Title
Hierarchical Clustering for Unstructured Volumetric Scalar Fields

Authors
Co, Christopher S.
Heckel, Bjoern
Hagen, Hans
et al.

Publication Date
2003

Peer reviewed

Hierarchical Clustering for Unstructured Volumetric Scalar Fields

Christopher S. Co*
University of California, Davis

Bjoern Heckel†
Plumtree Software, Inc.

Hans Hagen‡
University of Kaiserslautern, Germany

Bernd Hamann*
University of California, Davis

Kenneth I. Joy*
University of California, Davis

Center for Image Processing and Integrated Computing (CIPIC)
Department of Computer Science
University of California, Davis

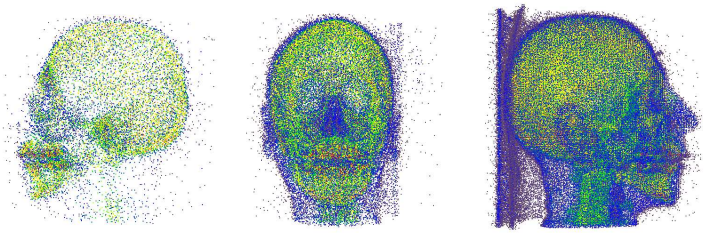


Figure 1: Scatter plots of cluster center points visualizing three levels of detail of a head data set.

Abstract

We present a method to represent unstructured scalar fields at multiple levels of detail. Using a parallelizable classification algorithm to build a cluster hierarchy, we generate a multiresolution representation of a given volumetric scalar data set. The method uses principal component analysis (PCA) for cluster generation and a fitting technique based on radial basis functions (RBFs). Once the cluster hierarchy has been generated, we utilize a variety of techniques for extracting different levels of detail. The main strength of this work is its generality. Regardless of grid type, this method can be applied to any discrete scalar field representation, even one given as a "point cloud."

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid and object representation I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Object hierarchies I.3.6 [Computer Graphics]: Methodology and Techniques—Graphics data structures and data types

Keywords: scalar field simplification, multiresolution data representation, hierarchical clustering, principal component analysis, radial basis function

*{co,hamann,joy}@cs.ucdavis.edu
†bjoern_heckel@hotmail.com
‡hagen@informatik.uni-kl.de

1 Introduction

Multiresolution techniques in computer graphics are commonly used for the manipulation of surfaces at various levels of detail. Although many of these surface methods can be generalized for the simplification of volume data, relatively few have been extended for the construction of volumetric multiresolution representations. Many of the existing multiresolution methods apply to hexahedral or tetrahedral meshes and typically require connectivity information. Thus, the direct application of such methods to scattered volumetric data, which contains no connectivity information, is usually not possible.

We propose a new approach for the construction of a hierarchical representation of any volumetric scalar data set. In a preprocessing step, we iteratively refine an initially coarse representation using clustering techniques to generate a hierarchy. At runtime, we extract levels of detail from this hierarchy to support interactive exploration.

We start with a coarse data representation, consisting of a single cluster containing all the sample points of the data set. This cluster is partitioned into two sub-clusters, which are inserted into a priority queue sorted by error. This procedure is applied iteratively; in each step, the cluster with the highest error is partitioned, and its sub-clusters are placed into the queue. Refinement terminates when a maximum number of iterations have been completed or when the maximum error in the priority queue is below some user-defined threshold. A hierarchy of clusters is built using the natural parent-child relationship created by this splitting procedure. We refer to the resulting hierarchy as a cluster binary tree (CBT). Field reconstruction and cluster partitioning are discussed in Section 3 and Section 4, respectively.

The level-of-detail extraction phase consists of a depth-first traversal over the CBT. We discuss two traversal methods: level-based and error-based. The level-based approach collects data in the hierarchy in a depth-first fashion, traversing the tree down to a user-defined maximum depth. The error-based approach gathers data in the cluster hierarchy based on an error threshold. The set of nodes collected by CBT traversal constitutes a level-of-detail rep-

resentation of the original data. As a result, multiple resolutions are represented by one compact binary tree. Level-of-detail extraction is discussed in Section 5.

Our goal is to build a multiresolution hierarchy without specific knowledge of the source of the volumetric scalar data. Although this information can often be employed to design more effective algorithms, it reduces the applicability of the method. The strength of our approach is that it can be applied to any volumetric scalar data set and avoids the costly generation of a grid.

2 Related Work

Although many simplification and multiresolution efforts have focused primarily on surface meshes, many techniques have been developed for volumetric data. Typically, multiresolution methods organize volumetric data based on regular or irregular grid structures. Regular grid structures include octrees, which have been used to provide adaptive levels of detail, see [Shekhar et al. 1996; Freitag and Loy 1999; Pinskiy et al. 2001]. Linsen et al. [2002; 2003] used wavelets and subdivision connectivity to represent and visualize regular grid data in a hierarchical fashion. Adaptive mesh refinement (AMR) techniques use a set of nested regular grids of varying resolution to represent volumes [Ohlberger and Rumpf 1997; Weber et al. 2001]. With respect to irregular grid methods, tetrahedral meshes have played an important role in constructing multiresolution hierarchies. Cignoni et al. [1994; 1997] described a system based on tetrahedral meshes to represent and visualize volumetric scalar data. Trotts et al. [1999] and Staadt and Gross [1998] used edge-collapse techniques to extend Hoppe's work [1996] for building progressive tetrahedrizations. Grosso and Greiner [1998] built hierarchical adaptive meshes using tetrahedra and octahedra.

