A Segmentation Approach To Scientific Visualization

Janine Bennett, CIPIC, University of California, Davis
Karim Mahrous, CIPIC, University of California, Davis
Bernd Hamann, CIPIC, University of California, Davis
Kenneth I. Joy, CIPIC, University of California, Davis

Abstract

We present a segmentation approach to scientific visualization that combines the definition of higher-level data, the efficient extraction of meaningful derived feature-like data from defined properties, and the effective visual representation of the extracted data. Our framework is aimed at multi-valued time-varying data sets, where, for example, grid vertices might have a multitude of associated scalar, vector and tensor quantities. This "segmentation" approach to massive data set exploration allows the user to focus upon regions, and interactively explore these regions efficiently. The challenge is to generate this segmented data from existing multi-valued data sets, store this data in an efficient scheme, generate the boundaries of each region, and display these boundaries to the user. We present an integrated scheme that allows a common representation for segmentation, allows it to be applied to a number of data types, and allows derived representations to be calculated. We illustrate this framework with examples from scalar- and vector-field visualization.

Keywords: visualization, segmentation, derived data, vector fields

1 Introduction

With the increase in computing power and our ability to gather more and more data via increasingly powerful imaging and sensor technology, the size of scientific data sets continues to grow. Data sets that represent physical phenomena now contain billions of elements representing multi-valued, multi-dimensional, time-varying data, and we are no longer able to fully analyze them. One can consider various paradigms when attempting to provide better tools for the exploration of these data sets, but the need is to drive these tools by the realization that the scientist rarely needs to examine an entire data set. Typically, the interest is in particular regions where certain properties hold. Tools must be developed that allow the scientist to identify and specify these properties, and segment the data set to determine the regions where these properties hold.

We firmly believe in the value of interactive data visualization as a tool for massive data set exploration, and we realize that it is no longer feasible to interactively explore a massive data set in its entirety. The approach we propose combines the definition of higher-level data, the efficient extraction of meaningful derived data from defined properties. Our framework is aimed at multi-valued time-varying data sets, where, for example, grid vertices might have a multitude of associated scalar, vector and tensor quantities. Our goal is to devise new algorithms that support the numerically robust extraction of regions (or boundaries of these regions) that represent similar qualitative or topological behavior. We propose this "segmentation" approach to massive data set exploration as we believe that it is possible to determine and interactively explore these focussed regions efficiently, with respect to both storage and computation requirements.

We are interested in multi-valued time-varying data sets that are based upon three-dimensional grids. We believe that these data sets present us with many fundamental unsolved problems in data exploration. Many researchers have developed algorithms for two-dimensional data (especially vector and tensor fields), but these methods do not scale to the additional dimension(s). Two-dimensional vector fields, for example, are easily characterized by their critical points. Classification of these points, and segmentation by separatrices, completely characterize these flow fields. However, three-dimensional vector fields can be arbrarily complex and often do not contain any critical point (*e.g.,* the NASA delta wing [9]). Exploration techniques for these fields are still in their infancy.

This research is motivated by the segmentation algorithms developed by Bonnell *et al.*[3], where a data set is segmented by utilizing "material fraction" information stored in each grid cell. In multi-fluid hydrodynamics simulations, the numerical algorithms produce percentages for each cell defining the relative amount of a fluid present in the cell. Of major interest is the extraction and visualization of interfaces between the various fluids, and how interfaces change over time. Bonnell *et al.*use the fraction data to construct the fluid interfaces. This approach is inherently a segmentation approach, where we segment the domain of the simulated data set into regions of same material type.

The challenge is to generate this segmented data for more general types of multi-valued data sets, store this data efficiently, generate the boundaries of each region, and display these boundaries. The segmentation approach will give us a more generally applicable scheme that supports a common representation for segmentation, that can be applied to a number of data types, and that enables us to calculate derived representations. These representations can be integrated with multiresolution techniques to reduce the sizes of the data representations, to support fast region/segment generation and visualization.

This framework emphasizes the exploration of massive scientific data through the computation of derived "higher-level data," data that will support interactive visual exploration considering regions of similar behavior. Due to increasing data set sizes we are convinced that such a framework is needed in order to create high-level views of data and to "steer" interactive visualization. Prior to providing a scientist with visual representations of original field variables we can generate views that show the various regions in domain space where a certain type of behavior are observed.

In Section 2 we discuss the motivation behind our framework, how it relates to previous work in the field, and how to adapt current algorithms to segment scalar fields. Section 3 and Section 4 discusses the segmentation of three-dimensional vector fields. Section 5 discusses the implementation issues in vector field segmentation and Section 6 shows the results of segmenting a complex three-dimensional vector field.

