Topological Volume Skeletonization and its Application to Transfer Function Design

Shigeo Takahashi[1], Yuriko Takeshima[2], and Issei Fujishiro[3]

[1]Graduate School of Arts and Sciences, University of Tokyo, Japan
[2]Institute of Fluid Science, Tohoku University, Japan
[3]Graduate School of Humanities and Sciences, Ochanomizu University, Japan

Abstract

Topological volume skeletonization is a novel approach for automating transfer function design in visualization by extracting the topological structure of a volume dataset. The skeletonization process yields a graph called a *volume skeleton tree*, which consists of volumetric critical points and their connectivity. The resultant graph provides critical field values whose color and opacity are accentuated in the design of transfer functions for direct volume rendering. Visually pleasing results of volume visualization demonstrate the feasibility of the present approach.

1 Introduction

Visualizing large-scale 3D datasets has become a challenging problem due to the recent developments of computer hardware and measurement equipments for scientific applications. While direct volume rendering is one of the popular tools for visualizing complicated inner structures of such large-scale datasets, it suffers from the lack of interactivity for adjusting visualizing parameters to generate informative images. The key to comprehensible volume rendering lies in the design of transfer functions (TFs), which map physical field values of given volume samples to optical properties such as color and opacity. However, the design requires excessive human interactions especially in the case of unknown large-scale 3D datasets because of the need to explore their involved features. Volume data mining (VDM) introduced by Fujishiro et al. [10, 11] is one of the concepts which allows us to extract such volume features effectively along with judicious design of TFs.

In recent years, several *data-centric* methods have been proposed for designing TFs semi-automatically for effective volume rendering [3, 16, 6]. Although these algorithms succeed in visualizing the features of volume datasets in some practical cases, they do not capture the associated global structures involved in volume datasets explicitly.

This paper, therefore, presents a novel approach for automating the TF design by explicitly extracting a rigorous volumetric structure even from an unknown volume dataset. Here, the volumetric structure means an abstract graph called a *volume skeleton tree* (VST), which represents the splitting and merging of isosurfaces of a varying scalar field value. The key idea in this approach is to extract the VST that symbolizes isosurface evolution over the domain of the field values, and then to emphasize significant isosurfaces having topological transitions in the design of TFs. Figure 1 illustrates such an example where two isosurfaces emerge and grow to adjoin each other. Tracing isosurface changes of each connected component produces a VST as shown on the right of Figure 1. The VST represents critical field values (CFVs) by its nodes, where significant topological changes occur in the isosurface evolution. In Figure 1, voxels around adjoining isosurfaces should be emphasized for informative volume rendering.

In this paper, we present a robust algorithm for constructing such VST even when handling noisy discrete volume samples. Since the definition of the VST is based on singularity theories on surfaces embedded in 4D hyper-space, it is well-defined in a mathematical sense, and offers a sound mechanism for analyzing the global structure of the volume dataset. The principle of our VST-based TF design is also provided with visually-appealing results of rendered images.

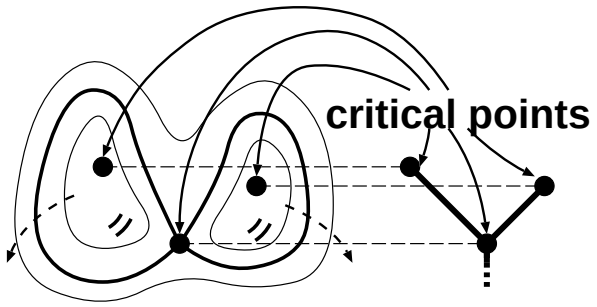


Figure 1: Isosurface evolution and its corresponding volume skeleton tree.

Although the fundamental concept has already presented in our previous papers [9, 10, 11], the present method is quite different from the previous one in that it analyzes topological structures directly from given volume datasets while the previous method employed a two-step process. Our previous method first extracts a series of isosurfaces at some discrete scalar field values, and then constructs a topological skeleton called a *Reeb graph* (RG) for every isosurface. The method extracts the CFVs by comparing the RG of an isosurface with its neighbor along the axis of the scalar field value. Actually, these extracted RGs are associated with this axis to construct a hierarchical graph representation called a *Hyper Reeb graph* (HRG). While this method produces good results, it takes a great deal of computation time because it seeks all the CFVs by using the bisection method to sample the domain of the scalar field value [11]. On the other hand, our new method reduces this computational cost considerably. This is enabled because our method directly extracts volumetric critical points by comparing the field value of each voxel only against those of its local neighbors. In addition, our method offers more advantages. The first advantage is that we can detect all the critical points embedded in the input volume so that they maintain the mathematical integrity, while we may miss some critical points when using our previous bisection-based method. Furthermore, we can discriminate significant CFVs from minor ones by considering the global structure of isosurface changes because, in this framework, we can directly trace the isosurface evolution over the domain of scalar field values. The present method also identifies the type of each topological change in isosurfaces, which allows us to take into account such types when emphasizing the significant isosurfaces in volume rendering.

This paper is organized as follows: After briefly reviewing previous work related to ours in Section 2, we present a robust algorithm for extracting the VST from a discrete volume dataset in Section 3. Section 4 describes a principle of TF design based on the VST, and presents a feasibility study of the proposed scheme by applying it to simulated and practical datasets. Finally we conclude this paper with related future extensions in Section 5.

2 Related Work

For convenience, we classify related work into two categories: level-set analysis and transfer function design.

2.1 Level-Set Analysis

Primary level-set analysis for topological skeletonization has been studied for discrete surface samples such as digital elevation models (DEMs). Such surface skeletons are represented by graphs, which are denoted as "contour trees" or "Reeb graphs," and well studied in the field of geographical information systems by Kweon and Kanade [19] and Takahashi et al. [29]. Lazarus and Verroust proposed an algorithm for constructing such structures from general polyhedral objects [20]. Another interesting framework for such topological analysis is proposed by Edelsbrunner's group [4], where they presented an algorithm for partitioning DEMs into topological primitives called Morse-complexes and introduced geometric measures [5] to control their multiresolution representations. The method seems to have the

potential for handling one-dimensional higher cases, i.e. volume datasets, after some sophistications, but it is still limited to surface cases.

On the other hand, several authors addressed methods for extracting similar skeletons from volume datasets. For example, Itoh et al. [14, 15] used an extrema graph to find starting points for isosurface propagation. While the graph is very efficient to accelerate such isosurface extraction, it only captures maxima and minima and discards saddles that play an indispensable role in the TF design. Bajaj's group developed an efficient algorithm for calculating such volume skeletons (i.e., contour trees) that also capture the changes in the number of connected isosurfaces [32], and then used them to explore feature isosurface transitions for efficient volume visualization [1]. The computational complexity of this algorithm was further improved by Tarasov and Vyalyi [30]. Furthermore, as an extension of the algorithms, Carr, Snoeyink, and Axen [2] developed an excellent algorithm that computes such topological skeletons from objects of any dimension. While these algorithms are elegant from a computational point of view, they do not pursue the change in topology (i.e., genera) of isosurfaces with respect to the scalar field value, and hence they cannot maintain the topological consistency of the extracted features.

More recently, Pascucci et al. [25] developed the algorithm in [2] so as to exactly identify the topology (i.e. genus) of an isosurface at any point of the skeleton graph. However, their algorithm still does not care about the topological consistency because they did not resolve degenerate singularities that are likely to be involved in the noisy large-scale datasets of limited precision.

On the other hand, the present algorithm retains this topological consistency by resolving all the possible degeneracies. Another difference from the conventional methods is that we fully identify the types of critical points including the isosurface embeddings in 3D space around them. This is why our method enjoys the global configuration of isosurface transitions with respect to the scalar field value without suffering from minor troublesome noise.

2.2 Transfer Function Design

Several methods for semi-automating TF design are in the literature, and these may be divided into the two major categories: image-centric and data-centric.

An *image-centric* method presents a broad selection of TFs and generates the corresponding visualization images so that a user can pick up the best one. The method then seeks the most suitable TF by iteratively refining the image selection according to the user's evaluation. He et al. [12] used stochastic algorithms to iteratively evolve the previous generation of TFs into the next one, while Marks et al. [21] introduced a special metric to locate the visualization images in the parametric space to calculate an optimal set of perceptually different images all at once. However, these image-centric methods do not take into account the meaning (i.e. significant features) involved in target volume datasets.

On the other hand, a *data-centric* method tries to find the best TF by rigorously analyzing such features of the input volume dataset. Castro et al. [3] presented a TF design method that is specific to medical data. They first segment a medical volume dataset into several components such as bones and muscles, and then assign a basis TF to each of the components so that they can obtain the final TF as a weighted sum of the basis TFs. Fang et al. [6] applied a sequence of 3D image processing procedures to an input volume in order to directly visualize the processed data with ordinary TFs.

One of the most outstanding work was recently reported by Kindlmann and Durkin [16], where they formalized a data structure called *a histogram volume* in a 3-dimensional space that is spanned by a scalar field value f and its first and second directional derivatives f' and f''. They used this data structure to design TFs that efficiently emphasize boundaries between different materials in the volume data. Kniss et al. [17] sophisticated this idea as a multi-dimensional TFs, and implemented a rich set of intuitive tools for designing such TFs efficiently. Hladuvka et al. [13] also proposed a similar method that employs the maximum and minimum principle curvatures of the volume function for defining such multi-dimensional TFs. Another generalization of the histogram volume can be found in the work by Tenginakai et al. [31]. They used a statistical model based on central moments of volume data, which are actually higher-dimensional versions of the histogram volumes.