Unfortunately, most of these methods cannot be applied directly to scattered data–i.e., data with no connectivity information–without first meshing the scattered data points or resampling the data to a regular grid. Tetrahedrizations can be expensive to compute and store, especially considering the increasing size of modern scientific data. Weber et al. [1999] proposed creating local triangulations at runtime in a given region of interest. While this approach provides a good solution for runtime visualization of scattered data, it does not lend itself to the construction of a multiresolution representation of volume data. Further, tetrahedrization algorithms can be difficult to implement in practice [Shewchuk 1997], as they require complex mesh data structures to be maintained [Pauly et al. 2002].

Resampling to a regular grid can produce many unnecessary redundancies, although adaptive sampling via an octree or an AMR representation can help. Regular grids typically sample in an axis-aligned fashion, which, though simple to implement and store, may not always produce a desirable partitioning of the volume. Grid data and multiresolution methods for grid data offer several advantages when exploring large data spaces, but these methods are not always easily generalized for all types of volume data. For example, methods for tetrahedral meshes can be applied to hexahedral meshes by decomposing hexahedra into tetrahedra, but the reverse is not true. The most general type of volumetric data is scattered data. Thus, any multiresolution method for scattered data could be applied to any gridded data by simply ignoring the mesh.

To create a multiresolution hierarchy for scattered scalar data, it is desirable to use methods that use "simple" connectivity, or no connectivity at all. Similar methods have been developed in surface simplification, surface reconstruction, and vector field hierarchy creation. These methods partition data points into similar sets, or clusters. Inspired by vector quantization methods, Brodsky and Watson [2000] used PCA to simplify models by refining an initially coarse representation. Their work prompted Shaffer and Garland [2001] to apply PCA-based vertex clustering and the dual quadric error metric to simplify models adaptively in an out-of-core fashion. Pauly et al. [2002] developed several extensions of multiresolution methods for point-sampled surfaces. Heckel et al. [1999a] used PCA to determine near-planar polygonal tiles for the reconstruction of surfaces from point cloud data. Vector field hierarchies were constructed using PCA in a similar way by clustering vectors that are locally similar [Heckel et al. 1999b].

One major issue in constructing a volumetric hierarchy from scattered data is the question of value approximation at arbitrary locations in the domain. In the absence of connectivity, radial basis functions (RBFs) are traditionally used to reconstruct a field. Hardy's multiquadric and reciprocal multiquadric methods have been used successfully for many scientific applications [Hardy 1990]. RBFs have been used in knot optimization for complex surfaces [Franke et al. 1995; Franke and Hagen 1999]. Carr et al. [2001] have demonstrated the usefulness of RBFs in surface reconstruction.

In our approach, we refine clusters of scattered data points using PCA to define partitioning planes intelligently. While the use of PCA is not new in surface simplification [Brodsky and Watson 2000; Shaffer and Garland 2001; Pauly et al. 2002], we extend PCA for use in volumetric scalar field simplification. We maintain a point hierarchy, similar in spirit to many multiresolution representations of surfaces [Rusinkiewicz and Levoy 2000; Pauly et al. 2002] with the exception that our points represent samples of a scalar field and not samples on a surface. We use RBFs defined using levels of detail from this point hierarchy for field reconstruction and value approximation.

3 The Scalar Field

Given a set of sample points S defined by a set of points $p_i = (x_i, y_i, z_i),\ \ i = 1, 2, \ldots, |S|$, let F_i be the scalar value associated with p_i. We define a cluster C to be a subset of points in S. The center of this cluster, p_c, is

$$p_c = \frac{1}{|C|} \sum_{p_j \in C} p_j.$$

We approximate a value F at $p = (x, y, z)$ using a multiquadric method [Hardy 1990; Franke and Hagen 1999]. This method effectively fits a function composed of a set of radial basis functions (RBFs) to the scattered data using least-squares approximation to derive function constants and a reparameterization of the data points.

To estimate the function values, we utilize an approximation of the form

$$F(x, y, z) = c + \sum_{j=1}^{N} a_j \, B_j(x, y, z), \tag{1}$$

where

$$B_j(x, y, z) = [\,(x_j - x)^2 + (y_j - y)^2 + (z_j - z)^2 + R^2\,]^{\pm 1/2},$$

subject to the constraints

$$F(x_j, y_j, z_j) = F_j \ \ j = 1, 2, \ldots, N\,. \tag{2}$$

We call c the *correction constant*, R the *multiquadric parameter*, and a_j the *blending coefficients*. The points p_j are often referred to as *knots*. The value of N is the number of data points used for value approximation. Classically, a global fit is performed by setting N to be the same as $|C|$, which can be computationally expensive for large clusters. We use a local scheme by considering only the N nearest neighbors to p, i.e., $N = min(\,|C|, k\,)$, where k is some

threshold. Often, equation (2) is too strict, and a least-squares fit is performed instead.

In the classic implementation of this algorithm, a minimization

$$\min_{\{knots,R,a_j,c\}} \sum_{i=1}^{N} \Big[\big(c + \sum_{j=1}^{N} a_j \, B_j(x_i, y_i, z_i) \big) - F_i \Big]^2$$

is performed, which is separated into two minimization steps:

$$\min_{\{knots,R\}} \min_{\{a_j,c\}} \sum_{i=1}^{N} \Big[\big(c + \sum_{j=1}^{N} a_j \, B_j(x_i, y_i, z_i) \big) - F_i \Big]^2.$$

The inner minimization step is performed by a least-squares operation. The outer minimization step is non-linear [Franke et al. 1995; Franke and Hagen 1999]. This optimization leads to values for c, R, a_j, and knots (x_j, y_j, z_j). The value at p is estimated using equation (1).

