volume of size around 1024^3 within seconds. Also, we introduce a GPU-based reduction scheme for getting estimated visibility for the data blocks in real time, while both of those methods in [12] and [22] used only the software raycasting approach. We use low resolution data for visibility estimation, which is more exact than just assuming a uniform opacity [12] or taking the simplified histogram [22] for each block. In Table I, we list the major differences between our approach and the two related ones.

III. MULTIRESOLUTION DATA HIERARCHY

To build a multiresolution data hierarchy from a large three-dimensional data set, we use wavelet transforms to convert the data into a hierarchical multiresolution representation, called the *wavelet tree* [13]. The wavelet tree construction algorithm starts with subdividing the original 3D volume into a sequence of blocks with the same size (assuming each has N voxels). These raw volume blocks form the leaf nodes of the wavelet tree. After performing a 3D wavelet transform on each block, a low-pass filtered subblock of size $N/8$ and wavelet coefficients of size $7N/8$ are produced. The low-pass filtered subblocks from eight adjacent leaf nodes in the wavelet tree are grouped into a single block of N voxels, which becomes the low resolution data stored in the parent node. We recursively apply this 3D wavelet transform and subblock grouping process in a bottom-up manner till the root of the tree is reached, where a single block of size N is used to represent the entire volume. To reduce the size of the coefficients stored in the wavelet tree, the wavelet coefficients in each tree node are set to zero if they are smaller than a user-specified threshold. These wavelet coefficients are then compressed using run-length encoding combined with a fixed Huffman encoder. Note that in the wavelet tree, the multiresolution data blocks associated with all the tree nodes have data of the same size, which is N. However, the spatial resolutions they represent may vary, depending on which level of the tree the corresponding nodes lie on.

Coupled with the construction of the wavelet tree, multiresolution error ε is evaluated for each of the tree nodes. We calculate the error as the summation of the errors between the parent block and its eight immediate child blocks. We also

take into account the maximum error of the child blocks, as an approximation of the error between the parent block and the original full-resolution data block it represents. Written in equation:

$$\varepsilon_{b_i} = \sum_{j=0}^{7} \varepsilon_{ij} + \max\{\varepsilon_{b_j}|_{j=0}^{7}\} \quad (1)$$

where ε_{ij} is the voxel-wise error between parent block b_i and its jth child block b_j (Note that b_j contributes one eighth of the low-pass filtered data to b_i. For each voxel value in b_j, we linearly interpolate its corresponding value from its low-pass filtered data in b_i.); ε_{b_i} and ε_{b_j} are the multiresolution errors of blocks b_i and b_j respectively. As a special case, if block b_i is associated with a leaf node in the hierarchy, we define its error as a small constant C. Depending on the need, ε_{ij} can be calculated in different ways. For example, we can directly calculate it in the scalar data space using MSE or SNRMSE (MSE of SNR), or in the RGB or the CIELUV color space. The multiresolution errors in the data hierarchy are then normalized for our use.

IV. LOD Selection and Rendering Algorithm

Our multiresolution LOD selection and rendering algorithm optimizes the quality of rendered images through the use of an image-based quality metric. Our quality metric evaluates the importance values of multiresolution data blocks by examining the contribution of data blocks to the final image, based on the *discretized volume rendering integral* (DVRI). The evaluation approximates the emission of each block, as well as takes into account the occlusion caused by the blocks in front of it. To capture the multiresolution error in the data hierarchy, we modulate the importance value with the distortion between low and high resolution data blocks, calculated in the roughly perceptually-uniform CIE $L^*u^*v^*$ (CIELUV) color space. To ensure a real-time update of the quality metric, we propose a summary table scheme to respond to changes of the transfer function, and a GPU reduction scheme for visibility estimation. At run time, for a given viewing direction, the LOD selection is made based on a priority queue scheme utilizing the importance values of multiresolution blocks as their priority values. The wavelet tree traversal maintains the LOD as a cut through the hierarchy, and the importance values dictate the sequence of LOD refinements. A certain number of blocks up to a user-specified budget are extracted and sent to the texture hardware for rendering.

A. Volume Rendering Integral

According to the emission-absorption optical model [23], the *volume rendering integral* (VRI) that calculates the amount of light I along a viewing ray r through the volume is given by:

$$I_r = \int_0^D \tilde{c}(s(\vec{x}(\lambda)))\ \exp(-\int_0^\lambda \tau(s(\vec{x}(\lambda')))\ d\lambda')\ d\lambda \quad (2)$$

where $s(\vec{x}(\lambda))$ is the scalar value at position $\vec{x}(\lambda)$ in the volume, parameterized by the distance λ to the viewpoint; $\tilde{c}(s)$ is the volume source term or intensity; $\tau(s)$ defines the attenuation function.

In general, the VRI cannot be evaluated analytically. Therefore, practical volume rendering algorithms discretize the VRI by numerical approximation. Using Riemann sum for n equal ray segments of length D/n, and further approximating the exponential function with the first two terms of the Taylor series expansion, we get the *discretized volume rendering integral* (DVRI) [24], also known as the compositing equation [20]:

$$I_r = \sum_{i=0}^{n} c(s_i)\alpha(s_i)\prod_{j=0}^{i-1}(1-\alpha(s_j)) \quad (3)$$

where $c(s)$ and $\alpha(s)$ define the color and opacity transfer function. This equation denotes that at each discrete sample position i along the viewing ray r in the volume, light is emitted according to the term $c(s_i)\alpha(s_i)$, which is absorbed by the volume at all positions along r in front of i according to the term $\alpha(s_j)$. Eqn. 3 serves as the foundation for our design of importance values for multiresolution data blocks.

B. Importance Value Design

The DVRI in Eqn. 3 evaluates the amount of light visible to the eye on a per-ray basis. It is also possible to look at the equation on a per-slice basis, which leads to the popular slice-based compositing technique for volume rendering. In this paper, the underlying entity for our LOD selection and rendering algorithm is a data block. Therefore, we evaluate the importance values of multiresolution data blocks by approximating Eqn. 3 on a per-block basis. The importance value of a data block b along the viewing direction r is calculated as follows:

$$I_b = (c(\mu)\alpha(\mu)\ \cdot\ t\ \cdot\ a)\ \cdot\ v \quad (4)$$

where μ is the mean scalar data value of block b; $c(\mu)$ and $\alpha(\mu)$ define the color and opacity transfer function (we actually calculate the magnitude of its corresponding CIELUV color); t is the average thickness (the length of the viewing ray segment inside the block) of block b; a is the screen projection area of the block, and v is its estimated visibility. Similar to Eqn. 3, here $((c(\mu)\alpha(\mu)\cdot t\cdot a)$ approximates the emission of block b along direction r, and v accounts for the attenuation. Given a viewing direction r, I_b essentially evaluates the contribution of block b to the final image.

If we record the mean scalar value μ of each block during the construction of the multiresolution data hierarchy, we can quickly compute $c(\mu)$ and $\alpha(\mu)$ on the fly. Also, given a viewing direction r, the average thickness t and projection area a of a block can be easily calculated at run time. However, to obtain the estimated visibility v of a block interactively is non-trivial, and we will describe our real-time GPU-based solution in Section IV-E.

Even if two multiresolution data blocks have the same approximate emission and absorption terms, the distortions between the blocks and their children can be different. Taking

into account the relative distortion, we modulate the importance value with the multiresolution error between low and high resolution data blocks. Eqn. 4 becomes:

$$I_b = (c(\mu)\alpha(\mu) \cdot \varepsilon \cdot t \cdot a) \cdot v \tag{5}$$

where ε is the distortion between block b and its higher resolution child blocks, normalized to $[0,1]$. The motivation behind this modulation is that if a block contains larger distortion, then it should receive a higher priority value for LOD refinements.

C. Multiresolution Error Evaluation

Previously, researchers have proposed various ways to calculate the multiresolution error ε in the scalar data space [1], [19], [29], and in the RGB [5], [12] or the CIELUV [22] color space. In this paper, we take an image-space approach and opt to evaluate the multiresolution error in the perceptually-adapted CIELUV color space, as suggested by Glassner [10].

Let us consider two data blocks b_i and b_j, where b_j is an immediate child block of b_i. We define the multiresolution error between b_i and b_j as follows:

$$\varepsilon_{ij} = \tilde{\sigma}_{ij} \cdot \frac{\tilde{\mu}_i^2 + \tilde{\mu}_j^2 + C_1}{2\tilde{\mu}_i\tilde{\mu}_j + C_1} \cdot \frac{\tilde{\sigma}_i^2 + \tilde{\sigma}_j^2 + C_2}{2\tilde{\sigma}_i\tilde{\sigma}_j + C_2} \tag{6}$$

where $\tilde{\sigma}_{ij}$ is the covariance between b_i and b_j; $\tilde{\mu}_i$ and $\tilde{\mu}_j$ are the mean values of b_i and b_j respectively; $\tilde{\sigma}_i$ and $\tilde{\sigma}_j$ are the standard deviations of b_i and b_j respectively (small constants C_1 and C_2 are included to avoid instability when $\tilde{\mu}_i\tilde{\mu}_j$ and $\tilde{\sigma}_i\tilde{\sigma}_j$ are very close to zero):

$$\tilde{\sigma}_{ij} = \frac{1}{N-1}\sum_{k=1}^{N}(\tilde{x}_{ik} - \tilde{\mu}_i)(\tilde{x}_{jk} - \tilde{\mu}_j) \ ; \tag{7}$$

$$\tilde{\sigma}_i = \frac{1}{N-1}\sum_{k=1}^{N}(\tilde{x}_{ik} - \tilde{\mu}_i)^2 \ ; \quad \tilde{\sigma}_j = \frac{1}{N-1}\sum_{k=1}^{N}(\tilde{x}_{jk} - \tilde{\mu}_j)^2 \ . \tag{8}$$

Here, N is the number of voxels in the block, and $\tilde{x}$ is the volume source term. Using Eqn. 1 and 6, we can calculate the multiresolution error for each tree node as we build up the multiresolution data hierarchy. Eqn. 6 consists of three parts, namely, *covariance*, *luminance distortion*, and *contrast distortion*. The first part is the covariance between b_i and b_j, which measures the degree of linear correlation between the two blocks ($\tilde{\sigma}_{ij}$ is always non-negative since we actually calculate it based on the CIELUV color differences of the pairs $(\tilde{x}_{ik}, \tilde{\mu}_i)$ and $(\tilde{x}_{jk}, \tilde{\mu}_j)$, as explained in Eqn. 9 and 10). Even though b_i and b_j are linearly related, there still might be relative distortions between them. Therefore, we add two more parts to the equation. The second one, measures how close the mean luminance is between b_i and b_j. The minimum value of 1.0 is achieved if and only if $\tilde{\mu}_i = \tilde{\mu}_j$. On the other hand, $\tilde{\sigma}_i$ and $\tilde{\sigma}_j$ can be viewed as estimate of the contrast of b_i and b_j, so the third part measures how similar the contrasts of the two blocks are. Also, the minimum value of 1.0 is achieved if and only if $\tilde{\sigma}_i = \tilde{\sigma}_j$. Collectively, these three parts capture the distortion between the two blocks. The luminance distortion and contrast distortion are originally from the image quality assessment literature [30], and have been shown to be consistent with the luminance masking and contrast masking features in the HVS respectively.

One should notice that the input source terms, $\tilde{x}$ and $\tilde{\mu}$, are CIELUV color values, rather than original scalar data values. Accordingly, we define $\tilde{x}_{ik} - \tilde{\mu}_i$ as follows ($\tilde{x}_{jk} - \tilde{\mu}_j$ can be defined similarly):

$$\tilde{x}_{ik} - \tilde{\mu}_i = \Delta E(f(c_{rgb}(x_{ik})\alpha(x_{ik})), \ f(c_{rgb}(\mu_i)\alpha(\mu_i))) \tag{9}$$

where x_{ik} is the scalar data value at the kth voxel position in block b_i; μ_i is the mean scalar value of b_i; c_{rgb} and α define the color and opacity transfer function; f is the function that converts an RGB color to its CIELUV color [6]; ΔE is the Euclidean distance between a pair of colors specified in the CIELUV color space:

$$\Delta E = \sqrt{\Delta L^{*2} + \Delta u^{*2} + \Delta v^{*2}} \tag{10}$$

where ΔL^*, Δu^*, and Δv^* are the differences of L^*, u^*, and v^* components for the pair of CIELUV colors.

D. Summary Table Scheme

As we can see, the calculation of multiresolution error ε_{ij} in Eqn. 6 requires the input of the color and opacity transfer function (Eqn. 9). At run time, whenever the user adjusts the transfer function, the multiresolution errors in the entire data hierarchy have to be recomputed all over again. This entails a considerable amount of computation overhead and makes the whole LOD selection and rendering process non-interactive. In the following, we describe a summary table scheme that ensures real-time update of the errors in response to transfer function changes.

Our summary table scheme is based on the observation that, for large data sets, the size of the data range is often many orders of magnitude smaller than the number of voxels in the volume. For instance, the RMI data set is byte (8-bit) data with a data range size of 256. However, the number of voxels in the volume is $2048 \times 2048 \times 1920$. Therefore, instead of calculating Eqn. 7 and 8 by going through the individual voxels, it suffices to count the frequencies of unique error terms, which is much faster (similar observations have been made and utilized in [5], [18]). In the case of byte data, there are $256^2 = 65536$ combinations for $\tilde{\sigma}_{ij}$, and only 256 cases for $\tilde{\sigma}_i$ or $\tilde{\sigma}_j$. To compute the error, rather than adding individual error terms voxel by voxel, we add the products of each unique error term and the frequency of that term.

To realize this, first of all, for each of the data blocks at the multiresolution hierarchy, we precompute the mean scalar value μ, and keep a local *histogram table* $\mathbb{H}$ (256 entries):

$$m = \mathbb{H}\ (x_i)$$

where x_i is the scalar value, m is the frequency of x_i in the block.

Next, for each data block associated with a non-leaf node in the hierarchy, we keep a local *correspondence table* $\mathbb{C}$ (65536 entries):

$$m = \mathbb{C}\ (x_i,\ x_j)$$

where x_i is the scalar data value in the current (parent) block; x_j is the data value in one if its eight immediate child blocks; m is the frequency of the data pair. We refer to these histogram and correspondence tables as *summary tables*. They are created during the construction of the data hierarchy and are precomputed only once. Besides this, we keep a global *distance table* $\mathbb{D}$ ($1+2+\ldots+255 = 32640$ entries):

$$\Delta E = \mathbb{D}\ (x_i,\ x_j)$$

where x_i and x_j are scalar data values, and $x_i < x_j$; ΔE is the distance between x_i and x_j in the CIELUV color space.

Finally, we keep a global *function table* $\mathbb{F}$ (the number of entries in the transfer function, usually 256):

$$L^*u^*v^* = \mathbb{F}\ (rgb\alpha)$$

where $rgb\alpha$ is the RGB color and opacity in the current transfer function, $L^*u^*v^*$ is the corresponding CIELUV color. The global distance table and function table are initialized at run time and are updated when the transfer function changes.

At run time, we can quickly calculate the multiresolution error ε for each block using Eqn. 1 and 6-10, by looking up the mean scalar value μ and summary tables ($\mathbb{H}$ and $\mathbb{C}$) stored in each of the blocks, as well as the global distance table $\mathbb{D}$ and function table $\mathbb{F}$. The lookup relationships are as follows:

$$\begin{aligned} \tilde{\mu}_i, \tilde{\mu}_j &\leftarrow \mu, \mathbb{F}\ ; \\ \tilde{\sigma}_i, \tilde{\sigma}_j &\leftarrow \mathbb{H}, \mathbb{F}\ ; \\ \tilde{\sigma}_{ij} &\leftarrow \mathbb{C}, \mathbb{D}\ . \end{aligned}$$

Whenever the user changes the transfer function, only the global distance table and function table need update.

For data sets other than byte data, quantization is necessary in order to reduce the size of summary tables (otherwise, the size of these tables could be even larger than the size of actual data blocks, and the time for error calculation would increase dramatically). For example, we can quantize the scalar data range into 256 levels either uniformly or based on the histogram of the whole data set. In this way, the total size of the summary tables will remain small regardless of the data type of the input volume.