The approach presented herein can be regarded as a novel one in this second category, since it relies heavily on the volume field topology delineated by the topological skeleton, which stretches a network over critical points in a volume. While the previous data-centric approaches are effective, they may all suffer from gentle gradients of scalar field values because they are sensitive only to local

sharp features. On the other hand, the present method extracts a global structure in the volume data as well as its local features. While Weber et al. [33] recently sophisticated our previous work [10] by developing the direct extraction of critical points from volume datasets, they considered only local features around the critical points and did not take into account its associated global connectivity.

3 Topological Skeletonization Algorithm

The topological skeleton provides us with key insights for visualizing the global structure of a large-scale 4D volume dataset. This section describes an algorithm for finding such a topological skeleton from a volume dataset that is represented as samples of a 4D hyper-surface. While our algorithm is an extension of the previous one for 3D surface cases [29], it has been newly developed for volumes because the behavior of 3D isosurfaces involved in the volume is much more complicated than that of 2D isocontours for 3D surfaces. This means that we have to detect changes in the topological type (genus) of each isosurface when skeletonizing a given volume data. The algorithm will select candidate field values to be emphasized in the design of TFs, hence it helps us generate comprehensible rendering of unknown volume datasets. Note that, in this paper, we always consider isosurface transitions as the scalar field value decreases.

The following is a list of steps in the algorithm:

1. Extract volumetric critical points that represent the positions of topological transitions of isosurfaces such as isosurface appearance (at maxima), merging, splitting, (at saddles) and disappearance (at minima).
2. Trace voxel flow lines from saddle critical points until maxima and minima to obtain a voxel flow network (VFN).
3. Convert the obtained VFN into a VST.
4. Simplify the VST in order to find the significant field values of the volume dataset.

Each step of the algorithm will be described in detail below.

Before going into details, we assume that a volume dataset is represented by sample points of a single-valued function,

$$w = f(x, y, z), \tag{1}$$

where x, y, and z represent ordinary 3D coordinates and w its corresponding scalar field value. This lets us to classify isosurfaces evolving in a volumetric domain into two categories: *solid isosurfaces* whose interior samples are larger in the field value than those on its boundary, and *hollow isosurfaces* whose interior is smaller. The same notation is used in [28] for surface cases. Here, we also assume that outermost solid isosurfaces of a decreasing field value always expand (i.e. non-shrinking) in 3D space defined by the volume dataset. This assumption is reasonable because we can consider the reverse ordering of the field values when the assumption is violated. Recall that we always decrease the field value when considering the topological changes of isosurfaces. In the remainder of this paper, we consider the following analytic volume function as an example:

$$\begin{aligned} f(x,y,z) \;=\; & 4c^2\big((x-R)^2+(z-R)^2\big) - \big((x-R)^2+y^2+(z-R)^2+c^2-d^2\big)^2 + \\ & 4c^2\big((x+R)^2+(z+R)^2\big) - \big((x+R)^2+y^2+(z+R)^2+c^2-d^2\big)^2, \end{aligned} \tag{2}$$

where $0 < d < c$ and $c^2 + d^2 \geq 6R^2$.

3.1 Definition of Critical Points

Cutting a volume dataset at different field values will produce topological changes of isosurfaces, including isosurface splitting and merging. A critical point will appear at a contact point between such isosurfaces when they split and merge. Figure 1 illustrates three critical points, where the upper two points activate isosurface growth and the lower one merges the two isosurfaces.

A more precise definition is stated as follows. A critical point p of the field function (1) is defined to be a point that satisfies:

$$\frac{\partial f}{\partial x}(p) = \frac{\partial f}{\partial y}(p) = \frac{\partial f}{\partial z}(p) = 0. \tag{3}$$

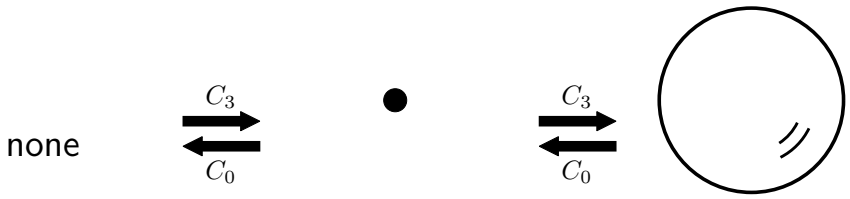


Figure 2: Isosurface changes at a maximum and a minimum. The arrows indicate the isosurface changes when the scalar field value decreases.

From Equation (3), for example, we can derive the following five critical points of the function (2):

$$\begin{aligned} p_{1,2} &: \left(\mp\sqrt{\frac{c^2+d^2-2R^2}{2}}, 0, \pm\sqrt{\frac{c^2+d^2-2R^2}{2}} \right), \\ p_{3,4} &: \left(\mp\sqrt{\frac{c^2+d^2-6R^2}{2}}, 0, \mp\sqrt{\frac{c^2+d^2-6R^2}{2}} \right), \\ p_5 &: (0,0,0)\,, \end{aligned} \tag{4}$$

and their corresponding CFVs:

$$\begin{aligned} f(p_{1,2}) &= 8c^2d^2, \\ f(p_{3,4}) &= 8c^2d^2 - 16R^2(c^2+d^2-4R^2), \\ f(p_5) &= -2(c^2-d^2)^2 + 8R^2(c^2+d^2-R^2). \end{aligned} \tag{5}$$

Here, note that a smaller subscript corresponds to a larger CFV.

The Morse lemma [8] claims that an infinitesimal neighborhood around a critical point of Equation (1) has a local coordinate where f has one of the following quadratic forms:

$$f = \begin{cases} -x^2-y^2-z^2 & \text{maximum (index 3)} \\ -x^2-y^2+z^2 & \text{saddle (index 2)} \\ -x^2+y^2+z^2 & \text{saddle (index 1)} \\ x^2+y^2+z^2 & \text{minimum (index 0).} \end{cases} \tag{6}$$

Here, the index represents the number of negative eigenvalues of the Hessian matrix at the critical point. As shown in Equation (6), critical points of the volume dataset are classified into four types: a maximum (index 3), a saddle (index 2), a saddle (index 1), and a minimum (index 0). In this paper, we use the symbols C_3, C_2, C_1, and C_0 to represent the above four types of critical points, where each subscript represents the corresponding index.

Figure 2 shows isosurface changes around a maximum (C_3) and a minimum (C_0) of the function (6) when the scalar field value decreases. At a maximum, a new topological sphere appears in 3D space. Conversely, an existing sphere disappears at a minimum.

At a saddle of index 2 two isosurfaces merge while an existing isosurface splits at a saddle of index 1 when the corresponding field value goes down. For such saddles, in particular, the topological changes become more complicated when we take into account spatial configuration of isosurfaces, i.e., isosurface embeddings in 3D space. Figure 3 classifies such changes in isosurfaces depending on their embeddings in 3D space. This classification is one of our contributions that discriminate the present work from our previous ones. This classification has a close relationship with the work by Shinagawa et al. [28] for the case of 3D surfaces.

Furthermore, the previous assumption of the non-shrinking property of outermost solid isosurfaces restricts isosurface changes at saddles as shown in Figure 3. Here, solid isosurfaces can follow only the four paths indicated by thick arrows because they expand as the field value decreases, while hollow isosurfaces the four paths indicated by outlined arrows.

The isosurfaces of the volume function (2) evolve as shown in Figure 4. Here, two isosurfaces appear at p_1 and p_2 first, and then they meet at p_3 and p_4 to become a torus. Finally, the hole of

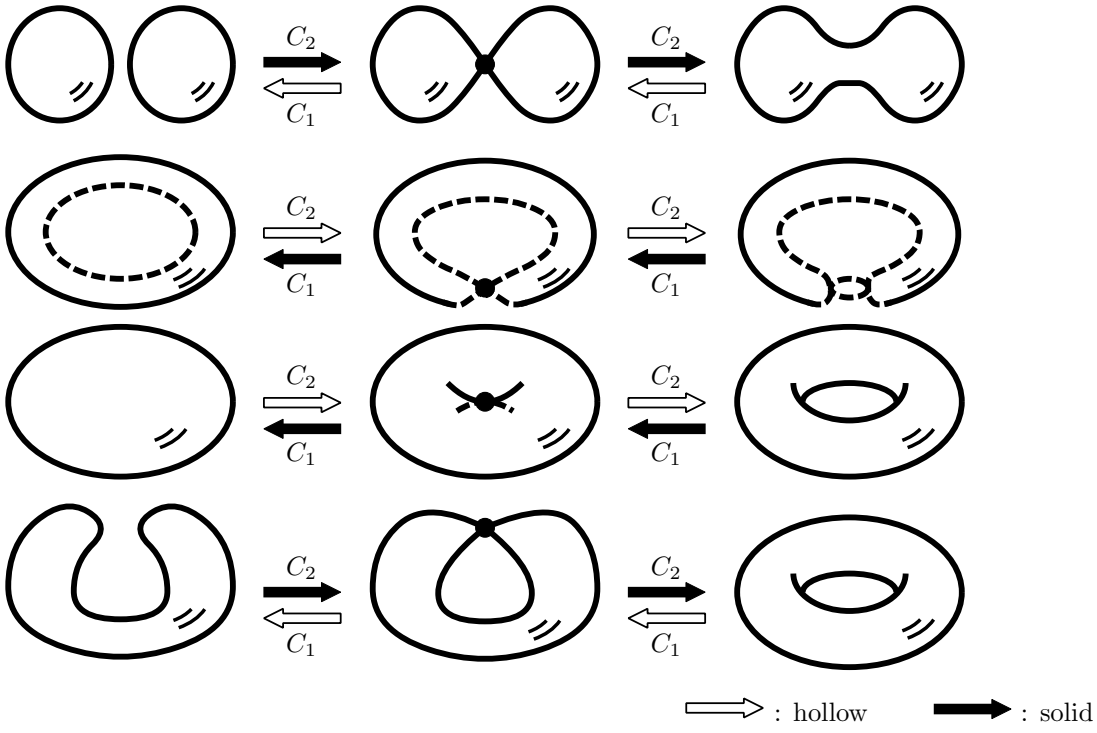


Figure 3: Classification of isosurface changes at saddles depending on the embedding in 3D space. The arrows indicate the isosurface changes when the scalar field value decreases. Thick arrows correspond to changes of solid isosurfaces while outlined arrows hollow isosurfaces.

the torus is filled at p_5, which makes the isosurface a sphere again. This isosurface evolution suggests that the critical points p_1 and p_2 are of index 3 (i.e. C_3), p_3 and p_4 are of index 2 (i.e. C_2), and p_5 is of index 1 (i.e. C_1).