Our experiments have shown that this method works well but is computationally intensive. The least-squares fitting procedure requires solving an $N \times (N + 1)$ linear system, i.e., an underdetermined system with an infinite number of solutions. A non-negativity constraint can be added, however one must be careful to avoid near-singular matrices. The major bottleneck is the non-linear optimization, which attempts to optimize $3N + 1$ variables: the x-, y-, and z-components for the N knot locations and R.

To reduce computational cost, we have applied simplifications to the approximation method that do not sacrifice overall approximation quality and allow the knots to stay fixed. First, we neglect the correction constant c. In geometric modeling, the correction constant is used to smooth "bumpy" surfaces. However, we wish to preserve "sharp" features and not smooth them away. By removing the correction constant c, we obtain an $N \times N$ linear system and no longer need to consider an underdetermined matrix problem. Second, we increase the local neighborhood for function value estimation. We have found that increasing the neighborhood eliminates the need to optimize knot locations. For instance, we obtain near-equivalent results using 25 nearest neighbors without knot optimization when compared to using five neighbors with knot optimization. Finally, the multiquadric parameter R is defined to be a constant. Our experience shows that optimization routines frequently set R to values so small that it is of negligible influence. Thus, setting the multiquadric parameter to a sufficiently low value eliminates the need for non-linear optimization. The blending coefficients, in many cases, compensate for any effect that the multiquadric parameter has.

Thus, the local approximation of the scalar field at (x, y, z) is given by

$$F(x, y, z) = \sum_{j=1}^{N} a_j \, B_j(x, y, z),$$

where we determine each a_j by solving

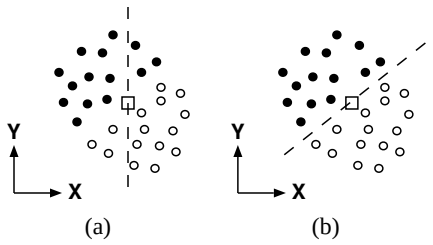
$$\min_{\{a_j\}} \sum_{i=1}^{N} \Big[\big(\sum_{j=1}^{N} a_j \, B_j(x_i, y_i, z_i) \big) - F_i \Big]^2.$$

For each cluster, we evaluate F at the cluster center p_c to compute F_c, the scalar field approximation at p_c.

4 Clusters

Given a cluster C, we define its error σ_c as

$$\sigma_c = \max_{p_j \in C} | F_j - F_c |$$



(a) (b)

Figure 2: Comparison of splitting schemes. Black dots represent points with scalar value one, and white dots represent points with scalar value zero. (a) Non-optimal splitting using an axis-aligned scheme, (b) near-optimal splitting.

where F_j is the scalar value associated with point p_j, and F_c is the value approximated at cluster center p_c. In other words, the error σ_c is the maximum deviation in scalar value between the approximated value and the values of all points in C. This error measure is simple to compute and suffices to identify clusters to be refined.

In each refinement step, the cluster with the highest error is obtained from the queue. We partition it into two smaller clusters by defining a splitting plane that divides the cluster into two distinct subsets. Center points, value approximations at center points, and errors are computed for the two resulting sub-clusters. The sub-clusters are then inserted into the queue. A binary tree is maintained during refinement by setting the sub-clusters to be children of the cluster just split. In this hierarchy, each cluster is interpreted as a data point of a given resolution in the hierarchy whose location is the cluster center p_c.

One possible refinement strategy uses an axis-aligned scheme. In this method, the cluster center and one coordinate axis determine the splitting plane. This k-D tree style splitting scheme [Samet 1990] is efficient and allows us to determine the splitting plane simply, but it may not produce a good decomposition of the data set. Figure 2 demonstrates this effect. Furthermore, we wish to define a splitting plane that reduces the error in the sub-clusters the most. We use the cluster center and a normal vector obtained from principal component analysis (PCA) on all four components of the scalar field (x_i, y_i, z_i, F_i) to define a more adaptive splitting plane. For a detailed explanation of PCA, we refer the reader to [Jolliffe 1986; Heckel et al. 1999a; Brodsky and Watson 2000; Shaffer and Garland 2001].

We first discuss how 3-D PCA can be used for bivariate scalar fields. For bivariate scalar field data (x_i, y_i, F_i), we can perform PCA in 3-D to obtain an orienting normal for a splitting line. PCA performs an eigen-decomposition of the covariance matrix of a set of samples producing, in the 3-D case, eigenvalues

$$\lambda_1 \geq \lambda_2 \geq \lambda_3$$

and corresponding eigenvectors

$$\vec{e}_1, \, \vec{e}_2, \text{ and } \vec{e}_3,$$

which define a local orthogonal coordinate system related to an ellipsoid induced by the data points. We use the vector corresponding to the dominant axis of this ellipsoid, i.e., $\vec{e}_1$, as the splitting plane's normal. However, to partition the 2-D points in the domain, we require a 2-D normal vector. We project the 3-D eigenvector to xy-space, see Figure 3. In most cases, we obtain a suitable normal by simply dropping the last component of the eigenvector.

We must consider the case when $\vec{e}_1$ is a multiple of the vector $\langle 0, 0, 1 \rangle$. Such a vector projected to xy-space produces the null vector. In this case, we choose the second dominant eigenvector,

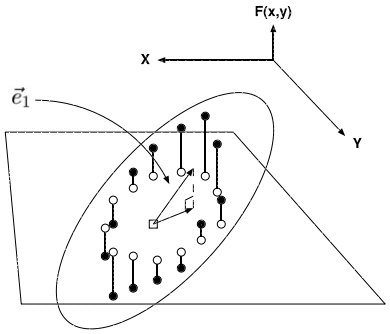


Figure 3: Example of 3-D PCA normal projected to xy-space. The white dots are data points in the xy-plane. The height of each black dot indicates the scalar value at the data point. The ellipsoid represents the local coordinate system computed by 3-D PCA. The dominant eigenvector $\vec{e}_1$ is shown with its projection onto the xy-plane.

