Joint EUROGRAPHICS - IEEE TCVG Symposium on Visualization (2003)
G.-P. Bonneau, S. Hahmann, C. D. Hansen (Editors)

Detecting Critical Regions in Scalar Fields

Gunther H. Weber[1,2] Gerik Scheuermann[1] and Bernd Hamann[2]

[1] AG Graphische Datenverarbeitung und Computergeomterie, FB Informatik, Universität Kaiserslautern, Germany
[2] Center for Image Processing and Integrated Computing, Department of Computer Science, University of California, Davis, U.S.A.

Abstract

Trivariate data is commonly visualized using isosurfaces or direct volume rendering. When exploring scalar fields by isosurface extraction it is often difficult to choose isovalues that convey "useful" information. The significance of visualizations using direct volume rendering depends on the choice of good transfer functions. Understanding and using isosurface topology can help in identifying "relevant" isovalues for visualization via isosurfaces and can be used to automatically generate transfer functions.

Critical isovalues indicate changes in topology of an isosurface: the creation of new surface components, merging of surface components or the formation of holes in a surface component. Interesting isosurface behavior is likely to occur at and around critical isovalues. Current approaches to detect critical isovalues are usually limited to isolated critical points. Data sets often contain regions of constant value (i.e., mesh edges, mesh faces, or entire mesh cells). We present a method that detects critical points, critical regions and corresponding critical isovalues for a scalar field defined by piecewise trilinear interpolation over a uniform rectilinear grid. We describe how to use the resulting list of critical regions/points and associated values to examine trivariate data.

1. Introduction

An isosurface is the set of all points in three-dimensional (3D) space where a trivariate scalar field has a certain isovalue. By varying this isovalue v it is possible to visualize an entire scalar field indirectly. Determining isovalues where "interesting" isosurface behavior occurs is difficult. Features of a scalar data set can easily be missed when certain isovalues are not considered.

Direct volume rendering visualizes a scalar field by mapping scalar values to optical properties using a transfer function and rendering images. The value of resulting visualizations highly depends on the transfer function. Transfer functions are commonly chosen in a cumbersome process based on trial-and-error. Automatic transfer function design is an area of ongoing research, see, for example, Pfister et al. [20] or Fujishiro et al. [9, 11].

Isosurface topology provides insight into the fundamental structure of isosurface behavior across isovalues. Topological changes occur at critical points with an associated critical isovalue. Three types of critical points exist. At a local minimum a closed surface component is created. At a saddle either the *genus* of an isosurface changes, i.e., holes appear/disappear in a surface component, or surface components separate or merge. At a maximum a closed surface component vanishes. Traditionally, critical points and values are the subject of Morse theory [15]. Any given position of a data set can be classified by considering a small neighborhood around it. Considering a C^2-continuous function f, critical points can be determined analytically as points where the gradient ∇f vanishes. The associated type of the critical point is determined by the eigenvalues of the Hessian at that point. For general C^0-continuous functions, including those given by piecewise trilinear interpolation (at mesh vertices, on mesh edges and on mesh faces), gradient and Hessian are undefined. Based on the work of Banchoff [4] it is possible to "emulate" these criteria for discrete data sets by looking at a neighborhood around a point and checking how many regions with a larger and how many regions with a smaller value than the value at the considered location exist. This definition has been used in computational geometry to detect critical points of piecewise linear scalar fields, [8].

To handle general data sets we further need to extend the concept of critical points to critical regions. While Morse theory requires that only isolated points can be critical, it is possible that a data set contains entire regions that are critical: A torus can form around a circle, which could be a min-

imum, a new component can appear around a constant value sub-volume, or two surfaces can merge along a line denoting a saddle. We extend the concept of isolated critical points to regions in a data set. This enables us to detect topology in data sets that contain regions of constant value.

We consider the common case of data sets with data values given at vertices of a uniformly spaced rectilinear grid. We define isosurface topology by assuming that trilinear interpolation is used within individual cells and use the topology of the level set of the trilinear interpolant in each cell to define level set topology for the whole data set. We detect both critical points and critical regions. For critical regions, isosurface topology changes occur over a region (a one-, two-, or three-dimensional manifold embedded in the domain). We detect critical regions, classify them, and use the resulting set of critical isovalues for interactive data exploration using isosurfaces and for automated transfer function design.

2. Related Work

Topology of level sets for data given on simplicial meshes using linear interpolation in cells has been a subject of research in computational geometry. Related to our work is the concept of *contour trees* that encode topological changes of level sets. A contour tree is a graph that describes how contours in a level set appear, join, split, or disappear. Van Kreveld et al. [21] used contour trees to speed up isosurface extraction. Using a contour tree, they compute a minimum seed set for isosurface reconstruction. Starting from the resulting seed points all isosurface components are tracked and computed. Bajaj et al. [1] also considered tetrahedral meshes. They determine a *contour spectrum* specifying properties like $2D$ contour length, $3D$ contour area and gradient integral as functions of the isovalue. The contour spectrum is displayed along with the contour tree of a data set aiding a user in identifying interesting isovalues. Bajaj et al. [3] also developed a technique that uses topology to enhance visualizations of trivariate scalar fields. Their method employs a C^1-continuous interpolation scheme for rectilinear grids and detects critical points of a scalar field. Subsequently, integral curves (tangent curves) are traced starting from locations close to saddle points. These integral curves are superimposed onto volume-rendered images to convey structure of the scalar field.

Fujishiro et al. [9] used a *hyper-Reeb graph* for exploration of scalar fields. A *Reeb graph* encodes topology of a surface, similarly to a contour tree. In addition to tracking the number of contours, it also encodes genus changes. A hyper-Reeb graph encodes changes of topology in an extracted isosurface. For each isovalue that corresponds to an isosurface topology change, a node exists in the hyper-Reeb graph containing a Reeb graph encoding the topology of that isosurface. Fujishiro et al. [9] constructed a hyper-Reeb graph using "focusing with interval volumes," an iterative approach that finds a subset of all critical isovalues, which was introduced by Fujishiro and Takeshima [10]. The hyper-Reeb graph can be used, for example, for automatic generation of transfer functions. Fujishiro et al. [11] extended this work and used a hyper-Reeb graph for exploration of volume data. In addition to automatic transfer function design, their extended method allows them to generate translucent isosurfaces between critical isovalues. Considering just the images shown in their paper [9], it seems that their approach does not detect all critical isovalues of a scalar field.