2 Segmenting Scalar Fields

Suppose we are given a data set and an associated set of properties $c_1, c_2, ..., c_m$. For each vertex $\mathbf{v}$ in the data set, we associate an m-tuple $(\alpha_1, \alpha_2, ..., \alpha_m)$, where α_i is the fraction of c_i present (or "valid") at $\mathbf{v}$. We assume that $0 \leq \alpha_i \leq 1$, for $i = 1, ..., m$, and $\sum_{i=1}^{m} \alpha_i = 1$. We will call data sets of this kind *segmented*. Thus, we consider a segmented data set to be one where each vertex of the data set has an associated barycentric coordinate. The challenge is to generate this segmented data from given multi-valued data sets, store this segmented data in an efficient scheme, generate the boundaries of each segment, and display these boundaries. Many problems in data exploration can be formulated using segmented data. These are a few examples:

- In a scalar field, an isosurface is determined by a single scalar value s. Here, each vertex $\mathbf{v}$ can be assigned a 2-tuple (α_1, α_2), where $\alpha_1 = 0$ if the scalar value associated with $\mathbf{v}$ is less than s, and where $\alpha_1 = 1$ if the scalar value is greater or equal to s, and $\alpha_2 = 1 - \alpha_1$. The isosurface is then the boundary of the region where $\alpha_1 = \frac{1}{2}$.

- In a scalar field, the region between two isosurfaces can be isolated. Given two scalar values $s_1 < s_2$, each vertex $\mathbf{v}$ is assigned a 3-tuple $(\alpha_1, \alpha_2, \alpha_3)$, where $\alpha_1 = 1$ if the scalar value s associated with $\mathbf{v}$ is less than s_1, where $\alpha_2 = 1$ if $s_1 \leq s \leq s_2$, and $\alpha_3 = 1$ if $s > s_2$. Points on the two boundaries of the region are characterized by $\alpha_1 = \alpha_2 = \frac{1}{2}$ and $\alpha_2 = \alpha_3 = \frac{1}{2}$.

- Considering vector fields, a streamsurface can be determined in a similar way. Here, each vertex $\mathbf{v}$ can be assigned a 2-tuple (α_1, α_2), where $\alpha_i = 0$ or 1 depending on whether a streamline emantating from $\mathbf{v}$ is on the "left" or "right" side of the streamsurface.

- Using slicing techniques, researchers have developed data sets that consist of a number of high-resolution slices, where each vertex $\mathbf{v}$ is assigned a set of probabilities that determine whether certain properties hold at the vertex. In this case, each data point is assigned an m-tuple $(\alpha_1, \alpha_2, ..., \alpha_m)$, where α_i represents the probability that property i is satisfied at the vertex. We can use segmentation techniques to determine the boundary of regions that have certain probabilities, and therefore satisfy certain property characteristics.

Those problems where the α values are real numbers are the most interesting and have the greatest application in data exploration. Generated surfaces will be smoother, and they will more accurately represent the segment boundary. We will extensively investigate the cases where the α values are real.

Müller [11] and Nielson and Franke [5] have defined methods to find the separating surface in an unstructured tetrahedral data set when each vertex is associated with a particular "type" (*i.e.*, exactly one of the $\alpha_i = 1$ for each vertex). Their methods follow the principle of the marching-cubes algorithm of Lorensen and Cline [10], generating a separating surface for each tetrahedron in the data set.

Bonnell *et al.*[3] have solved a more general problem for a specific case: In multi-fluid Eulerian hydrodynamics calculations, geometric approximations of fluid interfaces are used to form the equations of motion to advance these interfaces correctly over time. In this application, grid cells of the data set contain fractional volumetric information for each of the fluids. Thus, each cell C of a grid $\mathcal{S}$ has an associated m-tuple $(\alpha_1, \alpha_2, ..., \alpha_m)$ that represents the portions of each of m fluids in the cell. By considering a "dual grid," Bonnell *et al.*associate the m-tuples with vertices of the dual grid and develop a method that finds a (crack-free) piecewise two-manifold separating surface approximating the boundary surfaces between the various fluids. Figure 1 illustrates the basic mapping used in the two-dimensional case when three materials are considered ($m = 3$). Given a triangle, where each vertex has an associated barycentric coordinate, this triangle is mapped into a 2-simplex in 3D barycentric space. This barycentric-space simplex has the values $(1, 0, 0)$, $(0, 1, 0)$, and $(0, 0, 1)$ at the vertices. Given a Voronoi decomposition of the 2-simplex, using the points $(1, 0, 0)$, $(0, 1, 0)$, and $(0, 0, 1)$ as the centers of the Voronoi cells, we can map the barycentric coordinates of the vertices of

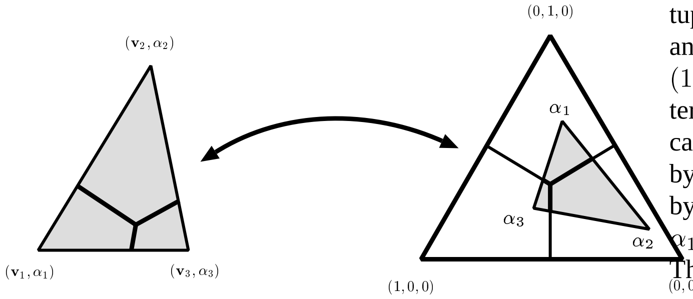
$(\mathbf{v}_2, \alpha_2)$
$(\mathbf{v}_1, \alpha_1)$
$(\mathbf{v}_3, \alpha_3)$
$(0,1,0)$
α_1
α_3
α_2
$(1,0,0)$
$(0,0,1)$

Figure 1: Given a triangle (shaded) in a two-dimensional grid, where each vertex $\mathbf{v}_i$ of the triangle contains a barycentric coordinate α_i, these barycentric coordinates are mapped a triangle contained in a 2-simplex in barycentric space. The intersections (dark lines) are calculated in barycentric space, and mapped back to the original triangle.