$\vec{e}_2$, which is guaranteed to be non-null when projected to xy-space, since it is orthogonal to $\vec{e}_1$. In practice, this occurs infrequently. (This never occurred in the results presented in Section 7.) Figure 4 illustrates the progression of the cluster splitting procedure.

This technique generalizes to trivariate scalar field data (x_i, y_i, z_i, F_i). PCA returns eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \lambda_4$ with corresponding eigenvectors $\vec{e}_1$, $\vec{e}_2$, $\vec{e}_3$, and $\vec{e}_4$. Again, we use the projection of $\vec{e}_1$ to xyz-space to define a splitting plane normal. When projection of $\vec{e}_1$ maps to the null vector in xyz-space, we choose $\vec{e}_2$ as our orienting normal.

Defining the splitting plane in this way divides a cluster across the axis of its greatest variation, thereby decreasing the cluster's error. Geometrically speaking, this partitioning splits one ellipsoid into two "rounder" child ellipsoids with reduced eccentricity. This shape can have an important positive side-effect on subsequent partitioning. When clusters become very thin in the split direction, it is possible that splitting a cluster produces a sub-cluster with no data points due to numerical error. Splitting across the dominant axis as defined by PCA avoids the creation of thin clusters by producing sub-clusters that are as "round" as possible. This numerical issue cannot be avoided completely; when it occurs, the "problem cluster" is not re-inserted into the error queue.

Cluster size can also have an effect on splitting. Eventually, clusters contain only a few data points. A minimum cluster size can be defined to set a "pseudo-compression ratio" for the finest resolution in the data hierarchy. Defining a minimum cluster size can also reduce the occurrences of the zero-size cluster problem.

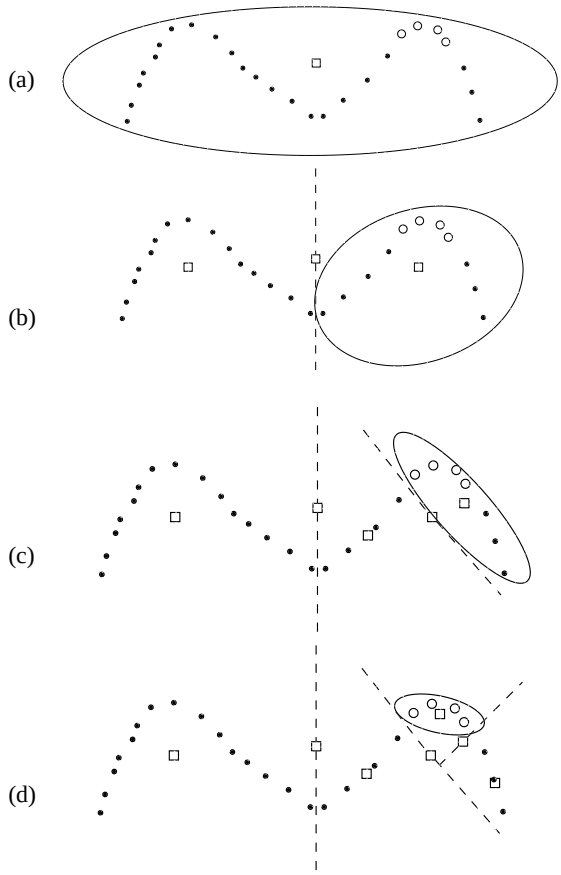


Figure 4: Example of clustering of 2-D scattered data. Black dots indicate points with scalar value one, and white dots indicate points with scalar value zero. Squares represent cluster centers, which become new data points in the generated hierarchy. (a) CBT generation begins with one initial cluster; (b) cluster is split into two sub-clusters; only the right cluster is chosen for splitting; (c) right cluster is split into two sub-clusters; (d) final split; all clusters have zero error.

5 Level-of-detail Extraction

Extraction of levels of detail from the CBT is performed by traversing the tree in a depth-first fashion, using either a *level-based* traversal that obtains the clusters in the hierarchy at a given level of the tree, or an *error-based* traversal that returns clusters in the hierarchy that have an error below a threshold. If a leaf node is encountered in the CBT during extraction, we use the cluster at the leaf and continue traversal. When traversing the CBT in a level-based manner, levels of detail are specified by the maximum depth to traverse the binary tree. When traversing the CBT in an error-based manner, levels of detail are specified by a maximum error that the level-of-detail should exhibit. (The hierarchy can only guarantee clusters with an error less than or equal to the maximum error at the termination of the preprocessing.) With the error metric we have defined, a low error threshold returns a high fidelity representation. Conversely, a higher error threshold returns a less faithful but memory efficient representation.

Error-based traversal offers a more compact representation, whereas the level-based approach provides better spatial distribution by covering the space spanned by the hierarchy with more data points. This phenomenon can be seen in the examples shown in Figure 5 and is further discussed in Section 7. In Figure 5, two levels of detail are shown for each traversal method. Between two resolutions of the hierarchy traversed in a level-based way, additional points are added in a spatially uniform manner. Between two resolutions of an error-based hierarchy, the distribution of the additional points depends more strongly on the nature of the field. Figure 5 demonstrates this effect for a silicium data set. In the error-based traversal examples (Figure 5, right column), only a few clusters are extracted to represent the volume outside the silicium structure, whereas several more clusters are used in that same region using a level-based approach (Figure 5, left column).