Topology is also important in the context of data simplification to preserve important features of a data set. Bajaj and Schikore [2] extended previous methods to develop a compression scheme preserving topological features. Their approach detects critical points of a piecewise linear bivariate scalar field $f(x,y)$. In this approach, "critical vertices" are those vertices for which the "normal space" of the surrounding triangle platelet contains the vector $(0,0,1)$. Integral curves are computed by tracing edges of triangles along a "ridge" or "channel." Bajaj and Schikore's method incorporates an error measure and can be used for topology-preserving mesh simplification.

Gerstner and Pajarola [12] defined a bisection scheme that enumerates all grid points of a rectilinear grid in a tetrahedral hierarchy. Using piecewise linear interpolation in tetrahedra, critical points can be detected. Data sets are simplified by specifying a traversal scheme that descends only as deep into the tetrahedral hierarchy as necessary to preserve topology within a certain error bound. The method incorporates heuristics that assign importance values to topological features, enabling controlled topology simplification.

Our method for detecting critical regions extends the work of Weber et al. [22]. They consider piecewise trilinear data sets and detect critical points at grid vertices, faces, and the interior of cells. To ensure that topological changes only occur at isolated points, they impose the constraint that two edge-connected vertices must differ in value. If that condition is violated the classification step skips that vertex and proceeds. Each vertex is classified by considering its six edge-connected neighbors. Using a precomputed case table with 64 entries that is manually precomputed using a criterion based on Gerstner and Pajarola's work [12], they classify each vertex.

Isosurface extraction is commonly used in scientific visualization. Isosurfaces of data defined on rectilinear, hexahedral grids are commonly extracted using the *marching cubes* (MC) method introduced by Lorensen and Cline [14]. Due to topological inconsistencies the original algorithm could produce holes in an extracted isosurface, see, for example, [7]. Several authors have proposed extensions [5, 6, 16, 17, 19] to the original algorithm with the aim of generating consistent and topologically correct isosurfaces. Most recently, Nielson [18] has provided an in-depth analysis of the trilinear interpolant resulting in a consistent and topologically

correct implementation of MC. In his doctoral dissertation, Lopes [13] proposed the use of additional points in a cell's interior to improve accuracy of isosurfaces extracted by the MC method. We use his approach with a minor correction motivated by the work of Nielson [18] to extract topologically meaningful isosurfaces.

3. Critical Points and Critical Regions

Considering a C^2-continuous function f, critical points occur where the gradient ∇f vanishes, i.e., where $\nabla f = 0$. The type of a critical point can be determined by the signs of the eigenvalues of the Hessian of f. Trilinear interpolation, when applied individually to the rectilinear grid cells discretizing a domain, produces, in general, only C^0-continuous functions.

Critical points can also be defined for discrete data sets. Scalar fields defined on simplicial meshes using piecewise linear interpolation have been subject of intensive research. By considering the definitions from calculus, it is possible to classify a point by considering its neighborhood. The resulting criteria lead to Definition 1, given in [22], which is based on the work of Banchoff [4]. It can also be used for discrete data. An extension allows us to define critical regions:

Definition 1 (Regular and Critical Points) *Let $M \subset \mathbb{R}^3$ be a mesh and $F : M \to \mathbb{R}$ be a C^0-continuous function that is C^∞-continuous in each grid cell. A point $\mathbf{x} \in \mathbb{R}^3$ is called (a) regular or ordinary, (b) minimum, (c) maximum, (d) saddle, (e) extended minimum, (f) extended maximum, (g) extended saddle, or (h) flat point of F, if for all $\varepsilon > 0$ there exists a neighborhood $U \subset U_\varepsilon(\mathbf{x})$ with the following properties: If $\dot{\bigcup}_{i=1}^{n_p} P_i$ is a partition of the preimage of $(F(\mathbf{x}), +\infty)$ in $U - \{\mathbf{x}\}$ into "positive" connected components, $\dot{\bigcup}_{j=j}^{n_n} N_j$ is a partition of the preimage of $(-\infty, F(\mathbf{x}))$ in $U - \{\mathbf{x}\}$ into "negative" connected components, and $\dot{\bigcup}_{k=1}^{n_z} Z_k$ is the partition of the preimage of $\{F(\mathbf{x})\}$ in $U - \{\mathbf{x}\}$ into "zero set" connected components, then (a) $n_p = n_n = n_z = 1$, (b) $n_p = 1$ and $n_n = n_z = 0$ (and U contains only ordinary points), (c) $n_n = 1$ and $n_p = n_z = 0$ (and U contains only ordinary points), (d) $n_p + n_n > 2$, $n_p, n_n \geq 1$, $n_z > 1$ (and U only contains ordinary points), (e) $n_p = 1$, $n_n = 0$, $n_z \geq 1$ (and U contains non-ordinary points), (f) $n_n = 1$, $n_p = 0$, $n_z \geq 1$ (and U contains non-ordinary points), (g) $n_p + n_n > 2$, $n_z = 1$ (and U contains non-ordinary points), and (h) $n_z = 1$ and $n_p = n_n = 0$.*

Remark 1 The zero set corresponds to the level set for the value at $\mathbf{x}$.

Remark 2 Definition 1 is a modified form of Definition 1 from [22]. The original definition added the zero set to both positive and negative sets. However, to extend the concept of critical points to critical regions it becomes necessary to consider the zero set individually.

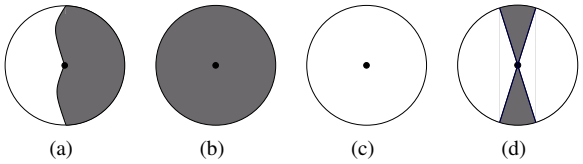


Figure 1: *(a) Around a regular point $\mathbf{x} \in \mathbb{R}^3$, the isosurface $F^{-1}(F(x))$ divides space into a single connected volume P with $F > 0$ (dark gray) and a single connected volume N with $F < 0$ (white). (b)/(c) Around a minimum/maximum, all points in U have a larger/smaller value than $F(\mathbf{x})$. (d) In case of a saddle, there are multiple (more than one) regions with values larger or smaller than the value $F(\mathbf{x})$.*

Remark 3 Cases (a) – (d) describe localized critical and regular points and are illustrated in Figure 1. These definitions are equivalent to those used in topology and Morse theory. Cases (e) – (g) extend these concepts to extended regions. An extended minimum/maximum/saddle corresponds to a region where each point in the region is a minimum/maximum/saddle according to the original definition in [22]. Concerning case (h), all points in U have the same value as $F(\mathbf{x})$, i.e., the region has constant value.