By adding a virtual minimum to the volume function (1), a volume dataset becomes a topological 3D sphere S^3. The Euler-Poincaré formula states that the critical points of a 3D sphere S^3 must satisfy the following formula [22]:

$$\#\{C_3\} - \#\{C_2\} + \#\{C_1\} - \#\{C_0\} = 0, \tag{7}$$

where $\#\{C_i\}$ represents the number of critical points of index i. In Figure 4, p_6 represents such a virtual minimum. In our framework, this representation is used to check the validity of the extracted critical points from a discrete volume dataset, which is also one of the main contributions of this paper.

3.2 Critical Point Extraction

The volume skeletonization begins with the extraction of critical points from discrete samples of a field function (1). For later analysis, the extracted critical points must be correct in the sense that they satisfy the Euler-Poincaré formula (7). Therefore, it is necessary to interpolate between the samples before extracting critical points. For this purpose, our algorithm fills tetrahedra into 3D grid samples to establish linear interpolation. This is because the tetrahedralization serves as the most reasonable method for volume interpolation in such a way that triangulation does for surface interpolation. If higher-order interpolants are used here, they may cause unpleasant oscillation that results in a number of uninvited critical points.

One simple way to fill tetrahedra is to decompose a cube confined by eight corner grids into five tetrahedra as shown in Figure 5(a). This is used by default in our algorithm, while the decomposed cube of Figure 5(a) and its rotated one are placed alternatively to avoid unmatched triangle faces between neighboring cubes. However, our scheme decomposes a cube differently if the cube has an

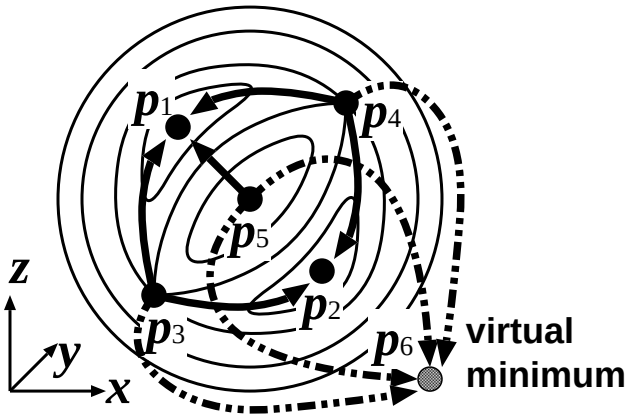


Figure 4: Critical points of the function (2) and voxel flows in between.

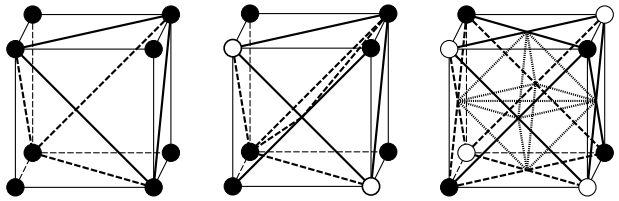


Figure 5: Cube decomposition: (a) decomposition by default, (b) decomposition when the front face is ambiguous, and (c) decomposition when all the faces are ambiguous.

ambiguous face where isocontours cannot be determined uniquely from its four corners. In such a case, we partition the ambiguous quadrilateral face with two diagonals in order to simulate the asymptotic decider [23]. Figure 5(b) illustrates a volume cube whose front face is ambiguous. The face is partitioned like a letter "X" and a new vertex is added so that its (x, y, z)-coordinates are identical with the center of the face. The field value of the new vertex, however, comes from the intersection between the two asymptotes of hyperbola defined by the four corner grid samples of the ambiguous face. This assures that our scheme can generate isosurface evolution topologically equivalent to that obtained by the asymptotic decider. Figure 5(c) shows the decomposition of a volume cube if all of its faces are ambiguous. Another scheme of tetrahedralization for this purpose is available from [33].

Extracting correct critical points requires careful handling of boundary sample vertices when filling tetrahedra into volume grids. For this reason, we connect all the boundary samples with the virtual minimum so that we can fill the gap with additional tetrahedra between the triangular faces on the volume boundary and the virtual minimum. Figure 6 gives a rough idea of this process. Here, we turn our attention not to the geometric position of the virtual minimum but to its topological connectivity with boundary samples. Actually, our algorithm only refers to the field value of each voxel and its associated connectivity in the tetrahedralization when extracting critical points; it ignores their geometric coordinates. Note that this process also allows us to regard the entire volume dataset as a 3D sphere S^3, and to extract critical points that satisfy the Euler-Poincaré formula (7).

Let us assume first that all the critical points in the dataset are non-degenerate, which means that a critical isosurface has no more than one topological change in itself. In the tetrahedral decomposition, each vertex has its adjacent vertices that constitute a triangulated sphere surrounding the original vertex. This can be easily constructed by putting together the tetrahedra incident to the current vertex, and then eliminating its incident edges. On the sphere, we define two classes of vertices; a plus vertex whose field value is larger than that of the original vertex, and a minus vertex whose value is smaller. This classification divides the sphere into several plus and minus connected regions by linking neighboring vertices of the same class. The process of extracting critical points proceeds by checking

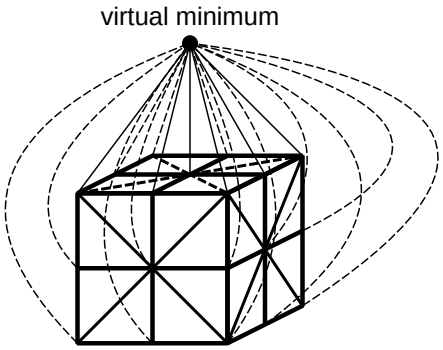


Figure 6: Connecting boundary samples with the virtual minimum.

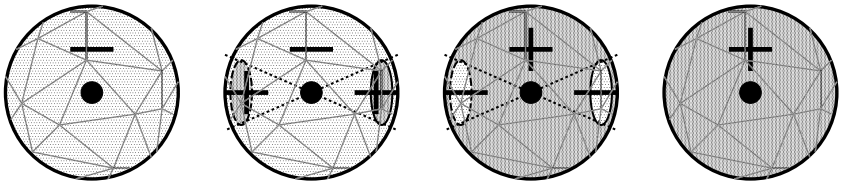


Figure 7: Plus and minus connected regions on the surrounding spheres and isosurfaces for (a) C_3, (b) C_2, (c) C_1, and (d) C_0.

whether each vertex meets the following criteria:

$$\begin{array}{llll} N_+ = 0, & N_- = 1 & \text{for} & C_3 \\ N_+ = 2, & N_- = 1 & \text{for} & C_2 \\ N_+ = 1, & N_- = 2 & \text{for} & C_1 \\ N_+ = 1, & N_- = 0 & \text{for} & C_0. \end{array} \tag{8}$$

Here, N_+ and N_- are the numbers of the plus and minus connected regions on the surrounding sphere. Figures 7(a), (b), (c), and (d) show the surrounding spheres and associated isosurfaces of C_3, C_2, C_1, and C_0, respectively. Note that at the saddles of indices 1 and 2, as shown in Figures 7(b) and (c), two cone-like parts of isosurfaces are in contact at the corresponding critical points.

A volume dataset usually contains degenerate critical points as well as non-degenerate ones. Such unusual critical points are classified into two categories: *multiple saddles* and *level field clusters*.

A multiple saddle is a vertex where three or more isosurfaces merge or one existing isosurface splits into three or more isosurfaces simultaneously. In this case, we count the multiplicity of the degenerate saddle point by decomposing it into non-degenerate ones. For example, the degenerate saddle having three isosurfaces in contact is considered to be a point where two isosurfaces adjoin first and another isosurface comes later. This means that the corresponding multiplicity becomes 2. In order to take into account these multiple saddles, the criteria (8) are rewritten as $N_+ = n+1$ for C_2 and $N_- = n+1$ for C_1, where n represents the multiplicity of the corresponding saddle point.

On the other hand, a level field cluster is defined to be a group where two or more neighboring vertices are at the same field value in the tetrahedral decomposition. The key idea of our approach is to introduce another ordering in addition to the primary ordering with respect to the scalar field value. This implies that two vertices having the same scalar field value are compared using the second ordering newly introduced. As the second ordering, our algorithm uses the lexicographical ordering with respect to the (x, y, z)-coordinates, which amounts to slightly distorting the scalar field values of the vertices in the level field cluster.