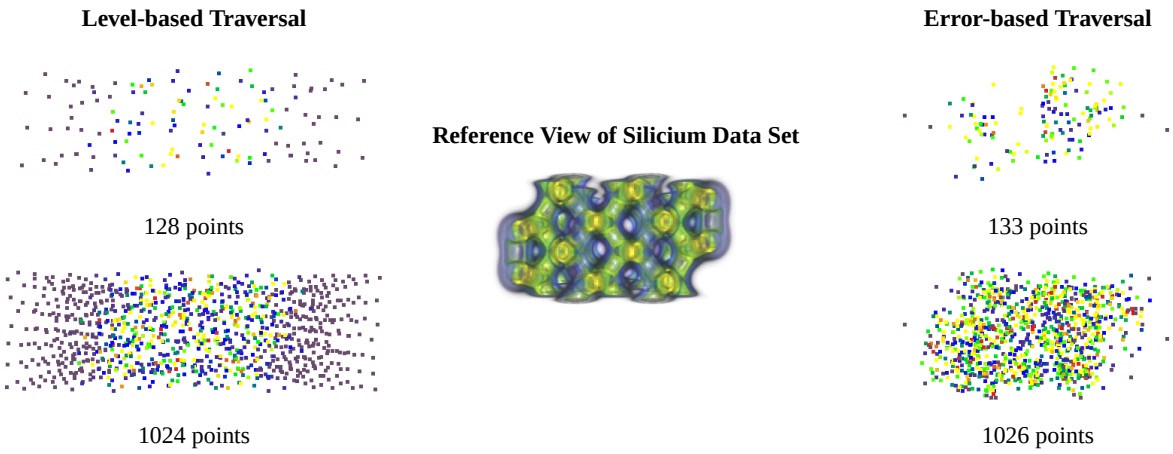


Figure 5: Demonstration of the difference between level- and error-based CBT traversal. Shown are 3-D cluster center point scatter plots visualizing levels of detail extracted from a CBT. The volume rendering (center) is provided for reference. Level-based CBT traversals (left column) provide better spatial data point distribution while error-based CBT traversals (right column) provide more detail using roughly the same number of data points.

6 Multiresolution Representation

The clusters extracted from the CBT define a level-of-detail representation of the volume by considering the p_c and F_c for each cluster as locations and values of a scattered scalar field. This set of data points, in conjunction with the field reconstruction basis functions described in Section 3, allows us to evaluate the function at any arbitrary location, and thus construct visualizations.

7 Results

We generated CBT hierarchies for three data sets. We extracted low- and high-resolution levels of detail from each CBT and volume rendered the fields to inspect the quality of the representation.

Table 1 summarizes the preprocessing results to generate the CBTs. For value approximation at the cluster centers and for sampling the data, we used 25 nearest neighbors and fixed the multiquadric parameter to 0.025. The minimum splittable cluster size was set to two. All function values are between zero and 255.

Figure 6 shows volume rendered visualizations of different levels of detail represented by the CBTs. The first column shows a volume rendering of the original data. Columns two and three provide volume visualizations of high and low resolutions, respectively. Information concerning the parameters used to extract these levels of detail and the number of data points used for rendering is provided in Table 2. To illustrate the quality of the various resolutions, "zooms" are given for the argon bubble data set in Figures 6(l), (m), and (n), and difference images are provided for the silicium data set in Figures 6(d) and (e). The difference images were obtained by differencing images in Figure 6(b) and (c) against image (a) of the original data set.

This method lends itself to data and computation parallelism. Once a cluster is split, its sub-clusters can be further split on separate machines. Since the hierarchy is based on a binary tree, merging the final results is low in cost. For each data set, preprocessing was performed in parallel on a hybrid PC cluster consisting of three machines with Pentium4 2.8 GHz processors with 2 Gb of main memory, and one machine with a Pentium4 2.2 GHz processor with 1 Gb of main memory.

As mentioned in Section 4, eventually clusters either converge to the granularity of the data set or suitable partitioning is not possible due to lack of precision. "Skipped splits" can result in the refinement, and their numbers are listed in Table 1.

Naturally, lower resolutions are less faithful to the original data set. This fact is seen in the results for the head data set, shown in Figures 6 (f), (g), and (h). A low-resolution representation produces a low-quality volume rendered result as seen in Figure 6(h), where the entire back plate from the scan is not represented in detail. Although less than one percent of the number of data points (17,528 data points) was used to construct the image, salient features of the head, such as the eye sockets and spikes around the teeth, are still discernible. Deficiencies of low resolutions are eliminated in higher resolutions, as seen in Figure 6(g).

The images shown in Figure 6 of the silicium data set demonstrate some of the differences between error-based and level-based hierarchies. The argon bubble and head results were generated using an error-based approach, whereas the silicium data set results were generated using level-based extraction. As can be seen in the difference images, Figures 6(d) and (e), successively higher resolutions of the data set improve the overall quality of the field representation but not any particular region. By contrast, the images of the head demonstrate that successively higher resolutions of the error-based hierarchy adaptively improve specific regions. The back plate and nose appear "blobby" in Figure 6(h) and not in (g), while the regions around the teeth and eye socket remain fairly consistent across resolutions.

Levels of detail of the argon bubble data set exhibit high-quality using few data points. In the low and high resolutions of the data set, shown in Figures 6(i), (j), and (k), less than five percent of the number of the original data points was used to reconstruct the field. Zooms shown in Figures 6(l), (m), and (n) illustrate the quality of the representation.