One can observe that, usually there is a strong degree of correlation between parent and child blocks in the data hierarchy. This means that in the correspondence table $\mathbb{C}$, when x_i is close to x_j (i.e., the entry is close to the major diagonal of the table), the frequency m is large. m is smaller, actually often zero, if the entry is further away from the major diagonal. Leveraging this observation, we can perform run-length encoding on the correspondence table $\mathbb{C}$ in a zigzag manner, as illustrated in Fig. 2. The zigzag run-length encoding not only reduces the storage of correspondence tables, but also improves the run-time performance. For instance, using the run-length encoded correspondence tables $\mathbb{C}$ for the RMI data set, the total size of summary tables reduces from 208MB to 44.1MB, and accordingly, the time to update multiresolution errors decreases from 43 seconds to 13 seconds.

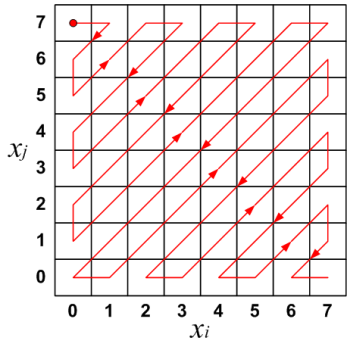


Fig. 2. Run-length encoding on the correspondence table $\mathbb{C}$ in a zigzag manner. An example of an 8×8 table is shown here. The encoding starts from the red circle, and follows the red arrows.

E. GPU-Based Visibility Estimation

Obtaining the exact visibility of the multiresolution data blocks requires rendering the blocks. This is similar to rendering the entire hierarchy, which could be rather slow and defeats the purpose of the visibility test. For coarse-grained multiresolution rendering, getting an approximate visibility of a block suffices. In this scenario, the visibility computation should be done prior to the actual rendering of blocks.

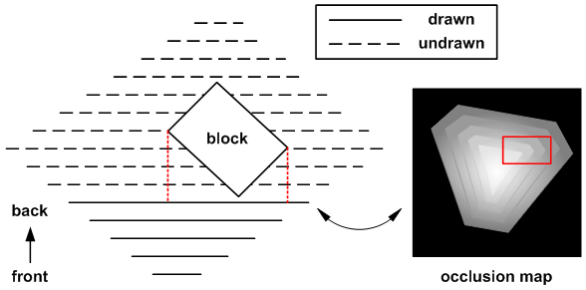


Fig. 3. Visibility estimation via rendering a low resolution of the data. The visibility of a block is acquired when its nearest vertex is in-between the current slice and the latest drawn one.

In our algorithm, we render a low resolution of the data (for example, we can use the root of the data hierarchy) by drawing front-to-back view-aligned slices, and evaluate the approximate visibility of all the blocks during the slice drawing, as illustrated in Fig. 3. The visibility of a block is computed as $(1-\alpha)$, where α is the *average* opacity within the block's screen projection on the occlusion map, accumulated right before the first slice intersecting the block (α is considered as the accumulated opacity in front of that block). Here we assume the visibility of a data block is independent of the resolutions of all the occluding blocks in front of it, because opacity correction is performed to compensate the varying slice distances within data blocks of different resolutions. Note that a conservative way of taking the *minimum* opacity, commonly used in occlusion culling, is unnecessary. For occlusion culling, the decision is to either

render or discard a block, and getting the minimum opacity is crucial to avoid producing incorrect images by leaving holes. For multiresolution rendering, the whole volume is rendered anyway, because the question is to select proper LODs for different regions within the volume, rather than to render or discard a region. Therefore, it is reasonable to get the average instead of the minimum opacity.

To compute the estimated visibility for a data block, a naive way is to read the alpha channel of the framebuffer to an off-screen buffer after a certain number of slices are rendered, and iterate through the pixels that the data block projects to and obtain an average of the opacities. This software approach is slow due to the framebuffer reads from the GPU to the CPU (refer to the timing in Table IV). The testing time is proportional to the size of output images and the number of pixels each block projects to. To minimize the transferring of pixels from the GPU to the CPU, we move all operations to the GPU. Our GPU reduction scheme is based on the *summed area tables* (SATs) [3]. The construction of a SAT is linear to the number of pixels on the area being considered, in our case the whole rendering screen. However, it only takes constant time to retrieve the sum over any rectangular area, which is done in one addition and two subtractions. This fits perfectly in getting the corresponding averages from the projections of the blocks. We build the SATs in multiple passes with the support of framebuffer objects having double auxiliary buffers (see the Appendix for the implementation detail). Getting the estimated visibility for a block is performed by another fragment program that looks up the four corners of its projection in the output auxiliary buffer holding the SAT.

Testing shows that the time to perform GPU-based visibility estimation is not negligible. For instance, with output image resolution of 512^2, each SAT takes around 10ms to compute on an nVidia GeForce 7800 GT graphics card. In total, it took about 0.3 second (recall that we need to recompute the SAT whenever a certain number of slices of the low resolution data are drawn) to update the visibility of 10499 non-empty blocks for the RMI data set. If we perform such a test for every frame, then the frame rate would be limited to around 3.3fps. To overcome this constraint, we incorporate the following two strategies to improve the rendering frame rates.

First, the number of block budget the user specifies is usually much smaller than the total number of blocks in the data hierarchy. For such a typical block budget and a given transfer function, normally a large portion of the updated visibility of blocks farther away from the viewpoint (more likely to be occluded from the blocks in front of them) never gets a chance to contribute to the current LOD decision. Actually, for the RMI data set, tests show that about 30-50% of the total number of blocks fall into this category. In view of this, we can only draw the front slices up to a certain percentage of the total number of slices, and update the corresponding visibility of blocks that are closer to the viewpoint. Any block whose visibility is not updated in this run uses whatever it has from the latest previous run. In this way, we can reduce the visibility estimation time to around 0.18 second for the RMI data set, if we only update 60% of the front slices and blocks.

Second, the visibility of the blocks only changes a little bit if the view does not change greatly. Therefore, if the angle between the current viewing direction and the latest one with the visibility updated is less than a threshold angle θ, we do not update the visibility and use whatever we have from the latest run. Otherwise, we need to update again. Here, θ is a predefined small angle (initialized to 5 degrees in our test), and is adaptive to the zooming of the data during the rendering.

By reducing the load to perform each run of visibility estimation and the frequency of performing such estimations, we can reuse visibility computation and utilize frame to frame coherence, achieving smoother rendering and better frame rates.

F. LOD Selection and Rendering

At run time, the user specifies the number of blocks as a budget for rendering. Given a viewing direction, the LOD selection is made based on a priority queue scheme. The priority values of blocks are their importance values calculated according to Eqn. 5 (where v is updated per view and ε per transfer function). Thus, a block with a higher importance value is more likely to be selected for refinement during the wavelet tree traversal. Constrained by the given budget, the traversal maintains the LOD as a cut through the multiresolution data hierarchy, and refines the blocks on the cut in a greedy manner.

The LOD selection and rendering works as follows: First, we initialize the priority queue with the data block of the lowest resolution, i.e., the root of the multiresolution data hierarchy. Then, we successively refine the block with the highest priority value in the queue until the budget is met. The refinement is performed by deleting the block b with the highest priority value, updating the importance values of b's eight child blocks, and inserting the child blocks into the queue. Finally, all the data blocks in the queue are sorted in front-to-back viewing order. These blocks are reconstructed, if necessary, and sent to texture hardware for rendering.

As we may anticipate reusing most of the reconstructed data blocks for subsequent frames due to the spatial locality and coherence exploited by the multiresolution data hierarchy, it is desirable to cache the data blocks that have already been reconstructed for better performance. The user can predefine a fixed amount of disk space and memory dedicated for the caching purpose. Upon requesting a data block for the rendering, we retrieve its data from memory, provided the block is cached in main memory. Otherwise, we need to load the data from disk if the reconstructed data block is cached on disk. If it is neither cached in memory nor on disk, then we need to reconstruct the data block and load it into main memory. When the system runs short of the available storage for caching the reconstructed blocks, our replacement scheme will swap out a data block that has been visited least often.

V. Results and Discussion

A. Results

We experimented with our LOD selection and rendering algorithm on the VisWoman and RMI data sets, as listed in

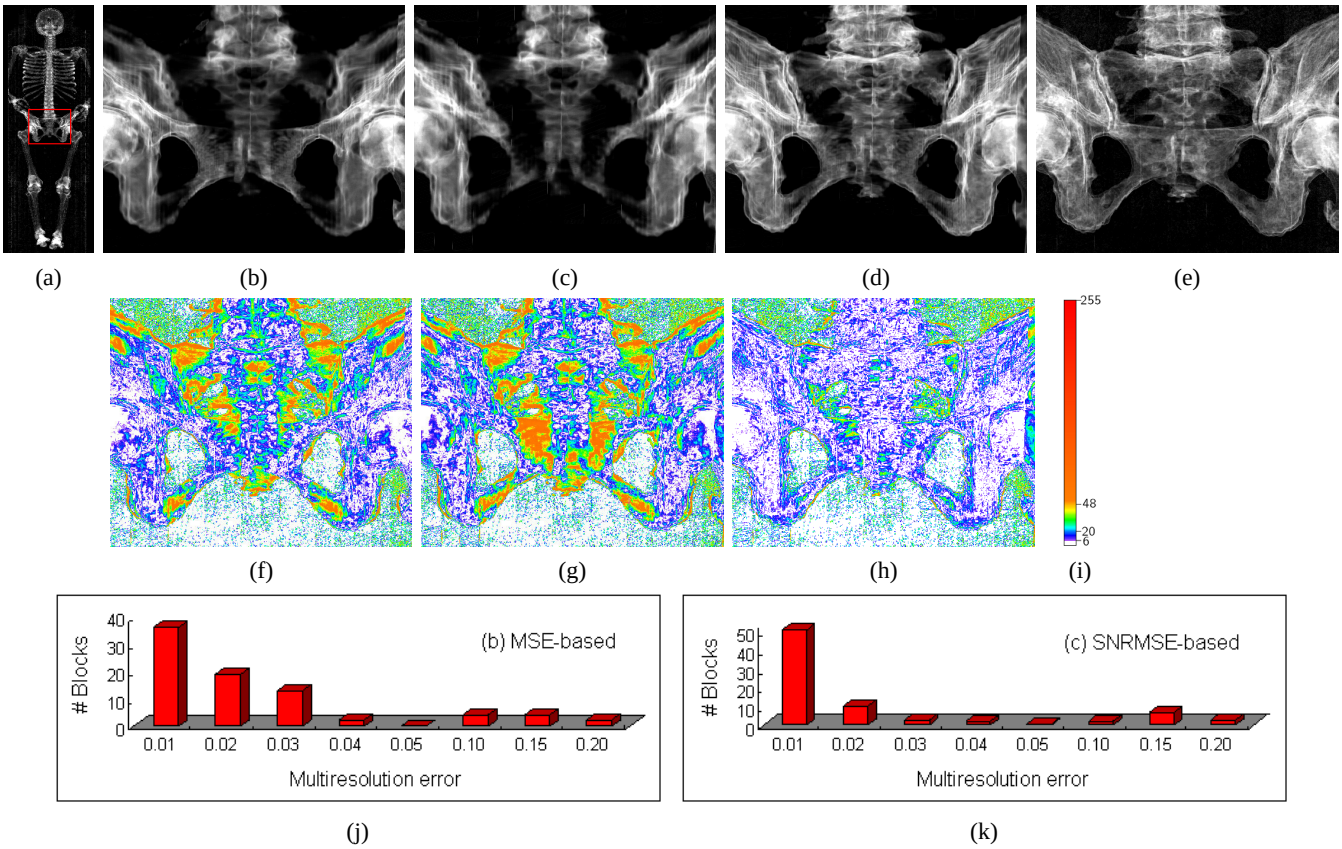


Fig. 4. First row: (a) shows an overview and (b)-(e) show a zoom to the pelvis. One can observe that (d) (image-based, 77 blocks, 8.29% of full data) shows more details than (b) (MSE-based, 80 blocks, 8.61%) and (c) (SNRMSE-based, 79 blocks, 8.50%). The reference image (e) is rendered with full resolution (929 blocks). Second row: objective image comparison in the CIELUV color space. (f), (g), and (h) show the difference between (b) and (e), (c) and (e), and (d) and (e) respectively. The color map (i) maps ΔE to color. Third row: (j) and (k) show the numbers of blocks selected in each of the multiresolution error levels for (b) and (c) respectively.

Table II. The decision for the block size was based on the cost of performing the wavelet transform for a single data block, and the rendering overhead for final image generation. We extended one voxel overlapping boundaries between neighboring blocks in each dimension when breaking the original volume data into blocks in order to produce seamless rendering. As a result, both hierarchies have a tree depth of six. For both data sets, the Haar wavelet transform with a lifting scheme was used to construct the data hierarchies. A *lossless* compression scheme was used with the threshold set to zero to compress the wavelet coefficients. For LOD rendering, we compared the images generated using our image-based quality metric and two data-based metrics: MSE and SNRMSE (MSE of SNR). For the MSE-based (SNRMSE-based) metric, we directly used the multiresolution error as the importance value, while ε_{ij} is the MSE (SNRMSE) of the scalar data values of blocks b_i and b_j in Eqn. 1. Similar block budgets were set for all three cases for fair comparison. All tests were performed on a 3.0GHz Intel Xeon processor with 3GB main memory, and an nVidia GeForce 7800 GT graphics card with 256MB video memory.

The first row of Fig. 4 shows the LOD rendering of the VisWoman data set using the three metrics. The full-resolution reference image is provided for comparison. We used a transfer function that highlights the skeleton. It can be observed that

TABLE II
THE VISWOMAN AND RMI DATA SETS.

data set (type)	VisWoman (short)	RMI (byte)
volume dimension	$512 \times 512 \times 1728$	$2048 \times 2048 \times 1920$
block dimension	$32 \times 32 \times 64$	$128 \times 128 \times 64$
volume (block) size	864MB (128KB)	7.5GB (1MB)
# non-empty blocks	9446	10499
compression ratio	2.37:1	5.60:1

when we rendered the data in low resolution, the LOD selection using the image-based quality metric shows more details than the MSE-based and SNRMSE-based metrics. In Fig. 4 (j) and (k), we compare the numbers of blocks selected in each of the multiresolution error levels for (b) and (c) respectively (the multiresolution errors have been normalized). Assuming that our multiresolution errors are able to capture the structural distortion of the data, we can infer that both MSE-based and SNRMSE-based metrics perform much worse due to their selections of not-so-highly prioritized multiresolution data blocks. An objective image comparison was also conducted to testify the visual quality gain obtained using our image-based quality metric. We calculated the pixel-wise differences between the low resolution image and the reference image in the CIELUV color space. The difference threshold $\Delta E \geq 6.0$

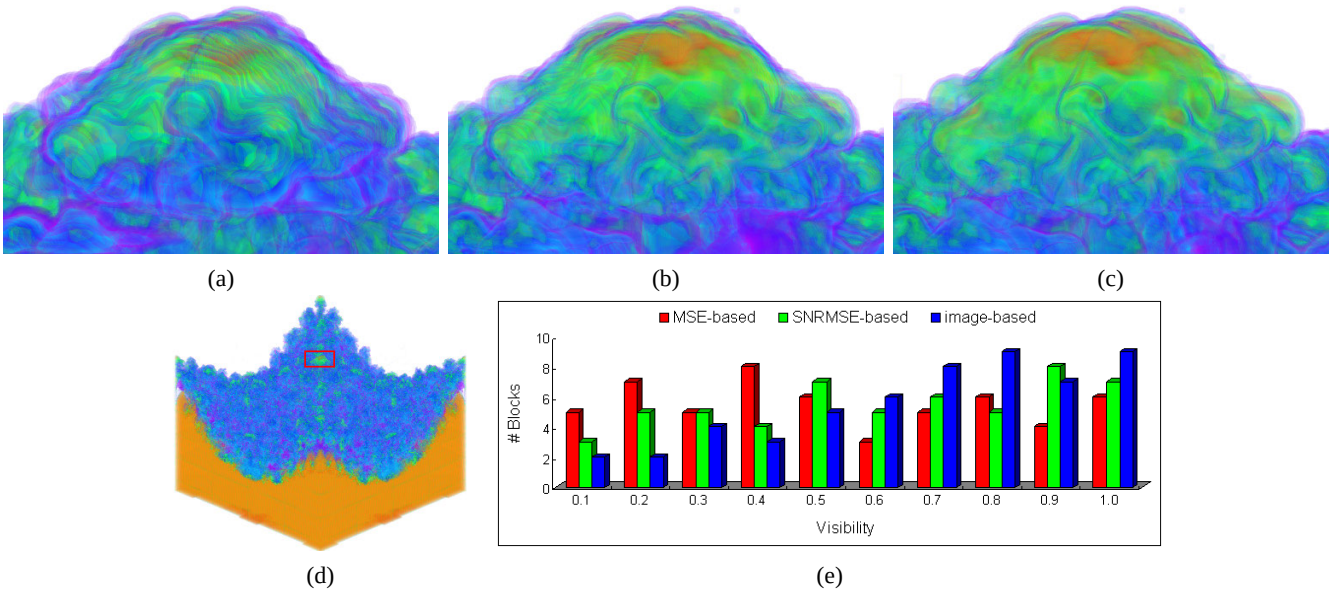


Fig. 5. (a) (MSE-based, 55 blocks), (b) (SNRMSE-based, 55 blocks), and (c) (image-based, 55 blocks) show a zoom to the center of the RMI data set, while an overview is shown in (d). (e) shows the numbers of blocks rendered in each of the ten visibility levels for (a)-(c) respectively.

