Construction of Vector Field Hierarchies

Bjoern Heckel[*]

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

Gunther Weber[†]

AG Graphische Datenverarbeitung and Computergeometrie
Fachbereich Informatik
Universität Kaiserslautern, Germany

Bernd Hamann[‡]
Kenneth I. Joy[§]

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

Abstract

We present a method for the hierarchical representation of vector fields. Our approach is based on iterative refinement using clustering and principal component analysis. The input to our algorithm is a discrete set of points with associated vectors. The algorithm generates a top-down segmentation of the discrete field by splitting clusters of points. We measure the error of the various approximation levels by measuring the discrepancy between streamlines generated by the original discrete field and its approximations based on much smaller discrete data sets. Our method assumes no particular structure of the field, nor does it require any topological connectivity information. It is possible to generate multiresolution representations of vector fields using this approach.

Keywords: vector field visualization; Hardy's multiquadric method; binary-space partitioning; data simplification.

1 Introduction

The rapid increase in the power of computer systems coupled with the improving precision of computational simulations now produce terascale data sets. The critical research problem encountered in the visualization of these data sets is the development of methods for storing, approximating, and rendering them. The crux of the problem is to reduce the size of the data sets while preserving essential features. We create different representations (or approximation levels) of a data set, each of which can be substituted for the complete set, depending on the requirements of a visualization technique. The given discrete vector field may be represented by a few data elements or by several billion elements if necessary, with each of the various representation levels containing as many as possible of the essential features contained in the original data set.

In this paper, we address simplification of vector fields. Our method creates a disjoint set of "clusters," and thereby a simplified discrete vector field, where a single point and vector is used to represent a cluster. We define a bisection strategy for clusters that utilizes a plane to effectively split a discrete data set. The "tiling" that is implied by this repeated bisection procedure creates a partitioning of space into convex regions. Each of these convex regions, bounded by certain split planes used during the split process, contains a particular discrete subset of the original data.

Each cluster has an associated error measure that depends on the differences in the streamlines generated from the points of the discrete vector field that are contained in the cluster. We measure the deviation of these simplified vector fields from the original discrete field and base our cluster generation process on these errors. Our algorithm splits clusters recursively, and splitting continues until a certain error condition is met.

Most standard hierarchical visualization and data representation schemes require a mesh defining the connectivity of data points at varying levels of resolution. Except in very specialized situations where the connectivity among points is implicit (e.g., rectilinear and curvilinear grids), one must store connectivity information explicitly. This is a major overhead concerning storage and also processing requirements. Our approach is among the very first stressing the concept of "gridless" representation. Our vector field data hierarchy is solely based on positional and vector information without the need to store or compute any connectivity information.

One of the main motivations for this work is to provide tools that enable interactive and real-time browsing of massive scientific data sets. Our method supplies a very flexible means for producing coarser levels of approximation of massive data, thus enabling interactive exploration. The vector field hierarchies generated with our approach can easily be adapted to support local level-of-detail rendering, or user-steered local refinement operations to focus on

[*]heckel@cs.ucdavis.edu
[†]weber@informatik.uni-kl.de
[‡]hamann@cs.ucdavis.edu
[§]joy@cs.ucdavis.edu

certain details present in a vector field.

In Section 2, we review mesh simplification algorithms that apply directly to our work. In Section 3, we survey approximation techniques that we use for vector field data. In Section 4, we discuss error measures that differentiate a simplified vector field from the original one. In Section 5, we describe the procedure that determines the bisection plane used to split a cluster. More practical aspects of our algorithm are covered in Section 6. Results of our algorithm are provided in Section 7. Conclusions and future work are presented in Section 8.

2 Related Work

Given a discrete vector field as a set of points $\{\mathbf{x}_i = (x_i, y_i, z_i) : i = 1, ..., n\}$ and a corresponding set of vectors $\{\mathbf{v}_i = (u_i, v_i, w_i) : i = 1, ..., n\}$, the goal of our simplification method is to construct a new discrete representation of the vector field containing fewer points. We present an algorithm based on a top-down refinement approach by using an adaptive clustering method. This type of simplification requires no topological knowledge or connectivity information.

A number of methods have been developed that simplify scalar fields, but research concerning simplification of vector fields is still in its infancy. Several simplification algorithms use a bottom-up or a top-down strategy for the construction of multiple approximation levels of a given field. In a bottom-up strategy, the given (high-resolution) grid is examined to identify regions where the mesh can be simplified, and the mesh is decimated in these areas. This process continues until an error threshold is reached. In a top-down strategy, an initial mesh is iteratively refined. One starts with a very coarse mesh and inserts points into the mesh until a desired error condition is met. Most of these methods require topological connectivity information for the data points.

Nielson *et al.* [12, 13] have used a wavelet approach to simplify vector fields over the sphere and over curvilinear grids. In [12], they define a class of Haar wavelets over triangular domains and apply these techniques to simplify a vector field over a sphere. In [13], they utilize "lifted Haar wavelets" and apply them to curvilinear grids.