Note that in our framework we keep the connectivity of the tetrahedral mesh when resolving the degeneracy for simplicity, while we can further subdivide the mesh to actually split a degenerate saddle point into non-degenerate ones. Readers can find such examples for surface cases in [30, 4].

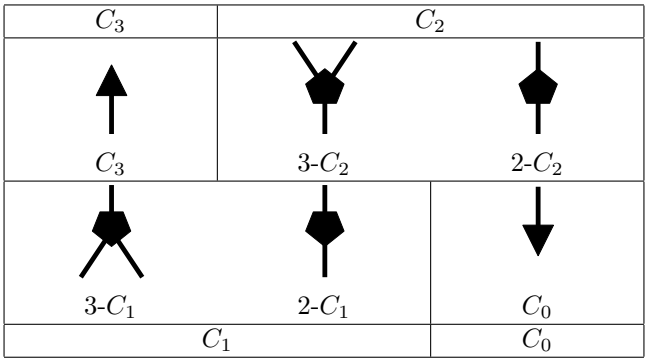


Figure 8: VST components around critical points.

3.3 Voxel Flow Network

Before finding the skeleton in a volume dataset, our algorithm constructs a graph called a *voxel flow network* (VFN) that works as an intermediate representation for the overall skeletonization process. Constructing the VFN begins with tracing voxel flows emanating from each saddle critical point. As shown in Figure 7, a saddle of index 2 has two plus regions and one minus region on its surrounding sphere, while that of index 1 one plus region and two minus regions. In each plus region, our algorithm selects a vertex of the largest field value as its representative. In the same way, a vertex having the smallest field value in a minus region is elected to be its representative. These representative vertices serve as the starting points for tracing the voxel flows in the tetrahedral volume dataset.

The process of tracing voxel flows is as follows. From each representative vertex of a plus region, we traverse from point to point by moving to the highest adjacent vertex in the tetrahedral decomposition. This tracing process terminates when reaching a maximum. Conversely, the representative vertex of a minus region leads us to a minimum through a descent tracing process. The VFN is then introduced to capture the field configuration between the critical points. Here, each node of the VFN represents a critical point and its link a voxel flow in between. Figure 4 shows the VFN of the analytic volume function (2). This VFN will provide enough information to construct a VST for later use.

3.4 Volume Skeleton Tree

A *volume skeleton tree* (VST) is defined as a graph that represents the splitting and merging of isosurfaces with respect to the field value, and effectively delineates the transition of such evolving isosurfaces. Each node of the graph represents a critical point, and its link one connected component of a varying isosurface in between. The VST is constructed by assembling components as shown in Figure 8, where each component contains one critical point. Here, we arrange these critical components so that the corresponding field value decreases from top to bottom in the figure. A C_3-type node has only one downward link, while a C_0-type node only one upward link. A C_2-type node has either of the two components: a "Y"-shaped branch if the corresponding critical point merges two separate isosurface components, or a simple joint if it makes a hole of a torus on an isosurface. For a node C_1, these components are turned upside down because its paths of isosurface evolution are the reverse of those of C_2. This argument is readily verified from Figures 2 and 3. In this paper, these four patterns of saddle components are denoted by 3-C_2, 2-C_2, 3-C_1, and 2-C_1, according to the degree (valence) of each saddle point, as shown in Figure 8.

In our framework, an algorithm for converting a VFN into a VST can be developed as an extension of the previous algorithm [29], which is used for topological analysis of a 3D surface such as DEMs. However, the volumetric version of the algorithm must be more sophisticated because an isosurface can become either a topological sphere or connected tori for 4D volumes while an isocontour is limited to a topological circle for 3D surfaces. This implies that the VST cannot fix the degree of its saddle node uniquely only from its type. Nevertheless our algorithm can construct a VST since the VST becomes a tree (i.e., the VST has no cycle in itself) due to the assumption that the volume dataset has a form of a single-valued function (1).

Our algorithm determines the VST from its extreme components to its center. Figure 9 illustrates

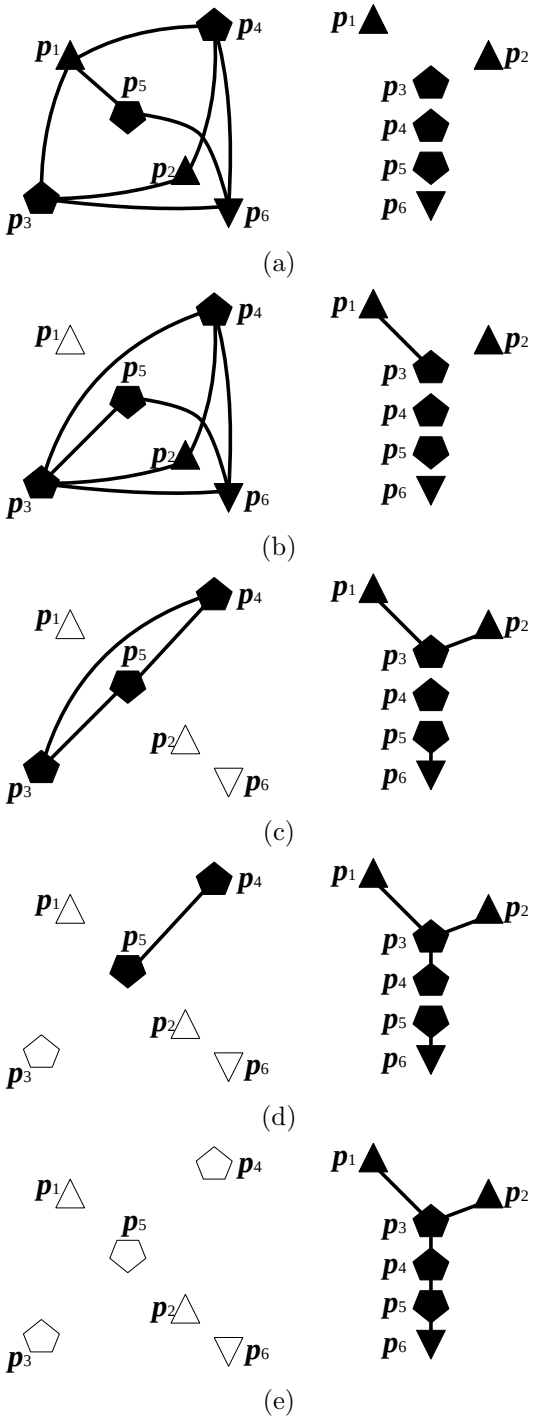


Figure 9: Process of constructing the VST of Equation (2): VFNs on the left and VSTs on the right. The critical points are arranged from top to bottom according to their scalar field values.

the VFNs of the analytic function (2) (on the left)[1] and its corresponding VSTs (on the right) in the conversion process. Here, a smaller subscript represents a critical point having a larger field value. Figure 9(a) gives the initial states of the VFN and VST.

The first node to be handled is a critical point of C_3 or C_0 because it always becomes a leaf in a VST. In Figure 9(a), our first target is the node p_1. As its adjacent node in the VST, our algorithm selects the node p_3 since it has the largest field value of all the nodes adjacent to p_1 in the VFN. In addition, in the VFN, the link $\overline{p_1p_3}$ is removed and the links emanating from p_1 are modified so that their endpoints become p_3 instead. Note that the VFN does not accept a new link if it already has an identical one. After this process, the node p_1 in the VFN is marked as finished because its incident link in the VST is now fixed. The resultant graphs are shown in Figure 9(b), where the finished nodes are indicated by outlined figures. The same process can be applied to the node p_2 of C_3 and the similar process will be done for the virtual minimum p_6 (Figure 9(c)).

After having processed all the maxima and minima, we turn our attention to saddles of degree 3 (i.e. 3-C_2 and 3-C_1) that already have two fixed links in the VST and some links in the VFN. In Figure 9(c), the point p_3 is the case where it has two fixed links in the VST and some links in the VFN. Here, the saddle has either upward or downward links in the VFN as it seeks an upward or downward link in the VST. Since p_3 has only downward links in the VFN, it looks for the adjacent point having the largest field value in the VFN in order to find its adjacent node in the VST. This process for the saddles of 3-C_2 and 3-C_1 is repeated until we cannot find any saddles having two fixed links in the VST. Figure 9(d) shows the states of the two graphs after this process.

If the VFN still has links at this point, there exist saddle points whose degrees are 2 (i.e. 2-C_2 and 2-C_1) in the VST. Such saddle points definitely appear as leaves in the determined parts of the VST, which means that such saddle points must have either the largest or smallest field value of all the undetermined nodes. In Figure 9(d), p_4 and p_5 are the cases. After having fixed the links of the saddles of 2-C_2 and 2-C_1 in the VST, the algorithm will be back to the previous stage where it again seeks nodes of C_2 and C_1 having two fixed links in the VST. In this case, however, the conversion process terminates when the link $\overline{p_4p_5}$ is added to the VST as shown in Figure 9(e).