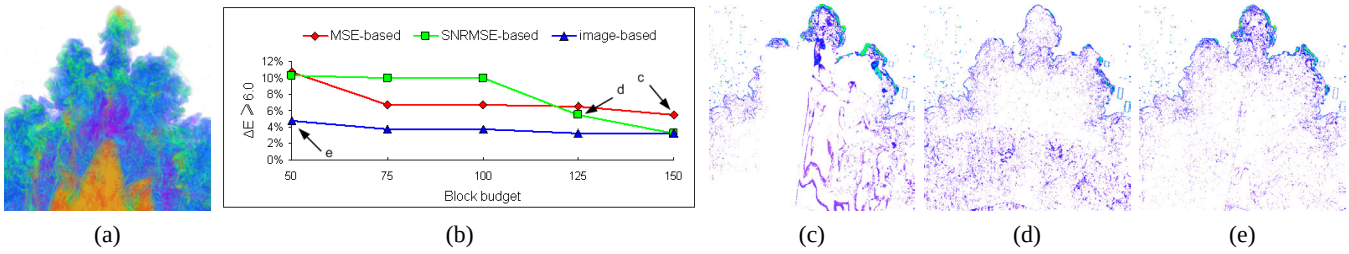
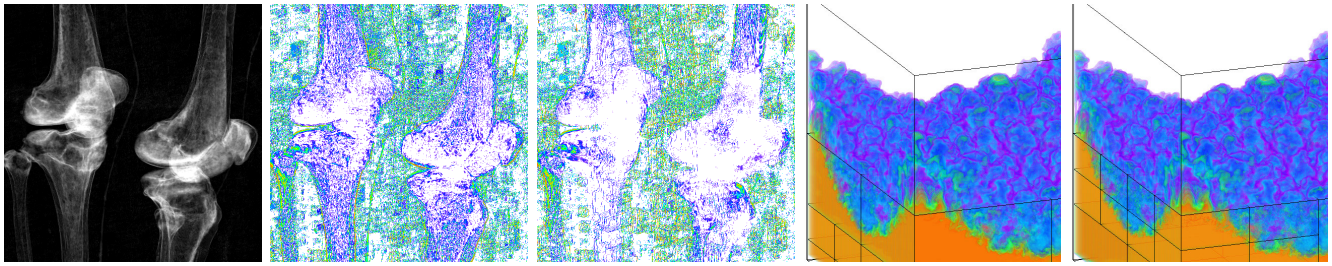


Fig. 6. (a) shows the reference image with full resolution (1237 blocks). (b) shows the percentage of pixels with $\Delta E \geq 6.0$ in the difference images for the three metrics, under five different block budgets. (c)-(e) show three difference images near 5%, as indicated in (b): (c) (MSE-based, 150 blocks), 5.51%, (d) (SNRMSE-based, 128 blocks), 5.48%, and (e) (image-based, 50 blocks), 4.75%.

gives the noticeable pixel distortion [22]. At the second row of Fig. 4, we show these difference images side by side. Clearly, the ones with the MSE-based and SNRMSE-based metrics contain larger visual distortion. Another rendering example of the VisWoman data set is shown in Fig. 1. We can see that the image-based LOD selection shows clearer structures along the spine.

Fig. 5 shows the LOD selection and rendering of the RMI data set using the three metrics. We zoomed into the center of the data and compared fine details after an overview. The image-based quality metric takes into account the multiresolution error and visibility of each data block, thus puts more refinement effort on the blocks that have larger visual contribution. Fig. 5 (e) shows the numbers of blocks rendered in each of the ten visibility levels for Fig. 5 (a)-(c) respectively. As we can see, compared with the ones with MSE and SNRMSE, the image-based one selected more blocks with higher visibility. Similar conclusions can be drawn from Fig. 6, where the image-based quality metric achieves near 5% noticeable pixel distortion with a block budget of only 50, as opposed to 150 and 125 for the MSE-based and SNRMSE-based ones respectively. To verify that including estimated visibility v in Eqn. 5 does help in LOD selections, we tested our image-based LOD selection algorithm without and with visibility information. The results in Fig. 7 show that adding visibility information in the LOD selection leads to more refinement on blocks closer to the viewpoint and with higher visibility. All the results in Fig. 1 and 4-7 confirm the effectiveness of our image-based LOD selection algorithm.

We also experimented with our summary table scheme for updating the multiresolution errors. With 256-level histograms and transfer functions, the statistics is shown in Table III. For both data sets, the zigzag run-length encoding scheme takes at least 70% less time to update the multiresolution errors with much smaller storage overhead than the one with no coding. The summary table scheme proved very efficient in response to the transfer function changes with negligible storage overhead. A rendering example of the VisWoman data set is shown in Fig. 8, where a different transfer function was used to highlight both the skin and the skeleton. We zoomed into the left foot and rendered in close to full resolution. The three methods generated similar results as we approached full resolution. Still, it can be seen that the MSE-based method contains much noise coming from the 3D test bed surrounding the cadaver

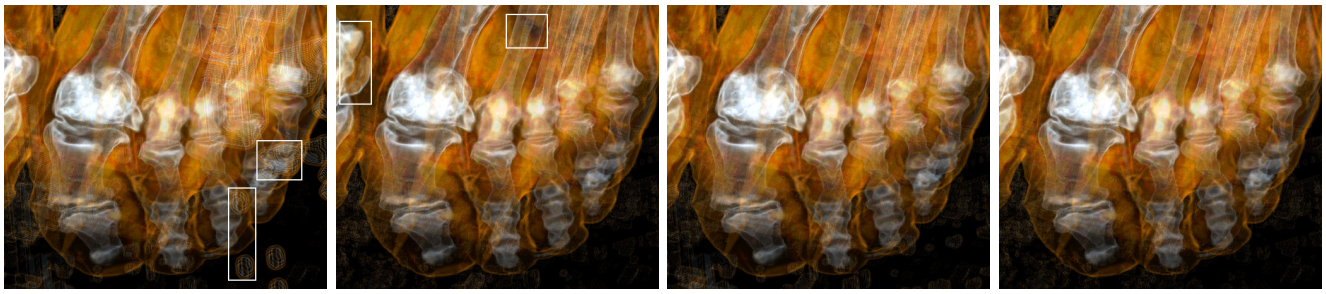


(a) (b) (c) (d) (e)

Fig. 7. (a) shows a zoom to the knee joints of the VisWoman data set, rendered with full resolution (1033 blocks). (b) (w/o visibility, 156 blocks) and (c) (with visibility, 154 blocks) show the different images. (d) (w/o visibility, 37 blocks) and (e) (with visibility, 32blocks) show a zoom of the RMI data set with block boundaries drawn to illustrate different LODs.

TABLE III
THE STATISTICS OF SUMMARY TABLE SCHEMES.

	no encoding		**zigzag run-length encoding**	
data set	**space (overhead)**	**update time**	**space (overhead)**	**update time**
VisWoman	175MB (20.25%)	34s	9.22MB (1.07%)	5s
RMI	208MB (2.71%)	43s	44.1MB (0.57%)	13s



(a) (b) (c) (d)

Fig. 8. (a) (MSE-based, 107 blocks), (b) (SNRMSE-based, 108 blocks), and (c) (image-based, 106 blocks) show a zoom to the left foot, rendered close to full resolution. The reference image (d) is rendered with full resolution (151 blocks). White frames are drawn in (a) and (b) to indicate some of the differences.

(the blocks corresponding to the test bed have larger MSEs yet less visual importance than the blocks corresponding to the foot.). Although the SNRMSE-based one generates the result with the least noise, it leaves some portion rendered in lower resolution, which is discernable when compared with the reference image.

After applying the two strategies on visibility estimation for improving the performance (Section IV-E), we compared the timing of visibility estimation for CPU and GPU solutions. We evaluated the visibility of the multiresolution blocks in the data hierarchy for a wide variety of block budgets (from 10 to 11000), together with different combinations of translation, scale, and rotation. Table IV gives the upper bounds of the timing for the two data sets with four output image resolutions. The timing results show that the solution with the GPU is about three to four times faster than the CPU one.

B. Discussion

Compared with the traditional MSE-based and SNRMSE-based metric, our experience shows that for most of the cases, the image-based quality metric gives LODs better visual quality. This is especially true when the block budget is small (usually below 20%) compared with the number of blocks for full resolution. As one gradually increases the block budget, the three metrics generate closer results as expected. However, as shown in Fig. 8, we still get some improvement over the data-based metrics. Our image-based quality metric performs quite well even when the data contains noise. For example, the VisWoman data set contains noise from the 3D test bed. Fig. 4 shows that the image-based quality metric is insensitive to the noise and captures the structural distortion of the data, since more refinement effort was put on the blocks corresponding to the pelvis. This result is consistent with the image quality measure using structural similarity [30]. Finally, we compared our image-based quality metric with Guthe's screen-space metric (refer to Section II-B). We used the same estimated visibility with an implementation of the screen-space error and tested both data sets. The results in Fig. 9 show the advantage of our image-based quality metric over the screen-space error metric.

Our summary table scheme works well when the space overhead for storing the tables is small, compared with the

TABLE IV
THE TIMING OF VISIBILITY ESTIMATION WITH DIFFERENT OUTPUT IMAGE RESOLUTIONS.

	VisWoman						
image resolution	**CPU**	**draw**	**FB read**	**get avg**	**GPU**	**draw**	**SAT**
256×256	0.219s	8.72%	63.37%	27.91%	0.072s	9.72%	90.28%
512×512	0.578s	3.84%	58.59%	37.57%	0.151s	17.21%	82.79%
768×768	1.078s	2.56%	52.96%	44.48%	0.312s	17.63%	82.37%
1024×1024	1.531s	1.79%	42.83%	55.38%	0.487s	21.56%	78.44%
	RMI						
256×256	0.359s	6.41%	56.02%	37.57%	0.103s	9.71%	90.29%
512×512	0.828s	3.53%	52.16%	44.31%	0.185s	17.84%	82.16%
768×768	1.594s	2.09%	44.30%	53.61%	0.372s	16.13%	83.87%
1024×1024	2.338s	1.35%	34.47%	64.18%	0.538s	20.45%	79.55%
FB read = framebuffer read, **get avg** = get average occlusion							

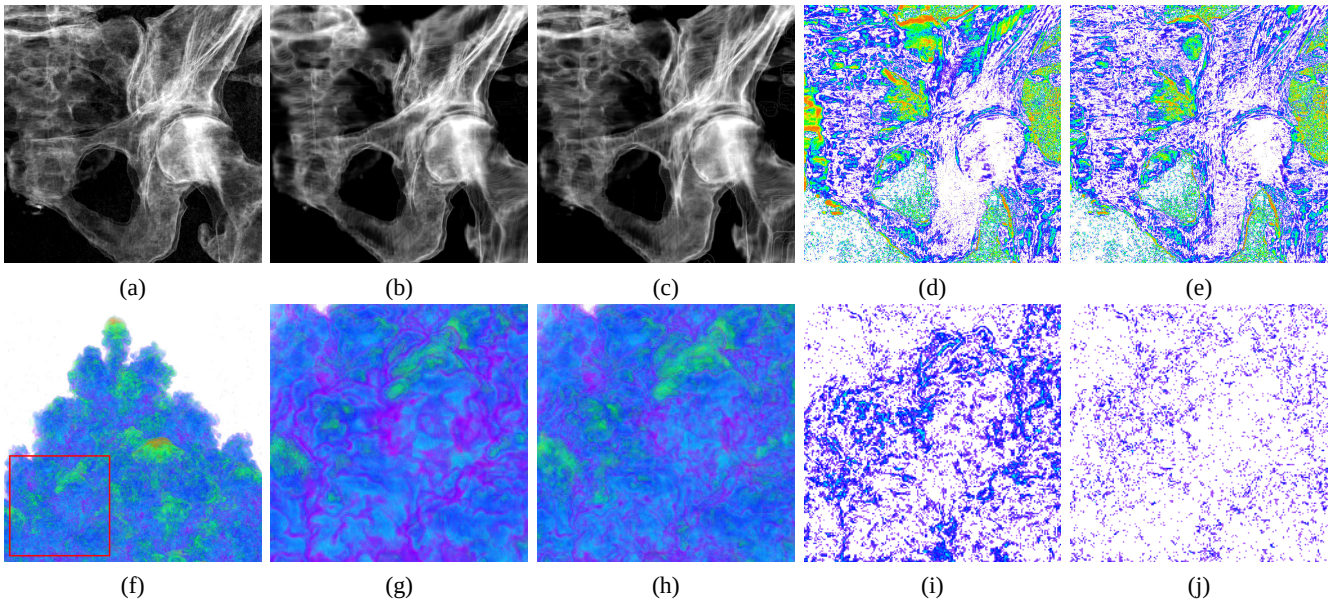


Fig. 9. First row: (a) (full resolution, 486 blocks), (b) (Guthe's screen-space metric, 56 blocks), and (c) (our image-based metric, 53 blocks) show a zoom to the left pelvis. (d) and (e) show the difference between (b) and (a), and (c) and (a) respectively. Second row: (f) shows a zoom rendered in high resolution (427 blocks). (g) (Guthe's screen-space metric, 43 blocks), and (h) (our image-based metric, 41 blocks) show a zoom of (f). (i) and (j) show the difference images for (g) and (h) respectively.

size of the input data set. In our experiments, good results were obtained with block sizes of 32^3 or 64^3 for a data set of size around 1024^3. This allows one to have histogram tables with sufficient entries (such as 256 in our experiments), while still keeping the scheme efficient. For a smaller block size, such as 16^3, we can reduce the number of entries in the histogram table, or use a simplified histogram to maintain an effective tradeoff between storage and processing requirements versus having enough precision for summary tables generation. On the other hand, our current solution is suitable for value-based transfer functions. There is a need of further research on generalizing the summary table scheme for multidimensional transfer functions.

To our knowledge, we are the first to utilize the GPU implementation of SATs for visibility estimation. We tested our GPU-based algorithm on the nVidia GeForce 7800 GT graphics card, which is based on the new generation PCI Express bus architecture. With PCI Express, the bandwidth between the CPU and the GPU increases to over 4GB per second in both upstream and downstream data transfers. In this case, framebuffer reads become less a constraint for the CPU-based solution. Still, it can be seen from Table IV that framebuffer reads take at least one third of the total time for the CPU solution. Our experiment reports that utilizing the GPU for visibility estimation, one can achieve a speedup up to four times.

For a typical output image resolution of 512^2, the summary of block classification is listed in Table V. For data-based metrics, the classification time is almost negligible since we only use the MSE or SNRMSE for block prioritization. For our image-based metric, taking into account the time for visibility estimation, we are able to prioritize all the multiresolution data blocks for LOD selection at a speed of 19.1kblocks/s for the VisWoman data set, and 14.0kblocks/s for the RMI data set. This result is comparable to the significance classification performance presented in [22], considering that we take a more

TABLE V
THE SUMMARY OF BLOCK CLASSIFICATION WITH 512^2 OUTPUT IMAGE RESOLUTION.

data set	VisWoman	RMI
block dimension	$32 \times 32 \times 64$	$128 \times 128 \times 64$
# blocks	9446	10499
data-based	0.028s	0.032s
visibility	0.151s	0.185s
prioritization	0.343s	0.563s
transfer function	5s	13s

refined and exact solution for block classification and visibility estimation. Note that the timing for visibility and prioritization gives the upper bound for all blocks. In actual rendering, the block budget is usually around tens to hundreds, and the classification time is much smaller than the upper bound (for instance, the prioritization time for the RMI data set is 0.016 second when the block budget is 1000). The classification of the blocks in the entire data hierarchy can be finished within seconds even if the user changes the transfer function at run time. This timing performance could be further improved with the support of transfer function preview at reduced resolutions. For example, the transfer function update time reduce to 1.7 seconds and 4.8 seconds from Table III for the VisWoman and RMI data sets respectively, if both use 64-level rather than 256-level transfer functions.