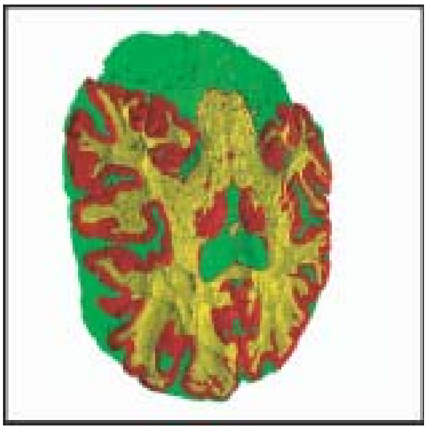

Figure 2: A brain segmentation obtained with Bonnell's algorithm.

the triangle into this simplex. Calculating the intersection of this mapped triangle with the Voronoi cells yields the corresponding segmentation of the triangle.

Bonnell's algorithm generalizes the work of a number of researchers [6, 14, 15, 24] who developed two-dimensional non-C^0 methods for calculating fluid interfaces. This method calculates a C^0, potentially non-manifold surface for a finite number of materials over a three-dimensional tetrahedral grid. Given m fluids, the method can potentially calculate $m-1$ separating surfaces for each cell. Figure 2 illustrates a brain data set. Here, each cell of the data set contains a 3-tuple, where the respective components of the tuple indicate the presence of gray matter, white matter, and other matter.

2.1 The General Case

In the case of four materials, it is sufficient to assume that each vertex of T has an associated barycentric coordinate tuple $\alpha = (\alpha_1, \alpha_2, \alpha_3, \alpha_4)$, where $\alpha_1+\alpha_2+\alpha_3+\alpha_4 = 1$, and $\alpha_i \geq 0$. By considering the 3-simplex having vertices $(1,0,0,0)$, $(0,1,0,0)$, $(0,0,1,0)$, and $(0,0,0,1)$ in material space, a partition of this simplex into Voronoi cells can be defined. The boundaries of these cells are bounded by the faces of the 3-simplex and six hyperplanes, defined by the set of α such that (i) $\alpha_1 = \alpha_2$, (ii) $\alpha_1 = \alpha_3$, (iii) $\alpha_1 = \alpha_4$, (iv) $\alpha_2 = \alpha_3$, (v) $\alpha_2 = \alpha_4$, and (vi) $\alpha_3 = \alpha_4$. The resulting Voronoi partition is shown in Figure 3a.

If T is a triangle in a two-dimensional unstructured grid, the barycentric coordinate tuples associated with the vertices of T are mapped into a triangle T_α in the material space 3-simplex. A clipping algorithm is applied to T_α to generate intersections with the boundaries of the Voronoi cells, by clipping against each of the six hyperplanes defining the Voronoi-cell boundaries.

Intersections can be found by a simple procedure. Suppose that an edge of T_α with endpoints $\alpha^{(1)}$ and $\alpha^{(2)}$ crosses the hyperplane defined by $\alpha_1 = \alpha_2$. If α is the intersection point, we can compute r such that

$$\alpha = (1-r)\alpha^{(1)} + r\alpha^{(2)}.$$

If the first two coordinates of α are equal, then

$$(1-r)\alpha_1^{(1)} + r\alpha_1^{(2)} = (1-r)\alpha_2^{(1)} + r\alpha_2^{(2)},$$

which allows us to calculate r directly. (See Hanson [7] for similar methods.) Once the intersections are determined by the clipping algorithm, the polygons in T_α are used to determine polygons in the Euclidean coordinates of T, which represent the material boundary.

In the k-material case, a tetrahedron (triangle) T has an associated k-simplex T_α in material space. The k-simplex is partitioned into Voronoi cells whose boundaries consist of the faces of the k-simplex and the $\binom{k}{2}$ hyperplanes defined by the equations $\alpha_i = \alpha_j$, where $1 \leq i < j \leq k$. The intersections of T_α with the boundaries of the Voronoi cells are calculated performing clipping. The polygons of T_α, determined by the clipping algorithm, are then used to determine polygons in the Euclidean coordinates of T in physical space, which represent the material boundary in T.