Running time depends on the nature of the data being processed and the parameters used. Although one could attempt to analyze the performance as a function of the input data size, the limiting factor with most any simplification algorithm is the inherent complexity of the data set itself. Compared with data sets with relatively small variation in scalar value, data sets with large variation in scalar value do not converge as quickly to meet the error threshold. In the CBT generation phase, the primary bottleneck is value approximation at the cluster centers. We provide timing results in Table 1 for the preprocessing of the data used. We emphasize that the running time of data preprocessing is strongly dependent on the

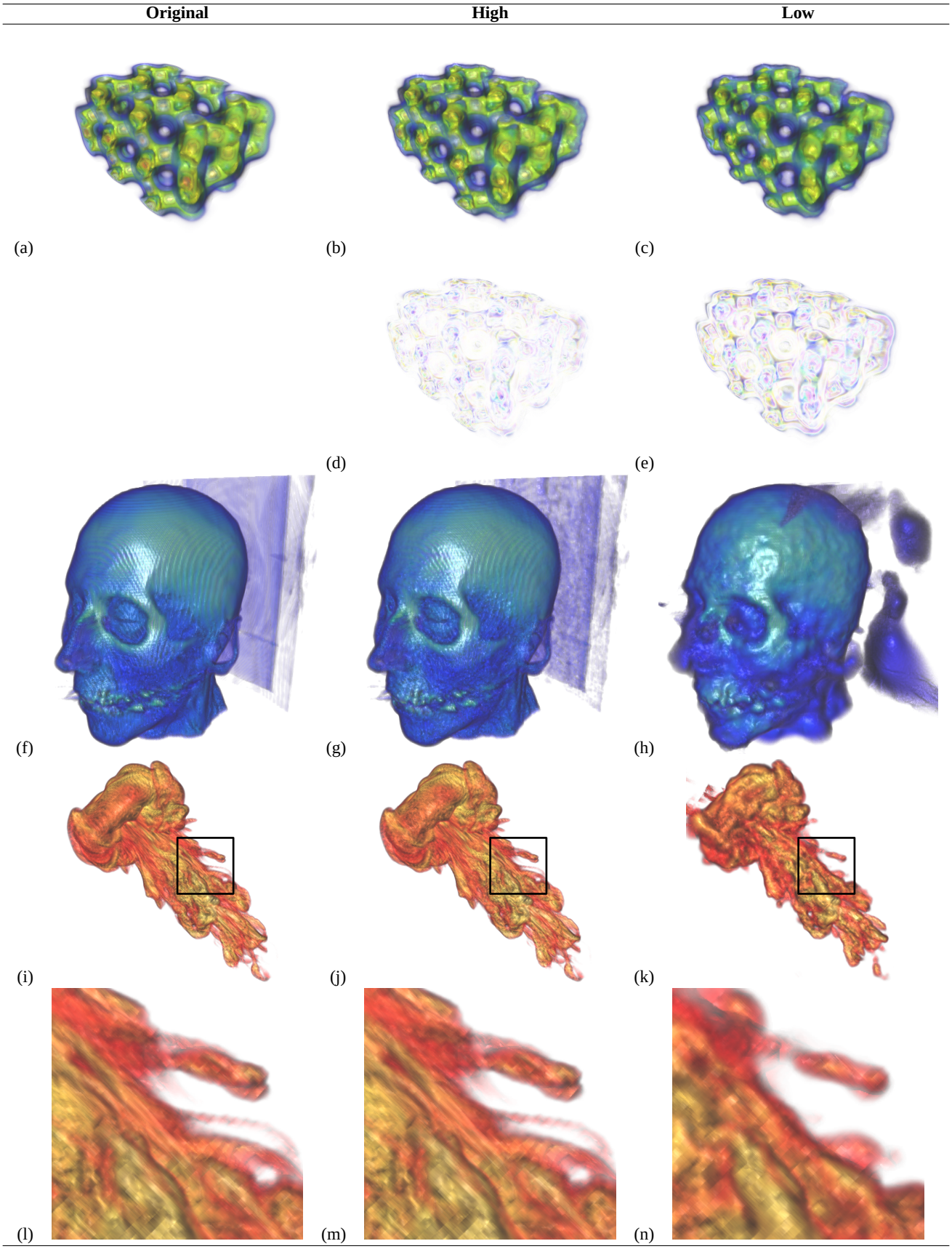


Figure 6: Quality comparisons for various resolutions of three data sets.

nature of the data set and the parameters used for preprocessing; it is much less affected by the input data size. As an example, the argon bubble data set, although larger in size, required much less time than the head data set to preprocess, see Table 1. However, the parallel nature of the method allows us to achieve better performance through the use of additional processors.

The CBT's space requirement is linear with respect to the number of iterations, since each iteration of the refinement process adds at most two new members to the cluster hierarchy. The time required to obtain a given resolution of the data from the cluster hierarchy is the time required to traverse the CBT, which is $O(\,2p+1\,)$, the time to perform a depth-first traversal in the worst case, where p is the number of nodes in the tree. Extracting lower resolutions from the data hierarchy requires less time due to the traversal termination conditions.

8 Conclusions

We have presented a method for the creation of a multiresolution hierarchy for discrete scalar field data through iterative refinement. The method uses clustering techniques to construct a data hierarchy. The refinement termination condition can be based on a user-defined upper bound for the number of iterations or on an error threshold. Levels of detail are extracted from the hierarchy using one of two depth-first tree traversal methods. Additional modes of data extraction could be defined by storing other parameters in the cluster hierarchy, such as gradient, function value range spanned by a cluster, or spatial range covered by a cluster. This flexibility allows one to customize the hierarchy to conform to application specific needs. To what degree these parameters can be used to enhance visualization is a topic for future work.