In the previous stage, it is possible to confuse a saddle of degree 2 (i.e. 2-C_2 and 2-C_1) with a saddle of degree 3 (i.e. 3-C_2 and 3-C_1) having only one fixed link in the VST. Our algorithm avoids this problem by tracing the incident links of the target saddle in the VFN. This means that, if the saddle point has one fixed link in the VST and only downward links in the VFN, we trace its links in a direction that reduces the scalar field value until reaching nodes having no downward links in the VFN. If the number of nodes we have reached is 1, the corresponding saddle is definitely a saddle of degree 2 (i.e. 2-C_2 or 2-C_1). Otherwise, we will skip and come back later to check the saddle again in the algorithm, because it may have three incident links in the VST (i.e., it may be a node of either 3-C_2 or 3-C_1). In the previous case, the tracing process finds the saddle p_4 to be of degree 2 (i.e. 2-C_2) because it finds only p_5 as its corresponding end of such a tracing process.

In this way, our algorithm successfully converts the VFN into the VST.

3.5 Skeleton Tree Simplification

Although the VST effectively captures the topological skeleton of a volume dataset, it may suffer from a large number of minor critical points because the input data involves high-frequency noise in some practical cases. Such minor critical points always hide an important global skeleton of the volume dataset. In order to extract the global structure, our algorithm simplifies the constructed VST by removing the noisy minor critical points. The resultant graph will offer important field values for the TF design by checking its remaining nodes.

Figure 10 shows three VST patterns to be removed in the simplification process. The patterns contain a (C_3, 3-C_2) branch, a (C_0, 3-C_1) branch, and a (2-C_2, 2-C_1) pair of joints. Note that the two nodes in the third pattern can have other nodes in between on the VST, while those in the former two branches are immediate neighbors each other. In the simplification process, our algorithm computes the difference in scalar field value between the two nodes of all the patterns, and then selects one pattern to be removed by finding the pair of nodes having the smallest difference. The number of simplification steps is controlled by a threshold that limits the acceptable difference. This simplification process is done one by one until no pattern cannot satisfy the given threshold.

[1]Note that the two links between p_5 and p_6 in Figure 4 are replaced with a single link in Figure 9(a).

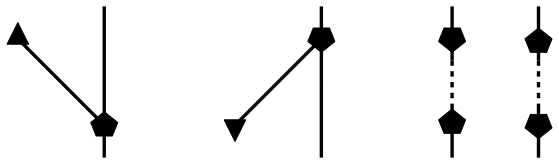


Figure 10: VST patterns to be eliminated in the simplification process.

Through this simplification process, we can distinguish a global skeleton of the volume dataset by cutting off unimportant parts. Note that the simplified VST still satisfies the Euler-Poincaré formula (7) because all the patterns in Figure 10 do not violate this topological consistency. This simplification process successfully reduces the number of CFVs on which we have to focus our attention in designing TFs.

4 Transfer Function Design

Although direct volume rendering can produce semi-transparent images of the entire volume dataset for us to peer inside of, the resultant images generally suffer from their obscure appearance, which stems intrinsically from the self-occluded volumetric projection of resamples along a viewing direction. One of the most significant factors for improving the "*visual clarity*" of such a volume-rendered image is the transfer function (TF), which maps physical fields of a given volume dataset to optical properties, such as color and opacity. In this section, we try to take advantage of the VST-based volumetric field topology description in order to design appropriate TFs semi-automatically for comprehensible volume rendering. See Section 2.2 for a survey of relevant TF designs.

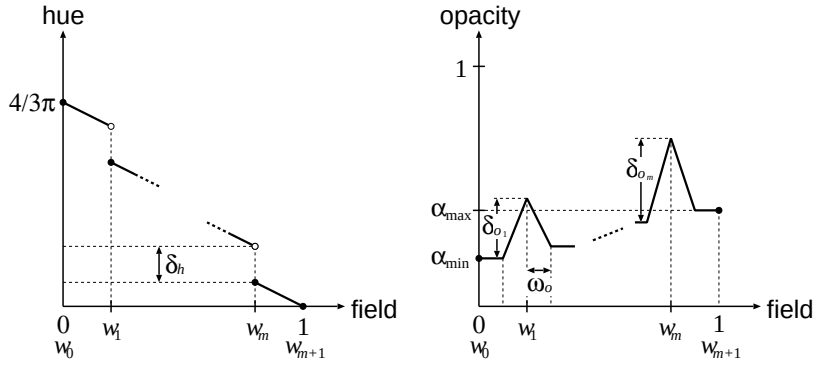
4.1 Principle

The basic idea of our VST-based TF design principle is to accentuate the topological change in volume fields in terms of both color (hue) and opacity. The major premise is that emphasized rendering of characteristic isosurfaces can make the "meaning" of volume rendered images more apparent to the viewer. The good and intuitive analogy can be found in traditional computer-aided design [18], where a variety of design-oriented edges, for example, highlights and outlines, are devised for parameterizing the design quality of surfaces. Our idea can be interpreted as a 3D extension of this.

Although multiple candidates for TF design principles could be proposed, a common principle adopted hereafter will be described below. This common principle helps us emphasize topological transitions contained in isosurfaces of CFVs, which are called *critical isosurfaces* in this paper. Recall that, in our framework, a sequence of non-shrinking isosurfaces can be obtained in a decreasing order of field values.

The color to be assigned in our principle is taken from the top of the HSV hexcone. The color TF is designed so that the rate of change in hue in $[\frac{4}{3}\pi, 0]$ is uniform for all normalized field values, except for a constant jump δ_h at each CFV w_i $(i = 1, \ldots, m)$, where m represents the number of CFVs except for the virtual minimum. Let w_0 and w_{m+1} be 0 and 1, respectively (Figure 11(a)). On the other hand, the opacity TF is designed to be elevated stepwise by some amount at each CFV w_i $(i = 1, \ldots, m)$. The minimum and maximum base opacity values are given by $\alpha_{\min}$ at w_0, and $\alpha_{\max}$, at w_{m+1}, respectively. The height of hat-like accentuation at each CFV can also be changed. The minimum and maximum height are given by $\delta_{o_{\min}}$ at w_1 and $\delta_{o_{\max}}$ at w_m, respectively, while their half-width is set to constant ω_o (Figure 11(b)).

Actually, we tested other design principles as described in our previous work [9, 10, 11]. For example, as presented in the references, the color TF can be designed to have steep linear slopes around CFVs and constant values elsewhere. While this is also effective, it may obscure isosurface transitions because the same color is assigned to isosurfaces of field values between neighboring CFVs. On the other hand, the present color TF has gaps around CFVs that are useful for visually grasping topologically-equivalent sub-volumes since the Gestalt *similarity* rule [7] is considered to hold in the



(a) Color (hue) TF (b) Opacity TF

Figure 11: Design principle of accentuating TFs.

above color (hue) assignment. The opacity TF could also have a different principle where the hat-like elevations are replaced with stepwise elevations. While this would track topological transitions of isosurfaces around CFVs more clearly, it would not fit to the color TF if we allow the constant gaps in color at CFVs. It should be noted here that what distinguishes this design principle from our previous ones [9, 10, 11] lies in the stepwise elevation in the opacity TF, whose purpose is to clarify topological changes in deeper positions. According to the assumption we made in Section 3, the VST makes it possible to detect isosurface embeddings without further manual examination of volumetric structure, which results in a keen field accentuation in the final visualization images.

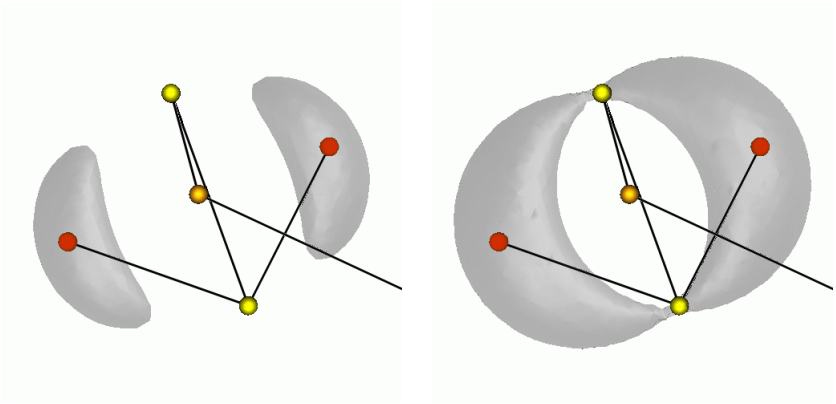
We are now in a right position to apply the above TF design principle to visualizing the analytic volume function defined in Equation (2). For the volume skeletonization, we use our volume analysis and TF design tools that have been implemented on an ordinal PC environment (CPU: Pentium IV, 2GHz, RAM: 1GB). For the purpose of volume rendering, we use AVS/Express 5.1 running on an SGI O2 (CPU: R5000, 180MHz, RAM: 192MB). All the experiments throughout the paper were performed on the same environments.

Figure 12 shows how the isosurfaces evolve in conjunction with the analyzed VST in the case of Equation (2) with $c = 0.60, d = 0.50, R = 0.20$. The volume dataset has $128 \times 128 \times 128$ voxels while its size was reduced to $64 \times 64 \times 64$ in constructing the VST. Note that, in this case, only five out of the six critical points are commonly displayed, and that a C_0-type node p_6 corresponding to the virtual minimum lies outside in the lower right direction (see Figures 4 and 9). Here, two C_3-type nodes p_1 and p_2 are colored in red, two C_2-type nodes p_3 and p_4 in yellow, and a C_1-type node p_5 in orange. The two isosurfaces appear at p_1 and p_2 at the CFV ($w = f(p_{1,2}) = 8c^2d^2 = 0.7200$), and meet at p_3 and p_4 to become a torus at the CFV ($w = f(p_{3,4}) = 8c^2d^2 - 16R^2(c^2 + d^2 - 4R^2) = 0.4320$). The hole of the torus is filled at p_5 (colored in orange) with the CFV ($w = f(p_5) = -2(c^2 - d^2)^2 + 8R^2(c^2 + d^2 - R^2) = 0.1582$) (See Equation (5)).