3 Vector Field Segmentation

Segmentation of the domains of trivariate vector fields is equally important and even more challenging than scalar field segmentation. Classical approaches used for bivariate vector field segmentation are based on constructing the separatrix structure of a field – a set of curves in the plane – defining regions that behave qualitatively similarly [8, 12, 17]. For example, all points in the domain of a bivariate vector field that, when used as seeds for the computation of streamlines, end at the same attracting focus define a region of same topological behavior. Separatrices are classically generated by computing the critical points

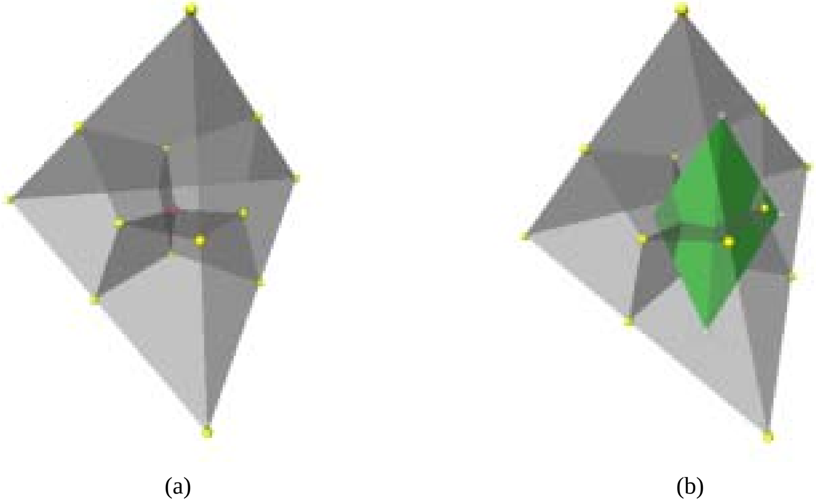

Figure 3: Voronoi cell decomposition for the four-material case. The figure illustrates a three-dimensional projection of the 3-simplex. The 3-simplex is segmented into four Voronoi cells in (a). A 3-simplex T_α, mapped from a tetrahedron T, is shown inside the 3-simplex in (b).

in the domain of the vector field, determining the types of these critical points, and using numerical methods to trace streamlines originating from so-called "originators" [8]. Due to the need to use numerical methods [2, 13] for streamline/separatrix approximation, results are usually subject to error, or are even wrong.

However, more interesting problems arise with three-dimensional vector fields. Two-dimensional vector fields, for example, are easily characterized by their critical points. Classification of these points, and segmentation by separatrices completely characterizes the flow of these fields. However, three-dimensional vector fields can be arbitrarily complex and might not contain any critical points.

Current three-dimensional vector field visualization techniques are based mainly upon streamline and streamsurface generation[8, 12]. These techniques use the definition of the vector field to trace massless particles through the field, tracking their progress by linking them in lines (streamlines) or surfaces (streamsurfaces). Level sets have been used [21] to enhance streamline generation. However, the most important segmentation method for vector fields is the generation of separatrices. Separatrices are streamsurfaces that separate the flow. Scheuermann *et al.*and others [18, 22, 19, 20, 17] have done research for separatrices in two dimensional fields, but little has been done for three-dimensional fields [16].

Three-dimensional vector fields are tremendously complex. They frequently have no critical points, and the characteristic features of these fields (vortices, sheer walls, etc.) are difficult to detect[9]. Extracting meaningful structural information for three-dimensional vector fields is the important problem, and we are a long way from solving it.

Classical approaches for bivariate vector field segmentation are based on constructing the separatrix structure of a field – a set of curves in the plane – defining regions that behave qualitatively similarly [8, 12, 17]. Separatrices are usually generated by computing the critical points in the domain of the vector field, determining the types of these critical points, and using numerical methods to trace streamlines originating from so-called "originators" in the field, see [8]. These streamlines form separatrices that segment the field into regions of similar topological behavior.

Unfortunately, similar techniques have not been developed for three-dimensional vector fields. Most visualization techniques for three-dimensional flow fields concentrate on streamline and streamsurface representations, however these techniques do not give a clear illustration of the field's topological behavior. More sophisticated techniques use streamline analysis to extract features of the field , such as attachment and separation lines on a boundary surface. However, separatrix methods have been unavailable for the analysis of three-dimensional fields. Two basic problems have prevented these studies: First, there are few critical points in three-dimensional fields, and second, the numerical marching methods to trace characteristic stream surfaces are difficult to implement and can create substantial numerical errors. In general, visualization and classification of three-dimensional fields continues to be an open problem.

The method presented in this paper is similar to those that generate separatrix structures in two-dimensional vector-fields: focusing on segmentation of the data set based on topological structure. The algorithm is a two-step process that partitions a vector field into regions of topologically similar flow. First, we sample the vector field using streamlines, and replace the original data by a "segmented" data set. Second, a segmentation algorithm generates separating surfaces in the field. We utilize a "local separatrix" concept introduced by Scheuermann *et al.*[18], which augments the separatrices generated from critical points with "local separatrices" originating from the boundary region of the data set. By segmenting the boundary region into "inflow" and "outflow" regions, a local separatrix is the streamline generated from a point on the boundary where the flow is tangential. In this way, the algorithm generates a complete separation of the field into regions of similar flow. For visualization of massive data sets, this algorithm can be used to determine similar flow regions within a small region, avoiding the problem of analyzing the complete data set.