VI. CONCLUSION AND FUTURE WORK

The focus of this work is to develop an image-based LOD selection algorithm for large volumetric data, and produce images of better visual quality compared with traditional data-based LOD selection algorithms, under similar block budgets. In this paper, we have presented an interactive LOD selection and rendering algorithm through the use of an image-based quality metric. Experimental results on large scientific and medical data sets demonstrate the effectiveness and efficiency of our image-based LOD selection algorithm.

Our approach is promising due to its generality and flexibility. The summary table scheme greatly alleviates the dependence of the error calculation on the transfer function, and thus allows one to update the errors within seconds whenever the transfer function changes. The GPU reduction scheme for visibility estimation is not limited to multiresolution volume rendering, and is readily applicable to other large volume visualization scenarios that capitalize on the visibility information. Moreover, the hierarchical data representation and the user-specified budget for rendering make our LOD selection scheme suitable for time-critical multiresolution volume rendering and remote visualization applications. Finally, one can have different definitions and thus different ways of measurement for the multiresolution error in Eqn. 5, which we plan to explore more. In the future, we also would like to extend our method for large-scale time-varying data visualization.

APPENDIX
THE GPU IMPLEMENTATION OF SATS

Nowadays, GPUs and fragment programs support *render-to-texture* (RTT) with 32-bit floating-point channels for *pixel buffers* (pbuffers) as well as *framebuffer objects* (FBOs). This is important for the construction of SATs since the sums require more precision. Our implementation uses the `GL_EXT_framebuffer_object` extension to avoid the context switching of pbuffers. Given an input FBO, a SAT can be built by successively adding the columns from left to right and then the rows from bottom to top. However, this requires that the FBO is treated as both an input and an output texture, which highly depends on the kinds of hardware and graphics library available. To date, most implementations do not have this capability. Therefore, we take an alternative and build the SATs in passes with the support of FBOs having double auxiliary buffers: one is the input, and the other is the output. Both auxiliary buffers have the same size as the rendering framebuffer. At the beginning, the alpha values from the rendering buffer is mapped to the input buffer.

The construction of a SAT in the GPU is as follows: First of all, we operate on the columns. At the ith pass, each texel $T_i(x,y)$ in the output buffer is updated using two texels from the input buffer according to the following equation:

$$T_i(x,y) = T_{i-1}(x,y) + T_{i-1}(x - 2^{i-1}, y) \tag{11}$$

where sampling off texture returns zero and does not affect the sum. At the end of each pass, the auxiliary buffers are swapped. For a rendering screen with resolution of n^2, the number of passes needed is $\log_2(n)$. Then, the process is repeated over the rows in a similar way to complete the SAT construction.

The GPU implementation of SATs was first given by Green [11] from nVidia, where a simple scanline-based algorithm was presented and used for antialiasing in the traditional way. For an input table of size n^2, the number of passes is $2n$. Our implementation requires $2\log_2(n)$ passes with two sample reads per pass. Actually, this could be further improved by performing up to 16 sample reads per pass. Hensley et al. [14] performed study on the tradeoff between the number of rendering passes and the number of samples per pass. The optimal tradeoff between the number of passes and the cost per pass largely depends on the overhead of render target switches and the design of the texture cache on the target platform.

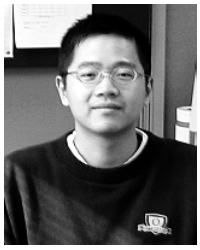
ACKNOWLEDGEMENTS

This work was supported by NSF ITR grant ACI-0325934, DOE Early Career Principal Investigator Award DE-FG02-03ER25572, NSF Career Award CCF-0346883, and Oak Ridge National Laboratory Contract 400045529. The VisWoman data set is courtesy of The National Library of Medicine, and the RMI data set is courtesy of Mark Duchaineau at Lawrence Livermore National Laboratory. The first author would like to thank Jian Huang, Ross J. Toedte, and Jinzhu Gao for initial discussions of this work. Finally, the authors would like to thank the reviewers for their helpful comments.

REFERENCES

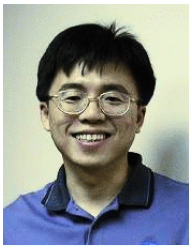
[1] I. Boada, I. Navazo, and R. Scopigno. Multiresolution Volume Visualization with a Texture-Based Octree. *The Visual Computer*, 17(3):185–197, 2001.

[2] D. Cohen-Or, Y. L. Chrysanthou, C. T. Silva, and F. Durand. A Survey of Visibility for Walkthrough Applications. *IEEE Trans. on Visualization & Computer Graphics*, 9(3):412–431, 2003.
[3] F. C. Crow. Summed-Area Tables for Texture Mapping. In *Proc. of ACM SIGGRAPH '84*, pages 207–212, 1984.
[4] J. El-Sana, N. Sokolovsky, and C. T. Silva. Integrating Occlusion Culling with View-Dependent Rendering. In *Proc. of IEEE Visualization '01*, pages 371–575, 2001.
[5] D. Ellsworth, L.-J. Chiang, and H.-W. Shen. Accelerating Time-Varying Hardware Volume Rendering Using TSP Trees and Color-Based Error Metrics. In *Proc. of IEEE Volume Visualization '00*, pages 119–129, 2000.
[6] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles & Practice in C (2nd Edition)*. Addison Wesley, 1995.
[7] A. Gaddipati, R. Machiraju, and R. Yagel. Steering Image Generation with Wavelet Based Perceptual Metric. In *Proc. of Eurographics '97*, 1997.
[8] J. Gao, J. Huang, H.-W. Shen, and J. A. Kohl. Visibility Culling Using Plenoptic Opacity Functions for Large Volume Visualization. In *Proc. of IEEE Visualization '03*, pages 341–348, 2003.
[9] M. H. Ghavamnia and X. D. Yang. Direct Rendering of Laplacian Pyramid Compressed Volume Data. In *Proc. of IEEE Visualization '95*, pages 192–199, 1995.
[10] A. S. Glassner. *Principle of Digital Image Synthesis, Volume 1*. Morgan Kaufmann, 1995.
[11] S. Green. Summed Area Tables using Graphics Hardware. In *Game Developers Conference '03*, 2003.
[12] S. Guthe and W. Straßer. Advanced Techniques for High-Quality Multi-Resolution Volume Rendering. *Computers & Graphics*, 28(1):51–58, 2004.
[13] S. Guthe, M. Wand, J. Gonser, and W. Straßer. Interactive Rendering of Large Volume Data Sets. In *Proc. of IEEE Visualization '02*, pages 53–60, 2002.
[14] J. Hensley, T. Scheuermann, G. Coombe, M. Singh, and A. Lastra. Fast Summed-Area Table Generation and its Applications. In *Proc. of Eurographics '05*, pages 547–555, 2005.
[15] C. E. Jacobs, A. Finkelstein, and D. H. Salesin. Fast Multiresolution Image Querying. In *Proc. of ACM SIGGRAPH '95*, pages 277–286, 1995.
[16] J. T. Klosowski and C. T. Silva. The Prioritized-Layered Projection Algorithm for Visible Set Estimation. *IEEE Trans. on Visualization & Computer Graphics*, 6(2):108–123, 2000.
[17] E. LaMar, B. Hamann, and K. I. Joy. Multiresolution Techniques for Interactive Texture-Based Volume Visualization. In *Proc. of IEEE Visualization '99*, pages 355–362, 1999.
[18] E. LaMar, B. Hamann, and K. I. Joy. Efficient Error Calculation for Multiresolution Texture-Based Volume Visualization. In *Hierarchical & Geometrical Methods in Scientific Visualization*, pages 51–62, 2003.
[19] D. Laur and P. Hanrahan. Hierarchical Splatting: A Progressive Refinement Algorithm for Volume Rendering. In *Proc. of ACM SIGGRAPH '91*, pages 285–288, 1991.
[20] M. Levoy. Efficient Ray Tracing of Volume Data. *ACM Trans. on Graphics*, 9(3):245–261, 1990.
[21] X. Li and H.-W. Shen. Time-Critical Multiresolution Volume Rendering Using 3D Texture Mapping Hardware. In *Proc. of IEEE Volume Visualization '02*, pages 29–36, 2002.
[22] P. Ljung, C. Lundström, A. Ynnerman, and K. Museth. Transfer Function Based Adaptive Decompression for Volume Rendering of Large Medical Data Sets. In *Proc. of IEEE Volume Visualization '04*, pages 25–32, 2004.
[23] N. Max. Optical Models for Direct Volume Rendering. *IEEE Trans. on Visualization & Computer Graphics*, 1(2):99–108, 1995.
[24] M. Meißner, J. Huang, D. Bartz, K. Muller, and R. Crawfis. A Practical Evaluation of Popular Volume Rendering Algorithms. In *Proc. of IEEE Volume Visualization '00*, pages 81–90, 2000.
[25] S. Muraki. Approximation and Rendering of Volume Data Using Wavelet Transforms. In *Proc. of IEEE Visualization '92*, pages 21–28, 1992.
[26] T. Porter and T. Duff. Compositing Digital Images. In *Proc. of ACM SIGGRAPH '84*, pages 253–259, 1984.
[27] N. Sahasrabudhe, J. E. West, R. Machiraju, and M. Janus. Structured Spatial Domain Image and Data Comparison Metrics. In *Proc. of IEEE Visualization '99*, pages 97–104, 1999.
[28] C. Ware. *Information Visualization: Perception for Design (2nd Edition)*. Morgan Kaufmann, 2004.
[29] C. Wang, J. Gao, and H.-W. Shen. Parallel Multiresolution Volume Rendering of Large Data Sets with Error-Guided Load Balancing. In *Proc. of Eurographics Parallel Graphics & Visualization '04*, pages 23–30, 2004.
[30] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Trans. on Image Processing*, 13(4):600–612, 2004.
[31] R. Westermann. A Multiresolution Framework for Volume Rendering. In *Proc. of IEEE Volume Visualization '94*, pages 51–58, 1994.
[32] J. Wilhelms and A. van Gelder. Multi-Dimensional Trees for Controlled Volume Rendering and Compression. In *Proc. of IEEE Volume Visualization '94*, pages 27–34, 1994.
[33] H. Zhou, M. Chen, and M. F. Webster. Comparative Evaluation of Visualization and Experimental Results Using Image Comparison Metrics. In *Proc. of IEEE Visualization '02*, pages 315–322, 2002.



Chaoli Wang received the BE and ME degrees in computer science from Fuzhou University, China in 1998 and 2001 respectively. Currently, he is a PhD candidate in the Department of Computer Science and Engineering at The Ohio State University. His research interests are computer graphics and visualization, with focus on developing algorithms for managing and rendering large-scale three-dimensional and time-varying data. He is a student member of the IEEE.

Antonio Garcia received the BS degree in systems engineering from Universidad Particular "Antenor Orrego" in 1993, the MS degree in computer science from The Ohio State University in 1999. He completed his PhD dissertation at The Ohio State University in 2005. His research was focused on parallel rendering, volume rendering, and GPU programming. Currently, he is working for Intel Corporation in the production, debugging and testing of display drivers for Intel's chipsets and GPUs.



Han-Wei Shen received his BS degree from Department of Computer Science and Information Engineering at National Taiwan University in 1988, the MS degree in computer science from the State University of New York at Stony Brook in 1992, and the PhD degree in computer science from the University of Utah in 1998. From 1996 to 1999, he was a research scientist at NASA Ames Research Center in Mountain View California. He is currently an Associate Professor at The Ohio State University. His primary research interests are scientific visualization and computer graphics. Professor Shen is a winner of US Department of Energy's Early Career Principal Investigator Award and National Science Foundation's CAREER award. He also won an Outstanding Teaching award in the Department of Computer Science and Engineering at The Ohio State University.

A Statistical Approach to Volume Data Quality Assessment

Chaoli Wang, *Member, IEEE,* and Kwan-Liu Ma, *Senior Member, IEEE*

***Abstract*— Quality assessment plays a crucial role in data analysis. In this paper, we present a reduced-reference approach to volume data quality assessment. Our algorithm extracts important statistical information from the original data in the wavelet domain. Using the extracted information as feature and predefined distance functions, we are able to identify and quantify the quality loss in the reduced or distorted version of data, eliminating the need to access the original data. Our feature representation is naturally organized in the form of multiple scales, which facilitates quality evaluation of data with different resolutions. The feature can be effectively compressed in size. We have experimented with our algorithm on scientific and medical data sets of various sizes and characteristics. Our results show that the size of the feature does not increase in proportion to the size of original data. This ensures the scalability of our algorithm and makes it very applicable for quality assessment of large-scale data sets. Additionally, the feature could be used to repair the reduced or distorted data for quality improvement. Finally, our approach can be treated as a new way to evaluate the uncertainty introduced by different versions of data.**

***Index Terms*— Quality assessment, reduced reference, wavelet transform, statistical modeling, generalized Gaussian density, volume visualization.**

I. Introduction

LEVERAGING the power of supercomputers, scientists can now simulate many things from galaxy interaction to molecular dynamics in unprecedented details, leading to new scientific discoveries. The vast amounts of data generated by these simulations, easily reaching tens of terabytes, however, present a new range of challenges to traditional data analysis and visualization. A time-varying volume data set produced by a typical turbulent flow simulation, for example, may contain thousands of time steps with each time step having billions of voxels and each voxel recording dozens of variables. As supercomputers continue to increase in size and power, petascale data is just around the corner.

A variety of data reduction methods have been introduced to make the data movable and enable interactive visualization, offering scientists options for studying their data. For instance, subsets of the data may be stored at a reduced precision or resolution. Data reduction can also be achieved with transform-based compression methods. A popular approach is to generate a multiresolution representation of the data such that a particular level of details is selected according to the visualization requirements and available computing resources. In addition, data may be altered in other fashions. Furthermore, it could be desirable to smooth the data or enhance a particular aspect of the data before rendering. Finally, the data may be distorted or corrupted during the transmission over a network.

C. Wang and K.-L. Ma are with the Visualization and Interface Design Innovation (VIDI) research group, Department of Computer Science, University of California, Davis, 2063 Kemper Hall, One Shields Avenue, Davis, CA 95616. E-mail:{wangcha, ma}@cs.ucdavis.edu.

Research has been conducted to evaluate the quality of rendered images *after* the visualization process [6], [29]. However, few studies focus on analyzing the data quality *before* the visualization actually takes place. It is clear that the original volume data may undergo various changes due to quantization, compression, sampling, filtering, and transmission. If we assume the original data has full quality, all these changes made to the data may incur quality loss, which may also affect the final visualization result. In order to compare and possibly improve the quality of the reduced or distorted data, it is important for us to identify and quantify the loss of data quality. Unequivocally, the most widely used data quality metrics are *mean square error* (MSE) and *peak signal-to-noise ratio* (PSNR). Although easy to compute, they do not correlate well with perceived quality measurement [18]. Moreover, these metrics require access to the original data and are *full-reference* methods. They are not applicable to our scenario, since the original data may be too large to acquire or compare in an efficient way. Therefore, it is highly desirable to develop a data quality assessment method that does not require full access of the original data.

In this paper, we introduce a *reduced-reference* approach to volume data quality assessment. We consider the scenario where a set of important statistical information is first extracted from the original data. For example, the extraction process could be performed at the supercomputer centers where the large-scale data are produced and stored, or ideally, in situ when the simulation is still running. We then compress the feature information to minimize its size. This makes it easy to transfer the feature to the user as "carry-on" information for volume data quality assessment, eliminating the need to access the original data again. Our feature representation not only serves as the criterion for data quality assessment, but also could be used as quality improvement to repair the reduced or distorted data. This is achieved by matching some of its feature components with those extracted from the original data. We have tested our algorithm on scientific and medical data sets with various sizes and characteristics to demonstrate its effectiveness.