Remark 4 When U is small enough, the non-ordinary points inside U all have the same value $F(x)$: Inside a cell, this stems from the fact that the derivative is zero at a non-ordinary point and one can separate the regions with zero derivative. At faces, this fact holds since critical points on a face are also critical points for the bilinear interpolant, and the derivative of the bilinear interpolant is used to provide the answer. On an edge, it suffices to consider the linear interpolant, which leads to the same result. Finally, vertices of the grid have a neighborhood of points on edges, faces and cell interiors.

For a localized minimum/maximum the minimum/maximum is completely surrounded by larger/smaller values. In both cases, n_z is zero. If n_z is one, each neighborhood contains points with the same function value. Each of these points of same value would also be a minimum according to the original definition in [22]. (In that definition the zero set is also included in the positive and negative components.) At a saddle, surface components merge at a point or along a region. The zero set corresponds to the level set of the saddle value with the currently considered location "cut out." For a localized saddle, the level set at a saddle corresponds to several surface components that meet in a point. If this point is removed, these surface components become disconnected, i.e., $n_z > 1$. If the saddle is along a region, the surface components meet along that region. If one location is cut out, these components remain connected, i.e., $n_z = 1$. Thus, the number of components of the zero set determines whether there are non-ordinary points in each neighborhood. This fact holds for the 3D case, but it

does not hold for the 2D case. In the 2D case, the zero set would be partitioned into more than one components when a saddle is "cut out," regardless whether it is a face saddle or not.

We can use the concept of an extended minimum, maximum, and saddle in combination with flat points to define larger regions that are critical, i.e., larger regions around which isosurface topology changes. We therefore extend the definitions of distance and neighborhood:

Definition 2 (Region Distance) *Let R be a connected region and $\|\mathbf{x}\|$ be a norm. The distance $\|\mathbf{x}-R\|$ of a point $\mathbf{p}$ to a region is the infimum of distances of $\mathbf{p}$ to all locations inside that region, i.e., $\|\mathbf{x}-R\| := \inf\{\|x-y\|, y \in R\}$.*

Definition 3 (Region Neighborhood) *Let R be a connected region. The ε neighborhood $U_\varepsilon(R)$ of R consists of all locations that have a distance less than ε from R, i.e., $U_\varepsilon(R) := \{x, \|\mathbf{x}-R\| < \varepsilon\}$.*

Definition 4 (Classification Region) *A connected region $R \subset \mathbb{R}^3$ in space is a* classification region *if the following statement holds: Each point $\mathbf{x} \in R$ is a flat point, an extended minimum, an extended maximum, or an extended saddle according to Definition 1, and R has maximal size, i.e., each point in an ε neighborhood around R is an ordinary point according to Definition 1.*

Remark 5 A Classification Region consists of points that cannot be classified locally but must be classified by also considering surrounding locations.

Definition 5 (Regular and Critical Regions) *Let $M \subset \mathbb{R}^3$ be a mesh and $F : M \to \mathbb{R}$ be a C^0-continuous function that is C^∞-continuous in each grid cell. A classification region $R \subset \mathbb{R}^3$ is called (a) regular, (b) minimum, (c) maximum, or (d) saddle of F, if for all $\varepsilon > 0$ there exists a neighborhood $U \subset U_\varepsilon(R)$ with the following properties: If $\dot{\bigcup}_{i=1}^{n_p} P_i$ is a partition of the preimage of $(F(R), +\infty)$ in $U - R$ into "positive" connected components, $\dot{\bigcup}_{j=j}^{n_n} N_j$ is a partition of the preimage of $(-\infty, F(R))$ in $U - R$ into "negative" connected components, and $\dot{\bigcup}_{k=1}^{n_z} Z_k$ is the partition of the preimage of $\{F(R)\}$ in $U - R$ into zero set connected components, then (a) $n_p = n_n = n_z = 1$, (b) $n_p \geq 1$ and $n_n = n_z = 0$, (c) $n_n \geq 1$ and $n_p = n_z = 0$, and (d) $n_p + n_n > 2$ and $n_z = 1$.*

4. Connected Components in a Single Cell

We classify a region by examining those hexahedral cells that contain its neighborhood. Within a cell, we are concerned with one trilinear function. If we need to determine connected components in a neighborhood around a classification region, we need to know its behavior in single cells. The following paragraphs provide proofs that document the correctness of our method.

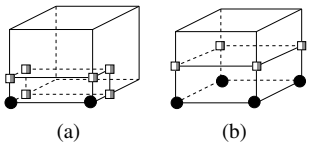


Figure 2: *With respect to the L^∞-norm, the intersection of the neighborhood with a cell is a collection of boxes. If the neighborhood is chosen sufficiently small, vertices of the boxes that do not coincide with cell vertices or faces have the same polarity as an edge-connected vertex, and their corresponding region is also connected to one of the cell's edges.*

Lemma 1 *Let C be a rectilinear cell with trilinear interpolation applied to it and v be an arbitrary value. Each maximum-extended connected positive region $R \subset f^{-1}((v, +\infty))$ contains at least one vertex of C.*

Proof A maximum-extended connected region with values $> v$ contains a local maximum within its compact closure. Since trilinear interpolation only allows a maximum in the vertices of a cell to exist, R contains a vertex. □
(The same lemma holds for negative connected regions.)

Lemma 2 *Let C be a trilinearly interpolated rectilinear cell that intersects the classification region R, i.e., a cell that contains a number of vertices that belong to the classification region with a value of v. Each connected positive or negative region in the neighborhood of R intersecting C, i.e., $U_\varepsilon(R) \cap C$, contains a part of an edge of the cell that starts at a vertex belonging to the classification region. (The vertex itself does not belong to the region.)*

Proof (sketch) Consider the L^∞-norm. A neighborhood around a point is a cube, and a neighborhood around an edge or a face is a (rectilinear) cuboid/box. The neighborhood around a collection of points, edges, and faces is a set of boxes. The intersection of the neighborhood with the cell is also a set of boxes, see Figure 2. Values are trilinearly interpolated within each box. Vertices of a box coincide with vertices of the cell, its edges, or lie in its interior. If we choose a sufficiently small neighborhood, the vertices in the interior have the same polarity as one of their edge-connected neighbor vertices. Each connected region in one of the boxes is connected to one of the vertices of that box. This vertex, in turn, is connected to a vertex that lies either on an edge of C or coincides with one of its vertices belonging to the classification region. Thus, each positive- and each negative-connected component within the neighborhood is connected to an edge emanating from a vertex of the classification region. □
Furthermore, if two edges of same polarity are connected within the intersection of a cell's face with the neighborhood of the classification region, their corresponding regions are also connected. Additionally, "being connected" is a transitive property, i.e., , if edge (region) A is connected to edge

(region) B and edge (region) B is connected to edge (region) C, then edge (region) A and edge (region) C are connected as well.