We utilize this concept to define a segmentation of a three-dimensional vector field into regions bounded by local separatrices. By manipulation of a "boundary box" in the field, we can create separating surfaces that define regions of similar flow depending on inflow/outflow regions on the boundary of the box. This allows us to place a box in the field and generate separatrices throughout the field, which can be visualized to determine characteristic features of the field.

Given a three-dimensional vector field and a rectangular boundary box B, we define a local separatrix as a stream surface within B that is tangent to the boundary of B. The algorithm generates these local separatrices by creating a derived "segmented" data set by sampling the original vector field with streamlines. These streamlines terminate either on the boundary or at a critical point. The general idea is to assign a different characteristic marker (or property) to each critical point in the field, and to each contiguous inflow or outflow region of the boundary of B – *i.e.,* regions bounded by lines where the flow is tangential on the boundary of B. Streamlines are initiated at points throughout the data set and traced until they either reach the boundary of B or come arbitrarily close to a critical point. Each of the streamlines is then marked appropriately. We apply the marker information to a vertex of the data set by considering streamlines that lie close to the vertex and assigning markers from these streamlines to the vertex. For example, given a point $\mathbf{p}$ on streamline S that has marker k, if $\mathbf{p}$ lies in tetrahedron T, then we can examine the vertices of the tetrahedron to see which is closest to $\mathbf{p}$. If the closest vertex contains an m-tuple $(m_1, m_2, ..., m_m)$, then we increment m_k. Most vertices will have only one non-zero marker incremented, as most of them will not have local separatrices passing near them. However, some will have multiple markers present.

The m-tuple of markers for each grid vertex is "normalized" to generate a m-dimensional barycentric coordinate tuple at each vertex. This resulting field where each point is associated with a barycentric coordinate tuple is called a *segmented* data set. We utilize "material interface" methods to calculate the boundaries between the regions, using a clipping procedure in barycentric space. The result is a set of local separatrices in the field, separating the flow field according to the inflow and outflow regions of the boundary.

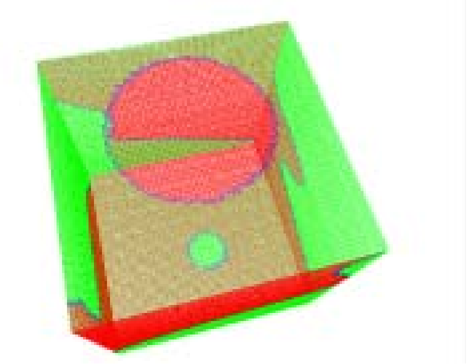

Figure 4: The inflow/outflow areas of the boundary box in a tornado data set. The terminating tetrahedra with inflow faces are colored red, and the ones with outflow faces are colored blue. The terminating tetrahedra that have regions of tangential flow are colored green. Note the circles on the top and bottom of the box boundary. These outline the center vortex of the tornado on the boundary of the box.

The separating surfaces generated by this method are local separatrices (much like those drawn by Dallmann [4]) of the field defined by the critical points and the field boundary. The user can use a slicing tool (or other technique) to browse through the separatrices, locating vortices and other features.

4 Segmenting Vector Fields

Given a vector field defined over a three-dimensional simplicial grid, we use streamlines to sample the vector field to create a segmentation of the field. The segment boundaries of this field will be approximations of the local separatrices. We first mark all critical points with a unique value (property), and identify and mark all connected inflow regions of the boundary. These marks are the property values of the field. We then sample the field using streamlines, tracing each streamline backward (see [21]) until it reaches either a critical point, or a boundary. Each streamline is associated with a unique marker considering its origin. Next, we transfer the marker information to the vertices, creating barycentric coordinate tuples at the vertices. By applying the segmentation algorithm presented previously, we generate the local separatrices of the field.

4.1 Marking Boundary Cells

Streamlines terminate at the boundary of the data set, or at a critical point. Tetrahedra lying on the boundary or tetrahedra that contain critical points are called *terminating tetrahedra*. When streamlines encounter a terminating tetrahedron, a marker is assigned to the streamline. Two types of terminating tetrahedra exist: internal and exter-

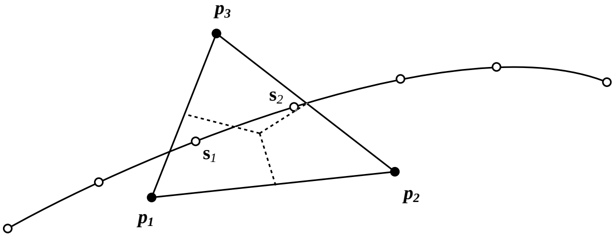

Figure 5: Streamline and Triangle. The points $\mathbf{s}_1$ and $\mathbf{s}_2$ cause the markers associated with $\mathbf{p}_1$ and $\mathbf{p}_3$, respectively, to be incremented.

nal. Internal terminating tetrahedra contain critical points [18]. External terminating tetrahedra have one or more triangular faces on the domain boundary. There exist three classifications of boundary triangles on an external terminating tetrahedron: A boundary triangle is either an *inflow* triangle, an *outflow* triangle, or it has a area of *tangential flow*. A boundary triangle is an inflow triangle if each vector $\vec{v_1}, \vec{v_2},$ and $\vec{v_3}$ associated with the vertices of the triangle satisfy

$$\vec{v_i} \cdot \vec{n} > 0$$

where $\vec{n}$ is the inward face normal of the boundary triangle. Similarly, a boundary triangle is an outflow triangle if

$$\vec{v_i} \cdot \vec{n} < 0.$$

If neither property holds, then the face contains points that have tangential flow.