The methods described do not rely on explicit connectivity information, and this fact is the primary strength of this approach. The algorithm can be applied to any type of volumetric scalar data regardless of the presence of connectivity information and without the use of complex mesh data structures. The multiresolution hierarchy can be constructed for large data sets as well, since cluster hierarchy generation can be done in parallel.

We plan to explore the effectiveness of different error metrics and optimal placement of the splitting plane. We used a simple error metric, effectively the L_∞ norm, to determine which clusters exhibit largest scalar error. A study of the effect on hierarchical clustering of different measures, such as the L_1 or L_2 norms, should be done. We used the cluster centers to specify a splitting plane. The use of other locations, such as the median point of the cluster, might have an interesting effect on the simplification process.

This method can be used to explore features and optimize sampling locations in arbitrary data. Since the cluster with the largest error is split at each iteration, dense clustering occurs in regions of high scalar value variation, and fewer clusters appear in lower variation regions. By virtue of this characteristic, this method has potential for feature detection. Different features could be tracked by applying, for example, a transfer function filter to the scalar values prior to hierarchy generation. Different error metrics and extraction routines could further enhance detection of features. This algorithm can be used to optimize sampling locations in volumetric data due to its adaptive nature. More splitting occurs in regions of high scalar variation. Thus, by observing where the hierarchical clustering method places more data points, one could intelligently resample a given data set using more optimal sampling locations obtained from the preprocessing step.

Acknowledgments

This work was supported by the National Science Foundation under contracts ACI 9982251 and ACI 0222909, and through the National Partnership for Advanced Computational Infrastructure (NPACI); the National Institutes of Health under contract P20 MH60975-06A2; the Lawrence Livermore National Laboratory under contract B523818; and the Lawrence Berkeley National Laboratory. We thank the members of the Visualization and Graphics Research Group at the Center for Image Processing and Integrated Computing (CIPIC) at the University of California, Davis. The time-varying argon bubble data set was provided by the Center for Computational Sciences and Engineering at the Lawrence Berkeley National Laboratory. We thank Zhaojun Bai for helping in the context of linear systems and numerical algorithms.

References

BRODSKY, D., AND WATSON, B. 2000. Model simplification through refinement. In *Proceedings of the Graphics Interface 2000*, Canadian Information Processing Society, Toronto, Ontario, 221–228.

CARR, J. C., BEATSON, R. K., CHERRIE, J. B., MITCHELL, T. J., FRIGHT, W. R., MCCALLUM, B. C., AND EVANS, T. R. 2001. Reconstruction and representation of 3D objects with radial basis functions. In *SIGGRAPH 2001 Conference Proceedings, August 12–17, 2001, Los Angeles, CA*, ACM Press, New York, NY 10036, USA, ACM, Ed., 67–76.

CIGNONI, P., DE FLORIANI, L., MONTONI, C., PUPPO, E., AND SCOPIGNO, R. 1994. Multiresolution modeling and visualization of volume data based on simplicial complexes. In *1994 Symposium on Volume Visualization*, A. Kaufman and W. Krueger, Eds., ACM SIGGRAPH, 19–26.

CIGNONI, P., PUPPO, E., AND SCOPIGNO, R. 1997. Multiresolution representation and visualization of volume data. *IEEE Transactions on Visualization and Computer Graphics 3*, 4 (Oct.), 352–369.

FRANKE, R., AND HAGEN, H. 1999. Least squares surface approximation using multiquadrics and parametric domain distortion. *CAGD 16*, 177–196.

FRANKE, R., HAGEN, H., AND NIELSON, G. M. 1995. Repeated knots in least squares multiquadric functions. In *Geometric modelling, Dagstuhl, Germany 1993*, Springer, Wien / New York, H. Hagen, G. E. Farin, H. Noltemeier, and R. Albrecht, Eds., vol. 10 of *Computing. Supplementum*, 177–187.

FREITAG, L. A., AND LOY, R. M. 1999. Adaptive, multiresolution visualization of large data sets using a distributed memory octree. In *Proceedings of SC99: High Performance Networking and Computing*, ACM Press and IEEE Computer Society Press, Portland, OR.

GROSSO, R., AND GREINER, G. 1998. Hierarchical meshes for volume data. In *Proceedings of the Conference on Computer Graphics International 1998 (CGI-98)*, IEEE Computer Society, Los Alamitos, CA, F. E. Wolter and N. M. Patrikalakis, Eds., 761–771.

HARDY, R. L. 1990. Theory and applications of the multiquadric-biharmonic method: 20 years of discovery 1968–1988. *Computers and Mathematics with Applications 19*, 163–208.

HECKEL, B., UVA, A. E., HAMANN, B., AND JOY, K. I. 1999. Surface reconstruction using adaptive clustering methods. In *Geometric Modelling: Dagstuhl 1999 ,Computing Suppl*, Springer-Verlag, G. Brunnett, H. Bieri, and G. Farin, Eds., vol. 14, 199–218.

HECKEL, B., WEBER, G. H., HAMANN, B., AND JOY, K. I. 1999. Construction of vector field hierarchies. In *Proceedings IEEE Visualization '99*, IEEE, San Francisco, CA, D. S. Ebert, M. Gross, and B. Hamann, Eds., IEEE, 19–26.

HOPPE, H. 1996. Progressive meshes. In *SIGGRAPH 96 Conference Proceedings*, Addison Wesley, H. Rushmeier, Ed., Annual Conference Series, ACM SIGGRAPH, 99–108.