Helman and Hesselink [7] have developed methods to simplify the visualization of two-dimensional vector fields by visualizing the topology of the field. The topology is visualized by specifying a collection of tangent curves that separate a flow into regions. The tangent curves connect critical points, where the flow is zero. These methods provide a good way to simplify two-dimensional vector fields, but these techniques have not been extended to three dimensions.

The algorithm we present in this paper utilizes clustering, similar to the work of Heckel *et al.* [5, 6], who use these techniques for the generation of surface triangulations for a given set of scattered surface data. In these works, they use adaptive clustering methods to construct near-planar point clusters that can be directly triangulated. The resulting cluster triangulations, together with a triangulation of the space between the clusters, provides a valid surface reconstruction of a point set. The authors utilize principal component analysis (PCA), see Hotelling [8], Jackson [9], or Manly [10], to find a best-fit plane to each cluster, and then split perpendicular to this plane.

Most of these methods are based on an error measure. However, error measures are difficult to define for vector fields. We have decided to utilize an error measure that pertains to the visualization itself, *i.e.*, we consider an approximation of a given vector field to be a good approximation if the visualizations of the original data set and the approximation are very similar.

The algorithm we use is based on a top-down approach. We utilize clustering [10] to generate a binary space partitioning (BSP) of the data set. Initially, all points of the given discrete vector field are placed in a single cluster. Clusters are split using a weighted best-fit plane that splits the clusters so that the variance of the error is reduced in the child clusters. Each cluster has an associated error measure that depends on the differences in the streamlines generated from the points of the original discrete vector field that are contained in the cluster. The algorithm splits clusters recursively, and splitting continues until a certain error condition is met. The approach is "gridless" in the sense that we never require any point connectivity.

3 Approximation of Unstructured Vector Fields

A discrete vector field in three-dimensional space is defined by a set of points $\mathbf{x}_i$, $i = 1, ..., n$, and a set of associated vectors $\mathbf{v}_i$. We assume that the data set is a scattered data set, with no mesh defining the connectivity of the points. We define an analytical approximation to the vector field by using *Hardy's multiquadric method*, see [2, 3, 4, 11], which is one of the most effective and most commonly used methods for scattered data interpolation.

Given n points $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n$ and associated vector values $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n$, Hardy's multiquadric interpolant is defined by the solution of the $n \times n$ linear system of equations

$$\mathbf{v}_i = \sum_{j=1}^{n} \alpha_j \sqrt{||\mathbf{x}_i - \mathbf{x}_j||^2 + R^2}, \quad i = 1, ..., n,$$

where the parameter R^2 is a positive constant. Once the unknown coefficients α_j are computed, one can approximate a vector value $\mathbf{v}$ at an arbitrary point $\mathbf{x}$ in space by

$$\mathbf{v}(\mathbf{x}) = \sum_{j=1}^{n} \alpha_j \sqrt{||\mathbf{x} - \mathbf{x}_j||^2 + R^2}.$$

The computation of the coefficients α_j requires the inversion of an $n \times n$ matrix. The function $\mathbf{v}(\mathbf{x})$ smoothly interpolates the vector values at the n data points and provides a vector estimate at any point $\mathbf{x}$ in space.

For large data sets, it is unreasonable to consider all data for the construction of Hardy's multiquadric interpolant. Franke and Nielson [1] have shown that for each point $\mathbf{x}_i$ of a scattered data set, an influence radius can be defined so that its contribution to the Hardy interpolant is zero outside this radius. A more common method is to utilize a fixed, small number of data points for the calculation of localized Hardy interpolants.

Concerning the identification of the data points closest to a particular point for which we have to compute a local Hardy interpolant, we utilize a kd-tree, see [14].

4 Clustering and Error Measurements

Traditionally, algorithms concerned with the approximation of scientific data consider the numerical deviation of the given data and its approximation as the error. In the context of our application, which is the visualization and analysis of a vector field, it is more important to ensure that the resulting imagery generated from a particular approximation varies very little from the imagery obtained from the original data set. We chose to use the deviation of streamlines as the criterion to measure the error of a discrete approximation of a given discrete field.

Our method separates a discrete vector field data set into disjoint clusters. A simplified vector field is defined by using a single point and associated vector for each cluster and using Hardy's

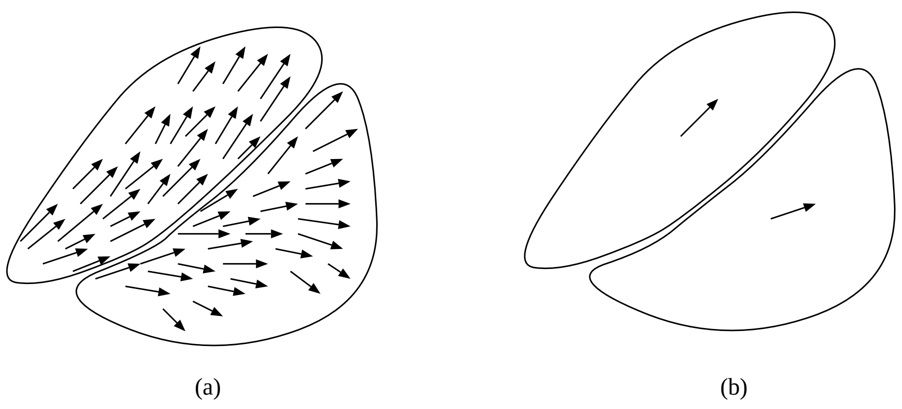


Figure 1: Cluster-based simplification of a vector field: (a) The original vector field segmented into two clusters. (b) By replacing each cluster with a single point and associated vector, a simplified field is obtained.