Terminating tetrahedra are marked using two methods: Internal terminating tetrahedra are detected and marked individually while boundary external terminating tetrahedra are marked using an area-growing approach. An unmarked boundary terminating tetrahedron T is identified and its type is determined. A unique mark is then generated and assigned to T. The boundary neighbors of T are identified and inherit the same mark if they have the same boundary flow characteristic as T. The algorithm progresses with the neighbor boundary tetrahedra until all boundary tetrahedra have been marked. Figure 4 illustrates the marking process for terminating tetrahedra on a tornado data set.

4.2 Segmentation

We sample the flow field using streamlines. As streamlines encounter a terminating tetrahedron T, they are assigned the marker k_T associated with T. We then retrace the points that generate the streamline, and for each point $\mathbf{s}$ increment the m_{k_T} property stored at the grid vertex nearest the point $\mathbf{s}$. This is illustrated in Figure 6. Given a streamline point $\mathbf{s}$ in tetrahedron T, we calculate the barycentric coordinate β of $\mathbf{s}$ in T, and increment the m_{k_T} property in the vertex of T that corresponds to the largest component of β.

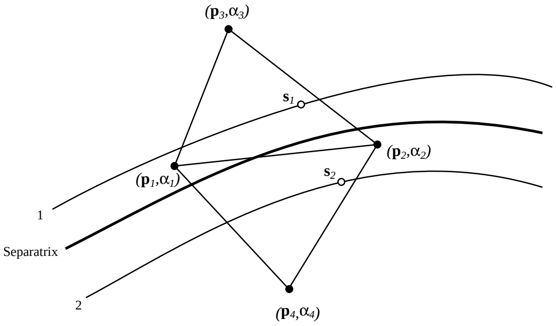

Figure 6: Separatrix and Triangle. Points on streamlines on both sides of the separatrix contribute to the barycentric coordinate tuple at a vertex. The streamline with marker 1 and the streamline with marker 2 both contribute to α_2, as the points $\mathbf{s}_1$ and $\mathbf{s}_2$ both lie closer to $\mathbf{p}_2$.

After sampling all streamlines, we normalize the property values at the vertices of the grid, creating the required barycentric coordinate tuples. The result is a segmented data set.

5 Implementation

The algorithm is straightforward to implement and is based upon streamline generation and interface construction.

Due to the fact that a large number of inflow/outflow boundary regions may occur on the boundary, the barycentric coordinates tuples may contain a large number of components. However, most of the data points will be associated with only one marker. Several of the data points may be associated with two markers near the local separatrices, and the case where three or more markers are present will be rare. Thus we never store the full barycentric coordinate, but only those components of each coordinate that are non-zero. This enables the algorithm to work with a large number of “properties.”

Many three-dimensional vector fields have *orbits*, *i.e.*, closed streamlines, that never enter a terminating cell, see [23]. In this case, a separate “property marker” must be used for each orbit. We have tested several heuristic algorithms to detect orbits and have implemented one that correctly identifies orbits when two-tetrahedron patterns are repeated along a streamline. This strategy seems to work well in practice and is illustrated in the results below.

6 Results

The segmentation of a three-dimensional vector field is illustrated by a computational simulation of a spherical argon bubble that is hit by a 1.25 Mach shock in the air. The bubble is deformed through interaction with the vorticity generated as the shock passes over the bubble. This data set was generated at the Center for Computational Sciences and Engineering at Lawrence Berkeley National Laboratory and has been used in a variety of adaptive mesh refinement methods (see Berger and Colella [1]). The data set is $128 \times 128 \times 256$ and we illustrate the field at time step 500. Here the argon bubble has deformed into a shape with a characteristic "smoke ring." The following figures show the result of the segmentation algorithm on this data set. The separatrices shown separate the flow between outflow regions on the boundary box, and also correctly identifies the ring with the vortices due to the turbulence. The ring is generated by identifying orbits in the streamlines.

Figure 7 and Figure 8 give a front view of the argon bubble flow field and the results of the segmentation algorithm.

Figure 9 and Figure 10 show close-up views of the ring identified by the segmentation algorithm.

7 Conclusions and Future Work

We have presented a new segmentation approach for scientific visualization. This approach defines a derived data set from higher-level data that allows the generation of boundaries defining regions of interest in the data set. The algorithm is simple to implement and can be used on both scalar and vector quantities. This "segmentation" approach to massive data set exploration allows the user to focus upon regions, and interactively explore these regions efficiently. The future challenge is to generate this segmented data from multi-valued, time-varying data sets containing scalar, vector and tensor quantities.