5. Detecting Critical Regions

5.1. Overview

Our algorithm detects critical regions in a two-pass approach. First, we perform a pass that classifies all grid vertices that can be classified locally, i.e., all grid vertices whose edge-connected neighbors differ in value. This local classification is performed as described in [22]. Using the polarities of the six edge-connected neighbor vertices, an index for a 64 entry look-up table (LUT) is computed. The type of vertex is then read from a manually generated LUT. Vertices that cannot be classified locally are marked by setting an associated flag, see Figure 3(a).

Second, this flag is used in a subsequent pass that handles global classification. The next vertex that belongs to a region that needs to be classified is located. Starting with this seed vertex we perform a "flood fill" operation that recursively adds all vertices having the same associated scalar value as the seed vertex and are edge-connected to the current vertex. While adding these vertices to the classification region we reset the associated flag that marks the vertex as belonging to an unclassified region. We also maintain an updated bounding box that contains all marked vertices. After the flood fill we have a bounding box, shown as dotted rectangle in Figure 3(b), containing all vertices belonging to the current classification region. All vertices belonging to this region are flagged. We extend the bounding box containing all vertices of the classification region to a bounding box of grid cells such that no vertex belonging to the classification regions lies on the boundary of this bounding box, shown as dashed rectangle in Figure 3(b).

We classify a region by constructing two graphs whose connected components correspond to connected regions in a neighborhood around that region. One graph corresponds to positive regions, and another graph corresponds to negative regions. These graphs are constructed by marching through all cells of the extended bounding box and adding nodes and edges to this graph cell-by-cell. It follows from our observations (Section 4) that for each cell a connected region in the neighborhood of the classification region contains at least one edge starting from a vertex of the classification region. Thus, we can represent connectivity of these regions by considering all edges that originate in a vertex belonging to the classification region. For each of those edges, we create a node in one of the graphs representing the classification region neighborhood. To add edges as nodes to one of the graphs, we assign a unique identification number to each edge in a grid by effectively numbering all edges of the grid. This step ensures that the same node in the neighborhood graph is accessed for two cells sharing that edge.

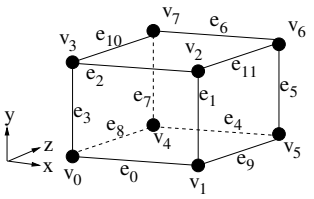


Figure 4: *Vertex (v_0–v_8)and edge (e_0–e_{11}) numbering scheme employed in this paper. Vertex numbers are also used to determine the topology case number.*

Once the two graphs representing the neighborhood of a region are constructed, a region can be classified by counting the connected components in them. The graph in Figure 3(c), for example, indicates that the classified region is a saddle. When a region is classified, the search for the next vertex belonging to an unclassified region proceeds until all vertices of the grid are checked.

5.2. Constructing the Connectivity Graphs within a Cell

Within a cell the graph is constructed based on the vertex configuration within the cell. A main case is determined based on which vertices of the current cell belong to the classification region. The vertices of a cell are numbered according to Figure 4 and a look-up index is computed by setting the corresponding bit for each vertex belonging to the classification region. This step results in an index referencing one of 256 possible cases. Considering rotational symmetry this number can be reduced to 23 base cases, shown in Figure 5. (This approach is similar to symmetry consideration, and our numbering of base cases corresponds to the numbering scheme used in [18].)

We reference an LUT that contains the base case number and permutations of vertices, edges, and faces that map vertex, edge, and face numbers of the current case to vertex, edge, and face numbers of the corresponding base case. We then construct the edge graph for a cell by calling a function that handles the corresponding base case. Whenever this function references a vertex, edge, or face of the cell it does so via the corresponding permutation function for the current topology case index.

We add all edges connecting the classification region to positive vertices as nodes to the positive-components graph and all edges connecting the classification region to negative vertices to the negative-components graph. If a pair of edges belongs to the same connected region in a cell, we connect the corresponding nodes in the positive or negative edge graph. When cell marching terminates, the connected components in one graph correspond to positive-connected regions, and the connected components in the other graph correspond to negative-connected regions.

We first consider cases $C0$ and $C22$. For case $C0$, a cell does not contain any vertices of the classification region and

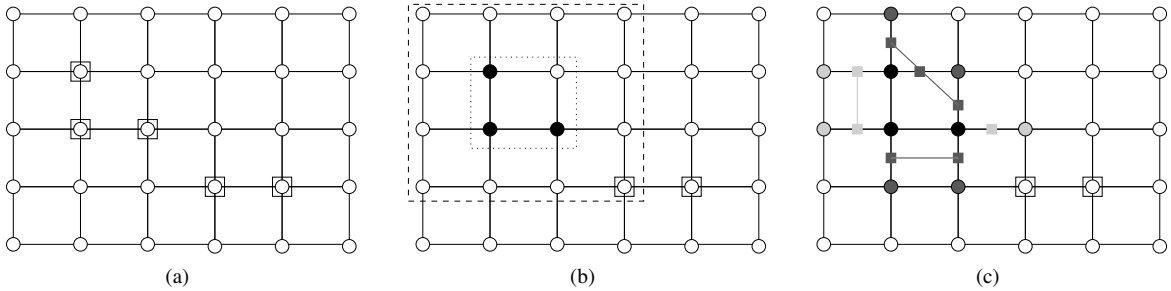


Figure 3: *Stages of the algorithm: (a) After the local pass all vertices that cannot be classified locally are marked with a flag (indicated by a rectangle around these vertices in the figure). Once the first flagged vertex is determined we perform a flood fill that marks all vertices belonging to the classification section (marked by solid black disks in the figure). The region is classified by constructing two graphs, one representing positive- and one representing negative-connected regions in the neighborhood around the classification region. For each edge originating in a vertex belonging to the classification region, a corresponding node in the graph exists (shown as dark and light gray solid rectangles). Two nodes in the graph are connected when the corresponding edges belong to the same connected region in a cell. A region can be classified by counting the connected components in the two graphs.*