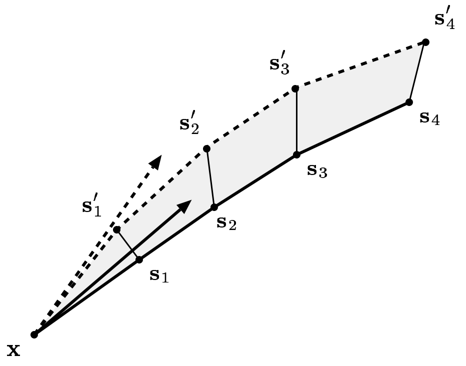


Figure 2: Calculation of the error measure at a point $\mathbf{x}$. The error is the sum of the distances between the sequence of point pairs $\mathbf{s}_i$ and $\mathbf{s}_i'$ obtained by a Runge-Kutta method. The streamline based on the original field is drawn as the solid line, and the dashed line corresponds to the simplified field.

multiquadric to approximate the field locally. Figure 1 illustrates a vector field subdivided into two clusters and the simplified vector field. The new data point associated with a cluster $\mathcal{C}$ is defined by the average of the coordinates of the points in $\mathcal{C}$. The single vector associated with $\mathcal{C}$ is calculated as the average of the vector values associated with the points of the original field contained in $\mathcal{C}$.

Suppose we have segmented a vector field $\mathcal{V}$ into a set of disjoint clusters $\mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_n$ and have determined the cluster center points $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n$ with associated vector values $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n$; the set of n points and n vectors defines a new, reduced vector field $\mathcal{V}'$. To measure the difference between $\mathcal{V}$ and $\mathcal{V}'$ at an arbitrary point $\mathbf{x}$, we compute a difference measure for the streamlines emantating from $\mathbf{x}$, one based on the field $\mathcal{V}$ and one based on $\mathcal{V}'$, see Figure 2.

A streamline is an integral curve. It is calculated by solving the equation

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{v}(\mathbf{x}(t)), \tag{1}$$

where t is the parameter along the streamline. Given an initial point $\mathbf{x}$ in a field, a streamline can be calculated by solving this initial-value problem. We apply a fourth-order Runge-Kutta scheme, see [15], to integrate the equation stepwise.

For a given point $\mathbf{x}$ in the field, we calculate two streamlines, one based on the simplified field and one based on the original one. We use the deviation of the two streamlines as an error measure for an approximation. The streamlines are calculated using a fixed number of Runge-Kutta steps (with a fixed step size), using the same point $\mathbf{x}$ as the initial position. We compare the difference between the two streamlines by calculating the sum of the distances between the corresponding Runge-Kutta points on the respective streamlines, see Figure 2.

Thus, we define the error at a point $\mathbf{x}$ to be

$$\epsilon(\mathbf{x}) = \sum_{i=1}^{n} ||\mathbf{s}_i - \mathbf{s}_i'||, \tag{2}$$

where $\mathbf{s}_i$ is the ith Runge-Kutta point in the generation of the streamline for $\mathcal{V}$, and $\mathbf{s}_i'$ is the ith point in the generation of the streamline for $\mathcal{V}'$.

To calculate an error measure for an entire cluster of points, we use the maximum calculated error value for those points belonging to the cluster, *i.e.*,

$$\epsilon(\mathcal{C}) = \max_{\mathbf{x} \in \mathcal{C}} \epsilon(\mathbf{x}). \tag{3}$$

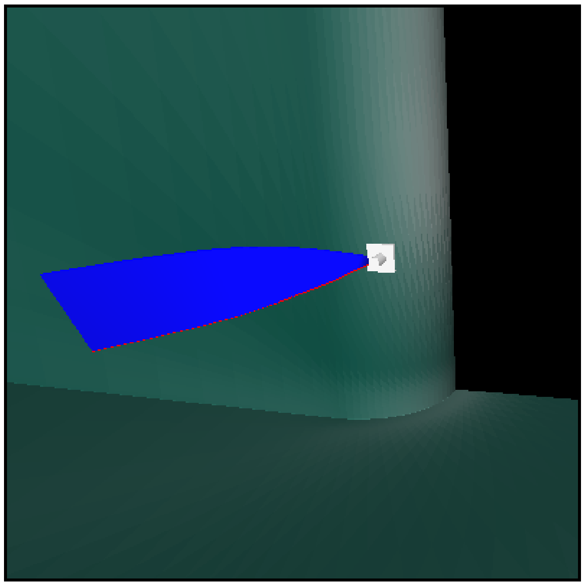
In Figure 3, we illustrate the difference between streamlines for the blunt-fin data set.