II. Background and Related Work

Unlike the Fourier transform with sinusoidal basis functions, the wavelet transform is based on small waves, called *wavelets*, of varying frequency and limited duration [7]. Wavelet transforms provide a convenient way to represent localized signals

simultaneously in space and frequency. The particular kind of dual localization makes many functions and operators using wavelets "sparse" when transformed into the wavelet domain. This sparseness, in turn, brings us a number of useful applications such as data compression, feature detection, and noise removal.

Besides sparseness, wavelets have many other favorable properties, such as multiscale decomposition structure, linear time and space complexity of the transformations, decorrelated coefficients, and a wide variety of basis functions. Studies of the *human visual system* (HVS) support a multiscale analysis approach, since researchers have found that the visual cortex can be modeled as a set of independent channels, each with a particular orientation and spatial frequency tuning [4], [21]. Therefore, wavelet transforms have been extensively used to model the processing in the early stage of biological visual systems. They have also gained much popularity, and have become the preferred form of representation for image processing and computer vision algorithms.

In volume visualization, Muraki [15] introduced the idea of using the wavelet transform to obtain a unique shape description of an object, where a 2D wavelet transform is extended to 3D and applied to eliminate wavelet coefficients of lower importance. Over the years, many wavelet-based techniques have been developed to compress, manage, and render three-dimensional [8], [11] and time-varying volumetric data [12], [23]. They are also used to support fast access and interactive rendering of data at runtime. In this paper, we employ the wavelet transform to generate multiscale decomposition structures from the input data for feature analysis.

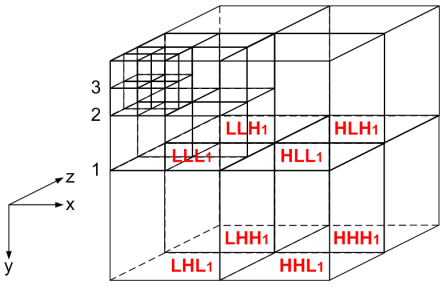


Fig. 1. Multiscale wavelet decomposition of a three-dimensional volumetric data. L = low-pass filtered; H = high-pass filtered. The subscript indicates the level and a larger number corresponds to a coarser scale (lower resolution). An example of three levels of decomposition is shown here.

The wavelet transform on a one-dimensional signal can be regarded as filtering the signal with both the scaling function (a low-pass filter) and the wavelet function (a high-pass filter), and downsampling the resulting signals by a factor of two. The extension of the wavelet transform to higher dimension is usually achieved using separable wavelets, operating on one dimension at a time. The three-dimensional wavelet transform on volume data is illustrated in Fig. 1. After the first iteration of wavelet transform, we generate one low-pass filtered wavelet subband (LLL_1) with one eighth of the original size, and seven high-pass filtered subbands (HLL_1, LHL_1, HHL_1, LLH_1, HLH_1, LHH_1, and HHH_1). We can then successively apply the wavelet transform to the low-pass filtered subband, thus creating a multiscale decomposition structure (a good introduction of wavelets for computer graphics can be found in [24]). In our experiment, the number of decomposition levels is usually between three and five, depending on the size of original data.

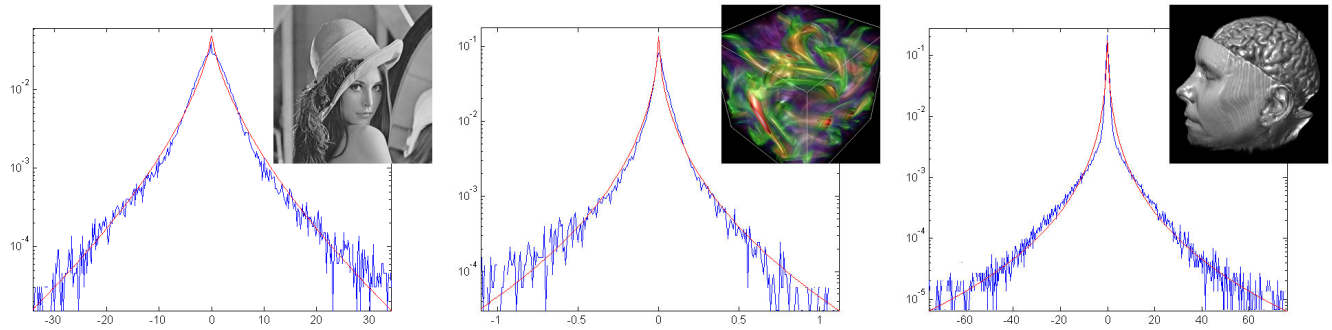
There is a wealth of literature on quality assessment and comparison in the field of image and video processing. Details on this are beyond the scope of this paper and we refer interested readers to [2] for a good survey. Here, we specifically review some related work in the field of graphics and visualization. Jacobs et al. [9] proposed an image querying metric for searching in an image database using a query image. Their metric makes use of multiresolution wavelet decompositions of the query and database images, and compares how many significant wavelet coefficients the query has in common with potential targets. In [6], Gaddipati et al. introduced a wavelet-based perceptual metric that builds on the subband coherent structure detection algorithm. The metric incorporates aspects of the HVS and modulates the wavelet coefficients based on the contrast sensitivity function. Sahasrabudhe et al. [18] proposed a quantitative technique which accentuates differences in images and data sets through a collection of partial metrics. Their spatial domain metric measures the lack of correlation between the data sets or images being compared. Recently, Wang et al. [26] introduced an image-based quality metric for interactive level-of-detail selection and rendering of large volume data. The quality metric design is based on an efficient way to evaluate the contribution of multiresolution data blocks to the final image.

In [29], Zhou et al. performed a study of different image comparison metrics that are categorized into spatial domain, spatial-frequency domain, and perceptually-based metrics. They also introduced a comparison metric based on the second-order Fourier decomposition and demonstrated favorable results against other metrics considered. In our work, we use the wavelet transform to partition the data into multiscale and oriented subbands. The study on volume data quality assessment is thus conducted in the spatial-frequency domain rather than the spatial domain.

III. Algorithm Overview

From a mathematical standpoint, we can treat volume data as three-dimensional arrays of intensity values with locally varying statistics that result from different combinations of abrupt features like boundaries and contrasting homogeneous regions. In line with this consideration, we advocate a statistical approach for volume data quality assessment. Given a volumetric data set, a first attempt may lead us to examine its statistics in the original spatial domain. However, even first-order statistics such as histograms would vary significantly from one portion of data to another, and from one data set to another. This defies simple statistical modeling over the entire data set, as well as the subsequent quality assessment.

Instead of spatial domain analysis, we can transform the volume data from the spatial domain to the spatial-frequency domain using the wavelet transform, and analyze its frequency



(a) Lena image, HL_1 subband (b) vortex data set, HHL_2 subband (c) brain data set, HLL_1 subband

Fig. 2. (a)-(c) are three wavelet subband coefficient histograms (blue curves) fitted with a two-parameter generalized Gaussian density model (red curves) for the Lena image, the vortex data set, and the brain data set, respectively. The estimated parameters (α,β) are $(1.2661, 0.6400)$, $(5.8881e-003, 0.4181)$, and $(7.1276e-003, 0.2709)$ for (a), (b), and (c), respectively. The overall fitting is good for all three examples.

statistics. Since frequency is directly related to rate of change, it is intuitive to associate frequencies in the wavelet transform with patterns of intensity variations in the spatial data. Furthermore, the wavelet transform allows us to analyze the frequency statistics at different scales. This will come in handy when we evaluate the quality of reduced or distorted data with different resolutions. Compared with the statistics of data in the spatial domain, the local statistics of different frequency *subbands* are relatively constant and easily modeled. This is realized using *generalized Gaussian density* (GGD) to model the marginal distribution of wavelet coefficients at different subbands and scales (Section IV-A). We also record information about selective wavelet coefficients (Section IV-C) and treat the low-pass filtered subband (Section IV-D) as part of our feature representation.

Our feature thus consists of multiple parts, and each part corresponds to certain essential information in the spatial-frequency domain. Note that this data analysis and feature extraction process can be performed when we have the access to the original data, or ideally, in situ where a simulation is running. Once we extract the feature from data, we are able to use it for quality assessment without the need to access the original data. Given a reduced or distorted version of data, we compare its feature components with those derived from the original data using predefined distance functions. This gives us an indication of quality loss in relation to the original data. We can also use the feature to perform a cross-comparison of data with different reduction or distortion types to evaluate the uncertainty introduced in different versions of data. Moreover, by forcing some of its statistical properties to match those of the original data, we may repair the reduced or distorted data for possible quality improvement.

IV. Wavelet Subband Analysis

In the multiscale wavelet decomposition structure, the low-pass filter subband corresponds to average information that represents large structures or overall context in the volume data. After several iterations of wavelet transforms, the size of the low-pass filter subband is small compared with the size of original data (already less than 0.2% for a three-level decomposition). Thus, we can directly treat it as part of the feature. On the other hand, the high-pass filtered subbands correspond to detail information that represents abrupt features or fine characteristics in the data. They spread across all different scales with an aggregate size nearly equal to the size of original data. The key issues are how to extract important feature information from these high-pass filtered subbands, and how to compress the feature.

A. Wavelet Subband Statistics

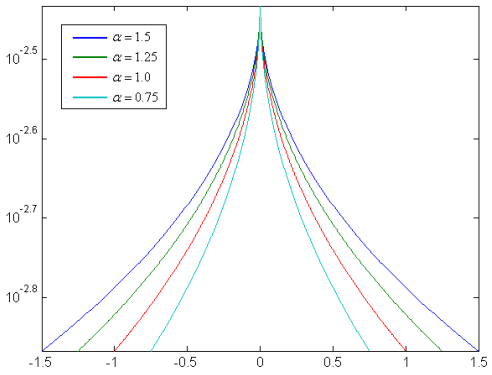
Studies on natural image statistics reveal that the histogram of wavelet coefficients exhibits a marginal distribution at a particular high-pass filtered subband. An example of the Lena image and the coefficient histogram of one of its wavelet subbands is shown in Fig. 2 (a). The *y*-axis is on a log scale in the histogram. As we can see, the marginal distribution of wavelet coefficients creates a sharp peak at zero and more extensive tails than the Gaussian density. The intuitive explanation of this is that natural images usually have large overall structures consisting of smooth areas interspersed with occasional abrupt transitions, such as edges and contours. The smooth areas lead to near-zero coefficients, and the abrupt transitions give large-magnitude coefficients. In [14], Mallat shows that such a marginal distribution of the coefficients in individual wavelet subbands can be well-fitted with a two-parameter generalized Gaussian density (GGD) model:

$$p(x) = \frac{\beta}{2\alpha\Gamma(\frac{1}{\beta})} \exp(-(\frac{|x|}{\alpha})^{\beta}), \tag{1}$$

where Γ is the Gamma function, i.e., $\Gamma(z) = \int_0^{\infty} e^{-t}t^{z-1}dt$, $z > 0$.

In the GGD model, α is the *scale* parameter that describes the standard deviation of the density, and β is the *shape* parameter that is inversely proportional to the decreasing rate of the peak. As an example, the plots of the GGD distribution under varied (α,β) values are illustrated in Fig. 3. The model parameter (α,β) can be estimated using the *moment matching* method [25] or the *maximum likelihood* rule [16]. Numerical experiments in [25] show that 98% of natural images satisfy this property. Even for the remaining 2%, the approximation of the real density by a GGD is still acceptable. Note that the

GGD model includes the Gaussian and the Laplacian distributions as special cases with $\beta = 2$ and $\beta = 1$, respectively. The GGD model provides a very efficient way for us to summarize the coefficient histograms of an image, as only two parameters are needed for each subband. This model has been used in previous work for noise reduction [22], image compression [3], texture image retrieval [5], and quality encoding [28]. In this paper, we use the moment matching method which takes the mean and the variance of wavelet coefficients in a subband to compute its GGD model parameters (α, β) (see the Appendix for the implementation detail). In Fig. 2 (a), the red curve is the GGD function with parameters estimated using the moment matching method. The result fits the original wavelet coefficient distribution quite well.



(a) $\beta = 0.5$

(b) $\alpha = 1.5$

Fig. 3. The GGD distribution and its model parameters (α, β). (a) α varies while $\beta = 0.5$. (b) β varies while $\alpha = 1.5$. The figure shows the sensitivity of the shape of GGD plots with respect to the model parameters.

We extend this statistical model to three-dimensional volume data since many scientific and medical data share the same intrinsic characteristics as natural images; i.e., homogeneous regions mixed with abrupt transitions. Moreover, the rate or proportion of homogeneous regions and abrupt transitions is also similar for image and volume: in 2D, we have area of homogeneous regions versus the edge length of abrupt transitions; and in 3D, we have volume of homogeneous regions versus the surface area of abrupt transitions. Initial experiments on two small data sets give very promising results. Fig. 2 (b) and (c) show one example of wavelet subband coefficient histograms for each data set and their respective well-fitted GGD curves. Thus, with only two GGD parameters, we are able to capture the marginal distribution of wavelet coefficients in a subband that otherwise would require at least hundreds of parameters using histogram. We shall see in Section V that this GGD model works well for larger data sets too. Next, we discuss the distance measure for wavelet subband statistics.

Let $p(x)$ and $q(x)$ denote the probability density functions of the wavelet coefficients in the same subband of the original and distorted data, respectively. Here, we assume the coefficients to be independently and identically distributed. Let $\mathbf{x} = \{x_1, x_2, \ldots, x_N\}$ be a set of randomly selected coefficients. The log-likelihoods of $\mathbf{x}$ being drawn from $p(x)$ and $q(x)$ are

$$l(p) = \frac{1}{N}\sum_{i=1}^{N} \log p(x_i) \quad \text{and} \quad l(q) = \frac{1}{N}\sum_{i=1}^{N} \log q(x_i) \qquad (2)$$

respectively. Based on the law of large numbers, when N is large, the log-likelihoods ratio between $p(x)$ and $q(x)$ asymptotically approaches the *Kullback-Leibler distance* (KLD) (also known as the *relative entropy* of p with respect to q):

$$d(p||q) = \int p(x) \log \frac{p(x)}{q(x)} dx. \qquad (3)$$

Although the KLD is not a true metric, i.e., $d(p||q) \neq d(q||p)$, it satisfies many important mathematical properties. For example, it is a convex function of p. It is always nonnegative, and equals zero only if $p(x) = q(x)$. In this paper, we use the KLD to quantify the difference between wavelet coefficient distributions of the original and distorted data. This quantity is evaluated numerically as follows:

$$d(p||q) = \sum_{i=1}^{M} P(i) \log \frac{P(i)}{Q(i)}, \qquad (4)$$

where $P(i)$ and $Q(i)$ are the normalized heights of the ith histogram bin, and M is the number of bins in the histogram. Note that the coefficient histogram Q is computed directly from the distorted data, while the coefficient histogram P is approximated using its GGD parameters (α, β) extracted from the original data.

Finally, the KLD between the distorted and original data over all subbands is defined as:

$$D_1 = \log(1 + \sum_{i=1}^{B} d(p^i||q^i)), \qquad (5)$$

where B is the total number of subbands analyzed, p^i and q^i are the probability density functions of the ith subbands in the original and distorted data respectively, and $d(p^i||q^i)$ is the estimated KLD between p^i and q^i.

B. Voxel Visual Importance

At runtime, a transfer function is applied to the input volume where the scalar data values are mapped to optical quantities such as color and opacity, and the volume is projected into 2D images. To capture the visualization-specific contribution

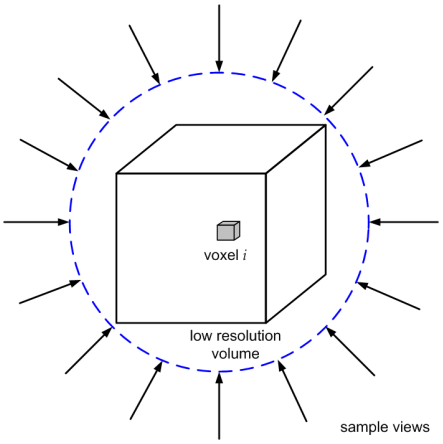


Fig. 4. A voxel's visual importance in the low resolution volume is the multiplication of its opacity and average visibility. The average visibility is calculated using a list of evenly-sampled views along the volume's bounding sphere.

for each voxel, we define a voxel's visual importance ω as follows:

$$\omega(i) = \alpha(i) \cdot \bar{v}(i) \tag{6}$$

where $\alpha(i)$ is the opacity of voxel i, $\bar{v}(i)$ is its average visibility. As sketched in Fig. 4, given an original large volume data, we use its low resolution form (for practical and performance concern) to calculate visual importance values of all voxels within the volume. In Eqn. 6, $\alpha(i)$ and $\bar{v}(i)$ account for the emission and attenuation of voxel i, respectively.