Figure 13 compares volume rendered images of the analytic volume function (2) with different designs of TFs. The volume rendered image with accentuated TFs in Figure 13(b) captures the inner structures of the volume much more clearly than the case with the naive TF (a linear color TF and an opacity TF with constant α) in Figure 13(a), which exactly reflects the isosurface evolution shown in Figure 12. Note that the scalar field values of the function (2) are normalized to a unit interval $[0, 1]$ in Figure 13.

Furthermore, there is no need to restrict our attention solely to critical isosurfaces. In place of such critical isosurfaces, we can also emphasize *representative isosurfaces* at a field value lying in each interval between two neighboring CFVs, which may be exactly the middle field value in the interval. This can be archived simply by replacing the CFVs with the above representative field values in our TF design principle. These representative isosurfaces reveal stepwise changes in the topology of connected isosurfaces rather than their evolutionary topological transitions around critical points. Figure 13(c) represents such a case where representative isosurfaces of the function (2) are accentuated in place.

Especially, the existence of even croissant-shaped twin isosurfaces occluded by outer thick subvolumes is understood visually thank to the definition of the stepwise opacity TFs for Figures 13(b)



(a) $w = 0.5606$ (b) $w = 0.4320$

(c) $w = 0.3117$ (d) $w = 0.1582$

Figure 12: VSTs and evolving isosurfaces: Analytic volume function. (a) Twin isosurfaces, (b) Twin isosurfaces merging into a torus, (c) A torus, (d) Filling a torus into a sphere.

and (c).

4.2 Applications to Real Datasets

In order to empirically prove the feasibility of the present TF design method, we try to design topologically-accentuated TFs for rendering comprehensibly three different types of volume datasets taken from quantum science and medical science. In all of these datasets, the field values are mapped affinely to the range [0, 1], and critical isosurfaces will be accentuated to clarify isosurface behavior around significant critical points.

For analyzing the fairly complex volume topology using the ordinal PC, we downsized the dataset by simple subsampling.

4.2.1 HIPIP

The first is a well-known testbed volume dataset called *HIPIP* (High Potential Iron Protein) [24], whose resolution is $64 \times 64 \times 64$. For analyzing the volume field topology, we downsized the dataset into $32 \times 32 \times 32$ voxels. The VST generation process yielded $1,268$ critical points, and followed VST simplification reduced the number into 24 ($\#\{C_0\} = 7$, $\#\{C_1\} = 6$, $\#\{C_2\} = 5$, $\#\{C_3\} = 6$), which appears to satisfy the Euler-Poincaré formula in Equation (7).

Figure 14(a) shows TFs which are accentuated on the basis of 17 CFVs, except for 0.5. Figure 14(b) uses the TFs to volume render an HIPIP in a topologically-accentuated manner, where 3D structures of positive and negative wave functions can be seen.

4.2.2 Proton – hydrogen atom collision

The next target is a large-scale 4D dataset, which consists of data for $61 \times 61 \times 61$ volumes over 10^4 time-steps for simulated intermediate-energy collisions of a proton and a hydrogen atom [10]. In this case, volume data at each time step is smooth enough to analyze its original resolution (i.e., 61^3) directly.

The simulation deals with a fundamental ion-atom collision problem, and is very important in that the problem has a wide spectrum of applications such as nuclear fusion, material sciences, and radiology. The purpose here is to investigate how the positive charge of an incident proton affects the behavior of an electron around the target hydrogen atom (Figure 15). A comprehensible illustration of the collision is obtained by visualizing the 3D distorted electron density distribution.

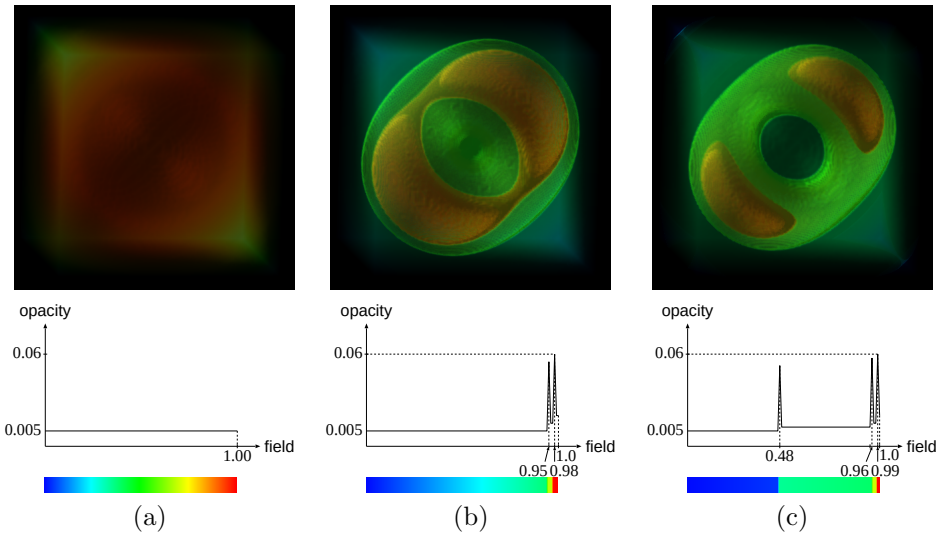
Figure 16 shows two typical volume-rendered snapshots before and just after the collision. Each snapshot was generated with its own TFs accentuated by the VST analysis, and allows us to understand more clearly the inner structures of the distorted electron density distribution. In particular, it appears from Figure 16(b) that a topologically enhanced TFs succeed in capturing an interesting arm-shaped electron density flux connecting the two topological spheres around the hydrogen nuclei, which suggests that the ionization occurs at the timing. This easy-to-miss phenomenon is not visible from an animation that renders the volume at each time step by using the same TFs over the entire time interval.

4.2.3 Tooth

The last example is a medical CT-scanned dataset, called *Tooth* hereafter, which consists of data for $128 \times 128 \times 80$ voxels with 8 bits. The original dataset ($256 \times 256 \times 161$, 16 bits) was used as one of the three target datasets in an exciting panel in *Vis2000* [26, 34]. The dataset suffers from high-frequency noises, and being downsized into $16 \times 16 \times 10$ voxels for VST generation.

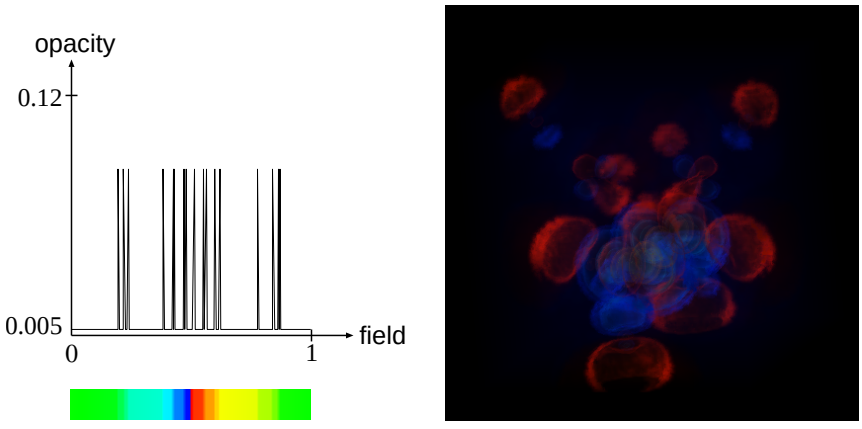
Topological analysis of the tooth volume is characterized by its semi-automaticity, involving three types of manual operations. Since a complete simplification of VST separated only the enamel and pulp materials from the tooth volume, we decided to interrupt the simplification process, producing 16 critical points and 11 CFVs. This effectively allows us to distinguish 4 anatomical regions (cement, enamel, dentin, and pulp) from the tooth. Figure 17(a) shows a tooth image which was volume rendered with the resulting TFs.

Figure 17(b) shows another comprehensible tooth image [11], which was volume rendered with TFs designed by the Kindlmann method [16]. The method relies on a field derivative analysis, and provides a robust and promising way to design boundary-reinforced opacity TF for medical volume



(a) (b) (c)

Figure 13: Visualization results: Analytic volume function. (a) With default TFs ($\alpha = 0.005$), (b) Critical isosurfaces with accentuated TFs ($\delta_h = 0.25$, $\alpha_{\min} = 0.005$, $\delta_{o_{\min}} = 0.05$, $\alpha_{\max} = 0.01$, $\delta_{o_{\max}} = 0.05$, $\omega_o = 0.004$), (c) Representative isosurfaces with accentuated TFs ($\delta_h = 0.25$, $\alpha_{\min} = 0.005$, $\delta_{o_{\min}} = 0.05$, $\alpha_{\max} = 0.01$, $\delta_{o_{\max}} = 0.05$, $\omega_o = 0.004$).



(a) (b)

Figure 14: Visualization result: HIPIP. (a) Accentuated TFs ($\delta_h = 0.05$, $\alpha_{\min} = 0.005$, $\delta_{o_{\min}} = 0.1$, $\alpha_{\max} = 0.005$, $\delta_{o_{\max}} = 0.1$, $\omega_o = 0.004$), (b) Volume rendered image.