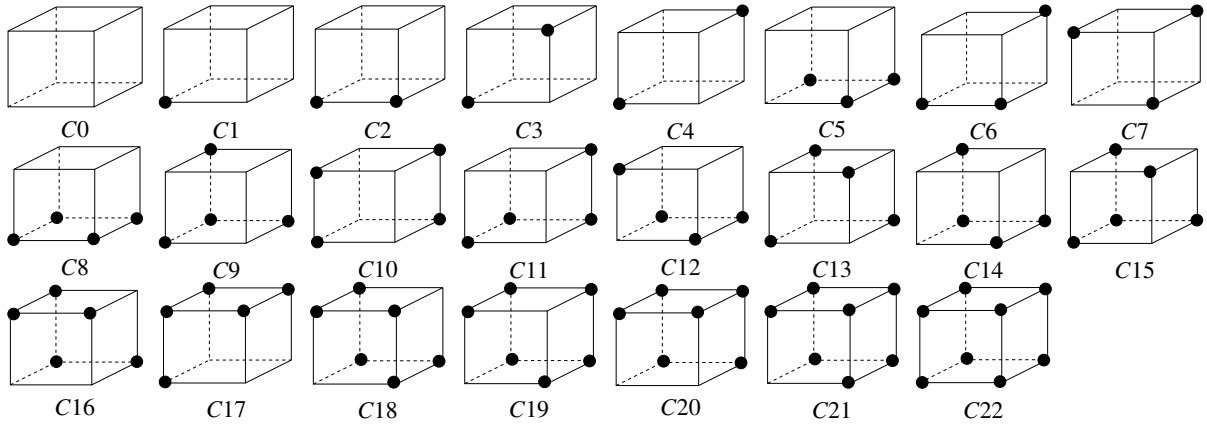


Figure 5: *Base cases for constructing graphs that represent positive and negative regions around a classification region. A case number is determined according to which vertices belong to the classification region (marked as solid black disks). The other vertices can assume arbitrary values different from the value of the classification region.*

thus does not intersect the ε-neighborhood of the classification region. A $C22$ cell belongs completely to the classification region and does not intersect the neighborhood either. Cells of these types can be skipped during graph construction.

Cases $C3$, $C4$, $C6$, $C7$, $C10$, and $C13$ are handled by considering them as a combination of single vertices or single edges belonging to the classification area. If we consider, for example, case $C3$, we observe that two vertices belong to the classification region. Both vertices have edge-connected neighbor vertices that do not belong to the classification region. Thus, it is possible to handle them independently and construct the neighborhood graphs for each vertex individually. Similarly, case $C6$, for example, can be handled by constructing the graphs for an edge and for a vertex.

Constructing the neighborhood graph for a single vertex corresponds to case $C1$. This is equivalent to the situation in a single cell as described in [22]. The neighborhood around the vertex in the classification region is partitioned in the same way as in a linearly interpolated tetrahedron. Thus, in this case all vertices of the same polarity are always connected, and we only need to add edges between all edges of equal polarities in the corresponding graph.

For the construction of the neighborhood graph of an edge, we must consider 16 cases. (The edge is connected to four vertices that do not belong to the classification region.) By using symmetry and reversing polarities, it is possible to reduce the number of cases to four possible sub-cases, see Figure 6. Figure 6 shows possible configurations for the edge-connected vertices that have either positive (gray) or negative (white) polarity. Each edge has a corresponding node in the classification graph (shown as white rectangles for the negative graph and gray rectangles for the positive

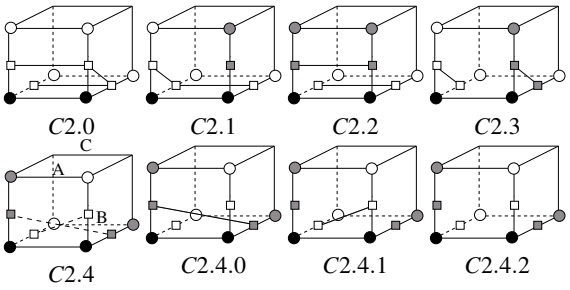


Figure 6: *Sub-cases for connecting an edge.*

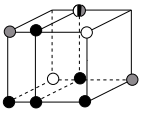


Figure 7: *Disambiguation for case C2.4.*

graph). Two nodes are connected by a line segment when they must be connected in their graph.

Cases $C2.0$ to $C2.3$ are derived from this observation: In these cases, nodes of the same polarity are on a face, and they are connected on the intersection of that face with the neighborhood of the classification region. Since connectivity is transitive, the actual connectivity graph is the transitive hull of the graphs shown in the figure. However, we are only interested in the number of connected components in a graph. Thus, it suffices to add those edges to the graphs that are shown in the figure. For case $C2.4$ two diagonally opposite vertices have the same polarity. In this case, we must determine whether the positive vertices are connected ($C2.4.0$), whether the negative vertices are connected ($C2.4.1$), or whether all regions are separated ($C2.4.2$). To distinguish between these cases, we first determine for which parameter values t_A and t_B along the edges "A" and "B" the value of the classification region is taken on. (For both parameter values, zero corresponds to the "left" end of the edge and one to the "right" end of the edge.) If $t_A < t_B$ then the regions belonging to v_2 and v_4, i.e., the negative regions, overlap. If $t_A > t_B$, the regions belonging to vertices v_3 and v_5, i.e., the positive regions, overlap. If $t_A = t_B$, the bilinear slice through the cell at that parameter value contains three vertices with the value of the classification region. The remaining vertex on edge "C" determines the polarity of the whole face. If it is positive, the positive vertices v_3 and v_5, along with their corresponding edges, are connected ($C2.4.0$); if it is negative, the negative vertices v_2 and v_4, along with their corresponding edges, are connected ($C2.4.1$). If it also has the value of the classification region, all vertices and edges are separated ($C2.4.2$). The remaining sub-cases can be derived from these base cases.

For the remaining base cases, those vertices that do not belong to the classification region, but are connected to one of its vertices by an edge, are numbered. We subsequently compute a topology sub-case number by setting the bit for each positive vertex. These sub-cases can be derived from a number of base sub-cases. Figure 8 shows base sub-configurations for topology base case $C5$. (Edges e_8 and e_0 lead to the same non-classification region vertex on a face and are connected within the intersection of a neighborhood of the classification region with that face. Thus, their corresponding nodes in the classification graph are always connected.)