8 Acknowledgments

This work was supported by the National Science Foundation under contracts ACI 9624034 (CAREER Award), through the Large Scientific and Software Data Set Visualization (LSSDSV) program under contract ACI 9982251, and through contract ACI 0222909, and through the National Partnership for Advanced Computational Infrastructure (NPACI); the National Institutes of Health under contract P20 MH60975-06A2, funded by the National Institute of Mental Health and the National Science Foundation; and the Lawrence Livermore National Laboratory under ASCI ASAP Level-2 Memorandum Agreement B347878 and under Memorandum Agreement B528318. We thank the members of the Visualization Group at the Center for Image Processing and Integrated Computing (CIPIC) at the University of California, Davis for their support. We express our special thanks to Oliver Kreylos and David Wiley.

References

[1] M. J. Berger and P. Colella. Local adaptive mesh refinement for shock hydrodynamics. *Journal of Computational Physics*, 82:64–84, 1989.

[2] W. Boehm and H. Prautzsch. *Numerical Methods*. Springer-Verlag, 1992.

[3] Kathleen Bonnell, Dan Schikore, Mark Duchaineau, Bernd Hamann, and Kenneth I. Joy. Constructing material interfaces from data sets containing volume fraction information. In T. Ertl, B. Hamann, and A. Varshney, editors, *Proceedings of IEEE Visualization 2000*, pages 367–372. IEEE Computer Society Press, October 2000.

[4] Uve Dallmann. Topological structures of three-dimensional flow separations. Technical Report 221-82, Deutsche Forschungs- und Versuchsanstalt für Luft- and Raumfahrt, 1983.

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

[6] Denis Gueyffier, Jie Li, Ali Nadim, Ruben Scardovelli, and Stéphanie Zaleski. Volume-of-fluid interface tracking with smoothed surface stress methods for three-dimensional flows. *Journal of Computational Physics*, 152:423–456, 1999.

[7] Andrew J. Hanson. Geometry for n-dimensional graphics. In Paul Heckbert, editor, *Graphics Gems IV*, pages 149–170. Academic Press, Boston, 1994.

[8] James Helman and Lambertus Hesselink. Representation and display of vector field topology in fluid flow data sets. In G. M. Nielson and B. D. Shriver, editors, *Visualization in Scientific Computing*, pages 61–73. IEEE Computer Society, 1990.

[9] D. N. Kenwright. Automatic detection of open and closed separation and attachment lines. In Hans Hagen David Ebert and Holly Rushmeier, editors, *IEEE Visualization '98*, pages 151–158, October 1998.

[10] W. E. Lorensen and H. E. Cline. Marching cubes: a high resolution 3D surface construction algorithm. In M. C. Stone, editor, *Computer Graphics (SIGGRAPH '87 Proceedings)*, volume 21 (4), pages 163–170, July 1987.

[11] Heinrich Müller. Boundary extraction for rasterized motion planning. In Horst Bunke, Tankeo Kanade, and Hartmut Noltemeier, editors, *Modeling and Planning for Sensor Based Intelligent Robot Systems*, pages 41–50. World Scientific Publishers, 1995.

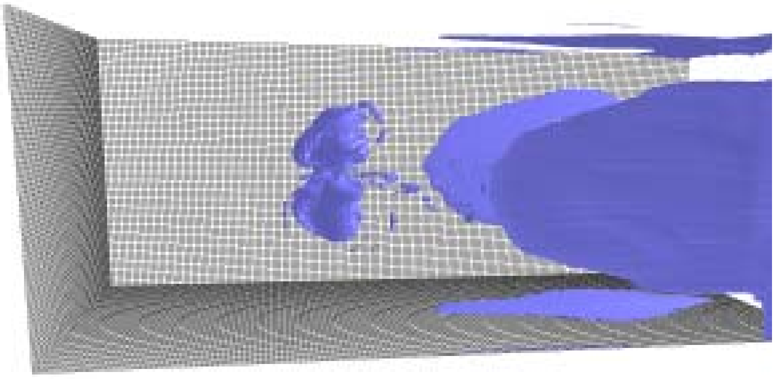

Figure 7: A side view of the result of the segmentation algorithm on the argon bubble data set. The shockwave (ring) is clearly identified along with closed surfaces trailing the shockwave due to the turbulence.

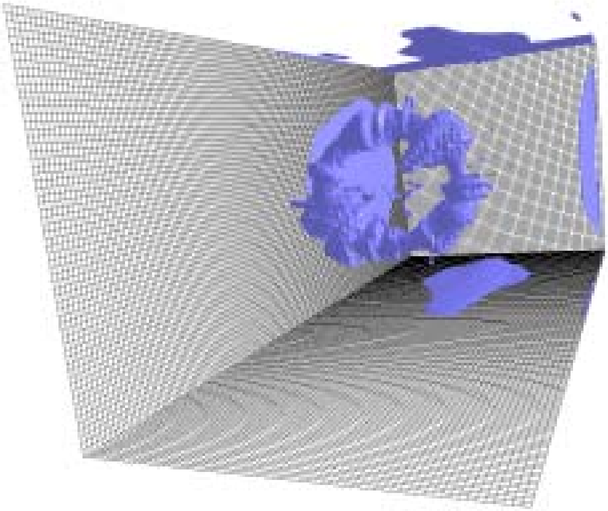

Figure 8: A front view of the results of the segmentation algorithm on the argon bubble data set.