5 Splitting Clusters

In order to generate the set of clusters whose cluster center points and associated vectors define a simplified vector field, we recursively split clusters until the maximum value of all cluster errors is less than a prescribed tolerance. Each cluster is split using a "bisection" plane, which is placed and oriented to reduce the error in the child clusters. Figure 4 provides an illustration of a cluster and a desirable splitting plane.

To generate the split plane for a cluster $\mathcal{C}$, we consider the points $\mathbf{x}_i$ of the original discrete vector field that are contained in $\mathcal{C}$ and the associated error values $\epsilon(\mathbf{x}_i)$ for these points. To generate position and orientation of the split plane, we focus on those points of the cluster where the error is low, and use a weighted least squares best-fit plane using these points.

First, we determine weights for each point, based on the approximation error. Those points $\mathbf{x}_i$ of the original vector field where the error is low should be weighted high for our split procedure, while


(a)

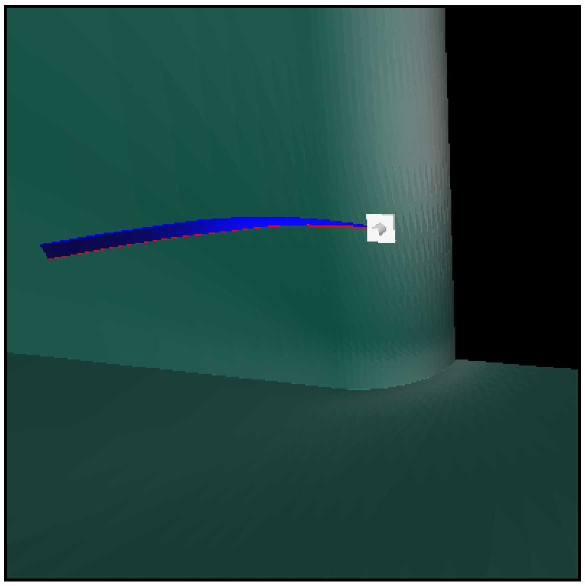

(b)

Figure 3: Error between two streamlines in the blunt-fin data set. (a) The simplified vector field based on 200 clusters, with relatively large error. (b) The simplified vector field based on 4000 clusters, with a substantially reduced error.

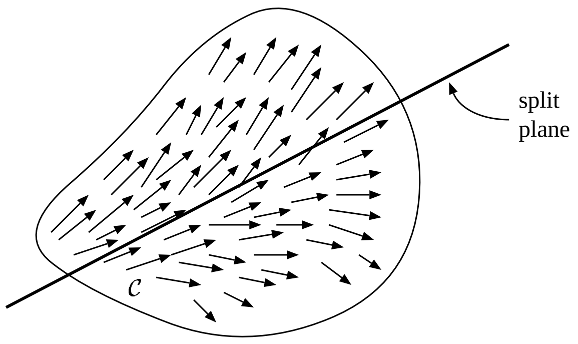


Figure 4: The split plane subdivides the cluster such that the errors in the child clusters are minimized.

points with high error should be weighted low. Thus, the weight $w(\mathbf{x}_i)$ of a point $\mathbf{x}_i$ is proportional to the reciprocal of the calculated error $\epsilon(\mathbf{x}_i)$ at the point. In particular, if

$$W = \sum_{\mathbf{x}_j \in \mathcal{C}} \epsilon(\mathbf{x}_j)^{-1},\ \epsilon(\mathbf{x}_j) \neq 0, \tag{4}$$

then we define the weight $w(\mathbf{x}_i)$ of a point $\mathbf{x}_i$ to be

$$w(\mathbf{x}_i) = \frac{1}{W\epsilon(\mathbf{x}_i)}. \tag{5}$$

We define the orientation of the split plane P as the best-fit plane, in the least squares sense, to the set of weighted points $\{w_i(\mathbf{x}_i)\mathbf{x}_i\ :\ \mathbf{x}_i \in \mathcal{C}\}$, passing through the point

$$\overline{\mathbf{x}} = \frac{1}{k}\sum_{\mathbf{x}_i \in \mathcal{C}} w_i(\mathbf{x}_i)\mathbf{x}_i, \tag{6}$$

where k is the cardinality of the cluster $\mathcal{C}$, see Hotelling [8] or Jackson [9].

The splitting process is done recursively, defining a binary space partitioning of the data set. At each step of the partitioning procedure, we concentrate on the cluster that has the maximum error. This results in simplified vector fields based solely on the differences between streamlines.

We illustrate the splitting process for a two-dimensional vector field defined by the function $(x, y) \rightarrow \langle x, xy \rangle$. We limit the vector field to the region $0 \leq x \leq 1$ and $-1 \leq y \leq 1$. This vector field is shown in Figure 5. Figure 6 shows the cluster centers and associated vectors after splitting.

The errors associated with clusters must be updated (locally) once a particular cluster has been split. The computation of a streamline emanating at a particular cluster center $\mathbf{x}$ depends on a certain number of nearby centers of "neighbor" clusters. Splitting one of these neighbor clusters will affect the streamline computation, and therefore the error for point $\mathbf{x}$. Thus, we locally update cluster values whenever splitting is performed.