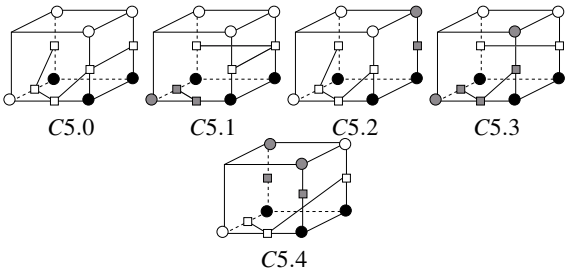


Figure 8: *Sub-cases for topology base case C5.*

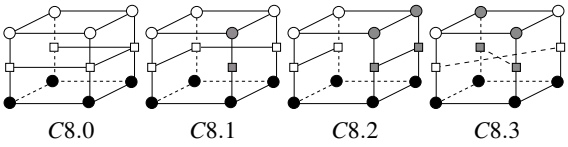


Figure 9: *Sub-cases for topology base case C8.*

Similarly to the construction of the classification graph for an edge, the graphs for most cases can be derived from the fact that nodes for edges on the same face are connected within the intersection of the region neighborhood and that face if they have the same polarity and using transitivity of the graph. Similarly to the edge case $C2$, only a minimum number of edges needed to obtain a correct number of connected components is given for the neighborhood graphs. The actual neighborhood graphs are the transitive closure of the graphs in the figure. Sub-configuration $C5.4$ is interesting: In contrast to sub-configuration $C2.4$ no ambiguity exists. The bottom face has always the same polarity as vertex v_0. Because trilinear interpolation is continuous, a bilinear slice close to the bottom face connects the vertices of the same polarity as v_0, i.e., v_0 and v_6. Thus, nodes for edges leading from the classification region to these vertices must be connected in the classification graph.

Case $C8$, see Figure 9, is similar to the case of handling an edge belonging to the classification region. For cases $C8.0$ – 8.2 the same arguments can be used to derive the classification graphs in each cell. It differs from the edge case in the way ambiguities ($C8.3$) are resolved: All vertices of the bottom face have the same value. Thus, contours on all bilinear slices parallel to the top face are topologically equivalent to contours on the top face. We use the asymptotic decider [19] to determine connectivity on that face and connect negative nodes, or positive nodes. If the saddle on the top face has the same value as the classification region, no nodes are con-

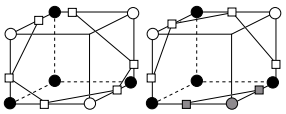


Figure 10: *Sub-cases for topology base case C9.*

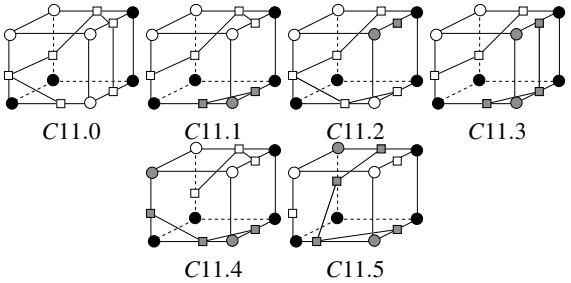


Figure 11: *Sub-cases for topology base case C11.*

nected. In this case, the saddle on that face is connected to the classification region via an "extended" internal cell saddle (see Section 5.3). To classify the region correctly in this case it would be necessary to trace that saddle and combine region classification with classification of extended face saddles. We currently do not do this. The neighborhood graphs for base case $C9$, see Figure 10, can be constructed using the same arguments as for case $C5$.

Figure 11 shows the classification graphs for sub-cases of topology case $C11$. The graphs for cases $C11.0$-$C11.4$ can be constructed using the same observations as for base case $C5$. For sub-case $C11.5$ we observe that all positive regions are connected since the complete bottom and back faces have positive polarity.

Case $C12$ is a combination of base case $C5$ and vertex case $C1$. Case $C14$ can be obtained by "mirroring" base case $C11$. Base case $C15$ is a combination of base case $C9$ and vertex case $C1$. Base case $C16$ has only three vertices that do not belong to the classification region. If a pair of vertices has the same polarity, it is alway connected in the neighborhood of the classification region. For base cases $C17$, $C19$, and $C20$, all vertices not belonging to the classification region lie on a single face. Furthermore, on this face they are always connected, when they have the same polarity. Thus, in these cases vertices of equal polarity are always connected. Base case $C18$ contains only two vertices that do not belong to the classification region. If these vertices have the same polarity, they are connected. Case $C21$ consists of only one vertex. In this case, it is necessary to connect the nodes in the classification graph that correspond to its incoming edges.

5.3. Detecting Saddles on a Cell Faces and within its Interior

We currently detect saddles on a cell's faces and within its interior using the same criteria that are described in [22]. On cell faces, saddles of the bilinear interpolant are detected. Subsequently we determine by considering the adjacent cells, whether that saddle is a saddle of the piecewise trilinear interpolant. We note, that internal saddle need not occur at isolated points anymore. The gradient of the interpolant within a cell can be zero along a line or a hyperbolic curve that connects two cell face saddles. In some cases, it is not possible, to determine locally, whether a face saddle is a saddle of the piecewise trilinear interpolant. We currently discard these cases. To handle them correctly, it is necessary to combine face saddles and extended interior saddles and classify them together.

6. Applications

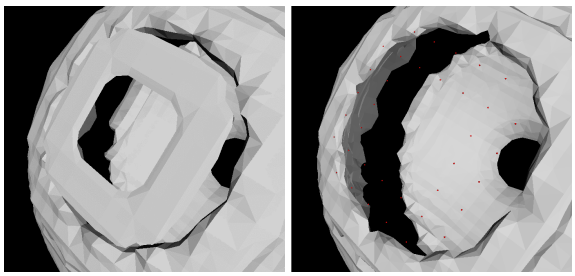
A convenient way to use critical isovalues is to provide a user with a navigational tool in addition to an isosurface viewer. Prior to starting an isosurface viewer, critical points, critical regions, and corresponding isovalues can be computed and displayed in an "isovalue navigator" window. In this window, critical isovalues are listed along with a corresponding type (vertex minimum, vertex maximum, vertex saddle, face saddle, interior saddle, region minimum, region maximum, region saddle). When a user selects a critical point, its corresponding position in space is marked by a sphere whose color depends on the type (blue, red, and green representing minimum, maximum, or saddle, respectively). For regions we mark its position by a point set with a color corresponding to type. Buttons allow a user to set the isovalue of a displayed isosurface to a value slightly below, equal to, or slightly above a critical isovalue. The isovalue offset for the isosurfaces below and above a chosen critical isovalue is specified in a text field.

In data sets containing several "nested isosurfaces," i.e., data sets where one isosurface component is completely contained within another, it can be difficult to locate a critical point, even if its position is marked. The "isovalue navigator" contains a button that positions the camera so that the viewing focus is the critical point. For critical regions we position the camera to view the centroid of the critical region.