Data set	Number of splits	Skipped splits	CBT depth	Processing time
Silicium	64,084	2	19	50 s 735 us
Head	300,663	19	23	44 min 32 s
Argon bubble	224,009	6	25	8 min 11 s

Table 1: CBT hierarchy generation statistics.

Data set	Figure	Extraction	Threshold	Size (points)	% Size
Silicium	(b)	level	15	20,982	18.52 %
	(c)	level	16	39,324	34.71 %
(original)	(a)	n/a	n/a	113,288	100 %
Head	(h)	error	60.00	17,528	0.84 %
	(g)	error	4.50	300,648	14.34 %
(original)	(f)	n/a	n/a	2,097,162	100 %
Argon bubble	(k), (n)	error	50.00	11,397	0.22 %
	(j), (m)	error	1.00	224,003	4.27 %
(original)	(i), (l)	n/a	n/a	5,242,880	100 %

Table 2: Statistics for visualizations shown in Figure 6. Column two lists the location(s) of images in Figure 6 to which a row's information applies.

JOLLIFFE, I. T. 1986. *Principal Component Analysis*. Springer-Verlag, New York, NY.

LINSEN, L., PASCUCCI, V., DUCHAINEAU, M. A., HAMANN, B., AND JOY, K. I. 2002. Hierarchical representation of time-varying volume data with 4th-root-of-2 subdivision and quadrilinear B-spline wavelets. In *Proceedings of Tenth Pacific Conference on Computer Graphics and Applications - Pacific Graphics 2002*, IEEE Computer Society Press, S. Coquillart, H.-Y. Shum, and S.-M. Hu, Eds.

LINSEN, L., GRAY, J. T., PASCUCCI, V., DUCHAINEAU, M. A., HAMANN, B., AND JOY, K. I. 2003. Hierarchical large-scale volume representation with $\sqrt[3]{2}$ subdivision and trivariate B-spline wavelets. In *Geometric Modeling for Scientific Visualization*, Springer-Verlag, Heidelberg, Germany, to appear, G. Brunnett, B. Hamann, H. Müller, and L. Linsen, Eds.

OHLBERGER, M., AND RUMPF, M. 1997. Hierarchical and adaptive visualization on nested grids. *Computing 59*, 4, 365–385.

PAULY, M., GROSS, M., AND KOBBELT, L. P. 2002. Efficient simplification of point-sampled surfaces. In *Proceedings of the 13th IEEE Visualization 2002 Conference (VIS-02)*, IEEE Computer Society, Piscataway, NJ, R. Moorhead, M. Gross, and K. I. Joy, Eds., 163–170.

PINSKIY, D. V., BRUGGER, E. S., CHILDS, H. R., AND HAMANN, B. 2001. An octree-based multiresolution approach supporting interactive rendering of very large volume data sets. In *Proceedings of The 2001 International Conference on Imaging Science, Systems, and Technology*, H. R. Arabnia, R. F. Erbacher, X. He, C. Knight, B. Kovalerchuk, M. M. O. Lee, Y. Mun, M. Sarfraz, J. Schwing, and M. H. N. Tabrizi, Eds., vol. 1, 16–22.

RUSINKIEWICZ, S., AND LEVOY, M. 2000. QSplat: A multiresolution point rendering system for large meshes. In *Siggraph 2000, Computer Graphics Proceedings*, ACM Press / ACM SIGGRAPH / Addison Wesley Longman, K. Akeley, Ed., Annual Conference Series, 343–352.

SAMET, H. 1990. *The Design and Analysis of Spatial Data Structures*, reprinted with corrections ed. Series in Computer Science. Addison-Wesley, Reading, MA, Apr.

SHAFFER, E., AND GARLAND, M. 2001. Efficient adaptive simplification of massive meshes. In *Proceedings of the Conference on Visualization 2001 (VIS-01)*, IEEE Computer Society, Piscataway, NJ, T. Ertl, K. Joy, and A. Varshney, Eds., 127–134.

SHEKHAR, R., FAYYAD, E., YAGEL, R., AND CORNHILL, J. F. 1996. Octree-based decimation of marching cubes surfaces. In *Proceedings of the Conference on Visualization*, IEEE, Los Alamitos, CA, R. Yagel and G. M. Nielson, Eds., 335–344.

SHEWCHUK, J. R. 1997. *Delaunay Refinement Mesh Generation*. PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA. Available as Technical Report CMU-CS-97-137.

STAADT, O. G., AND GROSS, M. H. 1998. Progressive tetrahedralizations. In *Proceedings of IEEE Visualization '98*, D. Ebert, H. Hagen, and H. Rushmeier, Eds., 397–402.

TROTTS, I. J., HAMANN, B., AND JOY, K. I. 1999. Simplification of tetrahedral meshes with error bounds. *IEEE Transactions on Visualization and Computer Graphics 3*, 5 (July/Sept.), 224–237.

WEBER, G. H., HECKEL, B., HAMANN, B., AND JOY, K. I. 1999. Procedural generation of triangulation-based visualizations. In *Proceedings of IEEE Visualization '99 (Late Breaking Hot Topics)*, A. Varshney, C. M. Wittenbrink, and H. Hagen, Eds., IEEE.

WEBER, G. H., KREYLOS, O., LIGOCKI, T. J., SHALF, J. M., HAGEN, H., HAMANN, B., JOY, K. I., AND MA, K.-L. 2001. High-quality volume rendering of adaptive mesh refinement data. In *Proceedings of the Vision Modeling and Visualization Conference 2001 (VMV-01)*, Aka GmbH, Berlin, T. Ertl, B. Girod, G. G. H. Niemann, and H.-P. Seidel, Eds., 121–128.