6 The Algorithm

Given a discrete vector field as a set of points and a corresponding set of vectors, we initially place all vectors in a single cluster, thus defining a single, constant vector field $\mathcal{V}'$. We calculate the error for all points of the cluster and define the maximum point error as the cluster error. Each cluster is placed in a priority queue ordered

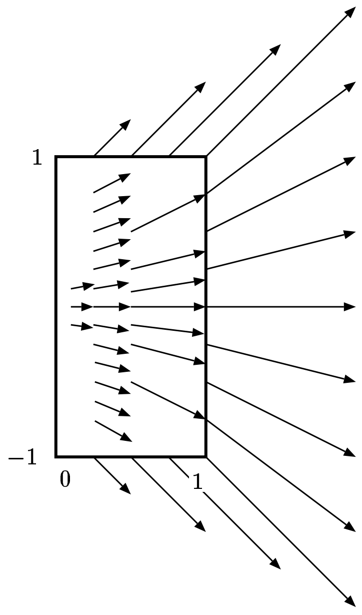


Figure 5: The vector field $(x, y) \rightarrow \langle x, xy \rangle$.

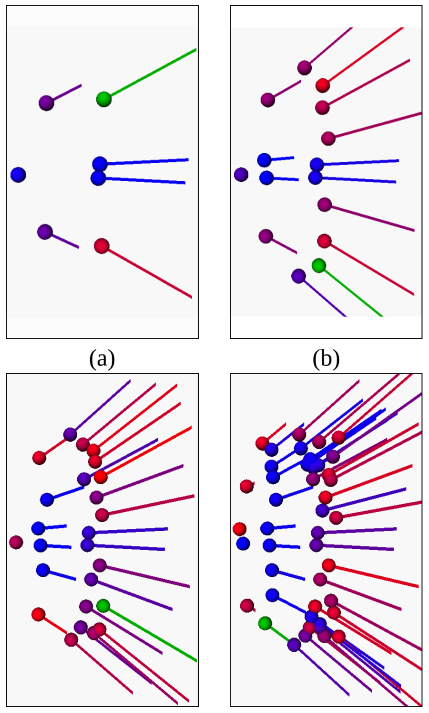


Figure 6: The splitting algorithm operating on the vector field $(x, y) \rightarrow \langle x, xy \rangle$. We have displayed the cluster centers after (a) 7 splits; (b) 15 splits; (c) 25 splits; and (d) 45 splits.

Data Set	Number of Points	Number of Clusters	Time in seconds
Spiral	4,000	500	45
Three Singularities	4,000	1,000	60
Blunt Fin	41,000	5,000	886

Table 1: Statistics for the three vector fields.

by decreasing cluster error. We repetitively use the cluster at the front of the priority queue to refine the vector field $\mathcal{V}'$.

Given a cluster $\mathcal{C}$ with maximal weight (the cluster at the front of the queue), we calculate the split plane for the cluster and split the cluster into two child clusters $\mathcal{C}_1$ and $\mathcal{C}_2$. We calculate the center points and average vectors that represent each of these clusters and obtain a modified vector field $\mathcal{V}'$ by replacing the point and vector representing $\mathcal{C}$ by the new points and vectors representing the child clusters. We recalculate the errors for $\mathcal{C}_1$ and $\mathcal{C}_2$ and insert them into the priority queue.

This process is repeated until a vector field approximation is obtained whose maximal cluster error is less than a prescribed error tolerance or until a prescribed number of clusters is generated.

7 Results

We have applied this algorithm to several complex data sets. Preprocessing of the data sets was done on an SGI Onyx2 computer system, using one 195MHz R10000 processor. The statistics for each data set are shown in Table 1.

The first data set is given by a discrete field of 4,000 samples taken from the vector field defined by the equation

$$(x, y) \rightarrow \langle -0.03x + 0.1y, -0.1x - 0.03y, 0 \rangle \,. \tag{7}$$

We have illustrated the simplified vector field at two resolutions in Figure 7. The streamlines for the original field and the simplified fields are nearly identical in each case. Here, the simplified fields represent the spiral well, even though few clusters are generated near the center.

The second data set is given by a discrete field of 4,000 samples taken from the vector field defined by the equation

$$(x, y) \rightarrow \left\langle -x^2 + xy + \frac{1}{2}y + \frac{1}{4}, x^2 + xy - \frac{1}{2}y - \frac{1}{4} \right\rangle . \tag{8}$$

This vector field contains three critical points. We have illustrated the simplified vector field at two resolutions in Figure 8. Here, the light-colored streamline represents the data of the original field, and the dark-colored streamline represents the data of the simplified field.

The third data set is the blunt-fin data set. This is a curvilinear data set containing 41,000 points. Figures 9a–d show streamsurfaces generated for this vector field and for simplified versions of it. In each figure, the streamsurface of the original vector field is shown in blue, while the streamsurface for the simplified field is shown in red. The simplified vector fields are based on (a) 200 clusters; (b) 500 clusters; (c) 1500 clusters; and (d) 2000 clusters.