Critical isovalues can guide in the construction of transfer functions. Given a list of critical isovalues we can construct a corresponding transfer function based on the methods described in [22], which, in turn, are based on the work of Fujishiro et al. [9, 11].

7. Results

Figure 12 shows a region maximum of value 190 in the "Nucleon" data set (courtesy of SFB 382 of the German Research Council (DFG), available at `http://www.volvis.org`) which was missed by the previous approach [22]. In the neighborhood of this maximum a torus-shaped isosurface component disappears.



(a) (b)

Figure 12: *Region maximum in "Nucleon" data set (courtesy of SFB 382 of the German Research Council (DFG)). (a) Isosurface for an isovalue slightly below the maximum. (b) Isosurface for an isovalue slightly above the maximum.*

Figure 13 shows results obtained by applying topological analysis to the "Hydrogen" data set (courtesy of SFB 382 of the German Research Council (DFG), available at `http://www.volvis.org`). Figures 13 (a) and 13 (b) show volume-rendered images of the data set using transfer functions that were automatically generated from a list of critical isovalues. Figures 13 (c) – (l) show a topological analysis of the same data set using isosurfaces.

8. Conclusions and Future Work

We have presented an algorithm that determines critical regions for minima, maxima, and saddles. Our approach detects exact regions for maxima and minima. Region saddles are detected, and our approach can mark a large region as a saddle, a region larger than the actual saddle region. We plan to extend our approach to detect exact regions of saddles. Handling of face saddles and internal saddles should be merged into a combined step that traces saddles in the interior of cells, even though this case rarely occurs. It should be possible to adapt our approach to detect critical regions for piecewise linear simplicial meshes. Developing additional automatic transfer function schemes could also help in gaining insight in data sets containing a large number of critical regions/points and isovalues.

Acknowledgments

This work was supported by the Stiftung Rheinland-Pfalz für Innovation; the National Science Foundation under contract ACI 9624034 (CAREER Award) and ACI 0222909, through the Large Scientific and Software Data Set Visualization (LSSDSV) program under contract ACI 9982251, and through the National Partnership for Advanced Computational Infrastructure (NPACI); the National Institute of Mental Health and the National Science Foundation under contract NIMH 2 P20 MH60975-06A2; the Lawrence Livermore National Laboratory under ASCI ASAP Level-2 Memorandum Agreement B347878 and under Memorandum Agreement B503159; and the Lawrence Berkeley National Laboratory.

We thank the members of the AG Graphische Datenverarbeitung und Computergeometrie at the Department of Computer Science at the University of Kaiserslautern, Germany, and the Visualization and Graphics Research Group at the Center for Image Processing and Integrated Computing (CIPIC) at the University of California, Davis.

References

1. Chandrajit L. Bajaj, Valerio Pascucci, and Daniel R. Schikore. The contour spectrum. In: Roni Yagel and Hans Hagen, editors, *IEEE Visualization '97*, pages 167–173, IEEE, ACM Press, New York, New York, October 19–24 1997.

2. Chandrajit L. Bajaj, Valerio Pascucci, and Daniel R. Schikore. Visualizing scalar topology for structural enhancement. In: David S. Ebert, Holly Rushmeier, and Hans Hagen, editors, *IEEE Visualization '98*, pages 51–58, IEEE, ACM Press, New York, New York, October 18–23 1998.

3. Chandrajit L. Bajaj and Daniel R. Schikore. Topology preserving data simplification with error bounds. *Computers & Graphics*, 22(1):3–12, 1998.

4. Thomas F. Banchoff. Critical points and curvature for embedded polyhedral surfaces. *American Mathematical Monthly*, 77(5):475–485, May 1970.

5. Evgeni V. Chernyaev. Marching cubes 33: Construction of topologically correct isosurfaces. Technical Report CN/95-17, CERN, Geneva, Switzerland, 1995. Available as `http://wwwinfo.cern.ch/asdoc/psdir/mc.ps.gz`.

6. Paulo Cignoni, Fabio Ganovelli, Claudio Montani, and Roberto Scopigno. Reconstruction of topologically correct and adaptive trilinear isosurfaces. *Computers & Graphics*, 24(3):399–418, June 2000.

7. Martin J. Dürst. Additional reference to "marching cubes" (letters to the editor). *Computer Graphics*, 22(2):72–73, April 1988.

8. Herbert Edelsbrunner, John Harer, and Afra Zomorodian. Hierarchical morse complexes for piecewise linear 2-manifolds. In: *Proceedings 17th Symposium on Computational Geometry*, pages 70–79, 2001.

9. Issei Fujishiro, Taeko Azuma, and Yuriko Takeshima. Automating transfer function design for comprehensible volume rendering based on 3D field topology analysis. In: David S. Ebert, Markus Gross, and Bernd Hamann, editors, *IEEE Visualization '99*, pages 467–470, IEEE, IEEE Computer Society Press, Los Alamitos, California, October 25–29, 1999.

10. Issei Fujishiro and Yuriko Takeshima. Solid fitting: Field interval analysis for effective volume exploration. In: Hans Hagen, Gregory M. Nielson, and Frits Post, editors, *Scientific Visualization Dagstuhl '97*, pages 65–78, IEEE, IEEE Computer Society Press, Los Alamitos, California, June 1997.

11. Issei Fujishiro, Yuriko Takeshima, Taeko Azuma, and Shigeo Takahashi. Volume data mining using 3D field topology analysis. *IEEE Computer Graphics and Applications*, 20(5):46–51, September/October 2000.

12. Thomas Gerstner and Renato Pajarola. Topology preserving and controlled topology simplifying multiresolution isosurface extraction. In: Thomas Ertl, Bernd Hamann, and Amitabh Varshney, editors, *IEEE Visualization 2000*, pages 259–266, 565, IEEE, IEEE Computer Society Press, Los Alamitos, California, 2000.

13. Adriano M. Lopes. *Accuracy in Scientific Visualization*. Ph.D. dissertation, University of Leeds, United Kingdom, March 1999. Available at `http://www.mat.uc.pt/~adriano/Publications/thesis.ps.gz`.

14. William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics (SIGGRAPH 87 Conference Proceedings)*, 21(4):163–169, July 1987.

15. John W. Milnor. *Morse Theory*. Princeton University Press, Princeton, New Jersey, May 1963.

16. Claudio Montani, Riccardo Scateni, and Roberto Scopigno. A modified look-up table for implicit disambiguation of marching cubes. *The Visual Computer*, 10(6):353–355, 1994.