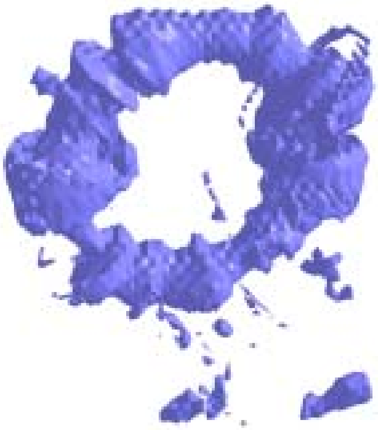

Figure 9: A close-up view of the “ring” identified by the segmentation algorithm.

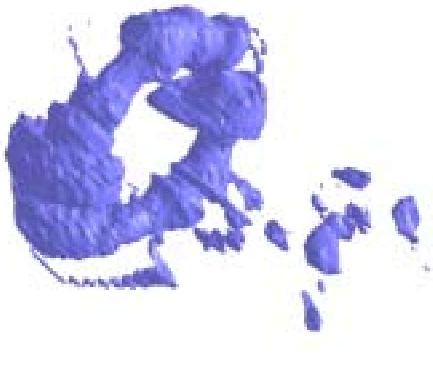

Figure 10: A close-up view of the “ring” identified by the segmentation algorithm.

[12] Gregory M. Nielson, Hans Hagen, and Heinrich Müller. *Scientific Visualization: Overviews, Methodologies, and Techniques*. IEEE Computer Society Press, 1997.

[13] Gregory M. Nielson and Il-Hong Jung. Tools for computing tangent curves for linearly varying vector fields over tetrahedral domains. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):360–372, 1999.

[14] W. F. Noh and Paul Woodward. SLIC (Simple line interface calculation). In A. I. van der Vooren and P. J. Zandbergen, editors, *Lecture Notes in Physics*, pages 330–340. Springer-Verlag, 1976.

[15] James Edward Pilliod and Elgridge Gerry Puckett. Second-order accurate volume-of-fluid algorithms for tracking material interfaces. Technical report, Lawrence Berkeley National Laboratory, 2000.

[16] Gerik Scheuermann, Thomas Bobach, Hans Hagen, Karim Mahrous, Bernd Hamann, and Kenneth I. Joy. A tetrahedra-based stream surface algorithm. In Thomas Ertl, Kenneth I. Joy, and Amitabh Varshney, editors, *Proceedings of IEEE Visualization 2001*, pages 83–91, October, 2001.

[17] Gerik Scheuermann, Hans Hagen, and Heinz Krüger. Clifford algebra in vector field visualization. In Hans-Christian Hege and Konrad Polthier, editors, *Mathematical Visualization*, pages 343–351. Springer Verlag, Heidelberg, 1998.

[18] Gerik Scheuermann, Bernd Hamann, Kenneth I. Joy, and Wolfgang Kollmann. Visualizing local vector field topology. *SPIE Journal of Electronic Imaging*, 9(4):356–367, October 2000.

[19] Gerik Scheuermann, Xavier Tricoche, and Hans Hagen. C1-interpolation for vector field topology visualization. In David Ebert, Markus Gross, and Bernd Hamann, editors, *Proceedings of IEEE Visualization '99*, pages 271–278, October, 1999.

[20] Xavier Tricoche, Gerik Scheuermann, and Hans Hagen. A topology simplification method for 2D vector fields. In Thomas Ertl, Bernd Hamann, and Amitabh Varshney, editors, *Proceedings Visualization 2000*, pages 359–366, 2000.

[21] Rüdiger Westermann, Christopher Johnson, and Thomas Ertl. Topology-preserving smoothing of vector fields. In *IEEE Transactions on Visualization and Computer Graphics*, volume 7, pages 222–229. IEEE Computer Society, 2001.

[22] Thomas Wischgoll and Gerik Scheuermann. Detection and visualization of closed streamlines in planar flows. In *IEEE Transactions on Visualization and Computer Graphics*, volume 7(2), pages 165–172. IEEE Computer Society, 2001.

[23] T. Wishgoll, G. Scheuermann, and H. Hagen. Distributed computation of planar-closed streamlines. In Robert Erbacher, Philip Chen, Matti Gröhn, Johnathan Roberts, and Craig Wittenbrink, editors, *Proceedings SPIE Visualization and Data Analysis 2002*, pages 41–50. SPIE, January 2002.

[24] D. L. Youngs. Time-dependent multi-matreial flow with large fluid distortion. In K. W. Morton and J. J. Baines, editors, *Numerical Methods for Fluid Dynamics*, pages 273–285. Academic Press, 1982.