We have found that the illustration of these simplified vector fields, where a short streamline is displayed at each cluster center, is a very useful tool for visualizing the entire field. Since the simplification routine generates clusters where the error is high, we obtain many streamlines in areas of high error, and few in regions of low error. Figures 9e and 9f illustrate this technique with two simplified fields at different resolutions for the blunt-fin data set.

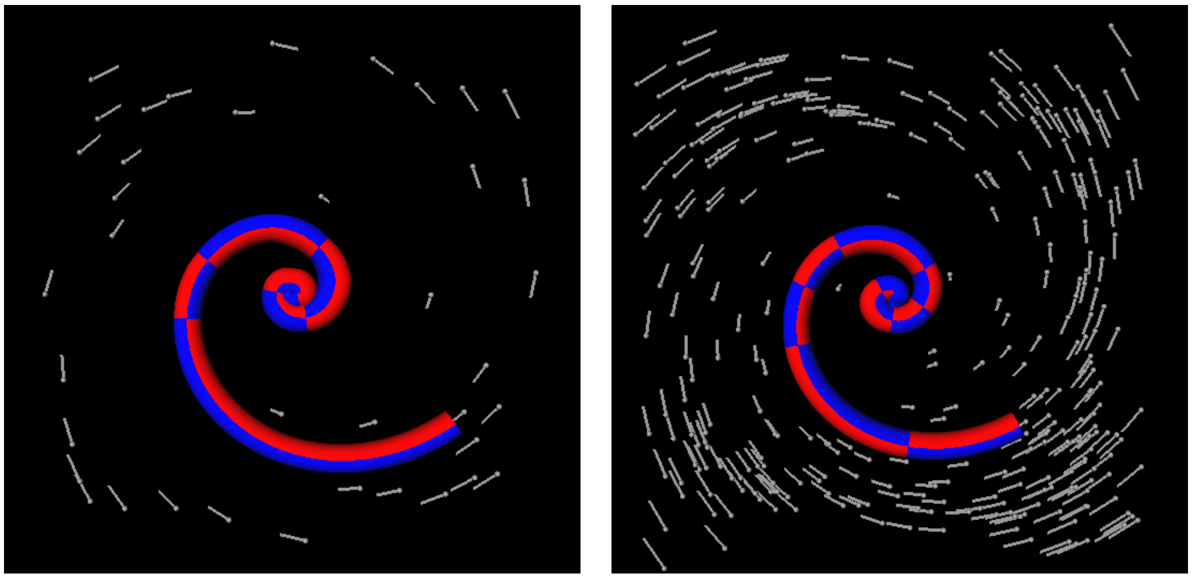


Figure 7: A two-dimensional spiral vector field. The streamlines (shown here as tubes) for the simplified fields and the original one are nearly identical. The simplified vector fields are based on (a) 40 clusters, and (b) 300 clusters.

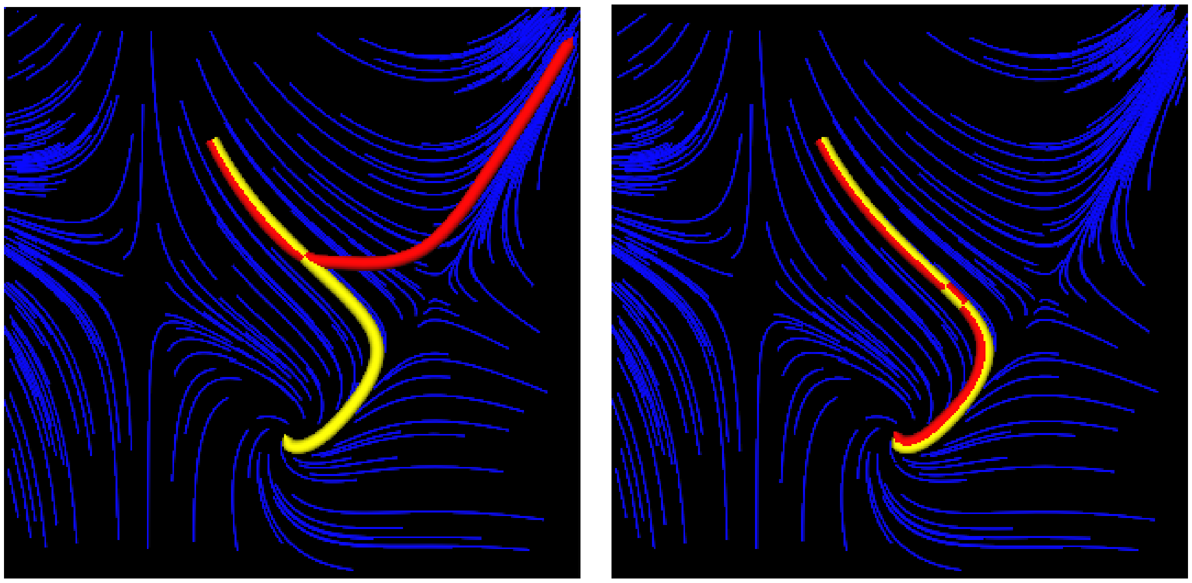


Figure 8: Segmentation of a two-dimensional vector field having three critical points. (a) Using 100 clusters, the simplified field cannot reproduce the streamline generated from the original field. The streamline generated from the original field is shown in light gray, and the streamline generated from the simplified field is shown in dark gray. (b) Using 500 clusters, the simplified field can reproduce the streamline generated from the original field.