17. Balas K. Natarajan. On generating topologically consistent isosurfaces from uniform samples. *The Visual Computer*, 11(1):52–62, 1994.

18. Gregory M. Nielson. On marching cubes. To appear in IEEE Transactions on Visualization and Computer Graphics, 2003.

19. Gregory M. Nielson and Bernd Hamann. The asymptotic decider: Removing the ambiguity in marching cubes. In: Gregory M. Nielson and Larry J. Rosenblum, editors, *IEEE Visualization '91*, pages 83–91, IEEE, IEEE Computer Society Press, Los Alamitos, California, 1991.

20. Hanspeter Pfister, Bill Lorensen, Chandrajit Bajaj, Gordon Kindlmann, Will Schroeder, Lisa Sobierajski Avila, Ken Martin, Raghu Machiraju, and Jinho Lee. The transfer-function bake-off. *IEEE Computer Graphics and Applications*, 21(3):16–22, May/June 2001.

21. Marc van Kreveld, René van Oostrum, Chandrajit Bajaj, Valerio Pascucci, and Daniel Osvaldo. Contour trees and small seed sets for isosurface traversal. In: Jean-Daniel Boissonnat, editor, *Proceedings of the Thirteenth ACM Symposium on Computational Geometry*, pages 212–219, ACM Press, New York, New York, June 4–6 1997.

22. Gunther H. Weber, Gerik Scheuermann, Hans Hagen, and Bernd Hamann. Exploring scalar fields using critical isovalues. In: Robert J. Moorhead, Markus Gross, and Kenneth I. Joy, editors, *IEEE Visualization 2000*, pages 171–178, IEEE, IEEE Computer Society Press, Los Alamitos, California, 2002.

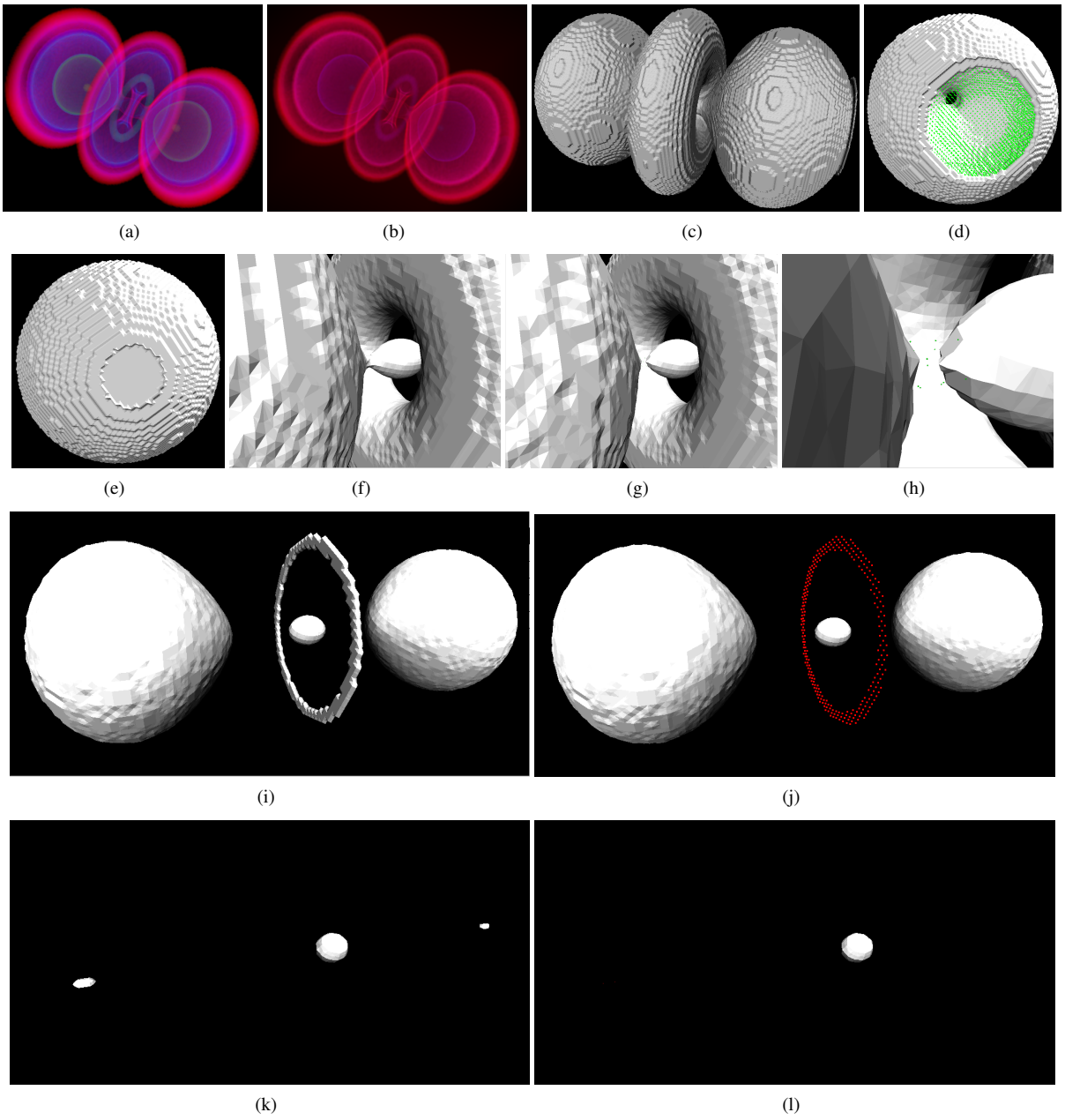


Figure 13: *Topological analysis of the "Hydrogen" data set: (a), (b) Volume rendered images with automatically generated transfer functions emphasizing topological changes (a) and zones of similar topological behavior (b); (c) – (l) topological structure; (c) around a region minim having a value of zero, two components appear simultaneously; (d), (e) at a saddle region having a value of* 3.5*, a hole in one surface component closes; (f), (g) the "inner" surface is separated into three disjoint components along two saddle regions having a value of* 12*; (h) a close-up of one of the two saddle regions having a value of* 12*; (i), (j) The "ring" component disappears around a region maximum having a value of* 36*; (k),(l) Two components disappear at two region maxima having a value of* 80*; The remaining component disappears around a localized maximum having a value of* 250*; (Data set courtesy of SFB 382 of the German Research Council (DFG).)*