To calculate the average visibility, we consider a list of evenly-sampled views along the bounding sphere that encloses the volume and take the average of the visibility from those sample views. Given a view along the bounding sphere, the visibility for each voxel in the low resolution data is acquired in this way: we render the low resolution data by drawing front-to-back view-aligned slices and evaluate the visibility of all the voxels during the slice drawing. The visibility of a voxel is computed as (1-α) right before the slice containing the voxel is to be drawn, where α is the accumulated opacity at the voxel's screen projection. This process repeats for each sample view. Finally, for each voxel in the volume, we use the average of its visibility from all sample views to calculate its visual importance. Essentially, the visual importance indicates the average contribution of a voxel in association with a given input transfer function. This visualization-specific term is then normalized and incorporated into the following wavelet coefficient selection.

C. Wavelet Coefficient Selection

The GGD model captures the marginal distribution of wavelet coefficients at each individual subband. Using the distance defined in Eqn. 5, we are able to know how close the coefficient distributions of distorted data are in relation to the original data. Nevertheless, the histogram itself does not tell the spatial-frequency positions of wavelet coefficients. This limits our ability to compare the data in finer detail and possibly repair the distorted data. Therefore, along with the *global* GGD parameters per wavelet subband, we also need to record *local* information about wavelet coefficients for data quality assessment and improvement.

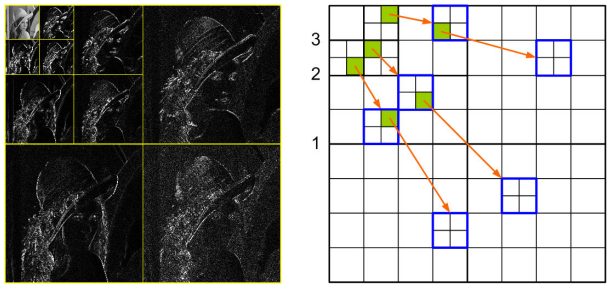


Fig. 5. The parent-child dependencies of wavelet coefficients in different subbands. In this 2D example, a coefficient in a coarse scale has four child coefficients in the next finer scale of similar orientation. The arrow points from the subband of the parents to the subband of the children.

An important observation is that, although the coefficients of wavelet subbands are approximately decorrelated, they are not statistically independent. For example, Fig. 5 shows a three-level decomposition of the Lena image. It can be seen that coefficients of large magnitude (bright pixels) tend to occur at neighboring spatial-frequency locations, and also at the same relative spatial locations of subbands at adjacent scales and orientations. Actually, in 2D, a coefficient c in a coarse scale has four child coefficients in the next finer scale. Each of the four child coefficients also has four child coefficients in the next finer scale. Furthermore, if c is insignificant with respect to some threshold ε, then it is likely that all of its descendant coefficients are insignificant too. This coefficient dependency has been exploited in several image compression algorithms, such as the embedded zerotree wavelet (EZW) encoding [20] and a following image codec based on set partitioning in hierarchical trees (SPIHT) [19]. These algorithms have also been extended to three-dimensional volumetric image compression in medical application [13]. In this paper, we utilize the coefficient dependency to store selective wavelet coefficients in an efficient manner.

There are two categories of wavelet coefficients that are of importance for the purpose of quality assessment and improvement. One category is the coefficients of large magnitude which correspond to abrupt features like edges or boundaries. As we can see in Fig. 2, they are along the tails of the marginal coefficient distribution where the perceptually-significant coefficients generally reside. The other category is neighboring near-zero coefficients which correspond to homogeneous regions. They are close to the zero peak of the distribution and are important indications of data regularity. Taking into account the visualization-related factor, we modulate the wavelet coefficients with the voxel visual importance values (Section IV-B) at their nearest spatial-frequency locations. In this case, a wavelet coefficient is large only if it has both large magnitude and high voxel visual importance; a wavelet coefficient is near zero if it has either near-zero magnitude or near-zero voxel visual importance.

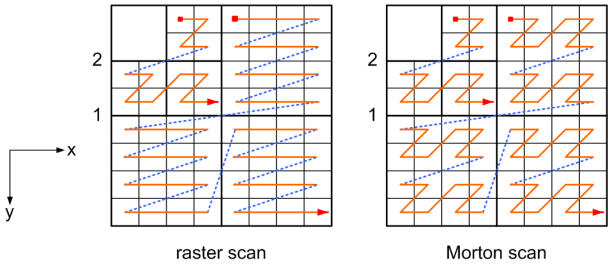


Fig. 6. Scan of wavelet coefficients in the raster order and the Morton order. The blue dashed line segments indicate discontinuities in the scan. Compared with the raster order, the Morton order preserves the spatial-frequency locality better.

Starting from the coarsest scale, we scan each wavelet subband and encode coefficients of interest. As illustrated with a 2D example in Fig. 6, we follow the *Morton order* (Z-curve order) as opposed to the ordinary raster order to better utilize the spatial-frequency locality. For neighboring near-zero coefficients (at least eight consecutive coefficients in 3D), we run-length encode their positions (i.e., the scan orders). For large-magnitude coefficients, we encode their positions and values as well. In general, most scientific and medical data have a low-pass spectrum. When the data are transformed into a multiscale wavelet decomposition structure, the energy in the subbands decreases from a fine scale (high resolution) to a coarse scale (low resolution). Therefore, the wavelet coefficients are, on average, smaller in the finer scales than in the coarser scales. Accordingly, we vary the thresholds of near-zero coefficients (ε) and large-magnitude coefficients (τ) for different scales. Finally, information of the selective coefficients is further compressed using the open source `zlib` compressor.

We define the distance for the selective wavelet coefficients as follows:

$$D_2 = \log(1+\sum_{i=1}^{B}\sqrt{\frac{\sum_{j=1}^{L_i}(\frac{c_j-c_j^{'}}{cmax_i})^2+\sum_{k=1}^{Z_i}(\frac{c_k^{'}}{cmax_i})^2}{L_i+Z_i}}), \quad (7)$$

where B is the number of subbands over all the scales, L_i and Z_i are the numbers of large-magnitude coefficients selected and near-zero coefficients selected in the ith subband, respectively. c_j and $c_j^{'}$ are the jth large coefficients selected from the original and distorted data respectively, and $cmax_i$ is the largest magnitude (modulated by visual importance) of all coefficients at the ith subband. For near-zero coefficients, we assume the original coefficients $c_k = 0$ and only consider coefficients in the distorted data with $|c_k^{'}| > \varepsilon$ for the calculation.

D. Low-Pass Filtered Subband

The low-pass filter subband in the multiscale wavelet decomposition structure corresponds to average information that represents large structures or overall context in the volume data. Compared with the size of original data, the size of this subband is usually small after several iterations of wavelet decomposition. Therefore, we directly incorporate it as part of the feature. Let b_i and b_j be the low-pass filter subbands of the original and distorted data, respectively. The similarity between b_i and b_j is defined as:

$$S = \frac{\sigma_{ij}}{\sigma_i\sigma_j}\cdot\frac{2\ \mu_i\mu_j}{\mu_i^2+\mu_j^2}\cdot\frac{2\ \sigma_i\sigma_j}{\sigma_i^2+\sigma_j^2} = \frac{4\ \sigma_{ij}\ \mu_i\mu_j}{(\sigma_i^2+\sigma_j^2)(\mu_i^2+\mu_j^2)}, \quad (8)$$

where σ_{ij} is the covariance between b_i and b_j, μ_i and μ_j are the mean values of b_i and b_j respectively, and σ_i and σ_j are their standard deviations. Eqn. 8 consists of three parts; namely, *loss of correlation*, *luminance distortion*, and *contrast distortion*. Collectively, these three parts capture the structure distortion of the low-pass filtered subband in the distorted data. This similarity measure comes from the image quality assessment literature [27], and has been shown to be consistent with the luminance masking and contrast masking features in the HVS, respectively. The dynamic range of S is $[-1,1]$. The best value of 1 is achieved when $b_i = b_j$. Hence, we define the distance between b_i and b_j as follows:

$$D_3 = \sqrt{1.0-(S+1.0)/2.0}. \quad (9)$$

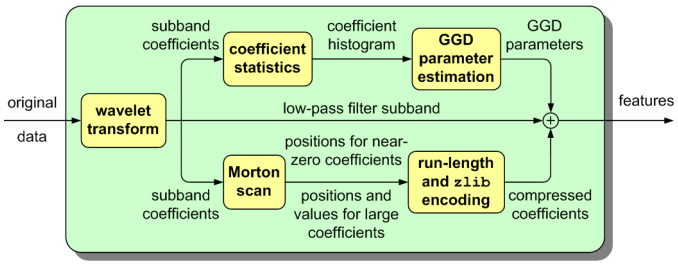


Fig. 7. Our feature representation of the original data in the wavelet domain.

E. Summary

In summary, as shown in Fig. 7, our feature representation of the original data in the wavelet domain includes three parts: the GGD model parameters from wavelet subband statistics, selective wavelet coefficients, and the low-pass filtered subband. Given a reduced or distorted version of data, we analyze the quality loss by calculating its distances to the original data for each of the feature components (Eqn. 5, 7, and 9). Each partial distance indicates some quality degradation with reference to the original data and the summation of all these partial distances gives the overall degradation. Thus, an overall distance could be computed heuristically as the weighted sum of the three individual distances:

$$D = k_1D_1 + k_2D_2 + k_3D_3, \quad (10)$$

where $k_i > 0$, $i = 1$, 2, and 3. Note that there is no need for normalizing this overall distance. For the purpose of data quality improvement, it is advantageous to keep each distance separate (or further at the subband level) so that we know which parts cause significant quality degradation. We can then repair accordingly using the feature extracted from the original data.

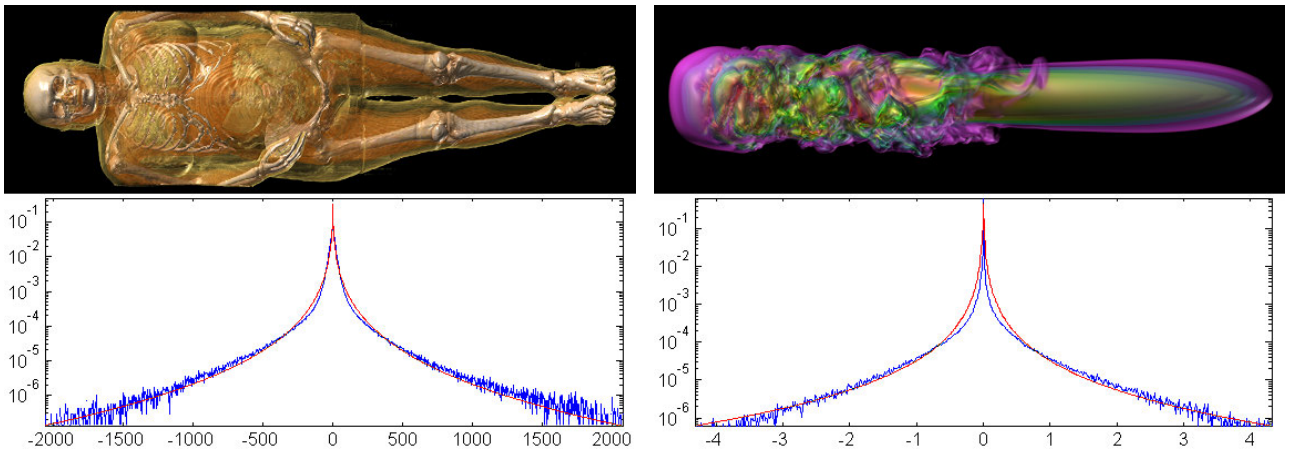
data set	dimension	data size	DL	LP dimension	LP size	HP size	feature size	ratio
turbulent vortex flow	128^3	8.0MB	3	16^3	16.0KB	42.2KB	58.2KB	0.710%
solar plume velocity magnitude	$512^2 \times 2048$	2.0GB	5	$16^2 \times 64$	64.0KB	376.0KB	440.0KB	0.021%
supernova angular momentum	864^3	2.40GB	5	27^3	76.9KB	852.6KB	929.5KB	0.037%
UNC brain	$128^2 \times 72$	4.5MB	3	$16^2 \times 9$	9.0KB	53.7KB	62.7KB	1.361%
head aneurysm	512^3	512MB	4	32^3	128.0KB	198.0KB	326.0KB	0.062%
visible woman	$512^2 \times 1728$	1.69GB	5	$16^2 \times 54$	54.0KB	753.4KB	807.4KB	0.046%

DL = decomposition levels

LP dimension (size) = low-pass filtered subband's dimension (size); **HP size** = all high-pass filtered subbands' feature size

TABLE I

THE SIX FLOATING-POINT DATA SETS AND THEIR FEATURE SIZES.



(a) visible woman data set, HLH_2 subband (b) solar plume data set, LLH_2 subband

Fig. 8. (a) and (b) are two wavelet subband coefficient histograms (blue curves) fitted with the GGD model (red curves) for the visible woman and the solar plume data sets, respectively. The estimated parameters (α, β) are $(1.6786e-002, 0.2425)$ and $(1.4922e-009, 0.1405)$, respectively. In general, the fitting works well for these two data sets.

V. RESULTS

We experimented with our algorithm on six floating-point data sets, as listed in Table I. Among the six data sets, three of them are from scientific simulation, and the remaining three are from medical application. These six data sets vary greatly in size, and exhibit quite different characteristics. For multiscale wavelet decomposition, we specifically restricted our attention to the Daubechies family of orthogonal wavelets, as evaluation of all possible wavelet transforms is out of the scope of our experiments. The decision for levels of wavelet decomposition is based on the size of input data, as well as the tradeoff between the size of feature and the robustness of GGD model parameters.

In our test, the threshold for near-zero wavelet coefficients ε_i at the ith subband was chosen as $cmax_i/(2^{L+3})$, where $cmax_i$ is the largest magnitude (modulated by visual importance) of all coefficients at the ith subband, and L is the total number of decomposition levels we have. The threshold for large-magnitude wavelet coefficients τ_i at the ith subband was chosen as $cmax_i/(2^{s+2})$, where s is the scale in which the ith subband locates. We varied τ_i according to the scale because the wavelet coefficients in a subband become more important as the scale increases. In Table I, the size of low-pass filtered subband is the uncompressed size of LLL_n, after n levels of decomposition. The feature size of all high-pass filtered subbands includes their respective GGD parameters and selective wavelet coefficients in the compressed form.

From the last column in Table I, we can see that for all six data sets, the size of feature is small compared with the original data. Note that the size of feature does not increase in proportion to the size of original data. This is mainly due to the increase of decomposition levels for larger data sets, as we can afford to have more levels of wavelet decomposition while still keeping the GGD parameters robust. Our experiment confirms that the GGD model generally performs well when the size of the input data becomes larger. For instance, Fig. 8 shows one of the wavelet subband coefficient histograms for the visible woman and the solar plume data sets, and their respective GGD curves. On the other hand, the feature size is also data and transfer function dependent. For example, the ratio for the brain data set is 1.361%, which is relatively high compared with the vortex data set having the same number of decomposition levels. Thus, it follows that we record a higher percentage of high-pass filtered subband feature information for the brain data set.

Next, we report results of volume data quality assessment and improvement using the extracted feature. To compare the quality of rendered images, we used a GPU raycaster for

volume rendering. All tests were performed on a 2.33GHz Intel Xeon processor with 4GB main memory, and an nVidia GeForce 7900 GTX graphics card with 512MB video memory.

A. Quality Assessment

First of all, we experimented with our quality measure on different data sets and observed how the quality of data changes for the same reduction or distortion type. We used the three smaller data sets (brain, vortex, and aneurysm) of their original resolutions and the other three data sets (visible woman, solar plume, and supernova) of their second highest resolutions in the test. To calculate the overall distance, we used Eqn. 10 with $(k_1, k_2, k_3) = (0.1, 1.0, 1.0)$. Please note that in this paper, we assume the original data set has full quality. Thus, any changes made to the data would involve possible quality loss, even though the desire is to enhance the data from a certain perspective.