The streamlines are generated at the centers of the clusters. The simplified vector fields are based on 170 clusters in Figure 9e and 300 clusters in Figure 9f.

8 Conclusions

We have presented an algorithm suitable for the generation of a hierarchical representation for discrete vector fields. This method uses a clustering approach, segmenting the vector field into a hierarchy of disjoint clusters. This allows us to represent the original vector field a varying resolutions. We utilize an error measure that measures the error between the streamlines generated from the points of the original discrete vector field and points of the approximate field, and we use this measure to generate split planes for iterative segmentation.

Our approach is innovative in two regards: (1) Our method is entirely gridless, leading to significant savings in memory and storage overhead, and (2) our error computation is based on the difference in streamlines between the original vector field and an approximation. The gridless paradigm is extremely space-saving, and an error metric based on the visual differences between approximations is definitely advantageous for visualization applications.

We have experimented with a variety of approaches for clustering. In the context of vector fields, the data is characterized by (1) positional information and (2) vector information for each individual datum. A multitude of different options exists concerning possible clustering criteria. Several approaches, including clustering merely based on vector information will, in general, lead to cluster whose members will lie in disjoint regions of the original field that are far apart from each other. The binary space partitioning approach that we have chosen creates spatially convex clusters and avoids most of the problems encountered with other clustering strategies.

We plan to develop other strategies for clustering vector field data. Our current algorithm generates a BSP tree, but there are many other hierarchical data structures that one could use. We plan to evaluate the changes in the vector field topology under simplification as well.

9 Acknowledgments

This work was supported by the National Science Foundation under contract ACI 9624034 (CAREER Award), the Office of Naval Research under contract N00014-97-1-0222, the Army Research Office under contract ARO 36598-MA-RIP, the NASA Ames Research Center through an NRA award under contract NAG2-1216, the Lawrence Livermore National Laboratory through an ASCI ASAP Level-2 under contract W-7405-ENG-48 (and B335358, B347878), and the North Atlantic Treaty Organization (NATO) under contract CRG.971628, and Silicon Graphics, Inc. The first author was also supported by the IBM Alamaden Research Center.

The blunt-fin data set is courtesy of NASA Ames Research Center.

We would like to thank the members of the Visualization Group at the Center for Image Processing and Integrated Computing at the University of California, Davis, for their support.

References

[1] Richard Franke and Gregory M. Nielson. Smooth interpolation of large sets of scattered data. *International Journal for Numerical Methods in Engineering*, 15(11):1691–1704, 1980.

[2] Richard Franke and Gregory M. Nielson. Scattered data interpolation and applications: A tutorial and survey. In H. Hagen, H. Mueller, and G.M. Nielson, editors, *Focus on Scientific Visualization*, pages 131–159. Springer-Verlag, New York, 1995.

[3] R. L. Hardy. Multiquadric equations of topography and other irregular surfaces. *Journal of Geophysical Research*, 76:1906–1915, 1971.

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

[5] Bjoern Heckel, Antonio E. Uva, and Bernd Hamann. Clustering-based generation of hierarchical surface models. In C.M. Wittenbrink and A. Varshney, editors, *Proceedings of Visualization 1998 (Hot Topics)*, pages 50–55. IEEE Computer Society Press, Los Alamitos, CA, October 1998.

[6] Bjoern Heckel, Antonio E. Uva, Bernd Hamann, and Kenneth I. Joy. Surface reconstruction using adaptive clustering methods. Technical Report CSE-99-1, Computer Science Department, University of California, Davis, January 1999.

[7] James Helman and Lambertus Hesselink. Representation and display of vector field topology in fluid flow data sets. *Visualization in scientific computing*, pages 61–73, 1990.

[8] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:pp. 417–441 and 498–520, 1933.

[9] J. E. Jackson. *A User's Guide to Principal Components*. Wiley, New York, 1991.

[10] B.F. Manly. *Multivariate Statistical Methods, A Primer*. Chapman & Hall, New York, New York, 1994.

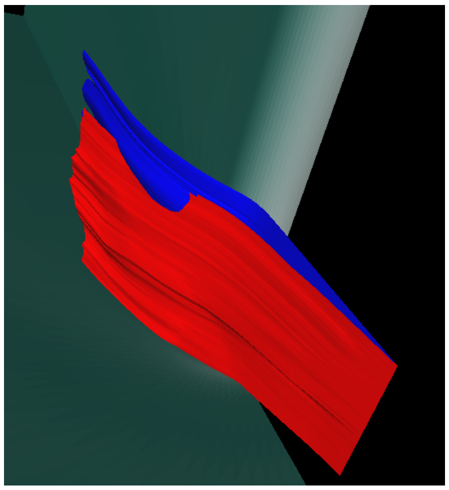
[11] Gregory M. Nielson, Thomas A. Foley, Bernd Hamann, and David A. Lane. Visualizing and modeling scattered multivariate data. *IEEE Computer Graphics and Applications*, 11(3):47–55, May 1991.