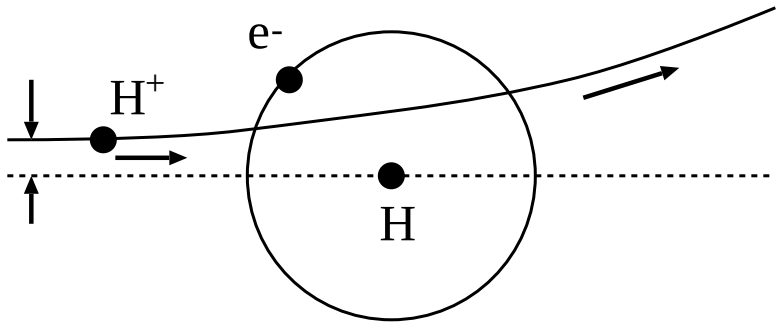


Figure 15: Proton–hydrogen atom collision process.

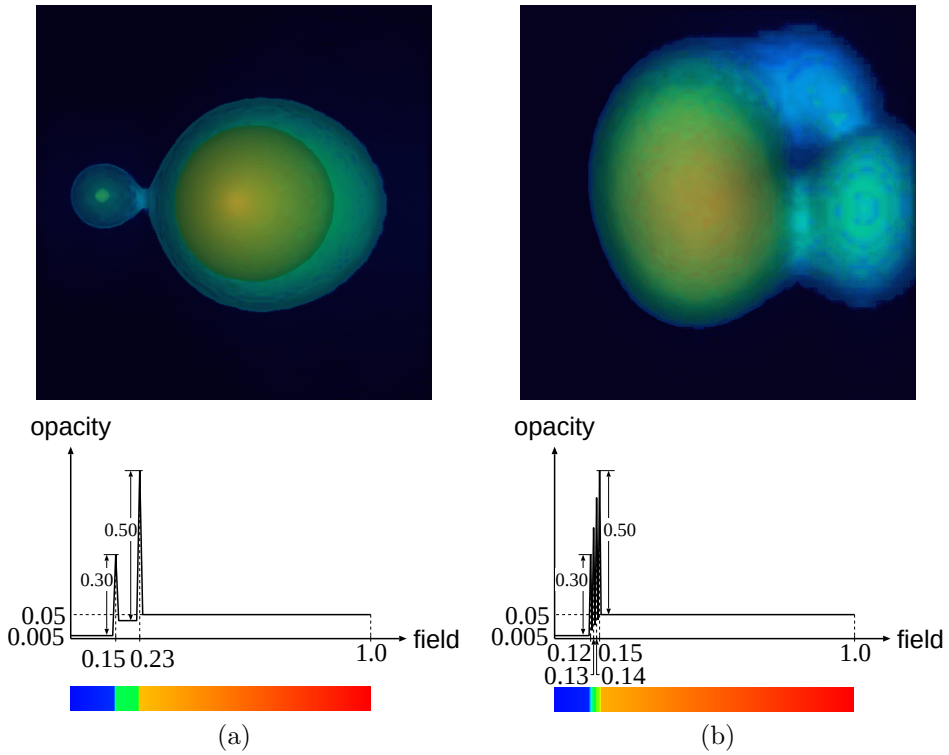


Figure 16: Visualization results: Hydrogen ion-atom collision. (a) Before collision and (b) just after collision. With accentuated TFs ($\delta_h = 0.25$, $\alpha_{\min} = 0.005$, $\delta_{o_{\min}} = 0.05$, $\alpha_{\max} = 0.01$, $\delta_{o_{\max}} = 0.05$, $\omega_o = 0.004$).

data. Indeed, the method won the first prize in the TF competition in the panel [26]. Comparing these images, it turned out that our method can detect almost the same material boundary structures of the tooth volume from the viewpoint of differential topology.

It should be noted here that semi-automatic TF design coupled with a priori domain-specific knowledge about target datasets deserves to be incorporated into the *image-centric* TF design framework [12, 21], because it can avoid the combinatorial explosion of organized sampling for preparing initial designs, and provides the user with the drastically improved serendipity.

4.2.4 Discussions

Computational costs Reasonable computing cost of the present method should be reported. Indeed, our tool installed on the fore-mentioned PC-based environment can usually analyze the volume topology and design accentuated TFs in a few minutes if we control the resolution of the given volume datasets appropriately.

Downsizing effects The algorithm extracts more critical points than expected when handling volume datasets having higher resolution and more complicated inner structures. In such a case, we used the datasets downsized through simple subsampling in place of original ones for the volume topology analysis, in order that we can limit the required amount of memory. It is thought that the detailed information of the dataset is lost with downsizing. However, we can still find significant CFVs of the volume dataset because the VST retains its global structure even after the data is downsized. Actually we applied our algorithm both to the original volume datasets and its downsized ones, and we could not find any information loss in most cases if we carefully control the resolution of downsized datasets because the corresponding VST will be simplified to capture the global structures of the datasets at last. Please note that the volume datasets with original resolution are used for rendering while downsized ones are used for analysis, and there are no fatal mismatches between them in our experiments.

It seems that we can alternatively apply some smooth pre-filtering such as Gaussian filtering, median filtering, or any anisotropic diffusions, before the topological analysis, in order to smooth out local noise that results in minor critical points. However, in our experiments, we confirmed that these pre-filters sometimes produce spurious critical points, which may hide the global structure of the volume data. This is why we simply downsize the volume data and reduce its resolution.

Justification for our TF design principle Topologically-accentuation of critical isosurfaces does not always produce the best TFs for arbitrary volume datasets. For example, the analysis of scalar fields that characterize an underlying flow fields, such as velocity, pressure, and helicity, requires their detailed behavior relatively far from the critical points. In such a case, our visualization system will alternatively accentuate representative isosurfaces instead of critical ones so that we can infer flow behavior around the critical points. In this way, the topological analysis provides us with a good initial guess, which can reduce the number of re-specification of TFs, and thus making the subsequent visual exploration process more efficient.

Another possibility of improving our design principle lies in judicious hue assignment for color (hue) TFs and its automatic computation. One of the keys to solving this problem is psychological factors of color science. Actually, in our implementation, we take advantage of the psychological fact that warm colors (e.g. red) advance while cold colors (e.g. blue) recede, because we clarify the inner structure of a given volume by assigning warm colors to inner isosurfaces while cold colors to outer isosurfaces. Although the hue assignment here has been commonly used for contemporary volume visualizers, this indeed takes into account the spatial configuration of isosurfaces derived from our volume skeletonization framework. A similar idea can be found in the work by Rheingans and Ebert [27], where they established a systematic color assignment in their non-photorealistic volume rendering schemes.

Comparison between results of different data types In general, volume datasets are classified into three categories according to the types of objects involved: simulated, acquired, and modeled datasets. Since the present method is distinguished by its ability to extract global topological skeletons robustly, it is most suitable to the simulated datasets where we cannot extract their significant features only by using local derivatives (See Figures 14 and 16). On the other hand, the acquired datasets usually contain much noise that is likely to hide the global structures embedded inside. However, our method still works well for this type of data because it can extract global transitions of isosurfaces by simplifying the obtained VST (See Figure 17). The modeled datasets are the most unsuitable of these three types because they involve many degenerate parts such as voxel clusters having the same scalar

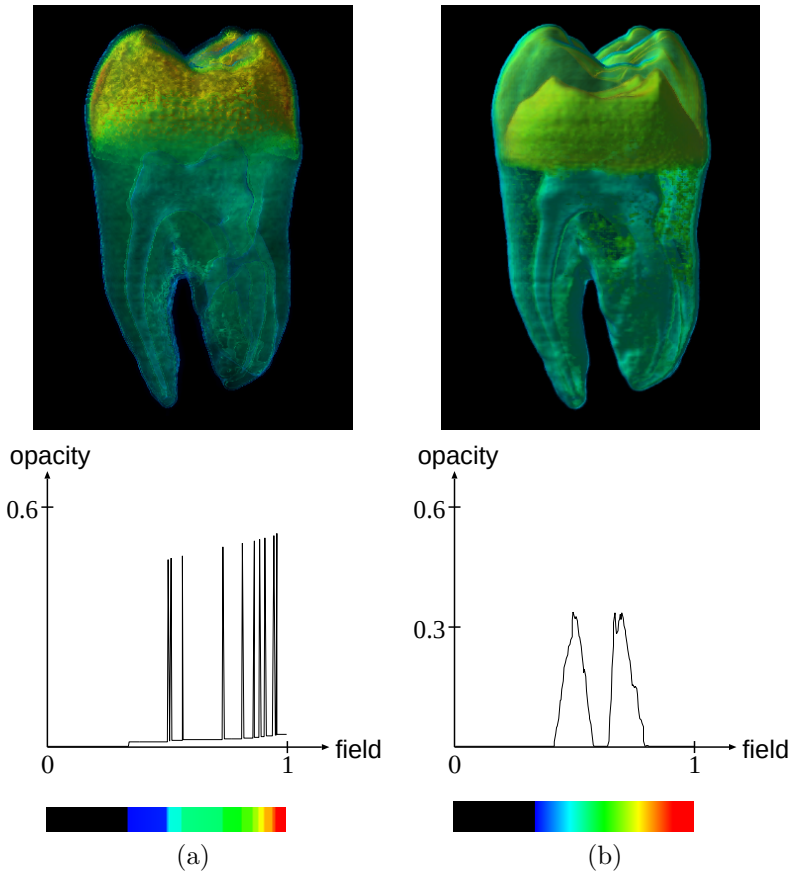


Figure 17: Visualization results: NLM Tooth volume. (a) Our method. With accentuated TFs ($\delta_h = 0.05$, $\alpha_{\min} = 0.01$, $\delta_{o_{\min}} = 0.4$, $\alpha_{\max} = 0.03$, $\delta_{o_{\max}} = 0.5$, $\omega_o = 0.004$), (b) the Kindlmann method.

field values, i.e., level field clusters (See Section 3.2). These clusters yield many unimportant critical points in the topological analysis of the volume datasets. Although our method is resistant to this type of volume dataset, it usually requires more memory storage and computation time. We are now investigating the multiresolution control of volume datasets that sophisticates our present framework.