Quantization is a commonly used approach for data reduction. Our first example studies the quality loss under the uniform quantization scheme. Fig. 9 shows the quality assessment result of all six data sets with six different quantization levels. A larger distance indicates a greater degree of quality degradation. More specifically, Table II lists all partial and overall distances for the aneurysm data set. Although different data sets have different responses of quality loss due to quantization, the overall trend is fairly obvious: the data quality gets increasingly worse as the number of quantization levels decreases.

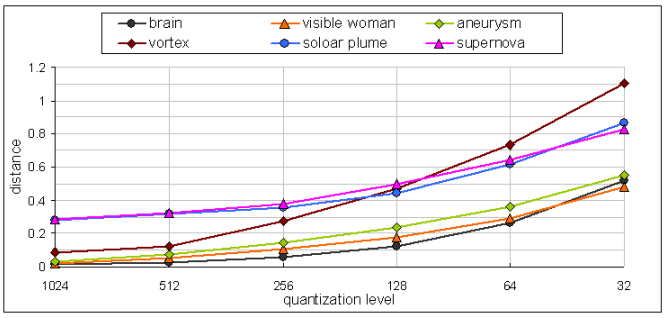


Fig. 9. Quality assessment on six test data sets with six different quantization levels. The data quality gets increasingly worse as the number of quantization levels decreases.

level	D_1	D_2	D_3	D
1024	0.1961	0.0071	0.0038	0.0305
512	0.5227	0.0144	0.0077	0.0744
256	0.9933	0.0277	0.0153	0.1423
128	1.5111	0.0551	0.0303	0.2365
64	1.9518	0.1065	0.0594	0.3611
32	2.2407	0.2089	0.1150	0.5480

TABLE II
PARTIAL AND OVERALL DISTANCES FOR THE ANEURYSM DATA SET WITH SIX DIFFERENT QUANTIZATION LEVELS.

Our second example studies the quality loss under the Gaussian smooth filtering. Fig. 10 shows how the data quality changes with six different Gaussian smooth filters for all six data sets. We applied a discrete Gaussian kernel of size 5^3 with different standard deviations. A larger standard deviation indicates a greater degree of smoothing since neighboring voxels carry more weight. Table III lists all partial and overall distances for the solar plume data set. It is clear that the data quality gets worse as the standard deviation increases. Unlike quantization, however, the rate of quality loss decreases gradually in the sequence.

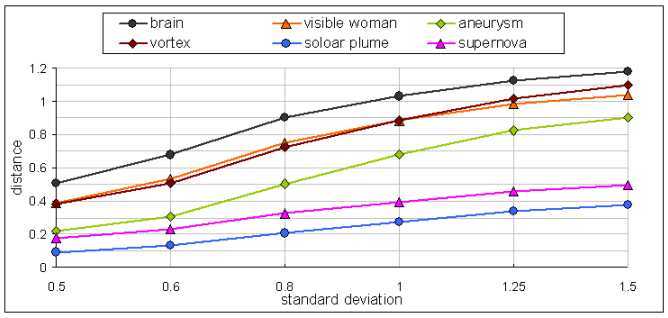
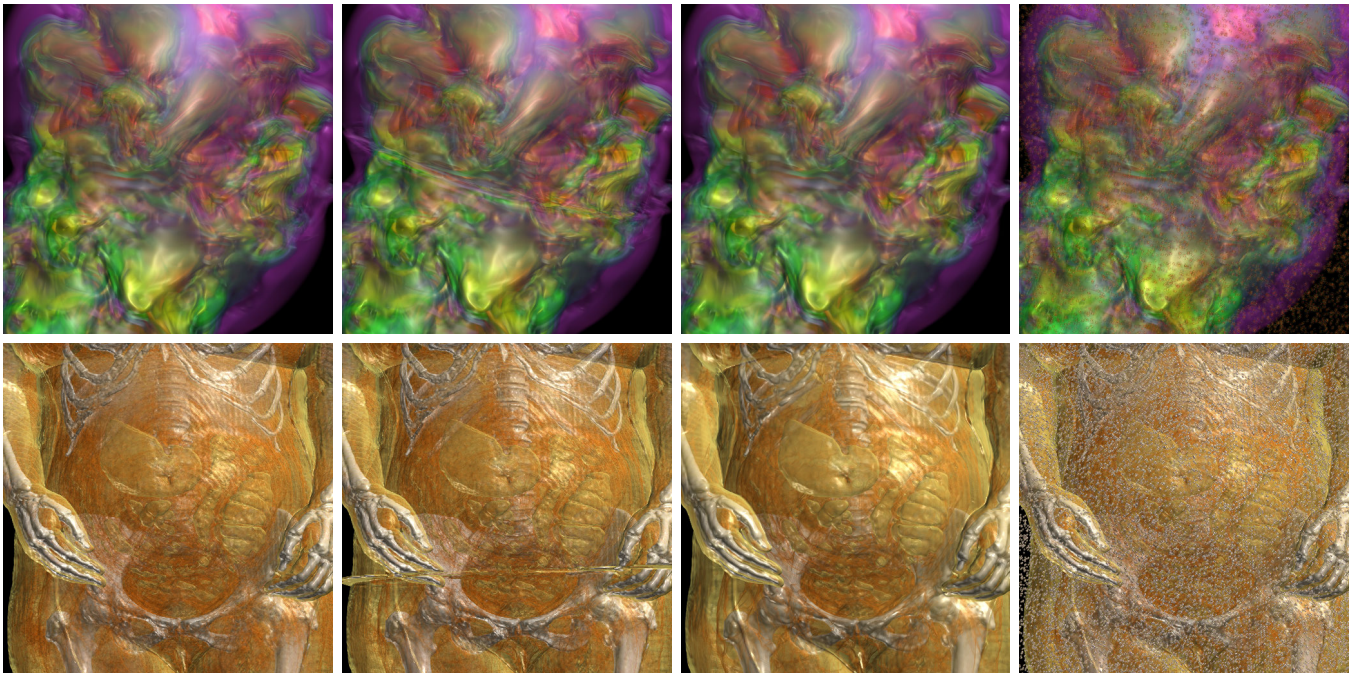


Fig. 10. Quality assessment on six test data sets with 5^3 Gaussian smooth filters of six different standard deviations. The data quality gets worse as the standard deviation increases.

σ	D_1	D_2	D_3	D
0.5	0.2687	0.0656	0.0004	0.0929
0.6	0.3793	0.0921	0.0006	0.1306
0.8	0.6325	0.1422	0.0010	0.2065
1.0	0.9225	0.1822	0.0015	0.2760
1.25	1.2039	0.2165	0.0020	0.3389
1.5	1.3548	0.2377	0.0023	0.3755

TABLE III
PARTIAL AND OVERALL DISTANCES FOR THE SOLAR PLUME DATA SET WITH SIX GAUSSIAN SMOOTH FILTERS OF DIFFERENT STANDARD DEVIATIONS.

Besides quality assessment of data with the same type of reduction or distortion, the feature also avails us to perform cross-type data quality comparison. For example, Fig. 11 gives quality assessment results on the solar plume and the visible woman data sets under four different distortion types: mean shift (of the data range over 256), voxel misplacement (with two slices of voxels misplaced), averaging filter (using a kernel of size 3^3), and salt-and-pepper noise (with an equal probability of $1/1024$ for the bipolar impulse). Table IV lists all partial and overall distances, MSE, and PSNR for these four distortion types. For both data sets, we can see that the mean shift introduces the minimum quality loss here, followed by the voxel misplacement. The salt-and-pepper noise incurs the most quality degradation. This result is consistent with perceived quality in rendered images. However, the MSE and the PSNR incorrectly recognize the mean shift as having a larger distortion than the voxel misplacement for both data sets. Note that they also give the opposite results on the averaging filter for the two data sets. This is due to the reason that the MSE and the PSNR metrics are only voxel-based and



(a) mean shift (b) voxel misplacement (c) averaging filter (d) salt-and-pepper noise

Fig. 11. Cross-type quality assessment on low resolution solar plume ($256^2 \times 1024$) and visible woman ($256^2 \times 864$) data sets. The data quality degrades as the overall distance (listed in Table IV) increases from (a) to (d). The rendered images are cropped for a closer comparison.

data set	**type**	D_1	D_2	D_3	D	**rank**	**MSE**	**PSNR**	**rank**
solar plume	mean shift	$2.0556e-4$	$4.5864e-7$	$5.8988e-2$	0.0590	4	$5.5252e-2$	48.1648	2
	misplacement	$1.3126e-1$	$9.4561e-2$	$5.5746e-3$	0.1133	3	$2.4239e-2$	51.7431	3
	averaging	$5.2393e-1$	$1.2551e-1$	$8.1216e-4$	0.1787	2	$5.2299e-3$	58.4033	4
	noise	$3.1198e+0$	$1.8137e+0$	$2.8900e-2$	2.1546	1	$6.7305e+0$	27.3078	1
visible woman	mean shift	$8.8428e-5$	$6.9770e-7$	$2.3914e-2$	0.0239	4	$1.8691e+3$	48.1648	3
	misplacement	$1.5366e-2$	$1.1612e-1$	$4.6497e-3$	0.1223	3	$1.5397e+3$	49.0066	4
	averaging	$1.6449e+0$	$5.4139e-1$	$1.4596e-3$	0.7073	2	$1.9289e+5$	28.0279	1
	noise	$1.7343e+0$	$7.8530e-1$	$9.7468e-3$	0.9685	1	$1.2152e+4$	40.0347	2

TABLE IV

PARTIAL AND OVERALL DISTANCES, MSE, AND PSNR FOR THE SOLAR PLUME AND THE VISIBLE WOMAN DATA SETS WITH FOUR DIFFERENT DISTORTION TYPES.

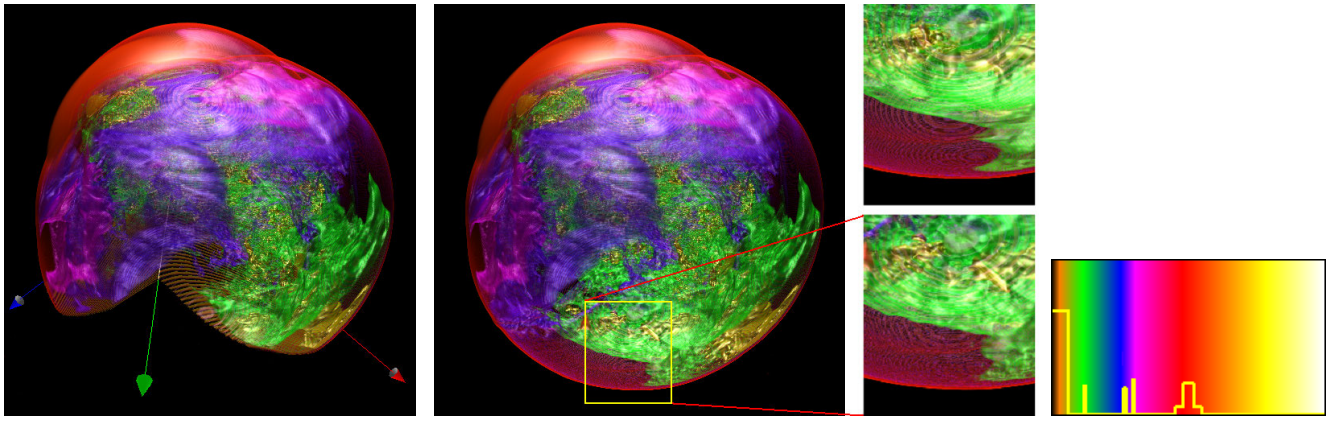
do not consider the overall structure distortion of the data.

B. Quality Improvement

Since the feature captures essential information from the original data, it can be utilized to improve the quality of distorted or corrupted data. In this paper, we do not show examples where the feature is used to construct a higher resolution data from a low resolution data, as it is the most common way of using the wavelet transform and compression.

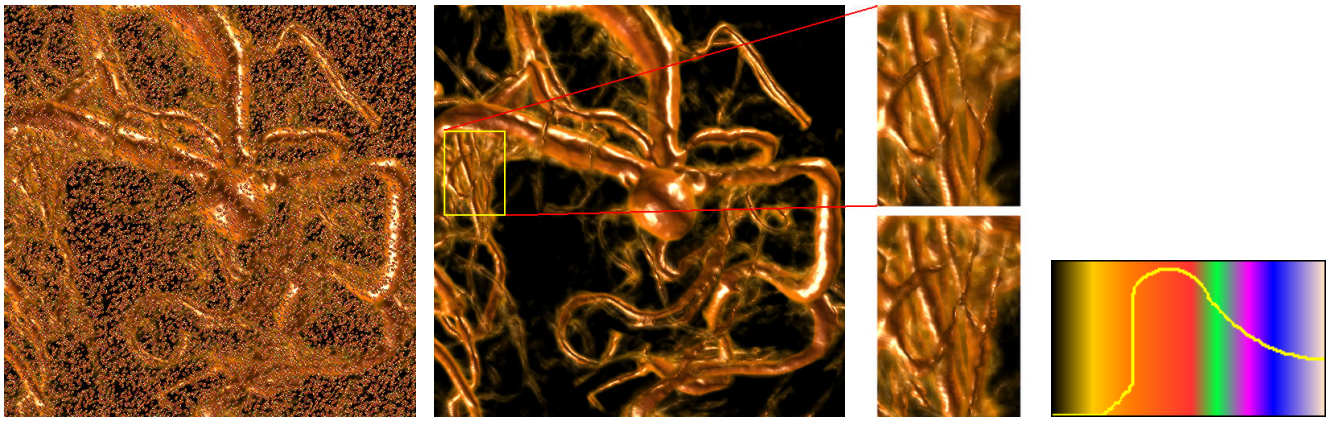
Our first example deals with missing data. Fig. 12 (a) shows the rendering of a low resolution supernova data set with one-eight (i.e., an octant) of data missing. The missing of data could result from incomplete data transmission, or even a bug in the data reduction source code. Recall that we keep each partial distance separate (and actually at the subband level). This helps us identify which parts introduce the dramatic change (in this case, the subband at the same orientation as the missing portion), and then repair accordingly using the feature information.

The repairing scheme works as follows: First, a multiscale wavelet decomposition structure is built from the corrupted data, where the size of low-pass filtered subband at the coarsest scale equals the size of low-pass filtered subband recorded in the feature (Section IV-D). Note that for the repairing purpose, we keep the low-pass filtered subbands in all scales. Then, we improve the high-pass filtered subbands across all scales by replacing the wavelet coefficients with their corresponding coefficients in the feature; that is, those large-magnitude and near-zero wavelet coefficients selected (Section IV-C). Next, starting from the coarsest scale (the lowest resolution), we reconstruct the low-pass filtered subband at the next finer scale, using the low-pass filtered subband recorded in the feature and improved high-pass filtered subbands. The reconstructed low-pass filtered subband is used to correct the missing part in



(a) before, $D = 1.3346$ (b) after, $D = 0.5536$ (c) comparison (d) transfer function

Fig. 12. Quality improvement on a low resolution supernova data set (432^3) with one-eight of data missing, as shown in (a). (b) is the result after an automatic repairing process using the feature information. In (c), a portion of (b) is zoomed in for comparison with the reference image displayed on the top. (d) shows the transfer function used.



(a) before, $D = 2.3094$ (b) after, $D = 0.7188$ (c) comparison (d) transfer function

Fig. 13. Quality improvement on the aneurysm data set (512^3) distorted by random noise, as shown in (a). (b) is the result after an automatic repairing process using the feature information. In (c), a portion of (b) is zoomed in for comparison with the reference image displayed on the bottom. (d) shows the transfer function used.

the same-scale low-pass filtered subband decomposed from the corrupted data. The corrected low-pass filtered subband is then used to reconstruct the next finer scale in an iterative manner. In this way, we are able to automatically repair the missing portion in the corrupted data scale by scale. Finally, an optional median filter is applied to the corrected portion of data at the finest scale in order to suppress potential noise and produce a better match with the original GGD model parameters. Fig. 12 (b) shows the result after this automatic repairing process. It is clear that the data quality improves as the overall distance decreases.

We can also apply a similar repairing process for noise reduction. For example, Fig. 13 (a) shows the rendering of the aneurysm data set distorted by random noise. This kind of distortion can be detected through the observation of a sequence of sudden spikes appearing in the wavelet coefficient subband histograms. The denoising process also follows a coarse-to-fine manner as usual, but there is no need to keep the low-pass filtered subband at every scale in the wavelet decomposition structure. Another difference is that for each high-pass filtered subband i, we first set large-magnitude wavelet coefficients to zero (if they are larger than the threshold τ_i) before improving them with their corresponding coefficients in the feature. Fig. 13 (b) shows the result after this repairing process. As we can see, the noise is eliminated, while the fine structure of the blood vessels is preserved.