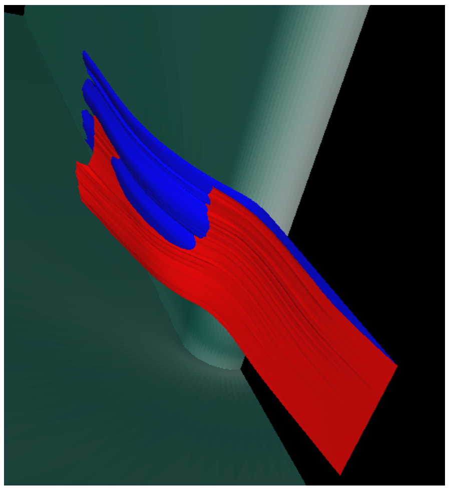
[12] Gregory M. Nielson, Il-Hong Jung, and Junwon Sung. Haar wavelets over triangular domains with applications to multiresolution models for flow over a sphere. In Roni Yagel and Hans Hagen, editors, *IEEE Visualization '97*, pages 143–149, Los Alamitos, October 1997. IEEE Computer Society Press.

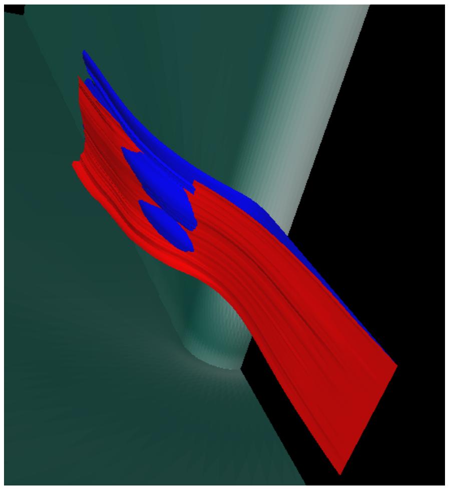
[13] Gregory M. Nielson, Il-Hong Jung, and Junwon Sung. Wavelets over curvilinear grids. In David S. Ebert, Hans Hagen, and Holly Rushmeier, editors, *Proceedings IEEE Visualization'98*, pages 313–317, Los Alamitos, 1998. IEEE Computer Society Press, Los Alamitos, CA.

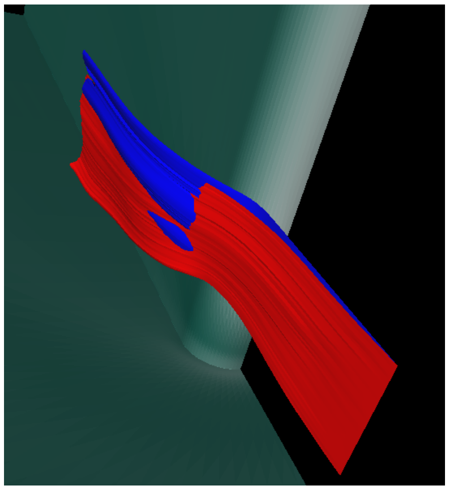
[14] Hanan Samet. *The design and analysis of spatial data structures*. Addison-Wesley, Reading, Mass., 1990.

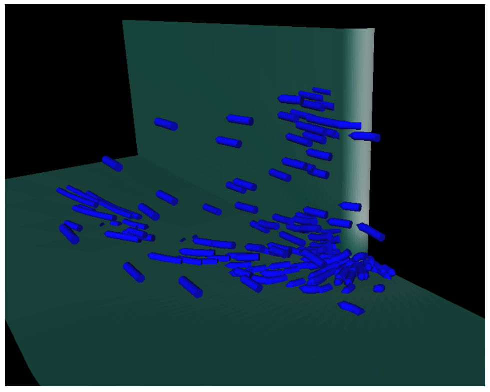
[15] Shyh-Kuang Ueng, Christopher Sikorski, and Kwan-Liu Ma. Efficient streamline, streamribbon, and streamtube constructions on unstructured grids. *IEEE Transactions on Visualization and Computer Graphics*, 2(2), June 1996. ISSN 1077-2626.


(a)


(b)


(c)


(d)


(e)

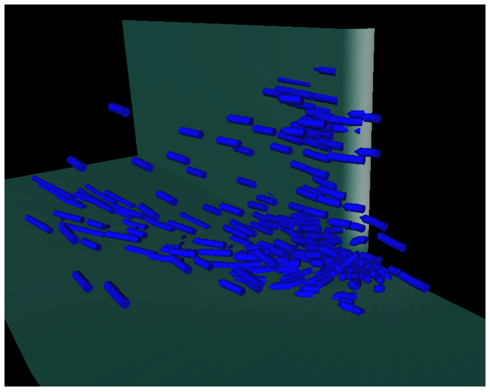

(f)

Figure 9: Construction of Vector Field Hierarchies.