5 Conclusions

This paper has presented a new data-centric approach to semi-automatic TF design for comprehensible volume rendering. The key idea of the approach is to extract a VST that characterizes the topological changes of isosurfaces varying over the domain of the scalar field value. Our contributions also include a robust algorithm for constructing a VST even from volume datasets with noise and restricted precision. The VST-based TF design successfully enhances our visualization capability by accentuating significant topological changes in volume fields using information about the global structure of isosurface evolution and its spatial embeddings.

When accentuating the color (hue) and opacity of TFs at CFVs, our scheme equally emphasizes voxels around isosurfaces associated with the corresponding CFVs. This sometimes obscures the clarity of rendered images because our scheme also exposes isosurfaces unrelated to the target topological changes. The VST allows us to differentiate connected components of isosurfaces associated with the topological change from unrelated ones because it explicitly describes how each individual isosurface evolves at different field values. It is also important to elaborate the simplification process of the VST to capture the global skeleton of volume fields. Automatic selection of a threshold must be developed so that it can control the simplification process by considering local field behavior in the dataset. Another advantage of the VST is that it can find the noisy field intervals where numerous topological changes occur in isosurfaces. This leads us to a concept of volume-skeleton-based smoothing that replaces the field values in such noisy field interval with those obtained using trilinear interpolation in the volume dataset. Our future extension also contains a sophisticated model to handle degeneracy especially involved in modeled volume datasets. This will require multiresolution control of VSTs, which is absolutely important for the analysis of large-scale volume datasets and their out-of-core visualization implementation.

Acknowledgements

We have benefited from continuous discussions with Xiaoyang Mao, Haruo Hosoya, Koji Koyamada, Sam Uselton, Yumi Yamaguchi, and Gregory M. Nielson. This work has been partially supported by Japan Society of the Promotion of Science under Grant-in-Aid for Scientific Research (C) No. 11680349 and No. 13680401, Young Scientists (B) No. 14780189 and No. 15700081, and Mitsubishi Precision Co., Ltd.

References

[1] C. L. Bajaj, V. Pascucci, and D. R. Schikore. The contour spectrum. In *Proceedings of IEEE Visualization '97*, pp. 167–173, 539, Oct. 1997.

[2] H. Carr, J. Snoeyink, and U. Axen. Computing contour trees in all dimensions. *Computational Geometry*, Vol. 24, No. 2, pp. 75–94, 2003.

[3] S. Castro, A. König, H. Löffelmann, and E. Gröller. Transfer function specification for the visualization of medical data. Technical Report TR–186–2–98–12, Vienna University of Technology, Mar. 1998. [http://www.cg.tuwien.ac.at/research/TR/98/TR-186-2-98-12Abstract.html].

[4] H. Edelsbrunner, J. Harer, and A. Zomorodian. Hierarchical Morse-Smale complexes for piecewise linear 2-manifolds. *Discrete & Computational Geometry*, Vol. 30, No. 1, pp. 87–107, 2003.

[5] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. *Discrete & Computational Geometry*, Vol. 28, No. 4, pp. 511–533, 2002.

[6] S. Fang, T. Biddlecome, and M. Tuceryan. Image-based transfer function design for data exploration. In *Proceedings of IEEE Visualization '98*, pp. 319–326, 546, Oct. 1998.

[7] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics Principles and Practice*, pp. 418–423. Addison-Wesley, 2nd edition, 1990.

[8] A. T. Fomenko and T. L. Kunii. *Topological Modeling for Visualization*, pp. 105–125. Springer-Verlag, 1997.

[9] I. Fujishiro, T. Azuma, and Y. Takeshima. Automating transfer function design for comprehensible rendering based on 3D field topology analysis. In *Proceedings of IEEE Visualization '99*, pp. 467–470, 563. IEE Computer Society Press, Oct. 1999.

[10] I. Fujishiro, T. Azuma, Y. Takeshima, and S. Takahashi. Volume data mining using 3D field topology analysis. *IEEE Computer Graphics and Applications*, Vol. 20, No. 5, pp. 46–51, 2000.

[11] I. Fujishiro, Y. Takeshima, S. Takahashi, and Y. Yamaguchi. Topologically-accentuated volume rendering. In F. H. Post, G. M. Nielson, and G.-P. Bonneau, editors, *Data Visualization: The State of the Art*, pp. 95–108. Kluwer Academic Publishes, 2003.

[12] T. He, L. Hong, A. Kaufman, and H. Pfister. Generation of transfer functions with stochastic search techniques. In *Proceedings of IEEE Visualization '96*, pp. 227–234, 489, Oct. 1996.

[13] J. Hladuvka, A. König, and E. Gröller. Curvature-based transfer functions for direct volume rendering. In G. Falcidieno, editor, *Proceedings of Spring Conference on Computer Graphics and its Applications 2000*, pp. 58–65, May 2000.

[14] T. Itoh and K. Koyamada. Automatic isosurface propagation using an extrema graph and sorted boundary cell lists. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 1, No. 4, pp. 319–327, 1995.

[15] T. Itoh, Y. Yamaguchi, and K. Koyamada. Fast isosurface generation using the volume thinning algorithm. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 7, No. 1, pp. 32–46, 2001.

[16] G. Kindlman and J.W. Durkin. Semi-automatic generation of transfer functions for direct volume rendering. In *Proceedings of 1998 IEEE Symposium on Volume Visualization*, pp. 79–86, 170, Oct. 1998.

[17] J. Kniss, G. Kindlmann, and C. Hansen. Multidimensional transfer functions for interactive volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 8, No. 3, pp. 270–285, 2002.

[18] K. Kondo, F. Kimura, and T. Tajima. An interactive rendering technique for 3-D shapes. In *Proceedings of EUROGRAPHICS'85*, pp. 341–352, 1985.

[19] I. S. Kweon and T. Kanade. Extracting topographic terrain features from elevation maps. *CVGIP : Image Understanding*, Vol. 59, No. 2, pp. 171–182, Mar. 1994.

[20] F. Lazarus and A. Verroust. Level set diagrams of polyhedral objects. In *Proceedings of the Fifth ACM Symposium on Solid Modeling and Applications*, pp. 130–140. ACM Press, 1999.

[21] J. Marks et al. Design galleries: A general approach to setting parameters for computer graphics and animation. In *Computer Graphics (Proceedings of Siggraph '97)*, pp. 389–400, Aug. 1997.

[22] W. S. Massey. *A Basic Course in Algebraic Topology.* Springer-Verlag, 1991.

[23] G. M. Nielson and B. Hamann. The asymptotic decider: Resolving the ambiguity in marching cubes. In *Proceedings of IEEE Visualization '91*, pp. 83–91, 413, Oct. 1991.

[24] L. Noodleman and D. Case. University of North Carolina at Chapel Hill.

[25] V. Pascucci and K. Cole-McLaughlin. Efficient computation of the topology of level sets. In *Proceedings of IEEE Visualization 2002*, pp. 187–194, Oct. 2002.

[26] H. Pfister et al. The transfer function bake-off. *Computer Graphics and Applications*, Vol. 21, No. 3, pp. 16–22, 2001.

[27] P. Rheingans and D. Ebert. Volume illustration: Nonphotorealistic rendering of volume models. *IEEE Transactions on Visualization and Computer Graphics*, Vol. 7, No. 3, pp. 253–264, 2001.

[28] Y. Shinagawa, Y. L. Kergosien, and T. L. Kunii. Surface coding based on Morse theory. *IEEE Computer Graphics and Applications*, Vol. 11, No. 5, pp. 66–78, Sep. 1991.

[29] S. Takahashi, T. Ikeda, Y. Shinagawa, T. L. Kunii, and M. Ueda. Algorithms for extracting correct critical points and constructing topological graphs from discrete geographical elevation data. *Computer Graphics Forum*, Vol. 14, No. 3, pp. 181–192, 1995.

[30] S. P. Tarasov and M. N. Vyalyi. Construction of contour trees in 3D in $O(n \log n)$ steps. In *14th ACM Symposium on Computational Geometry*, pp. 68–75, 1998.

[31] S. Tenginakai, J. Lee, and R. Machiraju. Salient iso-surface detection with model-independent statistical signatures. In *Proceedings of IEEE Visualization 2001*, pp. 231–238, Oct. 2001.

[32] M. van Kreveld, R. van Oostrum, C. Bajaj, V. Pascucci, and D. Schikore. Contour trees and small seed sets for isosurface traversal. In *13th ACM Symposium on Computational Geometry*, pp. 212–220, 1997.

[33] G. H. Weber, G. Scheuermann, H. Hagen, and B. Hamann. Exploring scalar fields using critical isovalues. In *Proceedings of IEEE Visualization 2002*, pp. 171–178, Oct. 2002.

[34] T. Woo. The National Library of Medicine of the National Institutes of Health [http://visual.nlm.nih.gov/data/].