VI. Discussion

A. Choice of Wavelets

To extract essential information from the original data, we decomposed the data into multiple scales using wavelet basis functions localized in spatial position, orientation, and spatial frequency. We used the Daubechies family of orthogonal wavelets in our experiment because they provide a good tradeoff between performance and complexity [5], [6]. Moreover, we found that the choice for the number of scaling and wavelet function coefficients has little effect on assessment accuracy. Therefore, we specifically used the Daubechies D4

transform for efficiency. Other separable wavelets (such as the Gabor wavelets) or redundant transforms (such as the steerable pyramid transform) could also be used in our algorithm. For example, the steerable pyramid transform decomposes the data into several spatial-frequency bands, and further divides each frequency band into a set of orientation bands. It can thus help to minimize the amount of aliasing within each subband. However, they are more expensive to compute and require more storage space.

B. Timing Performance

The timing of wavelet analysis on the original data includes the time for multiscale wavelet decomposition, GGD parameters estimation, and subband wavelet coefficients selection. This one-time preprocess may take anywhere from seconds to a total of several minutes on a single PC, depending on the size of input data. For data sets that could not be loaded into memory simultaneously, we employed a block-wise wavelet transform process and handled boundaries of neighboring blocks to guarantee seamless results. The timing of quality assessment on different versions of data includes the time for wavelet decomposition and distance calculation. At runtime, it usually takes less than one minute on a single PC to evaluate data with the size up to 512MB. For larger gigabytes data, the time to perform wavelet transforms becomes dominant in the quality assessment process. In the worst case, the assessment time would be similar to the preprocess time if the data we evaluate has the same size as the original data.

C. GGD Model

We note that as an approximation, the GGD model introduces a prediction error at each wavelet subband with respect to the corresponding wavelet coefficient distribution. For example, the fit near the center of the histogram in Fig. 8 (b) is not good. This error can be calculated as the KLD between the model histogram and the histogram of wavelet subband coefficients from the original data. Let $d(p_m^i||p^i)$ denote the prediction error at the ith subband. Accordingly, we use $d(p^i||q^i) = |d(p_m^i||q^i) - d(p_m^i||p^i)|$ to calculate the overall KLD (Eqn. 5). That is, we actually subtract the prediction error from the KLD between the model histogram and the histogram of wavelet subbband coefficients from the reduced or distorted data (denoted as $d(p_m^i||q^i)$), and use the absolute value in the calculation.

On the other hand, our experiment shows that the GGD model generally works well on scientific and medical data sets with different sizes. However, there are cases where this model fails to give good results. Such an example is shown in Fig. 14. For these failed cases, we can store the actual wavelet subband coefficient histograms (per scale) at the expense of increasing the storage, or fit each coefficient histogram with splines to smooth out the irregularities. Although there is a need of further research on why these cases fail, the apparent reason is that those data sets do not fall into the category of natural statistics. For this same reason, we can not partition the original data into blocks in an octree fashion, and analyze the individual blocks using the GGD model (each block does not necessarily exhibit the marginal coefficient distribution even though the whole data set does). Therefore, our solution is a multiscale, not a true multiresolution approach.

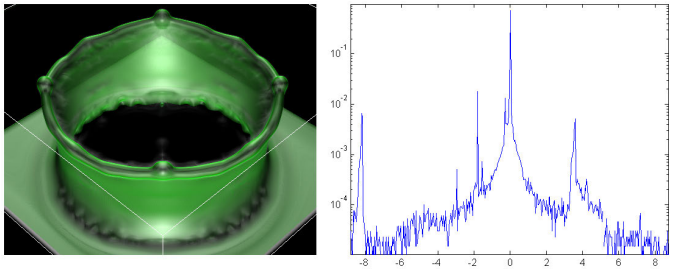


Fig. 14. The "milk crown" physical simulation data set ($512 \times 256 \times 512$) and its LHL_3 subband coefficient histogram, which does not exhibit the marginal distribution.

D. Transfer Function

In this work, different versions of a data set were rendered using the same transfer function for the purpose of subjective data quality comparison. Since our focus was on data quality assessment, we chose to fix rendering parameters so that the possible difference or uncertainty introduced by the visualization process could be minimized. Our current algorithm explicitly takes the input transfer function into consideration by modulating wavelet coefficients with voxel visual importance values at their nearest spatial-frequency positions. The voxel visual importance values were precomputed offline with a given transfer function. If the transfer function changes at runtime, the calculation can be performed online (in this case, we need to keep the low resolution data).

Our solution is a coarse approximation of voxel contribution to the visualization. The accuracy of voxel visual importance values depends on the resolution of data used and the number of sample views taken for average visibility calculation. There is a tradeoff between the update speed and the accuracy of visual importance values. In practice, we can update visual importance values within seconds for a low resolution volume of size around 64^3 with 16 sample views. In our solution, voxel visual importance values are only used to modulate and select wavelet coefficients (Section IV-C). An improvement of our implementation is to store selective wavelet coefficients offline by only considering their magnitudes. At runtime when the transfer function changes, the visual importance values are calculated and used to further pick visually important coefficients from stored coefficients.

Besides our current algorithm, another way to possibly improve wavelet subband analysis is to apply the idea presented in [1] that classifies the voxels into core, gradient, and unimportant voxels and assigns weight functions for wavelet coefficients accordingly. Alternatively, the users can also provide their own voxel visual importance volume, derived from volume classification or segmentation, for example, to modulate wavelet coefficients. Nevertheless, we understand that this voxel-based approach is not an optimal solution for large data analysis in terms of both efficiency and effectiveness. A better solution could be using some shape functions to approximate

the volume data and capture the visual importance aspect. Another direction is to perform a more rigorous study on data quality comparison in association with direct volume rendering algorithm specifications [10].

VII. Conclusion and Future Work

We introduce a reduced-reference approach to volume data quality assessment. A multiscale wavelet representation is first built from the original data which is well-suited for the subsequent statistical modeling and feature extraction. As shown in Section V, we extract minimum feature information in the wavelet domain. Using the feature and predefined distance measures, we are able to identify and quantify the quality loss in the reduced or distorted version of data. Quality improvement on distorted or corrupted data is achieved by forcing some of their feature components to match those from the original data. Finally, our approach can be treated as a new way to evaluate the uncertainty introduced by reduced or distorted data.

Our algorithm is flexible with data sets of different sizes, ranging from megabytes to gigabytes in the experiment. We believe that the general approach presented in this paper can be applied to quality assessment and improvement on larger scale data. As we move into the era of petascale computing, our work can help scientists perform in-situ processing so that low resolution data together with a set of features are saved to disk, which greatly reduces storage requirement and facilitates subsequent data analysis, quality assessment, and visualization.

Our current scheme is based on the GGD model which generally works well on data sets that exhibit natural statistics. We will investigate where and how well the GGD model works for different volume data. Furthermore, we can improve this model by augmenting it with a set of hidden random variables that govern the GGD parameters [17]. Such hidden Markov models may encompass a wider variety of data sets and yield better quality assessment results. On the other hand, we need to conduct a user study to suggest that the visual quality perceived by the users conforms to the quality assessment results obtained from our algorithm. In the future, we also would like to extend this reduced-reference approach to quality assessment of time-varying, multivariate data.

Appendix
The Calculation of GGD Parameters

The key MATLAB functions for calculating the GGD parameters (α, β) and for returning the GGD function values are provided as follows:

```
function f = fbeta(x)
% FBETA: an auxiliary function that computes beta

  f = exp(2 * gammaln(2 ./ x) - gammaln(3 ./ x)
          - gammaln(1 ./ x));
  % GAMMALN: logarithm of Gamma function


function [alpha, beta, K] = sbpdf(mean, variance)
% SBPDF: estimate generalized Gaussian probability
% density function of an wavelet subband using the
% moment matching method

  F = sprintf('fbeta(x) - %g', mean^2 / variance);

  try
    beta = fzero(F, [0.01, 5]);
    % FZERO: find zero of a function of one variable
  catch
    warning('(mean^2 / variance) is out of the range');
    if (mean^2 / variance) > fbeta(5)
      beta = 5;
    else
      beta = 0.01;
    end
  end

  alpha = mean * exp(gammaln(1/beta) - gammaln(2/beta));

  if (nargout > 2)
    K = beta / (2 * alpha * gamma(1/beta));
    % GAMMA: Gamma function
  end


function y = ggpdf(x, alpha, beta, K)
% GGPDF: return generalized Gaussian probability
% density function with parameters alpha and beta
% at the values in x

  if (alpha <= 0 | beta <= 0)
    tmp = NaN;
    y = tmp(ones(size(x)));
    % ONES: create an array of all ones
  else
    y = K * exp(-abs(x).^beta ./ (alpha^beta));
    y = y ./ sum(y);
  end
```

Acknowledgements

This research was supported in part by the U.S. National Science Foundation through grants CCF-0634913, CNS-0551727, OCI-0325934, and CCF-9983641, and the U.S. Department of Energy through the SciDAC program with Agreement No. DE-FC02-06ER25777, DOE-FC02-01ER41202, and DOE-FG02-05ER54817. The aneurysm data set was provided by Michael Meißner and made available by Dirk Bartz at http://www.volvis.org/. The visible woman data set was courtesy of the U.S. National Library of Medicine. The solar plume data set was provided by John Clyne (NCAR). The supernova data set was provided by John M. Blondin (NCSU) and Anthony Mezzacappa (ORNL). The milk crown data set was courtesy of Kenji Ono (RIKEN, Japan). Chaoli Wang would like to thank Hongfeng Yu and Shinho Kang for their helps in GPU and MATLAB programming, respectively. Finally, the authors would like to thank the reviewers for their helpful comments.

References

[1] C. L. Bajaj, S. Park, and I. Ihm. Visualization-Specific Compression of Large Volume Data. In *Proceedings of Pacific Graphics '01*, pages 212–222, 2001.

[2] A. C. Bovik. *Handbook of Image and Video Processing (2nd Edition)*. Academic Press, 2005.

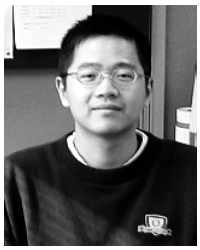
[3] R. W. Buccigrossi and E. P. Simoncelli. Image Compression via Joint Statistical Characterization in the Wavelet Domain. *IEEE Transactions on Image Processing*, 8(12):1688–1701, 1999.

[4] J. G. Daugman. Two-Dimensional Spectral Analysis of Cortical Receptive Field Profiles. *Vision Research*, 20:847–856, 1980.

[5] M. N. Do and M. Vetterli. Wavelet-Based Texture Retrieval Using Generalized Gaussian Density and Kullback-Leibler Distance. *IEEE Transactions on Image Processing*, 11(2):146–158, 2002.

[6] A. Gaddipati, R. Machiraju, and R. Yagel. Steering Image Generation with Wavelet Based Perceptual Metric. *Computer Graphics Forum*, 16(3):241–251, 1997.

[7] R. C. Gonzalez and R. E. Woods. *Digital Image Processing (2nd Edition)*. Prentice Hall, 2002.
[8] S. Guthe, M. Wand, J. Gonser, and W. Straßer. Interactive Rendering of Large Volume Data Sets. In *Proceedings of IEEE Visualization Conference '02*, pages 53–60, 2002.
[9] C. E. Jacobs, A. Finkelstein, and D. H. Salesin. Fast Multiresolution Image Querying. In *Proceedings of ACM SIGGRAPH '95*, pages 277–286, 1995.
[10] K. Kim, C. M. Wittenbrink, and A. Pang. Extended Specifications and Test Data Sets for Data Level Comparisons of Direct Volume Rendering Algorithms. *IEEE Transactions on Visualization and Computer Graphics*, 7(4):299–317, 2001.
[11] T.-Y. Kim and Y. G. Shin. An Efficient Wavelet-Based Compression Method for Volume Rendering. In *Proceedings of Pacific Graphics '99*, pages 147–157, 1999.
[12] L. Linsen, V. Pascucci, M. A. Duchaineau, B. Hamann, and K. I. Joy. Hierarchical Representation of Time-Varying Volume Data with $\sqrt[4]{2}$ Subdivision and Quadrilinear B-Spline Wavelets. In *Proceedings of Pacific Graphics '02*, pages 346–355, 2002.
[13] J. Luo and C. W. Chen. Coherently Three-Dimensional Wavelet-Based Approach to Volumetric Image Compression. *Journal of Electronic Imaging*, 7(3):474–485, 1998.
[14] S. Mallat. A Theory for Multiresolution Signal Decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693, 1989.
[15] S. Muraki. Volume Data and Wavelet Transforms. *IEEE Computer Graphics and Applications*, 13(4):50–56, 1993.
[16] H. V. Poor. *An Introduction to Signal Estimation and Detection (2nd Edition)*. Springer-Verlag, 1994.
[17] J. Portilla, V. Strela, M. J. Wainwright, and E. P. Simoncelli. Image Denoising Using Scale Mixtures of Gaussians in the Wavelet Domain. *IEEE Transactions on Image Processing*, 12(11):1338–1351, 2003.
[18] N. Sahasrabudhe, J. E. West, R. Machiraju, and M. Janus. Structured Spatial Domain Image and Data Comparison Metrics. In *Proceedings of IEEE Visualization Conference '99*, pages 97–104, 1999.
[19] A. Said and W. A. Pearlman. A New, Fast, and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(3):243–250, 1996.
[20] J. M. Shapiro. Embedded Image Coding Using Zerotrees of Wavelet Coefficients. *IEEE Transactions on Signal Processing*, 41(12):3445–3462, 1993.
[21] R. Shapley and P. Lennie. Spatial Frequency Analysis in the Visual System. *Annual Review of Neuroscience*, 8:547–83, 1985.
[22] E. P. Simoncelli and E. H. Adelson. Noise Removal via Bayesian Wavelet Coring. In *Proceedings of International Conference on Image Processing '96*, pages 16–19, 1996.
[23] B.-S. Sohn, C. L. Bajaj, and V. Siddavanahalli. Feature Based Volumetric Video Compression for Interactive Playback. In *Proceedings of IEEE Symposium on Volume Visualization '02*, pages 89–96, 2002.
[24] E. J. Stollnitz and D.H. Salesin. *Wavelets for Computer Graphics: Theory and Applications*. Morgan Kaufmann, 1996.
[25] G. Van de Wouwer, P. Scheunders, and D. Van Dyck. Statistical Texture Characterization from Discrete Wavelet Representations. *IEEE Transactions on Image Processing*, 8(4):592–598, 1999.
[26] C. Wang, A. Garcia, and H.-W. Shen. Interactive Level-of-Detail Selection Using Image-Based Quality Metric for Large Volume Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13(1):122–134, 2007.
[27] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.
[28] Z. Wang, G. Wu, H. R. Sheikh, E.-H. Yang, and A. C. Bovik. Quality-Aware Images. *IEEE Transactions on Image Processing*, 15(6):1680–1689, 2006.
[29] H. Zhou, M. Chen, and M. F. Webster. Comparative Evaluation of Visualization and Experimental Results Using Image Comparison Metrics. In *Proceedings of IEEE Visualization Conference '02*, pages 315–322, 2002.



Chaoli Wang received the BE and ME degrees in computer science from Fuzhou University, China, in 1998 and 2001, respectively, the PhD degree in computer and information science from The Ohio State University in 2006. Currently, he is a postdoctoral researcher in the Department of Computer Science at University of California, Davis. His research interests are computer graphics and visualization, with a focus on large-scale data analysis and visualization. He is a member of the IEEE.

Kwan-Liu Ma is a professor of computer science at the University of California-Davis, and he directs the DOE SciDAC Institute for Ultrascale Visualization. His research spans the fields of visualization, high-performance computing, and user interface design. Professor Ma received his PhD in computer science from the University of Utah in 1993. He received the NSF PECASE award in 2000, Schlumberger Foundation Technical Award in 2001, and UC Davis College of Engineering's Outstanding Mid-Career Research Faculty Award in 2007. He is the paper chair for the IEEE Visualization 2008 Conference, IEEE Pacific Visualization 2008 Symposium, and Eurographics Parallel Graphics and Visualization 2008 Symposium. Professor Ma also serves on the editorial boards of the IEEE Computer Graphics and Applications and the IEEE Transactions on Visualization and Graphics.